

FÓTHI ÁKOS

# BEVEZETÉS

A

# PROGRAMOZÁSHOZ<sup>1</sup>

Második, javított kiadás

ELTE Informatikai Kar, 2005.

A mű digitális megjelenítése az Oktatási Minisztérium támogatásával,  
a Felsőoktatási Tankönyv- és Szakkönyv-támogatási Pályázat keretében történt.

ý

---

<sup>1</sup>Az ELTE IK Elektronikus Könyvtár által közvetített digitális tartalmat a felhasználó a szerzői jogról szóló 1999. évi LXXVI. tv. 33. §(4) bekezdésében meghatározott oktatási, illetve tudományos kutatási célra használhatja fel. A felhasználó a digitális tartalmat képernyőn megjelenítheti, letöltheti, arról elektronikus adathordozóra vagy papíralapon másolatot készíthet, adatrögzítő rendszerében tárolhatja. Az ELTE IK Elektronikus Könyvtár weblapján található digitális tartalmak üzletszerű felhasználása tilos, valamint a szerzők írásbeli hozzájárulása nélkül kizárt a digitális tartalom módosítása és átdolgozása, illetve az ilyen módon keletkezett származékos anyag további felhasználása.



# Tartalomjegyzék

<b>1. Alapfogalmak</b>	<b>11</b>
1.1. Halmazok . . . . .	11
1.2. Sorozatok . . . . .	12
1.3. Relációk . . . . .	12
1.3.1. Műveletek . . . . .	14
1.3.2. Logikai relációk . . . . .	16
1.4. Direktszorzat . . . . .	17
1.5. Függvényterek . . . . .	18
1.6. Példák . . . . .	19
1.7. Feladatok . . . . .	24
<b>2. A programozás alapfogalmai</b>	<b>29</b>
2.1. Az állapottér fogalma . . . . .	29
2.2. A feladat . . . . .	30
2.3. A program . . . . .	32
2.4. A programfüggvény . . . . .	33
2.5. Megoldás . . . . .	34
2.6. Programozási feladat . . . . .	35
2.7. Példák . . . . .	36
2.8. Feladatok . . . . .	40
<b>3. Specifikáció</b>	<b>43</b>
3.1. A leggyengébb előfeltétel . . . . .	43
3.2. A feladat specifikációja . . . . .	45
3.3. A változó fogalma . . . . .	47
3.4. Példák . . . . .	50
3.5. Feladatok . . . . .	52
<b>4. Kiterjesztések</b>	<b>57</b>
4.1. A feladat kiterjesztése . . . . .	57
4.2. A program kiterjesztése . . . . .	58

4.3.	Kiterjesztési tételek . . . . .	59
4.4.	A feladat kiterjesztése és a specifikáció tétele . . . . .	65
4.5.	A paramétertér kiterjesztése . . . . .	65
4.6.	Példák . . . . .	66
4.7.	Feladatok . . . . .	68
<b>5.</b>	<b>A megoldás fogalmának általánosítása</b>	<b>69</b>
5.1.	A megoldás fogalmának kiterjesztése . . . . .	69
5.2.	Megoldás ekvivalens állapottéren . . . . .	70
5.3.	Reláció szerinti megoldás . . . . .	71
<b>6.</b>	<b>Programkonstrukciók</b>	<b>73</b>
6.1.	Megengedett konstrukciók . . . . .	73
6.2.	A programkonstrukciók programfüggvénye . . . . .	79
6.3.	Feladatok . . . . .	81
<b>7.</b>	<b>Levezetési szabályok</b>	<b>85</b>
7.1.	A szekvencia levezetési szabálya . . . . .	85
7.2.	Az elágazás levezetési szabálya . . . . .	86
7.3.	A ciklus levezetési szabálya . . . . .	88
7.4.	Feladatok . . . . .	92
<b>8.</b>	<b>Elemi programok</b>	<b>95</b>
8.1.	Elemi programok . . . . .	95
8.2.	Elemi programok leggyengébb előfeltétele . . . . .	97
8.3.	Az értékadás mint feladatspecifikáció . . . . .	99
8.4.	Feladatok . . . . .	100
<b>9.</b>	<b>A programkonstrukciók és a kiszámíthatóság</b>	<b>101</b>
9.1.	Parciálisan rekurzív függvények . . . . .	101
9.2.	A parciális rekurzív függvények kiszámítása . . . . .	103
9.3.	Relációk . . . . .	107
<b>10.</b>	<b>A típus</b>	<b>109</b>
10.1.	A típusspecifikáció . . . . .	109
10.2.	A típus . . . . .	110
10.3.	A típusspecifikáció tétele . . . . .	112
10.4.	Absztrakt típus . . . . .	112
10.5.	Példák . . . . .	113
10.6.	Feladatok . . . . .	117

<b>11. Típuskonstrukciók</b>	<b>121</b>
11.1. A megengedett konstrukciók . . . . .	121
11.2. Szelektorfüggvények . . . . .	124
11.3. Az iterált specifikációs függvényei . . . . .	126
11.4. A függvénytípus . . . . .	126
11.5. A típuskonstrukciók típusműveletei . . . . .	128
<b>12. Programozási tételek (Levezetés)</b>	<b>131</b>
12.1. Programozási tételek intervallumon . . . . .	131
12.1.1. Összegzés . . . . .	133
12.1.2. Számlálás . . . . .	134
12.1.3. Maximumkeresés . . . . .	136
12.1.4. Feltételes maximumkeresés . . . . .	137
12.1.5. Lineáris keresés . . . . .	138
12.2. Tételek „feltételig” változata . . . . .	139
12.2.1. Összegzés feltételig . . . . .	140
12.2.2. Számlálás feltételig . . . . .	141
12.2.3. Maximumkeresés feltételig . . . . .	142
12.2.4. Feltételes maximumkeresés feltételig . . . . .	142
12.2.5. Lineáris keresés . . . . .	143
12.2.6. Tételek másképpen . . . . .	145
12.3. Programozási tételek halmazon . . . . .	147
12.4. Bonyolultabb feladatok . . . . .	149
12.4.1. Logaritmikus keresés . . . . .	149
12.4.2. Visszalépéses keresés . . . . .	151
12.4.3. Visszalépéses számlálás . . . . .	155
12.5. Függvényérték kiszámítása . . . . .	156
12.5.1. Függvénykompozícióval adott függvény kiszámítása . . . . .	156
12.5.2. Esetszétválasztással adott függvény kiszámítása . . . . .	157
12.5.3. Rekurzív formulával adott függvény kiszámítása . . . . .	157
12.5.4. Elemenként feldolgozható függvény . . . . .	158
12.6. Feladatok . . . . .	164
<b>13. Transzformációk</b>	<b>167</b>
13.1. Visszavezetés . . . . .	168
13.2. Egyszerű programtranszformációk . . . . .	168
13.3. Típustranzformációk . . . . .	171
13.4. Állapottér-transzformáció . . . . .	176
13.4.1. Szekvenciális megfelelő . . . . .	176
13.4.2. Példa állapottér-transzformációra . . . . .	178
13.5. Programinverzió . . . . .	181
13.5.1. Egyváltozós eset . . . . .	181
13.5.2. Kétváltozós eset . . . . .	182

13.6. Példák . . . . .	184
13.7. Feladatok . . . . .	196
<b>14. Absztrakciós stratégia</b>	<b>203</b>
14.1. Az időszerűsítés definíciója . . . . .	204
14.2. Időszerűsítés egyértelmű módosítófajllal . . . . .	207
14.2.1. Visszavezetés halmazok uniójára . . . . .	207
14.2.2. Visszavezetés egyváltozós elemenkénti feldolgozásra . . . . .	209
14.2.3. Visszavezetés kétváltozós elemenkénti feldolgozásra . . . . .	211
14.3. Időszerűsítés nem egyértelmű módosítófajllal . . . . .	212
14.3.1. Megoldás adatabsztrakcióval . . . . .	212
14.3.2. Megoldás függvényabsztrakcióval . . . . .	215
14.4. Feladatok . . . . .	221
<b>15. Megoldások</b>	<b>225</b>
15.1. Alapfogalmak . . . . .	225
15.2. A programozás alapfogalmai . . . . .	236
15.3. Specifikáció . . . . .	239
15.4. Kiterjesztés . . . . .	249
15.5. Programkonstrukciók . . . . .	251
15.6. Levezetési szabályok . . . . .	262
15.7. Elemi programok . . . . .	265
15.8. Típus . . . . .	266
15.9. Levezetés . . . . .	272
15.10. Visszavezetés . . . . .	300
15.11. Absztrakciós stratégia . . . . .	345
<b>Tárgymutató</b>	<b>360</b>
<b>Irodalomjegyzék</b>	<b>363</b>

# Bevezetés

A könyvben ismertett programozási modell megfogalmazásának elsődleges célja az volt, hogy elméleti háttérrel adjon a lépésenkénti finomítás módszeréhez – az algoritmusok és az adatszerkezetek vonatkozásában egyaránt. Időközben ezen sikerült jelentősen túllépni, s a jelenlegi legfontosabb cél a tudatos programozás támogatása.

Bár a modell eszközei segítségével a feladat specifikációját helyettesíteni tudjuk olyan feladatok specifikációival, amely feladatok megoldása esetén a rendelkezésre álló matematikai eszközökkel belátható az eredeti feladat megoldásának helyessége, nem célunk az automatikus programszintézis, ezért nem fordultunk a formális eszközök, módszerek felé.

A programnyelvek, programok vizsgálata, helyességük bizonyítása már több mint negyven évre nyúlik vissza, sőt néhány fontos eredmény ennél is régebbi. Maga a programozás is nagyon hamar a vizsgálatok tárgyává vált, mind elméleti, mind gyakorlati szempontból. Itt csak utalunk a relációs modell számunkra legfontosabb előzményeire, és megpróbáljuk elhelyezni a véleményünk szerint legjelentősebb irányzatok között.

## Előzmények

A bemutatott modell négy fontos előzményre épít. Dijkstra „programozási diszciplínájá”-ra [Dij 76], Hoare típuskezelésére [Hoa 72], Jackson programtervezési módszerére [Jac 75] és Mills matematikai megközelítésére [Mills 72].

A 60-as években kibontakozó, szoftverkrízisnek nevezett jelenség jelentős lendületet adott a programozással kapcsolatos kutatásoknak. Ezeknek szinte szimbólumává vált Dijkstra nevezetes cikke [Dij 68], és a programnyelvek által a kezdetek óta támogatott moduláris programozás mellett vagy inkább után a strukturált programozásnak nevezett irányzat kiinduló pontja lett.

Az általa bevezetett fogalmakra, módszerekre [Dij 76] nagyon sokan építettek, így a relációs modellünk is. Számunkra legfontosabbak a leggyengébb előfeltétel, a levezetési szabályok, a nemdeterminisztikusság, az állapotér, a Dijkstra által használt, bár Hoare-tól származó [Hoa 72] elő- és utófeltétellel történő specifikáció.

Habár Dijkstra ebben a könyvében logikai jellegű megoldásokat használt, nem törekedett szigorú formalizálásra, ezt rövid időn belül nagyon sok cikk és könyv [Gri 81]

megtette. A formalizálás legtöbbször logikai eszközökkel történt, aminek következtében háttérbe szorult az állapotér fogalma és a leggyengébb előfeltételt predikátum-transzformátorként kezelték.

Jelentősen befolyásolta a relációs modellünk kialakítását Mills [Mills 72] matematikai megközelítési módja. A programfüggvény fogalma, a függvény lezártjának és a ciklusnak a kapcsolata, a memória modellezése az, amit elsősorban figyelembe vettünk. A nemdeterminisztikusság kezelésére kézenfekvő megoldás volt a relációk használata.

A programozás problémáinak más jellegű megközelítését adta Jackson [Jac 75], ami a napjainkban félformálisnak nevezett módszerek egyik ősénekin tekinthető, és a gyakorlatban sokszor alkalmazták. Ezeket a programtervezési módszereket is igyekeztünk kezelhetővé tenni a relációs modellben.

### Programozási paradigmák

A jelenlegi fő programozási irányzatok két nagy csoportba sorolhatók. Az egyikbe tartoznak a *formális módszerek*, amelyeknek az alapja egy-egy formális matematikai diszciplína.

A  $\lambda$  kalkuluson, illetve annak kiterjesztésein alapul a *funkcionális* paradigma. A nagy megbízhatóságú, bonyolult feladatokat megoldó programok iránti igény és a rendelkezésre álló erőforrások hihetetlen növekedése miatt a funkcionális megközelítés már nem elméleti, hanem elsősorban gyakorlati jelentőségű.

A *logikai alapú* programozás (modellezés) – különösen a logika különböző kiterjesztései, pl. temporális logika következtében – sok általános és speciális rendszer keretében jelenik meg.

Figyelemre méltó karriert fut be a *B-módszer* [Abr 96], ami Dijkstra és Hoare említett munkáin alapul. Az absztrakt gép fogalmára épül, és a Zermelo–Fraenkel axiomákon alapuló B nyelvet használja a rendszerkészítés minden fázisában.

A másik nagy csoportba tartoznak az úgynevezett *félformális rendszerek*. Ezek általában az objektumelvű modellalkotás támogatói, mint pl. az UML, de ide sorolhatjuk a *tervintaalapú* tervezést [GHJV 95] is, ami a klasszikus eljáráskönyvtár általánosításának is tekinthető. Ezeknek a rendszereknek lényeges gyakorlati eleme a kód-újrafelhasználás, amit szintén az erőforrás-növekedés tesz lehetővé.

A relációs modellt az első csoporthoz a *levezetés*, a másodikhoz a *visszavezetés* fogalma kapcsolja.

A legújabb irányzatok közé tartoznak a *multiparadigmális* megközelítések [Bud 95], amikor a módszer alapja nem egyetlen absztraháló fogalom, illetve absztrakciós stratégia. Ebbe a csoportba soroljuk relációs modellünket is, de mi nem az eszközökkel, hanem a mögöttük levő elméleti összefüggésekkel foglalkozunk.



## Az egyes fejezetekről

Az első részben összefoglaljuk a könyvben felhasznált legfontosabb matematikai fogalmakat, jelöléseket, és bevezetünk a relációkkal kapcsolatban néhány speciális fogalmat.

A következő két részben bevezetjük a relációalapú modell alapfogalmait: állapot-tér, feladat, program, megoldás. Megadjuk a megoldás egy elégséges feltételét, a gyakorlati szempontból is fontos specifikáció tételét, ehhez felhasználjuk a változó és a leggyengébb előfeltétel fogalmát.

A 4. és az 5. fejezetben általánosítjuk az eddig bevezetett fogalmakat. E fejezetek jelentősége elsősorban elméleti. Első olvasásra kihagyhatók.

A következő három részben a szokásos szekvenciális programkonstrukciókat vezetjük be, és megfogalmazzunk néhány velük kapcsolatos, a lépésenkénti finomítás szempontjából hasznos tételt.

A 9. fejezet szintén elméleti szempontból érdekes, és nem szükséges a továbbiak megértéséhez.

Ezután a programozás legfontosabb fogalmával, a típussal foglalkozunk a 10. és a 11. fejezetben. A típussal kapcsolatos fogalmak nemcsak fontosak, hanem elég bonyolultak is. Első olvasásra nem is kell törekedni alapos megismerésükre. Célszerű az egész könyv végigolvasása után visszatérni ezekre a fejezetekre.

A következő három fejezetben azt mutatjuk meg, hogyan lehet gyakorlati programozási feladatokat megoldani. A 12. fejezetben a levezetési szabályok alkalmazásával, a feladatot lépésenként finomítva jutunk el a bizonyítottan helyes megoldó programhoz. A 13. fejezetben a visszavezetést mutatjuk be, azaz meglévő megoldásokat, tételeket használunk fel. A 14. fejezetben egy összetettebb feladatot oldunk meg különböző módokon.

Az utolsó részben a fejezetek végén szereplő feladatok megoldásához adunk útmutatót, sőt sok esetben megadunk egy lehetséges megoldást is.

## Köszönetnyilvánítás

Ez a könyv az Eötvös Loránd Tudományegyetemen a programtervező matematikusok számára tartott előadásaim alapján készült. A több mint húsz éve megjelent jegyzethez képest sok minden változott. Ez alatt az idő alatt sok volt és jelenlegi kollégám vett részt a tárgy oktatásában, és segítette fejlődését. A húsz év alatt több ezer hallgatóval találkoztam ezen előadások kapcsán, az ő negatív és néha pozitív véleményük is sok segítséget jelentett nekem.

Jelenlegi és volt kollégáim: Fekete István, Gregorics Tibor, Horváth Zoltán, Konczné Nagy Márta, Kozics Sándor, Kozsik Tamás, Nyékyné Gaizler Judit, Sike Sándor, Steingart Ferenc, Szabóné Nacs Rozália, Tőke Pál, Varga Zoltán, Vargyas Miklós, Venczel Tibor segítségéért köszönettel tartozom.

Külön köszönöm Steingart Ferencnek azt a nagy munkát, amelyet a könyv első elektronikus jegyzet változatának elkészítésével végzett, nélküle nem vágtam volna bele ennek a könyvnek a megírásába.

Köszönöm Bozsik József, Kovács Péter, Riskó Gergely és Sztupák Szilárd Zsolt hallgatóknak, hogy elkészítették a feladatok megoldását. Kovács Péter, Bozsik József segítségével a 15.1.–15.8., Riskó Gergely a 15.9. és 15.11., Sztupák Szilárd Zsolt a 15.10. rész megoldásait készítette el.

Köszönöm Hunyadvári László gondos lektori munkáját, rengeteg hasznos észrevételét.

### **Ajánlás**

Ajánlom e könyvet elsősorban a programtervező matematikus, programtervező informatikus hallgatóknak. Ajánlom azoknak is, akik programozással foglalkoznak, szeretik a kihívásokat és nyitottak. Ez a könyv nem könnyű olvasmány és nem is hasznos ismeretek, receptek gyűjteménye, de aki figyelmesen, gondolkodva elolvassa, valószínűleg más szemmel fogja látni az eddigi munkáját is.

A járt út biztonsága helyett ajánlom a járatlan út izgalmát és reményét.

# 1. fejezet

## Alapfogalmak

Ebben a részben bevezetjük azokat a jelöléseket és alapvető definíciókat, amelyeket a továbbiakban gyakran fogunk használni. Ezek legnagyobb része középiskolából vagy bevezető jellegű matematikai tanulmányokból ismert.

### 1.1. Halmazok

Először bevezetjük a matematikában gyakran használt halmazok jelöléseit.

$\mathbb{N}$	–	a természetes számok halmaza,
$\mathbb{N}_0$	–	a nemnegatív egészek halmaza,
$\mathbb{Z}$	–	az egész számok halmaza,
$\mathbb{L}$	–	a logikai értékek halmaza,
$\emptyset$	–	az üres halmaz.

Szokás a természetes számok halmazába a nullát is beleértetni, de ebben a könyvben erre külön jelölést ( $\mathbb{N}_0$ ) használunk.

A halmazokat gyakran vagy az elemeik felsorolásával

$$\mathbb{L} ::= \{igaz, hamis\},$$

vagy egy tulajdonság megfogalmazásával

$$[a..b] ::= \{x \in \mathbb{Z} \mid x \geq a \text{ és } x \leq b\}$$

adjuk meg.

Természetesen használni fogjuk a matematikában megszokott halmazelméleti műveleteket

$\cup$	–	unió,
$\cap$	–	metszet,
$\setminus$	–	különbség

és relációkat is

$\in$	–	elem,
$\subseteq$	–	részalmaz,
$\subset$	–	valódi része.

Egy  $H$  halmaz számosságát  $|H|$  jelöli. Azt, hogy egy  $H$  halmaz véges, néha így is írjuk:  $|H| < \infty$ .

Egy  $H$  halmaz részalmazainak halmazát, azaz a hatványalmazát  $\wp(H)$ -val, véges részalmazainak halmazát  $\mathfrak{F}(H)$ -val jelöljük.

## 1.2. Sorozatok

Ha  $A$  egy adott halmaz, akkor az  $\alpha = \langle \alpha_1, \alpha_2, \dots \rangle$ ,  $\alpha_i \in A$  egy  $A$ -beli véges vagy végtelen sorozatot jelöl.

Az  $A$ -beli véges sorozatokat  $\alpha = \langle \alpha_1, \alpha_2, \dots, \alpha_n \rangle$ ,  $\alpha_i \in A$  alakban írhatjuk le. Egy  $\alpha$  véges sorozat hosszát  $|\alpha|$  jelöli.

Az  $A$ -beli véges sorozatok halmazát  $A^*$ -gal, a végtelen sorozatok halmazát  $A^\infty$ -nel jelöljük. Az előző két halmaz uniójaként előálló  $A$ -beli véges vagy végtelen sorozatok halmazára az  $A^{**}$  jelölést használjuk.

Egy  $\alpha \in A^{**}$  sorozat értelmezési tartományát  $\mathcal{D}_\alpha$ -val jelöljük, és a következő halmazt értjük rajta:

$$\mathcal{D}_\alpha ::= \begin{cases} [1..|\alpha|], & \text{ha } \alpha \in A^*; \\ \mathbb{N}, & \text{ha } \alpha \in A^\infty. \end{cases}$$

Legyenek  $\alpha^1, \alpha^2, \dots, \alpha^{n-1} \in A^*$  és  $\alpha^n \in A^{**}$ . Ekkor azt a sorozatot, amelyet az  $\alpha^1, \alpha^2, \dots, \alpha^{n-1}, \alpha^n$  sorozatok egymás után írásával kapunk, a fenti sorozatok *konkatenációjának* nevezzük, és  $\text{kon}(\alpha^1, \alpha^2, \dots, \alpha^{n-1}, \alpha^n)$ -nel jelöljük.

Egy  $A^{**}$ -beli sorozat *redukáltjának* nevezzük azt a sorozatot, amelyet úgy kapunk, hogy az eredeti sorozat minden azonos elemekből álló véges részsorozatát a részsorozat egyetlen elemével helyettesítjük. Egy  $\alpha \in A^{**}$  sorozat redukáltját  $\text{red}(\alpha)$ -val jelöljük.

Bevezetjük még a  $\tau$  függvényt, amely egy véges sorozathoz hozzárendeli annak utolsó elemét:  $\tau : A^* \rightarrow A, \forall \alpha \in A^*$ :

$$\tau(\alpha) ::= \alpha_{|\alpha|}.$$

## 1.3. Relációk

Legyenek  $A$  és  $B$  tetszőleges halmazok, ekkor az

$$A \times B ::= \{(a, b) \mid a \in A \text{ és } b \in B\}$$

definícióval adott halmaz az  $A$  és  $B$  halmazok *direktszorzata*.

Egy  $R \subseteq A \times B$  halmazt *bináris relációnak* nevezünk. A továbbiakban, ha nem okoz félreértést, a bináris jelzőt elhagyjuk.

Az  $R$  reláció *értelmezési tartománya*:

$$\mathcal{D}_R ::= \{a \in A \mid \exists b \in B : (a, b) \in R\},$$

a reláció *értékkészlete*:

$$\mathcal{R}_R ::= \{b \in B \mid \exists a \in A : (a, b) \in R\},$$

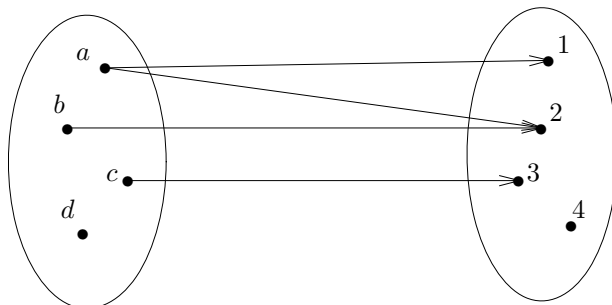
a reláció *értéke* egy  $a \in A$  helyen, vagy másképpen  $a$   $R$  szerinti *képe*:

$$R(a) ::= \{b \in B \mid (a, b) \in R\},$$

egy  $H \subseteq A$  halmaz  $R$  szerinti *képe* a halmazt alkotó elemek képeinek uniója, azaz

$$R(H) ::= \bigcup_{h \in H} R(h).$$

Az  $R \subseteq A \times B$  relációt felfoghatjuk egy nemdeterminisztikus leképezésnek, megfeleltetésnek is az  $A$  és a  $B$  halmaz elemei között. Az értelmezési tartomány jelenti azokat az  $A$ -beli elemeket, amelyekhez hozzárendelünk legalább egy  $B$ -beli elemet. Hasonlóan, az értékkészlet a legalább egy elemhez hozzárendelt elemek halmaza. Az 1.1. ábrával egy relációt szemléltetünk.  $A = \{a, b, c, d\}$ ,  $B = \{1, 2, 3, 4\}$ ,  $R = \{(a, 1), (a, 2), (b, 2), (c, 3)\}$ . Az  $a$  ponthoz hozzá van rendelve az 1 és a 2 is,  $b$ -hez



1.1. ábra. Reláció szemléltetése

csak a 2,  $c$ -hez a 3,  $d$ -hez nincs hozzárendelve semmi, és 4 nincs hozzárendelve semmihez.

Egy  $a \in A$  elemnek a képe azoknak a  $B$ -beli elemeknek a halmaza, amelyeket a reláció hozzárendel az  $a$ -hoz. Egy  $H$  halmaz képét az elemek képeinek uniójával definiáltuk. Ez másképpen úgy fogalmazható meg, hogy a  $H$  képe azokból az elemekből áll, amelyek legalább egy  $H$ -beli elemhez hozzá vannak rendelve, azaz

$$R(H) = \{b \in B \mid \exists a \in H : (a, b) \in R\}.$$

Azt mondjuk, hogy egy reláció *determinisztikus*, ha

$$\forall a \in A : |R(a)| \leq 1.$$

A determinisztikus relációkat másképpen *parciális függvényeknek* hívjuk. *Függvénynek* nevezünk egy  $R$  relációt akkor, ha

$$\forall a \in A : |R(a)| = 1.$$

Egy  $A$ -ból  $B$ -be képező  $f$  függvényt általában  $f : A \rightarrow B$ -vel, míg egy parciális függvényt  $f \in A \rightarrow B$ -vel jelölünk. E jelölés használata esetén  $f(a)$  nem az egyelemű halmazt, hanem annak elemét jelenti.

Legyen  $R \subseteq A \times B$ . Ekkor az  $R^{(-1)}$  reláció az  $R$  inverze, ha

$$R^{(-1)} ::= \{(b, a) \in B \times A \mid (a, b) \in R\}.$$

Legyen  $H \subseteq B$  tetszőleges halmaz.  $H$ -nak az inverz reláció szerinti képét,  $R^{(-1)}(H)$ -t a  $H$  halmaz  $R$  reláció szerinti *inverz képének* nevezzük. A definíciókból látszik, hogy

$$R^{(-1)}(H) = \{a \in A \mid R(a) \cap H \neq \emptyset\}.$$

Fontos megkülönböztetni az inverz képet a  $H$  halmaz  $R$  reláció szerinti *ősképletől*, amelynek a definíciója a következő:

$$R^{-1}(H) ::= \{a \in \mathcal{D}_R \mid R(a) \subseteq H\}.$$

Az őskép általában nem egyezik meg az inverz képpel, de mindig része annak. A két kép kapcsolatát mutatja az 1.2. ábra. Megjegyezzük azonban, hogy függvény, illetve parciális függvény esetén a két kép megegyezik.

Legyen  $P \subseteq A \times B$  és  $Q \subseteq A \times B$ . Ha  $P \subseteq Q$ ,  $P$ -t  $Q$  *leszűkítésének* nevezzük. Ha  $R \subseteq A \times B$  és  $H \subseteq A$ , akkor

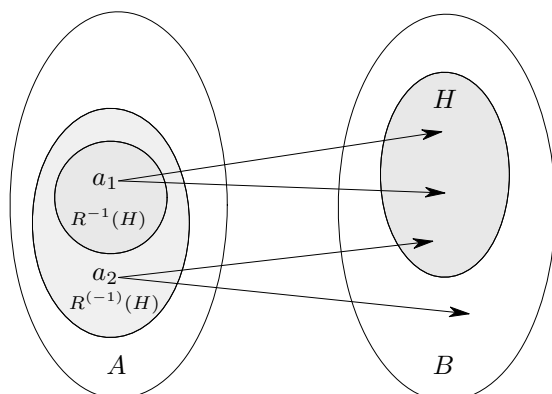
$$R|_H ::= R \cap (H \times B)$$

$R$  egy leszűkítése, amit úgy is nevezünk, hogy  $R$  *megszorítása*  $H$ -ra.

### 1.3.1. Műveletek

A relációk között értelmezünk műveleteket is. Legyen  $P \subseteq A \times B$  és  $Q \subseteq B \times C$ . Az  $R \subseteq A \times C$  relációt a  $P$  és  $Q$  relációk *kompozíciójának* nevezzük, ha

$$R = \{(a, c) \in A \times C \mid \exists b \in B : (a, b) \in P \text{ és } (b, c) \in Q\},$$



1.2. ábra. Inverz kép és ősképek

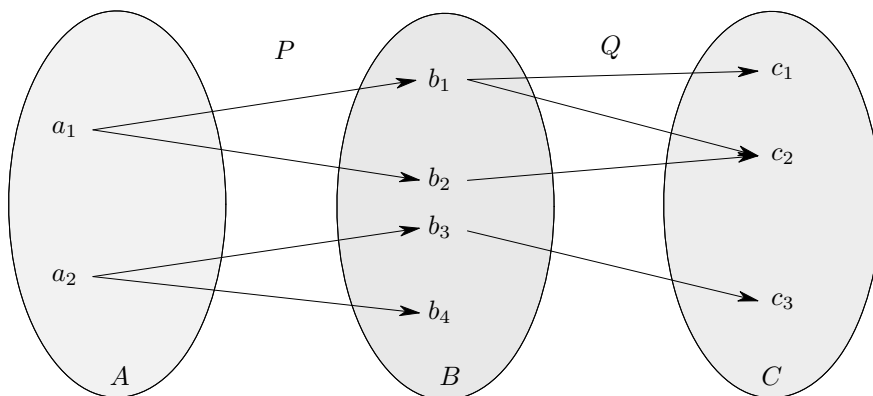
és  $Q \circ P$ -vel jelöljük.

Az  $S \subseteq A \times C$  relációt a  $P$  és  $Q$  relációk *szigorú kompozíciójának* nevezzük, ha

$$S = \{(a, c) \in A \times C \mid \exists b \in B : (a, b) \in P \text{ és } (b, c) \in Q \text{ és } P(a) \subseteq \mathcal{D}_Q\},$$

és  $Q \odot P$ -vel jelöljük.

Mint az az 1.3. ábrán is látható, a kompozíció és a szigorú kompozíció általában nem egyezik meg,  $(a_2, c_3) \in Q \circ P$ , de nem eleme  $Q \odot P$ -nek. Könnyű belátni, a szigorú kompozíció mindig része a kompozíciónak. Azt sem nehéz belátni, hogy ha a két reláció közül legalább az egyik függvény, vagy ha az első parciális függvény, a kétféle kompozíció megegyezik.



1.3. ábra. Kompozíció és szigorú kompozíció

Ha  $R \subseteq A \times A$ , akkor azt mondjuk, hogy  $R$  *homogén* reláció. Egy homogén reláció önmagával komponálható. Ennek alapján definiáljuk  $R$  *hatványait*:

$$R^0 ::= \{(a, a) \mid a \in A\},$$

és ha  $n \in \mathbb{N}$ ,

$$R^n ::= R \circ R^{n-1}.$$

Az  $\{(a, a) \mid a \in A\}$  relációt *identitás* relációnak is nevezzük, és  $id_A$ -val jelöljük.

Az  $R \subseteq A \times A$  reláció *lezártja* az az  $\bar{R} \subseteq A \times A$  reláció, amelyre:

$$(1) \mathcal{D}_{\bar{R}} ::= \{a \in A \mid \nexists \alpha \in A^\infty : \alpha_1 = a \text{ és } \forall i \in \mathbb{N} : \alpha_{i+1} \in R(\alpha_i)\} \text{ és}$$

$$(2) \forall a \in \mathcal{D}_{\bar{R}} : \bar{R}(a) ::= \{b \in A \mid \exists k \in \mathbb{N}_0 : b \in R^k(a) \text{ és } b \notin \mathcal{D}_R\}.$$

A lezárt tehát olyan pontokban van értelmezve, amelyekből kiindulva a relációt nem lehet végtelen sokszor egymás után alkalmazni, és ezekhez a pontokhoz olyan pontokat rendel, amelyeket úgy kapunk, hogy a reláció véges sokszori alkalmazásával kikerülünk az eredeti reláció értelmezési tartományából. Tehát  $\mathcal{D}_R \cap \mathcal{R}_{\bar{R}} = \emptyset$  mindig teljesül, és  $\forall a \notin \mathcal{D}_R$ -ra  $a \in \mathcal{D}_{\bar{R}}$  és  $\bar{R}(a) = \{a\}$ . Felhívjuk a figyelmet arra, hogy ez a definíció nem egyezik meg azzal, amit *tranzitív lezárt*nak szoktak nevezni.

Az  $R \subseteq A \times A$  reláció *korlátos lezártja* az az  $\overline{\bar{R}} \subseteq A \times A$  reláció, amelyre:

$$(1) \mathcal{D}_{\overline{\bar{R}}} ::= \{a \in A \mid \exists k_a \in \mathbb{N}_0 : R^{k_a}(a) = \emptyset\} \text{ és}$$

$$(2) \forall a \in \mathcal{D}_{\overline{\bar{R}}} : \overline{\bar{R}}(a) ::= \bar{R}(a).$$

Vegyük észre, hogy egy reláció lezártja és korlátos lezártja különbözhet. A definíciókból látható, hogy ez a különbözőség csak az értelmezési tartományok között jelentkezhet. Ennek azonban szükséges feltétele, hogy egy alkalmas pontból kiindulva a relációt ne lehessen végtelen sokszor alkalmazni, de a véges sokszori alkalmazások hosszára ne tudjunk korlátot mondani. Természetesen ez csak akkor lehetséges, ha végtelen sok véges alkalmazás-sorozatot tudunk a pontból indítani. Nézzünk erre egy egyszerű példát: legyen  $R \subseteq \mathbb{Z} \times \mathbb{Z}$  és

$$R(a) = \begin{cases} \{a-1\}, & \text{ha } a > 0; \\ \mathbb{N}, & \text{ha } a < 0. \end{cases}$$

Ekkor  $\mathbb{Z} = \mathcal{D}_{\bar{R}} \neq \mathcal{D}_{\overline{\bar{R}}} = \mathbb{N}_0$ .

### 1.3.2. Logikai relációk

Az  $R \subseteq A \times \mathbb{L}$  típusú relációkat – ahol  $A$  tetszőleges halmaz – *logikai relációknak* nevezzük. A logikai relációkra bevezetünk néhány jelölést:

Legyen  $R \subseteq A \times \mathbb{L}$ . Ekkor az  $R$  *igazsághalmaza*:

$$[R] ::= R^{-1}(\{\text{igaz}\}),$$



gyenge igazsághalmaza:

$$[R] ::= R^{(-1)}(\{igaz\}).$$

Ha  $R$  függvény, akkor az igazsághalmaz és a gyenge igazsághalmaz megegyezik.

Legyen  $H \subseteq A$ , azt az  $A \rightarrow \mathbb{L}$  függvényt, amelynek az igazsághalmaza  $H$ , a  $H$  halmaz *karakterisztikus függvényének* nevezzük, és  $\mathcal{P}(H)$ -val jelöljük. A fenti definíciókból következik, hogy tulajdonképpen mindegy, egy halmaz részhalmazairól vagy a halmazon értelmezett logikai függvényekről (állításokról) beszélünk-e, hiszen ezek a fogalmak kölcsönösen egyértelműen megfelelnek egymásnak.

Gyakran fogjuk használni az azonosan igaz és az azonosan hamis függvényeket:  $Igaz_A : A \rightarrow \mathbb{L}$  és  $[Igaz_A] = A$ ,  $Hamis_A : A \rightarrow \mathbb{L}$  és  $[Hamis_A] = \emptyset$ . Ha nem okoz félreértést, az  $A$  indexet nem írjuk ki.

Legyen  $R \subseteq A \times A$  és  $\pi : A \rightarrow \mathbb{L}$ . Az

$$R|\pi = R|_{[\pi]} \cup \{(a, a) \in A \times A \mid a \in [\pi] \setminus \mathcal{D}_R\}$$

relációt *feltételes relációnak* nevezzük, ami a reláció leszűkítése és kiterjesztése a feltétel igazsághalmazára, azaz  $\mathcal{D}_{R|\pi} = [\pi]$ .

A továbbiakban egy feltételes reláció lezártját, illetve korlátos lezártját a reláció *feltételre vonatkozó lezártjának*, illetve *feltételre vonatkozó korlátos lezártjának* fogjuk nevezni.

## 1.4. Direktszorzat

Az előzőekben definiáltuk két halmaz direktszorzatát, most ezt a fogalmat általánosítjuk.

Legyenek  $A_i, i \in I$  tetszőleges halmazok, és  $X = \{x \mid x : I \rightarrow \cup_{i \in I} A_i\}$ , azaz  $X$  az  $I$ -t az  $A_i$ -k uniójába képező függvények halmaza. Ekkor az

$$A = \prod_{i \in I} A_i ::= \{x \in X \mid \forall i \in I : x(i) \in A_i\}$$

halmazt az  $A_i, i \in I$  halmazok *direktszorzatának* nevezzük.

Az  $I$  halmaz gyakran az  $[1..n]$  intervallum, ekkor az

$$A = \prod_{i=1}^n A_i$$

jelölést használjuk. Ebben az esetben azt mondhatjuk, hogy a direktszorzat elemei rendezett  $n$ -esek. Az általános esetben is a direktszorzat elemeit gyakran mint rendezett  $n$ -eseket adjuk meg, feltételezve, hogy egyértelműen el tudjuk dönteni, hányadik komponens melyik  $i \in I$ -hez tartozik.

Megjegyezzük, hogy  $I$  üres is lehet, ebben az esetben a direktszorzat csak az üres halmazt tartalmazza.

Ha  $J \subseteq I$  és  $K = I \setminus J$ , akkor a

$$B = \prod_{j \in J} A_j$$

direktszorzat *altere* és a

$$B' = \prod_{k \in K} A_k$$

direktszorzat *kiegészítő altere*  $A$ -nak.

A  $pr_B : A \rightarrow B$  függvényt *projekciónak* nevezzük, ha

$$\forall a \in A : pr_B(a) = a|_J.$$

Ha  $J = \{j\}$ , a  $pr_B$  függvényt  $j$ -edik projekciónak vagy  $j$  változónak is nevezzük.

Értelmezzük a projekciót párokra, sorozatokra és halmazokra is:

$$pr_B((a_1, a_2)) ::= (pr_B(a_1), pr_B(a_2))$$

$$pr_B(\langle a_1, a_2, \dots \rangle) ::= \langle pr_B(a_1), pr_B(a_2), \dots \rangle$$

$$pr_B(\{a_1, a_2, \dots\}) ::= \{pr_B(a_1)\} \cup \{pr_B(a_2)\} \cup \dots$$

A párok vetülete tehát a komponensek vetületéből álló pár, a sorozat vetülete egy, az eredetivel azonos hosszúságú sorozat, amelynek elemei rendre az eredeti sorozat elemeinek vetületei. A halmaz vetülete halmaz, de lehet, hogy a vetülethalmaz elemeinek száma kisebb, mint az eredeti volt, például egy végtelen halmaz vetülete lehet véges is.

Azt mondjuk, hogy az  $A = \prod_{i \in I} A_i$  direktszorzat *ekvivalens* a  $B = \prod_{j \in J} B_j$  direktszorzattal, ha létezik olyan  $f : I \rightarrow J$  kölcsönösen egyértelmű megfeleltetés (bijekció), hogy  $\forall i \in I : A_i = B_{f(i)}$ . Úgy is fogalmazhatunk, hogy az értelmezési tartomány elemeinek átnevezésével kapjuk meg  $A$ -ból  $B$ -t. Könnyű belátni, hogy valóban ekvivalencia relációt definiáltunk a direktszorzatok között.

## 1.5. Függvényterek

Ha  $f, g : A \rightarrow B$  függvények és  $\oplus$  egy művelet  $B$ -n, azaz  $\oplus : B \times B \rightarrow B$ , akkor definiálhatjuk az  $f \oplus g : A \rightarrow B$  függvényt is, úgy, hogy

$$\forall a \in A : f \oplus g(a) = f(a) \oplus g(a).$$

Parciális függvények esetén

$$\mathcal{D}_{f \oplus g} = \{a \in A \mid a \in \mathcal{D}_f \text{ és } a \in \mathcal{D}_g \text{ és } (f(a), g(a)) \in \mathcal{D}_\oplus\}.$$

Az  $f, g$  függvények és egy  $\sim \subseteq B \times B$  reláció logikai függvényt definiálnak:  
 $f \sim g : A \rightarrow \mathbb{L}$  és

$$\forall a \in A : f \sim g(a) = \begin{cases} igaz, & \text{ha } a \in \mathcal{D}_f \text{ és } a \in \mathcal{D}_g \text{ és } (f(a), g(a)) \in \sim; \\ hamis & \text{egyébként.} \end{cases}$$

Az így kapott logikai függvényekre a fentebb definiált módon alkalmazhatjuk a szokásos logikai műveleteket:  $\wedge, \vee, \rightarrow, \equiv, \neg$ . Megjegyezzük, hogy a definícióból rögtön adódnak a következő összefüggések. Legyen  $p, q : A \rightarrow \mathbb{L}$ , ekkor:

$$\begin{aligned} [p \wedge q] &= [p] \cap [q], \\ [p \vee q] &= [p] \cup [q], \\ [\neg p] &= A \setminus [p], \\ [p \rightarrow q] &= (A \setminus [p]) \cup [q]. \end{aligned}$$

A  $p = q$  jelölést, ha nem okoz félreértést az  $p \equiv q$  értelmében is használjuk.

$$[p \equiv q] = [p] \cap [q] \cup [\neg p] \cap [\neg q].$$

Az implikáció jelölésére gyakran használják a  $\Rightarrow$  jelet a  $\rightarrow$  helyett, mi az előbbit a „következik” reláció jelölésére használjuk, azaz

$$p \Rightarrow q \Leftrightarrow [p] \subseteq [q].$$

Megjegyezzük, hogy  $p \Rightarrow q$  és  $p \rightarrow q = \text{Igaz}$  ugyanazt jelenti.

A  $\forall$  és a  $\exists$  jeleket eddig is használatuk mint a „minden” és „létezik” szavak rövidítéseit. A fenti logikai kifejezések esetén  $\forall i \in I : \dots$  jelentése  $\bigwedge_{i \in I} \dots$ , és  $\exists i \in I : \dots$  jelentése  $\bigvee_{i \in I} \dots$ . Ha  $I = \emptyset$ ,  $\bigwedge_{i \in I} \dots$  azonosan igaz,  $\bigvee_{i \in I} \dots$  pedig azonosan hamis.

## 1.6. Példák

**1.1. példa:** Írjuk fel az  $A \times B$ ,  $A \times C$ ,  $(A \times B) \times C$ , és  $A \times B \times C$  halmazok elemeit, ha  $A = \{0, 1\}$ ,  $B = \{1, 2, 3\}$ ,  $C = \{p, q\}$ !

**Megoldás:**

$$\begin{aligned} A \times B &= \{(0, 1), (0, 2), (0, 3), (1, 1), (1, 2), (1, 3)\}, \\ A \times C &= \{(0, p), (0, q), (1, p), (1, q)\}, \\ (A \times B) \times C &= \{((0, 1), p), ((0, 2), p), ((0, 3), p), ((1, 1), p), ((1, 2), p), ((1, 3), p), \\ &\quad ((0, 1), q), ((0, 2), q), ((0, 3), q), ((1, 1), q), ((1, 2), q), ((1, 3), q)\}, \\ A \times B \times C &= \{(0, 1, p), (0, 2, p), (0, 3, p), (1, 1, p), (1, 2, p), (1, 3, p), \\ &\quad (0, 1, q), (0, 2, q), (0, 3, q), (1, 1, q), (1, 2, q), (1, 3, q)\}. \end{aligned}$$

**1.2. példa:** Legyen  $R \subseteq \{1, 2, 3, 4, 5\} \times \{1, 2, 3, 4, 5\}$ .

$$R = \{(1, 2), (1, 4), (2, 1), (3, 4), (3, 3), (3, 5), (4, 5)\}.$$

a) Mi a reláció értelmezési tartománya és értékkészlete?

- b) Determinisztikus-e, illetve függvény-e a reláció?
- c) Mi  $R$  0., 2. hatványa, mi  $R$  inverze?
- d) Mi a  $\{4, 5\}$  halmaz inverz képe, illetve ősképe?
- e) Hány eleme van  $R$  értékészlete hatványhalmazának?

**Megoldás:**

a)  $\mathcal{D}_R = \{1, 2, 3, 4\}$ ,  
 $\mathcal{R}_R = \{1, 2, 3, 4, 5\}$ .

- b) A reláció nem determinisztikus, ugyanis pl.  $|R(1)| = 2!$  Mivel a reláció nem determinisztikus, függvény sem lehet.
- c) A reláció 0. hatványa az identikus leképezés, azaz:

$$R^0 = \{(1, 1), (2, 2), (3, 3), (4, 4), (5, 5)\}.$$

Mivel  $R^2 = R \circ R$ , azt kell megvizsgálnunk, hogy mely pontokból hogyan lehet a relációt egymás után kétszer alkalmazni:

$$\begin{aligned} (1, 2) &\longrightarrow (2, 1) \\ (1, 4) &\longrightarrow (4, 5) \\ (2, 1) &\longrightarrow (1, 2) \\ (2, 1) &\longrightarrow (1, 4) \\ (3, 4) &\longrightarrow (4, 5) \\ (3, 3) &\longrightarrow (3, 4) \\ (3, 3) &\longrightarrow (3, 3) \\ (3, 3) &\longrightarrow (3, 5) \end{aligned}$$

A fenti táblázat alapján:

$$R^2 = \{(1, 1), (1, 5), (2, 2), (2, 4), (3, 5), (3, 4), (3, 3)\}.$$

$R^{(-1)}$  a reláció inverzének definíciója alapján:

$$R = \{(2, 1), (4, 1), (1, 2), (4, 3), (3, 3), (5, 3), (5, 4)\}.$$

- d) Írjuk fel, hogy mit rendel a reláció az értelmezési tartomány egyes pontjaihoz:

$$\begin{aligned} R(1) &= \{2, 4\} \\ R(2) &= \{1\} \\ R(3) &= \{3, 4, 5\} \\ R(4) &= \{5\} \end{aligned}$$

Az inverz kép definíciója alapján:

$$R^{(-1)}(\{4, 5\}) = \{1, 3, 4\}.$$

Az ősképp definíciója alapján:

$$R^{-1}(\{4, 5\}) = \{4\}.$$

e)  $|\wp(\mathcal{R}_R)| = 2^{|\mathcal{R}_R|} = 2^5 = 32.$

**1.3. példa:** Megadható-e valamilyen összefüggés egy  $H$  halmaz inverz képének képe és a  $H$  halmaz között?

**Megoldás:**

Legyen  $R \subseteq A \times B$ ,  $H \subseteq B$ . Ekkor

$$\begin{aligned} R(R^{(-1)}(H)) &= R(\{a \in A \mid R(a) \cap H \neq \emptyset\}) = \\ &= \bigcup_{R(a) \cap H \neq \emptyset} R(a). \end{aligned}$$

Vegyük észre, hogy általános esetben nem tudunk mondani semmit a két halmaz viszonyáról, ugyanis

1. ha  $H \not\subseteq \mathcal{R}_R$ , akkor  $H \not\subseteq R(R^{(-1)}(H))$ , és
2. ha  $\exists a \in R^{(-1)}(H) : R(a) \not\subseteq H$ , akkor  $R(R^{(-1)}(H)) \not\subseteq H$ .

Tekintsük e fenti esetet egy egyszerű példán: Legyen  $A = B = \{1, 2, 3\}$ ,  $R = \{(1, 1), (1, 2)\}$ . Ekkor  $H = \{2, 3\}$  esetén  $R(R^{(-1)}(H)) = \{1, 2\}$ , azaz egyik irányú tartalmazás sem áll fenn.

**1.4. példa:** Legyen  $R \subseteq A \times B$ ,  $P, Q \subseteq B$ . Hogyan lehetne jellemezni az  $R^{-1}(P \cup Q)$  és az  $R^{-1}(P \cap Q)$  halmazt az  $R^{-1}(P)$  és  $R^{-1}(Q)$  halmaz segítségével?

**Megoldás:**

$$\begin{aligned} R^{-1}(P \cup Q) &= \{a \in \mathcal{D}_R \mid R(a) \subseteq (P \cup Q)\} \\ &\supseteq \{a \in \mathcal{D}_R \mid R(a) \subseteq P\} \cup \{a \in \mathcal{D}_R \mid R(a) \subseteq Q\}. \end{aligned}$$

A másik irányú tartalmazás azonban nem áll fenn, ugyanis lehet olyan  $a \in \mathcal{D}_R$ , amelyre

$$R(a) \not\subseteq P, \text{ és } R(a) \not\subseteq Q, \text{ de } R(a) \subseteq P \cup Q.$$

Nézzük ezt egy példán: Legyen  $A = B = \{1, 2\}$ ,  $R = \{(1, 1), (1, 2)\}$ ,  $P = \{1\}$ ,  $Q = \{2\}$ . Ekkor  $R^{-1}(P)$  és  $R^{-1}(Q)$  üres, de  $R^{-1}(P \cup Q) = \{1\}$ .

Vizsgáljuk most meg a metszetet!

$$\begin{aligned} R^{-1}(P \cap Q) &= \{a \in \mathcal{D}_R \mid R(a) \subseteq (P \cap Q)\} \\ &= \{a \in \mathcal{D}_R \mid R(a) \subseteq P\} \cap \{a \in \mathcal{D}_R \mid R(a) \subseteq Q\} \\ &= R^{-1}(P) \cap R^{-1}(Q). \end{aligned}$$

Tehát bebizonyítottuk, hogy két tetszőleges halmaz metszetének ősképe egyenlő a két halmaz ősképeinek metszetével.

**1.5. példa:** Legyenek  $F \subseteq A \times B, G \subseteq B \times C$ . Igaz-e, hogy

$$(G \circ F)^{(-1)} = F^{(-1)} \circ G^{(-1)}?$$

**Megoldás:**

$$\begin{aligned} (G \circ F)^{(-1)} &= \{(c, a) \in C \times A \mid \exists b \in B : (a, b) \in F \text{ és } (b, c) \in G\} \\ &= \{(c, a) \in C \times A \mid \exists b \in B : (b, a) \in F^{(-1)} \text{ és } (c, b) \in G^{(-1)}\} \\ &= F^{(-1)} \circ G^{(-1)}. \end{aligned}$$

**1.6. példa:** Legyenek  $F \subseteq A \times B, G \subseteq B \times C$ . Igaz-e, hogy

$$\forall Y \subseteq C : (G \circ F)^{-1}(Y) = F^{-1}(G^{-1}(Y))?$$

**Megoldás:** A szigorú kompozíció definíciójából azonnal adódik, hogy ha  $a \in A$  és  $a \in \mathcal{D}_{G \circ F}$ , akkor  $G \circ F(a) = G(F(a))$ , ezt felhasználva:

$$\begin{aligned} (G \circ F)^{-1}(Y) &= \{a \in A \mid a \in \mathcal{D}_{F \circ G} \text{ és } (G \circ F)(a) \subseteq Y\} \\ &= \{a \in A \mid a \in \mathcal{D}_F \text{ és } F(a) \subseteq \mathcal{D}_G \text{ és } G(F(a)) \subseteq Y\} \\ &= \{a \in A \mid a \in \mathcal{D}_F \text{ és } F(a) \subseteq \{b \in B \mid b \in \mathcal{D}_G \text{ és } G(b) \subseteq Y\}\} \\ &= F^{-1}(G^{-1}(Y)). \end{aligned}$$

**1.7. példa:**  $W = N_1 \times N_2 \times N_3$ .  $\alpha \in W^{**}$ , ahol  $N_i = \mathbb{N}$  ( $i = 1, 2, 3$ ).  $\alpha_1 = (1, 1, 1)$ . Az  $\alpha$  sorozat további elemeit úgy kapjuk meg, hogy a pontok koordinátáit az első koordinátával kezdve ciklikusan 1-gyel növeljük.  $red(pr_{N_1 \times N_3}(\alpha)) = ?$

**Megoldás:**

Írjuk fel először a sorozat első néhány tagját:

$$\alpha = \langle (1, 1, 1), (2, 1, 1), (2, 2, 1), (2, 2, 2), (3, 2, 2), (3, 3, 2), \dots \rangle$$

Az  $\alpha$  sorozat projekciója  $N_1 \times N_3$ -ra:

$$pr_{N_1 \times N_3}(\alpha) = \langle (1, 1), (2, 1), (2, 1), (2, 2), (3, 2), (3, 2), \dots \rangle$$

A fenti sorozat redukáltja:

$$\text{red}(pr_{N_1 \times N_3}(\alpha)) = \langle (1, 1), (2, 1), (2, 2), (3, 2) \dots \rangle$$

A fentiekből jól látható, hogy a redukció pontosan azokat az elemeket hagyja ki a sorozatból, amelyekben a növelés a második komponensben történt, így az eredményssorozat elemeit is a koordináták ciklikus eggyel növelésével kapjuk meg, az  $(1, 1)$  pontból kiindulva.

**1.8. példa:** Legyen  $A = \{1, 2, 3, 4, 5\}$  és  $R \subseteq A \times A$ .

$R = \{(1, 2), (1, 4), (2, 1), (3, 4), (3, 3), (3, 5), (4, 5)\}$ . Mi lesz  $R$  lezártja és korlátos lezártja?

**Megoldás:**

Mivel  $\mathcal{D}_R = \{1, 2, 3, 4\}$ , azt kell csak megvizsgálni, hogy honnan jutunk el biztosan a reláció ismételt alkalmazásával 5-be.  $R(1) = \{2, 4\}$  és  $R(2) = \{1\}$ , ezért  $1, 2 \notin \mathcal{D}_{\overline{R}}$ , a 3 sem, mert a 3-ból akárhányszor eljuthatunk 3-ba, 4-ből egy lépésben, 5-ből nulla lépésben jutunk 5-be. Tehát  $\overline{R} = \{(4, 5), (5, 5)\}$  és  $\overline{\overline{R}} = \overline{R}$ .

**1.9. példa:**  $A = \{1, 2, 3, 4, 5\}$ ,  $R \subseteq A \times A$ .  $R = \{(1, 2), (1, 5), (2, 5), (3, 2), (3, 4), (5, 2)\}$ .

$[\pi] = \{1, 2, 3, 4\}$ . Írjuk fel a reláció feltételre vonatkozó lezártját! **Megoldás:**

$R|\pi = \{(1, 2), (1, 5), (2, 5), (3, 2), (3, 4), (4, 4)\}$ . Az  $(5, 2)$  kimaradt a szűkítés miatt, a  $(4, 4)$  pedig bekerült a bővítés miatt.  $\overline{R|\pi} = \{(1, 5), (2, 5), (5, 5)\}$ .

**1.10. példa:** Van-e olyan nem üres reláció és  $\pi$  feltétel, hogy a reláció lezártja üres halmaz, és a  $\pi$  feltételre vonatkozó lezártja azonos a relációval?

**Megoldás:**

Legyen  $A$  tetszőleges halmaz. Nyilván  $\overline{id_A} = \emptyset$ . Ha  $\pi = \text{Hamis}$ ,  $id_A|\pi = \emptyset$ , aminek a lezártja  $id_A$ .

**1.11. példa:**  $R \subseteq \mathbb{N}_0 \times \mathbb{N}_0$ .

$$R(a) = \begin{cases} \{a - 2\}, & \text{ha } a > 1; \\ \{2^k \mid k \in \mathbb{N}\}, & \text{ha } a = 1. \end{cases}$$

Mi az  $R$  reláció lezártja és korlátos lezártja?

**Megoldás:**

$\mathcal{D}_R = \mathbb{N}$ . Minden  $a$ -hoz, ha  $a$  páros, a reláció  $a/2$  lépésben hozzárendeli a 0-t és csak azt, ha pedig páratlan, az 1-et, aminek a képe az összes páros kettőshatvány. A kettőshatványokból, mivel mind páros, az  $R$  ismételt alkalmazása 0-ba vezet. Tehát  $\mathcal{D}_{\overline{R}} = \mathbb{N}_0$ , és természetesen  $\forall a \in \mathbb{N}_0 : \overline{R}(a) = \{0\}$ .

Mivel nincs felső korlátja a kettőshatványoknak, ezért  $\mathcal{D}_{\overline{\overline{R}}}$  nem tartalmazza a páratlan számokat.

Megjegyezzük, hogy ha a feladatban  $\{2^k \mid k \in \mathbb{N}\}$  helyett  $\{2^k \mid k \in \mathbb{N}_0\}$  szerepelne,  $\mathcal{D}_{\overline{R}}$  sem tartalmazná a páratlan számokat.

## 1.7. Feladatok

1.1. Legalább, illetve legfeljebb hány eleme van egy  $m$  elemű és egy  $n$  elemű halmaz

- metszetének;
- egyesítésének;
- Descartes-szorzatának;
- különbségének?

1.2. Bizonyítsa be, hogy  $H \subseteq A \times B$  esetén

- $(\forall (a, b), (c, d) \in H : (a, d) \in H) \Leftrightarrow (\exists K \subseteq A : \exists L \subseteq B : H = K \times L)$ ;
- ha  $H$  nem üres, akkor  $K$  és  $L$  egyértelmű.

1.3.  $R \subseteq A \times B$ . Mivel egyenlő  $R^{-1}(B)$ ?

1.4.  $R = \{((x, y), (x+y, y)) \mid x, y \in \mathbb{N}\}$ . Mi a  $H = \{(a, b) \mid a, b \in \mathbb{N} \text{ és } a+b < 5\}$  halmaz inverz képe, illetve ősképe?

1.5.  $R = \{((x, y), (x+y, y)) \mid x, y \in \mathbb{N}\} \cup \{((x, y), (x-y, y)) \mid x, y \in \mathbb{N}\}$ . Mi a  $H = \{(a, b) \mid a, b \in \mathbb{N} \text{ és } a+b < 5\}$  halmaz inverz képe, illetve ősképe?

1.6.  $R = \{((x, y), (f(x, y), y)) \mid x, y \in \mathbb{N}\}$ , ahol  $f : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ . Mi a  $H = \{(a, b) \mid a, b \in \mathbb{N} \text{ és } a+b < 5\}$  halmaz ősképe, illetve inverz képe?

1.7.  $R \subseteq A \times B, Q \subseteq B$ . Van-e valamilyen összefüggés az  $R^{-1}(B \setminus Q)$  halmaz és az  $A \setminus R^{-1}(Q)$  halmaz között?

1.8. Készítsen olyan nem üres relációt, amelyre igaz, hogy értékkészlete minden valódi részhalmazának ősképe üres halmaz!

1.9. Legyen  $F, G \subseteq \mathbb{N} \times \mathbb{N}, Y = \{1, 2\}$ .  $F = \{(a, b) \mid b|a \text{ és } b \neq 1 \text{ és } b \neq a\}$ .  
 $G = \{(a, b) \mid 2|a \text{ és } a = 2b\}$ .

$$\begin{aligned} G \circ F &=? & G \odot F &=? \\ F^{(-1)} \circ G^{(-1)} &=? & F^{-1}(G^{-1}(Y)) &=? \\ (G \circ F)^{-1}(Y) &=? & (G \circ F)^{(-1)} &=? \end{aligned}$$

1.10. Legyen  $A = \{1, 2, 3, 4, 5\}, R \subseteq A \times A, R = \{(1, 2), (1, 4), (2, 1), (3, 4), (3, 3), (3, 5), (4, 5)\}, f \subseteq A \times \mathbb{L}$  és  $f = \{(1, i), (2, i), (3, i), (4, h), (5, i)\}$ . Mi  $f$ , illetve  $(f \circ R)$  igazsághalmaza és gyenge igazsághalmaza?

1.11.  $R, Q \subseteq A \times A$ . Igaz-e, hogy  $(R \odot Q)^{(-1)} = Q^{(-1)} \circ R^{(-1)}$ ?

1.12.  $F \subseteq A \times B, G \subseteq B \times C$ . Igaz-e, hogy  $\forall Y \subseteq C : (G \circ F)^{-1}(Y) = F^{-1}(G^{-1}(Y))$ . Igaz-e az állítás, ha  $G$  vagy  $F$  függvény?



**1.13.**  $F \subseteq A \times B, G \subseteq B \times C$ . Igaz-e, hogy  $(G \odot F)^{(-1)} = F^{(-1)} \odot G^{(-1)}$ . Igaz-e az állítás, ha  $G$  vagy  $F$  függvény?

**1.14.** Mi az összefüggés két reláció kompozíciójának értelmezési tartománya és ugyanezen két reláció szigorú értelemben vett kompozíciójának értelmezési tartománya között?

**1.15.** Készítsen olyan nem üres  $R$  relációt és  $f$  logikai függvényt, hogy  $f \circ R$  igazhalmaza üres legyen!

**1.16.**  $R \subseteq A \times A$ . Igaz-e, hogy  $(R^{(-1)})^2 = (R^2)^{(-1)}$ ?

**1.17.**  $R \subseteq A \times A$ . Igaz-e, hogy  $\forall H \subseteq A : R^{-1}(R^{-1}(H)) = (R^2)^{-1}(H)$ ?

**1.18.**  $P, Q \subseteq \mathbb{N} \times \mathbb{N}$ .  $Q = \{(a, b) \mid 2 \mid a \text{ és } b \mid a \text{ és } b \text{ prím}\}$ .

a)  $P = \{(a, b) \mid b \mid a \text{ és } b \neq 1 \text{ és } b \neq a\}$

b)  $P = \{(a, b) \mid b \mid a\}$

Adja meg a  $Q^{(-1)}, Q \circ P$  és  $Q \odot P$ -t relációt!

**1.19.**  $H \subseteq A \times B, G \subseteq B \times C, F \subseteq C \times D$ . Igazak-e az alábbi állítások?

a) Ha  $a \in \mathcal{D}_{G \circ H} \cap \mathcal{D}_{G \odot H}$ , akkor  $G \circ H(a) = G \odot H(a)$ .

b)  $\mathcal{D}_{G \odot H} \subseteq \mathcal{D}_{G \circ H}$ .

c)  $(\forall a \in \mathcal{D}_H : |H(a)| = 1) \Rightarrow G \circ H = G \odot H$ .

d)  $\mathcal{D}_G = B \Rightarrow G \circ H = G \odot H$ .

**1.20.** Asszociativitás:  $H \subseteq A \times B, G \subseteq B \times C, F \subseteq C \times D$ . Igazak-e:

$$(F \circ G) \circ H = F \circ (G \circ H),$$

$$(F \odot G) \odot H = F \odot (G \odot H)?$$

**1.21.** Legyen  $Q, R, S \subseteq A \times A$ , és vezessük be az alábbi jelölést: ha  $X \subseteq A \times A$  tetszőleges reláció, akkor  $X$  komplementere:

$$\widehat{X} = \{(a, b) \in A \times A \mid (a, b) \notin X\}.$$

Igaz-e, hogy

$$Q \odot R \subseteq S \iff Q^{(-1)} \odot \widehat{S} \subseteq \widehat{R}?$$

Igaz-e a fenti állítás nemszigorú kompozíció esetén?

**1.22.** Legyen  $Q, R, S \subseteq A \times A$ . Igaz-e, hogy

$$R \subseteq S \Rightarrow R \odot Q \subseteq S \odot Q,$$

$$R \subseteq S \Rightarrow Q \odot R \subseteq Q \odot S?$$

**1.23.** Legyen  $R$  és  $Q$  két reláció a természetes számok halmazán.  $R$  egy természetes számhoz rendeli önmagát és a kétszeresét,  $Q$  egy páros természetes számhoz a felét.

- Írja fel a két relációt, és adja meg az értelmezési tartományukat!
- Írja fel az  $R$  reláció  $k$ . hatványát ( $k \geq 1$ ) és ennek az értelmezési tartományát!
- Írja fel a  $Q \circ R$  relációt és annak értelmezési tartományát!
- $F = Q \circ R$ . Írja fel az  $F$  relációt és az értelmezési tartományát!

**1.24.**  $F \subseteq A \times B$ ,  $G \subseteq B \times C$ . Igaz-e, hogy:

- $\mathcal{D}_{G \circ F} = F^{-1}(\mathcal{D}_G)$ ,
- $\mathcal{D}_{G \circ F} = F^{-1}(\mathcal{D}_G)$ ?

**1.25.**  $P \subseteq \mathbb{N}_0 \times \mathbb{N}_0$ .  $P = \{(a, b) \mid b|a \text{ és } b \neq 1 \text{ és } b \neq a\}$ . Mi lesz  $P$  lezártja és korlátos lezártja?

**1.26.**  $R \subseteq \mathbb{N} \times \mathbb{N}$ .  $R = \{(a, b) \mid b|a \text{ és } b \neq 1 \text{ és } b \neq a\}$ .  $[\pi] = \{x \mid \exists k \in \mathbb{N}_0 : x = 2^k\}$ . Írjuk fel az  $R|_{\pi}$  relációt, lezártját és korlátos lezártját!

**1.27.** Adjunk példát olyan nem üres relációra, amelynek lezártja üres halmaz, és van olyan  $\pi$  feltétel, hogy a reláció feltételre vonatkozó lezártjának értelmezési tartománya megegyezik az eredeti reláció értelmezési tartományával!

**1.28.**  $R \subseteq A \times A$ . Tegyük fel, hogy az  $R$  értelmezési tartománya egyenlő az  $R$  értelmezési tartományának  $R$ -re vonatkozó ősképeivel. Mit mondhatunk  $R$  lezártjáról?

**1.29.**  $R \subseteq \mathbb{N}_0 \times \mathbb{N}_0$ .

$$R(a) = \begin{cases} \{a - 3\}, & \text{ha } a > 2; \\ \{3k \mid k \in \mathbb{N}\}, & \text{ha } a = 1. \end{cases}$$

Mi az  $R$  reláció lezártja és korlátos lezártja?

**1.30.**  $R \subseteq \mathbb{N} \times \mathbb{N}$ . Az  $R$  reláció minden összetett számhoz a legnagyobb valódi osztóját rendeli. Legyen  $q$

- egy rögzített összetett természetes szám,
- egy rögzített prímszám.

Legyen  $[P_q] = \{a \in \mathbb{N} \mid \exists k \in \mathbb{N} : a = q^k\}$ . Mi lesz az  $R$  reláció  $P_q$  feltételre vonatkozó lezártjának értelmezési tartománya?

**1.31.**  $R \subseteq \mathbb{N}_0 \times \mathbb{N}_0$ .

$$R(x) = \begin{cases} \{b \mid \exists k \in \mathbb{N}_0 : b = 2k + 1\}, & \text{ha } x \neq 0 \text{ és } x \text{ páros;} \\ \{x - 7\}, & \text{ha } x \geq 7 \text{ és } x \text{ páratlan;} \\ \{0\}, & \text{ha } x = 1; \\ \{7\}, & \text{ha } x = 0. \end{cases}$$

Mi lesz  $R$  lezártja és korlátos lezártja?

**1.32.**  $R$  legyen az 1.29. feladatban adott reláció.  $\lceil \pi \rceil = \{k \in \mathbb{N} \mid 2 \nmid k\}$ . Adja meg az  $R \lceil \pi \rceil$  relációt, lezártját és korlátos lezártját!

**1.33.** Igazak-e az alábbi állítások?

- Ha  $a \in \mathcal{D}_{\overline{R}} \cap \mathcal{D}_{\overline{\overline{R}}}$ , akkor  $\overline{R}(a) = \overline{\overline{R}}(a)$ .
- $\mathcal{D}_{\overline{R}} \subseteq \mathcal{D}_{\overline{\overline{R}}}$ .
- Ha az  $A$  halmaz véges és  $R \subseteq A \times A$ , akkor  $\overline{R} = \overline{\overline{R}}$ .
- Ha  $A$  megszámlálhatóan végtelen,  $R \subseteq A \times A$ , és  $\forall a \in A : (\exists n(a) \in \mathbb{N}_0 : |R(a)| \leq n(a)) \Rightarrow \overline{R} = \overline{\overline{R}}$ .

**1.34.** Legyen  $R \subseteq \mathbb{N}_0 \times \mathbb{N}_0$ .

$$R(x) = \begin{cases} \{b \mid b > 0 \text{ és } b < x \text{ és } 2 \mid b\}, & \text{ha } x \text{ páratlan;} \\ \{x - 1\}, & \text{ha } x \text{ páros;} \end{cases}$$

$\lceil \pi \rceil = \{x \in \mathbb{N} \mid 2 \mid x\}$ . Mi az  $R$  reláció  $\pi$  feltételre vonatkozó lezártja és korlátos lezártja?

**1.35.** Legfeljebb, illetve legalább milyen hosszú egy  $m$  és egy  $n$  hosszúságú sorozat redukáltjának konkatenációja, illetve konkatenációjának redukáltja?

**1.36.** Igaz-e, hogy egy  $\alpha$  sorozat redukáltjának projekciója ugyanolyan hosszú, mint az  $\alpha$  sorozat redukáltja?

**1.37.** Igaz-e, hogy egy  $\alpha$  sorozat projekciójának redukáltja ugyanolyan hosszú, mint az  $\alpha$  sorozat redukáltja?

**1.38.** Legyen  $A = N_1 \times N_2 \times N_3 \times N_4$ ,  $B = N_4 \times N_1$ , ahol  $N_i = \mathbb{N}$  ( $i \in [1..4]$ ).

$$\alpha = \langle (1, 1, 1, 1), (1, 2, 1, 1), (1, 2, 3, 1), (1, 2, 3, 4), (5, 2, 3, 4), (5, 7, 3, 4), (5, 7, 10, 4), (5, 7, 10, 14), \dots \rangle$$

- $pr_B(\alpha) = ?$
- $red(pr_B(\alpha)) = ?$



## 2. fejezet

# A programozás alapfogalmai

Ebben a fejezetben a programozás legfontosabb alapfogalmait vezetjük be. Nagyon fontos szempont, hogy figyelmünket nem a programok tulajdonságainak vizsgálatára, hanem a programok előállítására fordítjuk. Ezért feltesszük a kérdést, miért is írunk programot? Az erre a kérdésre adott válasz meghatározza gondolkodásunk irányát. A válaszuk az, hogy azért, mert van valami megoldandó feladatunk, problémánk. Tehát a gondolkodásunk kiinduló pontja az, hogy kell lennie valaminek, amit feladatnak hívunk, s ez a feladat határozza meg az elérendő célt.

A gyakorlatban nagyon sokféle feladat van. Mi bennük a közös? Ez a kérdés is többféleképpen közelíthető meg. A mi megközelítésünk szerint a feladat lényege az, hogy meghatározzuk, milyen állapotban vagyunk és milyen állapotba akarunk jutni. Az, hogy mik az állapotok, a konkrét problémától függ. Például, ha egy autót akarunk egy hosszabb útra felkészíteni, az állapotát jellemezheti az, hogy mennyi a tankban a benzin, mennyi az ablakmosó folyadék, mekkora a nyomás a kerekekben, működik-e az irányjelző és így tovább. A lényeg az, hogy van a rendszernek valahány jellemzője, ezen jellemzők lehetséges értékei egy-egy halmazt alkotnak. Egy ilyen halmaz állhat a mennyiséget kifejező számokból, lehet akár a  $\{működik, nem működik\}$  halmaz is.

Ha mindegyik jellemző lehetséges értékeinek halmazából választunk egy-egy értéket megkapjuk az autó egy lehetséges állapotát. Már csak az hiányzik, hogy észrevegyük, a lehetséges állapotok halmaza matematikailag egy direktszorzat.

### 2.1. Az állapottér fogalma

Az elsőként bevezetendő absztrakt fogalom a fent említett lehetséges állapotok halmaza.

#### 2.1. DEFINÍCIÓ: ÁLLAPOTTÉR

Legyen  $I$  egy véges halmaz és legyenek  $A_i, i \in I$  tetszőleges véges vagy meg-

számlálható, nem üres halmazok. Ekkor az  $A = \prod_{i \in I} A_i$  halmazt *állapotter-  
nek*, az  $A_i$  halmazokat pedig *típusérték-halmazoknak* nevezzük.

Amikor egy modellt készítünk, el kell döntenünk, hogy a valóság mely részét kívánjuk modellezni, és melyek azok a jellemzők – és milyen értékeket vehetnek fel –, amelyeket a modellünkben figyelembe akarunk venni.

Az állapotter fenti definíciójában az egyes komponenseket tekintjük úgy, mint egyes jellemzők lehetséges értékeinek halmazát. A típusérték-halmaz elnevezés arra utal, hogy ezek a halmazok bizonyos közös tulajdonsággal rendelkező elemekből állnak. A későbbiekben majd kitérünk arra is, hogy ez a közös tulajdonság mit is jelent. Mivel a jellemzők érték-halmaza lehet azonos, az állapotter komponensei között egy halmaz többször is szerepelhet.

Kikötöttük, hogy az állapotternek csak véges sok komponense legyen. Lehetne általánosabb definíciót is adni úgy, hogy nem kötjük ki az  $I$  halmaz végességét, ekkor a fent definiált állapotteret az általánosított állapotter egy (véges) nézetének nevezzük.

Az, hogy a komponensek legfeljebb megszámlálhatók, azt is jelenti, hogy a komponensek között nem szerepelhet pl. a valós számok halmaza. Természetesen ettől még egy típusérték-halmaz tartalmazhatja akár  $\sqrt{2}$ -t is. Az  $\{x \mid \exists n \in \mathbb{N} : x = \sqrt{n}\}$  lehet állapotter-komponens.

## 2.2. A feladat

Az állapotter fogalmának segítségével könnyen megfogalmazhatjuk, hogy mit értünk feladaton. Azt kell megfogalmaznunk, hogy egy adott állapotból (azaz az állapotter egy eleméből, pontjából) milyen állapotba (azaz az állapotter mely pontjába) akarunk eljutni.

### 2.2. DEFINÍCIÓ: FELADAT

*Feladatnak* nevezzük egy  $F \subseteq A \times A$  relációt.

A feladat fenti definíciója természetes módon adódik abból, hogy a feladatot egy leképezésnek tekintjük az állapotterén, és az állapotter minden pontjára megmondjuk, hova kell belőle eljutni, ha egyáltalán el kell jutni belőle valahova.

Az, hogy egy feladatnak mi lesz az állapottere, természetesen magától a feladattól függ, ám még a feladat ismeretében sem egyértelmű. Például egy pont síkbeli koordinátáit megadhatjuk derékszögű koordináta-rendszerben, de megadhatjuk polárkoordinátákkal is.

Mégis, az, hogy mit választunk állapotternek, nagyon fontos, hiszen meghatározza, hogy a továbbiakban mit és hogyan tudunk leírni. Ha túl kevés jellemzőt vizsgálunk – azaz az állapotter túl kevés komponensből áll –, akkor lehetnek olyan fogalmak, amelyeket nem tudunk benne leírni, ha túl sok a komponens, akkor fölöslegesen túl bonyolult lesz a modell.

Tekintsük azt az egyszerű feladatot, hogy össze kell adni két természetes számot. Az állapotteret elég kézenfekvő módon három komponensűnek választhatjuk. A három komponens a két összeadandó és az összeg. Tehát  $A = \mathbb{N} \times \mathbb{N} \times \mathbb{N}$ , s a feladat

$$F = \{((a, b, c), (x, y, z)) \in A \times A \mid a + b = z\},$$

vagy

$$G = \{((a, b, c), (x, y, z)) \in A \times A \mid b = x \text{ és } c = y \text{ és } a + b = z\}.$$

A két feladat nem azonos, bár mindkettő két természetes szám összegéről szól. A különbség köztük az, hogy az  $F$  feladat nem mond semmit arról, mi legyen az összeadandókkal, a  $G$  pedig kiköti, hogy maradjanak változatlanok.

Felvetődik, hogy nem lenne elég a két komponensű állapotter is? Legyen  $A = \mathbb{N} \times \mathbb{N}$  és

$$H = \{((a, b), (x, y)) \in A \times A \mid a + b = x\}.$$

Ezt a feladatot azonban nem úgy interpretálnánk, hogy „adjunk össze két természetes számot”, hanem úgy, hogy „növeljük meg egy természetes számot egy természetes számmal”.

Megjegyezzük, gyakran fogalmazzuk meg feladatot úgynevezett „input-output” modellben is, azaz milyen bemenő adatokhoz milyen kimenő adatokat rendelünk. Ez a szétválasztás az  $F$  és  $G$  feladat esetében nem okozna gondot, de a  $H$ -hoz hasonló feladatok esetében már problémás lehetne, nem is beszélve a bevezetőben említett autós feladatról. Az állapotter modell igazi előnyeit a későbbiekben még tapasztalni fogjuk.

Felhívjuk a figyelmet arra, hogy a definíció szerint a feladat reláció, azaz általában nem determinisztikus, például  $G$  és  $H$ . A nondeterminisztikusság azonban még „érdemibb” is lehet. Legyen a feladat a következő: határozzuk meg egy természetes szám egy valódi osztóját (a szám megváltoztatása nélkül)! Ebben az esetben  $A = \mathbb{N} \times \mathbb{N}$  és

$$F = \{((a, b), (x, y)) \in A \times A \mid a = x \text{ és } y \mid x \text{ és } x \neq y \text{ és } y \neq 1\}.$$

Például  $(6, 5)$  pont  $F$  szerinti képe  $\{(6, 2), (6, 3)\}$ . A 6-nak 2 is, meg 3 is valódi osztója, azaz  $|F(6, 5)| = 2$ .

Nagyon fontos, hogy pontosan lássuk a különbséget az előző feladat és a következő között: határozzuk meg egy természetes szám összes valódi osztóját! Ebben az esetben az állapotter is más lesz, hiszen egy természetes szám összes valódi osztója nem egy szám, hanem egy halmaz. Tehát  $A' = \mathbb{N} \times \mathfrak{F}$ , ahol  $\mathfrak{F}$  az  $\mathbb{N}$  véges részhalmazainak halmaza.

$$G = \{((a, b), (x, y)) \in A' \times A' \mid a = x \text{ és } y = \{n \in \mathbb{N} \mid n \mid x \text{ és } x \neq n \text{ és } n \neq 1\}\}.$$

Most  $|G(6, \{5\})| = 1$  és  $G(6, \{5\}) = \{(6, \{2, 3\})\}$ .

Megjegyezzük még, hogy  $\mathcal{D}_F \neq A$ , például  $(5, 5) \notin \mathcal{D}_F$ , de  $\mathcal{D}_G = A'$ , például  $(5, \{5\}) \in \mathcal{D}_G$  és  $G(5, \{5\}) = \{(5, \emptyset)\}$ .

### 2.3. A program

Amikor a program fogalmát igyekszünk tisztázni, a számítógépen futó programokra és az általuk megvalósított algoritmusokra figyelünk. Ha egy számítógépen egy program „fut”, az abban jelentkezik, hogy a számítógép memóriájának tartalma folyamatosan változik. Itt most a „memóriát” általánosan értelmezzük, beleértünk a szűken vett memóriától a regisztereken keresztül a képernyőig mindent, ami információt hordoz.

Egy program jellemző tulajdonsága, hogy „működik”, azaz időben dinamikus folyamat. A dinamikus rendszerek általában nehezebben kezelhetők, vizsgálhatók, mint a statikusak. Ezért arra gondolunk, lehet-e helyettesíteni egy dinamikus folyamatot egy statikus modellel?

Tekintsük például az alábbi – a programozástól igazán messze eső – problémát. Adott egy kémiai kísérlet, amely túl gyorsan játszódik le ahhoz, hogy az ember pontosan regisztrálni tudja az egymás utáni eseményeket. Ez a programfutáshoz hasonlóan egy időben dinamikus lejátszódó folyamat. Hogyan követhető nyomon mégis a kísérlet? Például úgy, hogy a kísérletet filmre vesszük, és a továbbiakban a képkockák által rögzített statikus állapotokat vizsgáljuk. Így az időben változó folyamatot egy statikus állapotsorozattal írjuk le.

A fenti példa szemléletesen mutatja, hogyan adhatunk statikus modellt egy dinamikus folyamat leírására.

A program definíciójában a program időbeni futásának jellemzésére az előbbi példával analóg módon vezetünk be egy statikus modellt: a futást állapotértékes sorozatokkal írjuk le. Ahogy a program futása során a memóriatartalom változik, úgy jutunk az állapottér újabb és újabb pontjaiba, így ezeket a pontokat egy sorozatba fűzve valójában „filmre vesszük” a programfutást.

#### 2.3. DEFINÍCIÓ: PROGRAM

Programnak nevezzük az  $S \subseteq A \times A^{**}$  relációt, ha

1.  $\mathcal{D}_S = A$ ,
2.  $\forall \alpha \in \mathcal{R}_S : \alpha = red(\alpha)$ ,
3.  $\forall a \in A : \forall \alpha \in S(a) : |\alpha| \neq 0$  és  $\alpha_1 = a$ .

A fenti definícióval a „működés” fogalmát akarjuk absztrakt módon megfogalmazni, ez magyarázza a sorozatokra vonatkozó kikötéseket. Az első tulajdonság azt jelenti, hogy a program az állapottér minden pontjához hozzárendel legalább egy sorozatot, azaz a program minden pontban „csinál” valamit. A „rosszul működést” is a működéssel, azaz valamilyen tulajdonságú sorozattal, sorozatokkal írjuk le.

A második tulajdonság azt fejezi ki, hogy a program értékkészlete olyan sorozatokból áll, amelyekben nem szerepel egymás után ugyanaz az elem. Egész pontosan, ha ez mégis előfordul, akkor ez az elem ismétlődik végtelen sokszor. A működés abban nyilvánul meg, hogy megváltozik az állapot, vagy ha mégsem, az az abnormalis működés jele. Emlékeztetünk arra, hogy az állapottér komponensei között minden előfordul, tehát ha bármi is történik, az más állapotértékes pontot jelent. Az, hogy az



állapottér egy pontjához a program végtelen sorozatot rendel, azt jelenti, hogy a program futása nem fejeződik be.

A harmadik tulajdonság csak annyit jelent, hogy a sorozat a működés teljes történetét leírja, beleértve a kiinduló állapotot is, ezért azt, hogy egy program egy pontból elindítva nem csinál semmit, egy ebből az egy pontból álló, egy hosszúságú sorozat jellemzi.

A programot is relációként definiáltuk, vagyis egy-egy állapottérbeli ponthoz több sorozat is hozzá lehet rendelve, azaz a működés nemdeterminisztikus. Ez első pillanatra talán meglepő, valójában természetes. Természetes akkor is, ha számítógépen, számítógéprendszeren futó programra gondolunk, hiszen egy program sok proceszszorból, sok, akár egymástól nagy távolságban levő komponensből álló rendszeren fut. De természetes akkor is, ha figyelembe vesszük, hogy a program-fogalom nemcsak a gépen futó program, hanem az algoritmus absztrakciója is, amelyeknek lehetnek „nyitva” hagyott részei is.

## 2.4. A programfüggvény

Most visszatérünk a fejezet elején szereplő autós példához. A feladat az volt, hogy készítsük fel az autót egy hosszabb útra. Attól függően, hogy az autó milyen állapotban van, azaz a jellemzői milyen értékekkel rendelkeznek: mennyi benne a benzin, mekkora a nyomás a kerekekben, működik-e az irányjelző, tevékenységek sorozatát hajtjuk végre, felpumpáljuk a kerekeket, kicserélünk egy izzót és így tovább, lépésről lépésre változik az állapot, működik a program. Ha végül olyan állapotba jut az autó, hogy most már nyugodtan el lehet vele indulni egy hosszabb útra, akkor a program megoldotta a feladatot.

Ahhoz, hogy egy program és egy feladat viszonyát megvizsgáljuk, elegendő, ha a programról tudjuk, hogy az állapottér egy adott pontjából kiindulva az állapottér mely pontjába jut, mert a megoldás szempontjából a közbülső állapotok lényegtelenek.

Az előbbieket miatt bevezetjük a programfüggvény fogalmát, amely a program futásának eredményét jellemzi.

### 2.4. DEFINÍCIÓ: PROGRAMFÜGGVÉNY

A  $p(S) \subseteq A \times A$  reláció az  $S \subseteq A \times A^{**}$  program *programfüggvénye*, ha

1.  $\mathcal{D}_{p(S)} = \{a \in A \mid S(a) \subseteq A^*\}$ ,
2.  $p(S)(a) = \{b \in A \mid \exists \alpha \in S(a) : \tau(\alpha) = b\}$ .

Az első követelmény azt fogalmazza meg, hogy csak azokban a pontokban van értelme azt vizsgálni, hogy hova jut egy program, ahonnan kiindulva a program nem „száll el”. A második pont értelemszerűen azt írja le, hogy ahova a program eljut, az a sorozat utolsó eleme.

Ha két program programfüggvénye megegyezik, az azt jelenti, hogy a két program működésének eredménye ugyanaz. Ezért mondjuk ebben az esetben azt, hogy a két program ekvivalens.

A programfüggvény elnevezés megtévesztő lehet, hiszen egy program programfüggvénye nem feltétlenül függvény, sőt az sem biztos, hogy determinisztikus reláció (parciális függvény). Jobban kifejezi a fogalom tartalmát a *hatásreláció* elnevezés. Mindkettőt használni fogjuk.

Megjegyezzük, hogy a programfüggvényen túl természetesen vannak a programnak olyan jellemzői, melyek a program minőségére vonatkoznak, és ezek szempontjából egyáltalán nem mindegy, hogy a program hogyan oldja meg a feladatot (ilyen lehet például a hatékonyság, a program idő- és tárigénye), de a továbbiakban ezekkel egyelőre nem foglalkozunk.

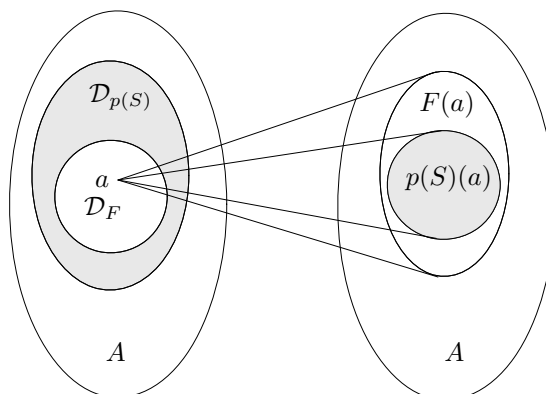
## 2.5. Megoldás

Fontos, hogy a programfüggvény ugyanolyan típusú reláció, mint a feladat volt. Így tehát a programfüggvény fogalmának bevezetésével lehetőségünk nyílik arra, hogy kapcsolatot teremtsünk egy adott feladat és egy adott program között. Természetesen ennek a kapcsolatnak azt kell leírnia, hogy mikor mondjuk egy programról azt, hogy megold egy adott feladatot.

### 2.5. DEFINÍCIÓ: MEGOLDÁS

Azt mondjuk, hogy az  $S$  program *megoldja* az  $F$  feladatot, ha

1.  $\mathcal{D}_F \subseteq \mathcal{D}_{p(S)}$ ,
2.  $\forall a \in \mathcal{D}_F : p(S)(a) \subseteq F(a)$ .



2.1. ábra. Megoldás

Ezzel a definícióval végül is azt kívánjuk meg, hogy az állapotér olyan pontjaihoz, ahol a feladat értelmezve van, a program csak véges sorozatokat rendeljen (termináljon), és a sorozatok végpontjait a feladat hozzárendelje a kezdőponthoz.

Néha gondot okoz ennek a definíciónak a megértése. Miért a programfüggvény szerinti kép része a feladat szerinti képnek? „Így nem kapunk meg minden megoldást!” Pedig elég csak az autós példára gondolni. Például a fékfolyadék szintjének a minimum és a maximum szint jelzése között kell lennie. Ezt a karbantartás eredményeképpen teljesítjük, de egyáltalán nem egyéltelmű, hogy mi lesz a beállított szint. Annak meg nincs is értelme, hogy „minden lehetséges szintet” beállítsunk.

Sokszor felmerül a kérdés, hogy van-e összefüggés két feladat között a megoldás szempontjából. Ezzel kapcsolatos a következő definíció.

### 2.6. DEFINÍCIÓ: SZIGORÍTÁS

Azt mondjuk, hogy az  $F_1 \subseteq A \times A$  feladat *szigorúbb*, mint az  $F_2 \subseteq A \times A$  feladat, ha

1.  $\mathcal{D}_{F_2} \subseteq \mathcal{D}_{F_1}$ ,
2.  $\forall a \in \mathcal{D}_{F_2} : F_1(a) \subseteq F_2(a)$ .

A szigorítás definícióját összevetve a megoldás definíciójával könnyen adódik a következő egyszerű, de fontos állítás:

**2.1. állítás:** Ha  $F_1$  szigorúbb, mint  $F_2$ , és  $S$  megoldása  $F_1$ -nek, akkor  $S$  megoldása  $F_2$ -nek is.

Felhívjuk a figyelmet arra is, hogy a megoldás definíciójában nem kötöttük ki a program és a feladat állapotterének azonosságát. Ezzel kapcsolatos a következő állítás:

**2.2. állítás:** Ha  $S$  program megoldása  $F_1 \subseteq A \times A$  feladatnak, az  $F_2 \subseteq B \times B$  szintén feladat és  $F_1 = F_2$ , akkor  $S$  megoldása  $F_2$ -nek is,

## 2.6. Programozási feladat

Létezik-e tetszőleges feladathoz megoldóprogram? Legyen  $F = A \times A$ . Definiáljuk  $S$ -et a következőképpen:  $\forall a \in \mathcal{D}_F : S(a) = \{red(\langle a, b \rangle) \mid b \in F(a)\}$  és  $\forall a \notin \mathcal{D}_F : S(a) = \{\langle a, \dots \rangle\}$ . Nyilvánvaló, hogy  $p(S) = F$ , tehát  $S$  megoldása  $F$ -nek. Vagyis tetszőleges feladathoz könnyen tudunk megoldóprogramot csinálni. Ebből kiindulva azt gondolhatnánk, hogy a programozás nagyon egyszerű feladat. Ez persze nem igaz. A programozás feladata valójában sokkal összetettebb, mert egy programot adott programokból, rögzített szabályok szerint kell összeraknunk, például egy programnyelv eszközeit kell használnunk.

**2.7. DEFINÍCIÓ: PROGRAMOZÁSI FELADAT**

Legyen  $A = \prod_{i \in I} A_i$ . *Programozási feladatnak* nevezzük az  $(F, \mathbb{P}, \mathbb{K})$  hármast, ahol  $F \subseteq A \times A$  egy feladat;  $\mathbb{P}$  a *primitív* (megengedett) programok véges halmaza,  $\forall S \in \mathbb{P} : S \subseteq A \times A^{**}$ ;  $\mathbb{K}$  a megengedett programkonstrukciók véges halmaza,  $\forall K \in \mathbb{K}$  egy az  $A$ -n értelmezett programok halmazán értelmezett művelet.

**2.8. DEFINÍCIÓ: PROGRAMOZÁSI FELADAT MEGOLDÁSA**

Az  $(F, \mathbb{P}, \mathbb{K})$  programozási feladatnak az  $S$  program megoldása, ha  $S$  a primitív programokból a megengedett konstrukciókkal előállítható, és megoldása  $F$ -nek.

A programozási feladat két értelemben is általánosítható: egyrészt kibővíthető a program működésére vonatkozó feltételekkel [Hor 98], ezzel ebben a könyvben nem foglalkozunk. Másrészt nem követeljük meg az azonos állapotteret, ehhez általánosítjuk a megoldás fogalmát, illetve bevezetjük a típus-specifikáció, típus, megfelelés és a típuskonstrukciók fogalmát.

Az, hogy milyen programkonstrukciókat engedünk meg, sok mindentől függ, mi a következőkben csak a legegyszerűbbekkel fogunk foglalkozni, mivel ezek is elégségesek egy programozási feladat megoldásához.

Már most felhívjuk a figyelmet arra a fontos szempontra, hogy valójában nagyon gyakran nem azt az utat követjük, hogy meglévő programokból rakjuk össze a megoldóprogramot, hanem a feladatot bontjuk fel részfeladatokra, úgy, hogy az ezeket megoldó programokból „automatikusan” megkapjuk az eredeti feladatot megoldó programot.

**2.7. Példák**

**2.1. példa:** Legyen  $A_1 = \{1, 2\}$ ,  $A_2 = \{1, 2\}$ ,  $A_3 = \{1, 2, 3, 4, 5\}$ ,  $A = A_1 \times A_2 \times A_3$ .  $F = \{(a, b, c), (d, e, f) \mid f = a + b\}$ .  $F(1, 1, 1) = ?$  Hány olyan pontja van az állapottérnek, amelyekhez a feladat ugyanazt rendeli, mint az  $(1, 1, 1)$ -hez?

**Megoldás:**

$$F(1, 1, 1) = \{(1, 1, 2), (1, 2, 2), (2, 1, 2), (2, 2, 2)\}.$$

Mivel a feladat hozzárendelése nem függ az állapottér harmadik komponensétől, a feladat ugyanezeket a pontokat rendeli az összes  $(1, 1, *)$  alakú ponthoz. Más pontokhoz viszont nem rendelheti ugyanezeket a pontokat, mert akkor az összeg nem lehetne 2! Tehát öt olyan pontja van az állapottérnek, amelyhez a feladat ugyanazt rendeli, mint az  $(1, 1, 1)$ -hez.

**2.2. példa:** Legyen  $A = \{1, 2, 3, 4, 5\}$ ,  $S \subseteq A \times A^{**}$ .

$$S = \left\{ \begin{array}{llll} (1, \langle 1251 \rangle), & (1, \langle 14352 \rangle), & (1, \langle 132 \dots \rangle), & (2, \langle 21 \rangle), \\ (2, \langle 24 \rangle), & (3, \langle 333333 \dots \rangle), & (4, \langle 41514 \rangle), & (4, \langle 431251 \rangle), \\ (4, \langle 41542 \rangle), & (5, \langle 524 \rangle), & (5, \langle 534 \rangle), & (5, \langle 5234 \rangle) \end{array} \right\}$$
$$F = \{(2, 1) (2, 4) (4, 1) (4, 2) (4, 5)\}.$$

a) Adjuk meg  $p(S)$ -t!

b) Megoldja-e  $S$  a feladatot?

**Megoldás:**

- a) Mivel a program az 1-hez és a 3-hoz végtelen sorozatot is rendel, a programfüggvény értelmezési tartománya:

$$\mathcal{D}_{p(S)} = \{2, 4, 5\}.$$

Ekkor a programfüggvény:

$$p(S) = \{(2, 1), (2, 4), (4, 4), (4, 1), (4, 2), (5, 4)\}.$$

- b) A megoldás definíciója két pontjának teljesülését kell belátnunk.

$$i. \mathcal{D}_F = \{2, 4\} \subseteq \{2, 4, 5\} = \mathcal{D}_{p(S)}.$$

$$ii. p(S)(2) = \{1, 4\} \subseteq \{1, 4\} = F(2),$$

$$p(S)(4) = \{4, 1, 2\} \not\subseteq \{1, 2, 5\} = F(4),$$

tehát az  $S$  program nem megoldása az  $F$  feladatnak.

**2.3. példa:** Fejezzük ki a programok uniójának programfüggvényét a programok programfüggvényeivel!

**Megoldás:** Legyenek  $S_1, S_2 \subseteq A \times A^{**}$  programok. Ekkor a programfüggvény értelmezési tartományának definíciójából kiindulva:

$$\begin{aligned} \mathcal{D}_{p(S_1 \cup S_2)} &= \{a \in A \mid (S_1 \cup S_2)(a) \subseteq A^*\} = \\ &= \{a \in A \mid S_1(a) \subseteq A^* \text{ és } S_2(a) \subseteq A^*\} = \\ &= \mathcal{D}_{p(S_1)} \cap \mathcal{D}_{p(S_2)}. \end{aligned}$$

Legyen  $a \in \mathcal{D}_{p(S_1)} \cap \mathcal{D}_{p(S_2)}$ . Ekkor

$$\begin{aligned} p(S_1 \cup S_2)(a) &= \{\tau(\alpha) \mid \alpha \in (S_1 \cup S_2)(a)\} = \\ &= \{\tau(\alpha) \mid \alpha \in S_1(a) \text{ vagy } \alpha \in S_2(a)\} = \\ &= p(S_1)(a) \cup p(S_2)(a). \end{aligned}$$

**2.4. példa:** Legyen  $F_1$  és  $F_2$  egy-egy feladat ugyanazon az állapottéren. Igaz-e, hogy ha minden program, amelyik megoldása  $F_1$ -nek, az megoldása  $F_2$ -nek is, és minden program, amelyik megoldása  $F_2$ -nek, az megoldása  $F_1$ -nek is, akkor  $F_1$  és  $F_2$  megegyeznek?

**Megoldás:** A leggyakoribb hiba, amit ennek a feladatnak a megoldásakor el szoktak követni, az az, hogy összekeverik az állítás feltételrendszerét magával a bizonyítandó állítással, és azt próbálják bebizonyítani, hogy valamelyik feladatnak minden program megoldása. Természetesen általában ez nem igaz, de nem is ez a feladat! Abból kell tehát kiindulnunk, hogy pontosan ugyanazok a programok oldják meg mindkét

feladatot, és meg kell vizsgálnunk, hogy következik-e ebből az, hogy a két feladat megegyezik.

Induljunk ki abból, hogy minden program, amely megoldása  $F_1$ -nek, az megoldása  $F_2$ -nek, és válasszunk egy olyan programot, amelynek programfüggvénye megegyezik az  $F_1$  relációval. Ekkor a választott program triviálisan megoldja az  $F_1$  feladatot, tehát meg kell oldania  $F_2$ -t is, azaz:

$$\begin{aligned} i. & \quad \mathcal{D}_{F_2} \subseteq \mathcal{D}_{F_1}, \\ ii. & \quad \forall a \in \mathcal{D}_{F_2} : F_1(a) \subseteq F_2(a). \end{aligned}$$

Most felhasználva, hogy minden program, amely megoldása  $F_2$ -nek, az megoldása  $F_1$ -nek is, és egy olyan program választásával, amelynek programfüggvénye megegyezik  $F_2$ -vel, az előzőekkel analóg módon adódnak a fordított irányú állítások:

$$\begin{aligned} iii. & \quad \mathcal{D}_{F_1} \subseteq \mathcal{D}_{F_2}, \\ iv. & \quad \forall a \in \mathcal{D}_{F_1} : F_2(a) \subseteq F_1(a). \end{aligned}$$

Az *i.* és *iii.* állításokból következik, hogy a két feladat értelmezési tartománya megegyezik, míg az *ii.* és *iv.* állítások garantálják, hogy e közös értelmezési tartomány egyes pontjaihoz mindkét feladat ugyanazokat a pontokat rendeli, azaz  $F_1 = F_2$ .

**2.5. példa:**  $F_1 \subseteq F_2$ . Az  $S$  program megoldja  $F_2$ -t. Igaz-e, hogy  $S$  megoldja  $F_1$ -et is?

**Megoldás:** Próbáljuk meg bebizonyítani az állítást. Ehhez a megoldás definíciójának két pontját kell belátnunk.

$$\begin{aligned} i. & \quad \mathcal{D}_{F_1} \subseteq \mathcal{D}_{p(S)}, \\ ii. & \quad \forall a \in \mathcal{D}_{F_1} : p(S)(a) \subseteq F_1(a). \end{aligned}$$

Az *i.* pont teljesülése könnyen látható, ugyanis  $S$  megoldása  $F_2$ -nek, tehát

$$\mathcal{D}_{F_1} \subseteq \mathcal{D}_{F_2} \subseteq \mathcal{D}_{p(S)}.$$

Az *ii.* pont bizonyításánál azonban gond van, hiszen az alábbi két állítás áll rendelkezésünkre:

$$\begin{aligned} \forall a \in \mathcal{D}_{F_1} : & \quad p(S)(a) \subseteq F_2(a), \\ \forall a \in \mathcal{D}_{F_1} : & \quad F_1(a) \subseteq F_2(a), \end{aligned}$$

és ezekből a kívánt állítás nem bizonyítható. Elakadtunk a bizonyításban, lehet, hogy nem igaz az állítás? Készítsünk ellenpéldát felhasználva azt, hogy hol akadtunk el a bizonyításban!

Legyen  $A = \{1, 2\}$ ,  $F_1 = \{(1, 1)\}$ ,  $F_2 = \{(1, 1), (1, 2)\}$ , és  $p(S)$  egyezzen meg az  $F_2$  feladattal. Ekkor  $S$  triviálisan megoldja  $F_2$ -t, de nem megoldása  $F_1$ -nek, ugyanis

$$1 \in \mathcal{D}_{F_1} \text{ és } p(S)(1) = F_2(1) = \{1, 2\} \not\subseteq \{1\} = F_1(1).$$

Tehát az állítás nem igaz.

**2.6. példa:** Legyenek  $S_1$  és  $S_2 \subseteq A \times A^{**}$  programok,  $F \subseteq A \times A$  pedig feladat. Tegyük fel továbbá, hogy  $S_1 \subseteq S_2$  és  $S_2$  megoldja az  $F$  feladatot. Igaz-e, hogy  $S_1$  megoldja  $F$ -et?

**Megoldás:** Ha  $S_1 \subseteq S_2$ , akkor mit tudunk a programfüggvényükről? Nézzük először az értelmezési tartományokat! A definíció szerint minden állapottérbeli ponthoz minden program hozzárendel legalább egy sorozatot, így  $S_1$  és  $S_2$  is. Mivel  $S_1 \subseteq S_2$  ezért csak az fordulhat elő, hogy egy adott ponthoz  $S_2$  olyan sorozatokat is rendel, amit  $S_1$  nem. Ha ezek a sorozatok mind végesek, akkor az adott pont vagy benne van mindkét program programfüggvényének az értelmezési tartományában, vagy egyikében sem; ha van közöttük végtelen is, az adott pont biztosan nem eleme  $p(S_2)$  értelmezési tartományának, de eleme lehet  $\mathcal{D}_{p(S_1)}$ -nek. Tehát  $\mathcal{D}_{p(S_2)} \subseteq \mathcal{D}_{p(S_1)}$  és  $\forall a \in \mathcal{D}_{p(S_2)} : p(S_1)(a) \subseteq p(S_2)(a)$ .

Mivel  $S_2$  megoldása  $F$ -nek, ezért  $\mathcal{D}_F \subseteq \mathcal{D}_{p(S_2)}$  és  $\forall a \in \mathcal{D}_F : p(S_2)(a) \subseteq F(a)$ . A fentiek miatt  $\mathcal{D}_F \subseteq \mathcal{D}_{p(S_1)}$  is és  $\forall a \in \mathcal{D}_F : p(S_1)(a) \subseteq F(a)$  is teljesül, vagyis  $S_1$  is megoldása  $F$ -nek.

## 2.8. Feladatok

**2.1.** Legyen  $A = \{\Omega, \Phi, \Psi, \Theta, \Gamma\}$ ,  $S \subseteq A \times A^{**}$ .

$$S = \left\{ \begin{array}{lll} (\Omega, \langle \Omega\Phi\Gamma\Omega \rangle), & (\Omega, \langle \Omega\Theta\Psi\Gamma \rangle), & (\Omega, \langle \Omega\Psi\Phi \dots \rangle), \\ (\Phi, \langle \Phi\Omega \rangle), & (\Psi, \langle \Psi\Theta \rangle), & (\Psi, \langle \Psi\Psi\Psi\Psi\Psi \dots \rangle), \\ (\Theta, \langle \Theta\Omega\Gamma\Omega\Theta \rangle), & (\Theta, \langle \Theta\Psi\Omega\Phi\Gamma\Omega \rangle), & (\Theta, \langle \Theta\Omega\Gamma\Theta\Phi \rangle), \\ (\Gamma, \langle \Gamma\Phi\Psi \rangle), & (\Gamma, \langle \Gamma\Psi \rangle), & (\Gamma, \langle \Gamma\Phi\Psi\Omega \rangle) \end{array} \right\}$$

$$F = \{(\Phi, \Omega) (\Phi, \Psi) (\Theta, \Omega) (\Theta, \Phi) (\Theta, \Theta)\}.$$

- Adjuk meg  $p(S)$ -t!
- Megoldja-e  $S$  az  $F$  feladatot?

**2.2.** Legyen  $S$  program,  $F$  olyan feladat, hogy  $S$  megoldása  $F$ -nek. Igaz-e, hogy

- ha  $F$  nemdeterminisztikus, akkor  $S$  sem az?
- ha  $F$  determinisztikus, akkor  $S$  is az?
- ha  $F$  nemdeterminisztikus, akkor  $p(S)$  sem az?
- ha  $p(S)$  determinisztikus, akkor  $F$  is az?
- ha  $F$  determinisztikus, akkor  $p(S)$  is az?
- ha  $S$  nemdeterminisztikus, akkor  $p(S)$  sem az?

**2.3.** Igaz-e, hogy  $p(S)$  értelmezési tartománya éppen  $A^*$  ősképe  $S$ -re nézve?



**2.4.** Mondhatjuk-e, hogy az  $S$  program megoldja az  $F$  feladatot, ha igaz a következő állítás:

$$q \in \mathcal{D}_F \Rightarrow S(q) \subseteq A^* \text{ és } p(S)(q) \subseteq F(q)?$$

**2.5.** Legyen  $A = \mathbb{N} \times \mathbb{N}$ ,  $F_1, F_2 \subseteq A \times A$ .

$$F_1 = \{((u, v), (x, y)) \mid y|u\},$$

$$F_2 = \{((u, v), (x, y)) \mid x = u \text{ és } y|u\}.$$

Ugyanaz-e a két feladat? (Van-e valamilyen összefüggés közöttük?)

**2.6.**  $F \subseteq A \times A$ .  $S_1, S_2$  programok  $A$ -n. Az  $S_1$  és az  $S_2$  is megoldja az  $F$  feladatot. Igaz-e, hogy az  $S = (S_1 \cup S_2)$  program is megoldja az  $F$  feladatot?

**2.7.** Tekintsük a következő szövegesen megadott feladatot: Adott egy sakktábla és két rajta lévő bástya helyzete. Helyezzünk el a táblán egy harmadik bástyát úgy, hogy az mindkettőnek az ütésében álljon! Készítsük el a modellt: írjuk fel az állapotteret és az  $F$  relációt!

**2.8.** Tudjuk, hogy  $S$  megoldja  $F$ -et (az  $A$  állapottéren). Igaz-e, hogy ha  $a \in A$ , akkor

$$S(a) \not\subseteq A^* \text{ vagy } p(S)(a) \not\subseteq F(a) \Rightarrow a \notin \mathcal{D}_F?$$

**2.9.** Legyen  $F \subseteq A \times A$  egy feladat és  $S \subseteq A \times A^{**}$  egy program. Jelöljük  $FP$ -vel azt a relációt, amely  $F$  és  $p(S)$  metszeteként áll elő. Igaz-e, hogy

- a) ha  $\mathcal{D}_{FP} = \mathcal{D}_F$ , akkor  $S$  megoldja  $F$ -et?
- b) ha  $S$  megoldja  $F$ -et, akkor  $\mathcal{D}_{FP} = \mathcal{D}_F$ ?



## 3. fejezet

# Specifikáció

A megoldás definíciója közvetlenül elég nehézkesen használható a programok készítése során, hiszen az, hogy egy program megold-e egy feladatot, a megoldás eddigi definíciója alapján csak nehezen ellenőrizhető. Ezért bevezetünk néhány új fogalmat, majd ezek segítségével megadjuk a megoldás egy elégséges feltételét.

### 3.1. A leggyengébb előfeltétel

Először a program működésének eredményét adjuk meg egy, a programfüggvénynél kényelmesebben használható jellemzővel.

#### 3.1. DEFINÍCIÓ: LEGGYENGÉBB ELŐFELTÉTEL

Legyen  $S \subseteq A \times A^{**}$  program,  $R : A \rightarrow \mathbb{L}$  állítás. Ekkor az  $S$  program  $R$  utófeltételhez tartozó *leggyengébb előfeltétele* az az  $lf(S, R) : A \rightarrow \mathbb{L}$  függvény, amelyre:

$$[lf(S, R)] = \{a \in A \mid a \in \mathcal{D}_{p(S)} \text{ és } p(s)(a) \subseteq [R]\}.$$

A leggyengébb előfeltétel tehát pontosan azokban a pontokban igaz, ahonnan kiindulva az  $S$  program biztosan terminál, és az összes lehetséges végállapotra igaz  $R$ .

Természetesen a leggyengébb előfeltétel igazsághalmazán kívül is lehetnek olyan pontok, amelyből a program egy futása eljut az utófeltétel igazsághalmazába, csak azokból a pontokból nem garantált, hogy oda jut.

Egy program működése úgy is jellemezhető, hogy megadjuk a program tetszőleges utófeltételhez tartozó leggyengébb előfeltételét. A feladat megoldása során az a célunk, hogy olyan programot találjunk, amelyik bizonyos feltételeknek eleget tevő pontokban terminál. Ezért azt mondhatjuk, hogy ha a számunkra kedvező végállapotokra megadjuk a program leggyengébb előfeltételét, akkor a programfüggvény meghatározása nélkül jellemezzük a program működését.

Emlékeztetünk arra, hogy definiáltuk a reláció szerinti ősképfogalmát is, ennek felhasználásával azonnal látszik, hogy

$$\lceil lf(S, R) \rceil = p(S)^{-1}(\lceil R \rceil).$$

Felhasználva az igazsághalmaz definícióját és a szigorú kompozíció szerinti ősképtulajdonságát:

$$p(S)^{-1}(\lceil R \rceil) = p(S)^{-1}(R^{-1}(\{igaz\})) = (R \odot p(S))^{-1}(\{igaz\}) = \lceil R \odot p(S) \rceil.$$

Mivel  $R$  függvény, a kompozíció és a szigorú kompozíció megegyezik, tehát

### 3.1. állítás:

$$\lceil lf(S, R) \rceil = \lceil R \circ p(S) \rceil.$$

Abban az esetben, ha  $p(S)$  is függvény,  $lf(S, R) = R \circ p(S)$ .

A fenti összefüggésekre gyakran fogunk hivatkozni.

A most következő tétel a leggyengébb előfeltétel néhány nevezetes tulajdonságáról szól.

### 3.1. TÉTEL: AZ $lf$ TULAJDONSÁGAI

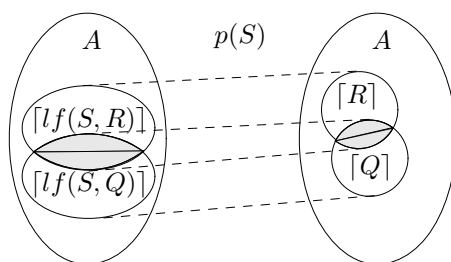
Legyen  $S \subseteq A \times A^{**}$  program,  $Q, R : A \rightarrow \mathbb{L}$  állítások. Ekkor

- (1)  $lf(S, Hamis) = Hamis$ ,
- (2) ha  $Q \Rightarrow R$ , akkor  $lf(S, Q) \Rightarrow lf(S, R)$ ,
- (3)  $lf(S, Q) \wedge lf(S, R) = lf(S, Q \wedge R)$ ,
- (4)  $lf(S, Q) \vee lf(S, R) \Rightarrow lf(S, Q \vee R)$ .

Az első tulajdonságot a csoda kizárása elvének, a másodikat monotonitási tulajdonságnak nevezzük.

#### Bizonyítás:

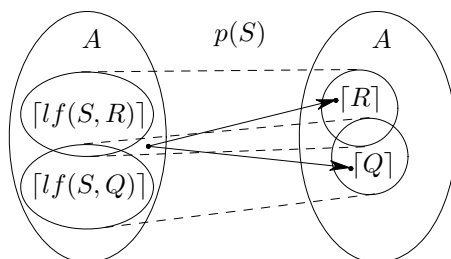
1. Indirekt: Tegyük fel, hogy  $\exists a \in \lceil lf(S, Hamis) \rceil$ . Ekkor a leggyengébb előfeltétel definíciója szerint:  $a \in \mathcal{D}_{p(S)}$  és  $p(S)(a) \subseteq \lceil Hamis \rceil = \emptyset$ . Ez nyilvánvaló ellentmondás.
2. Indirekt: Tegyük fel, hogy  $\exists a \in \lceil lf(S, Q) \rceil \setminus \lceil lf(S, R) \rceil$ . Ekkor  $a \in \mathcal{D}_{p(S)}$  és  $p(S)(a) \subseteq \lceil Q \rceil$  és  $p(S)(a) \not\subseteq \lceil R \rceil$ . Ez viszont ellentmond annak a feltételnek, mely szerint  $\lceil Q \rceil \subseteq \lceil R \rceil$ .
3. Az állítást két részben, a mindkét irányú következés belátásával bizonyítjuk.
  - (a)  $lf(S, Q) \wedge lf(S, R) \Rightarrow lf(S, Q \wedge R)$ , ugyanis:  
Legyen  $a \in \lceil lf(S, Q) \wedge lf(S, R) \rceil$ . Ekkor  $a \in \lceil lf(S, Q) \rceil$  és  $a \in \lceil lf(S, R) \rceil$ , azaz  $a \in \mathcal{D}_{p(S)}$  és  $p(S)(a) \subseteq \lceil Q \rceil$ , illetve  $p(S)(a) \subseteq \lceil R \rceil$ . Ekkor azonban  $p(S)(a) \subseteq \lceil Q \rceil \cap \lceil R \rceil = \lceil Q \wedge R \rceil$ , azaz  $a \in \lceil lf(S, Q \wedge R) \rceil$ .



3.1. ábra. A leggyengébb előfeltétel és a metszet kapcsolata

(b)  $lf(S, Q \wedge R) \Rightarrow lf(S, Q) \wedge lf(S, R)$ , ugyanis:

Legyen  $a \in [lf(S, Q \wedge R)]$ . Ekkor a leggyengébb előfeltétel definíciója alapján  $a \in \mathcal{D}_{p(S)}$  és  $p(S)(a) \subseteq [Q \wedge R]$ . Felhasználva, hogy  $[Q \wedge R] = [Q] \cap [R]$ , adódik, hogy  $p(S)(a) \subseteq [Q]$  és  $p(S)(a) \subseteq [R]$ , azaz  $a \in [lf(S, Q)]$  és  $a \in [lf(S, R)]$ , tehát  $a \in [lf(S, Q) \wedge lf(S, R)]$ .



3.2. ábra. A leggyengébb előfeltétel és az unió kapcsolata

4. Legyen  $a \in [lf(S, Q) \vee lf(S, R)]$ . Ekkor  $a \in [lf(S, Q)]$  vagy  $a \in [lf(S, R)]$ .  
Ha  $a \in [lf(S, Q)]$ , akkor – a monotonitási tulajdonság alapján –  $a \in [lf(S, Q \vee R)]$ . Hasonlóan, ha  $a \in [lf(S, R)]$ , akkor  $a \in [lf(S, Q \vee R)]$ .

□

A (4) tulajdonság visszafelé nem igaz.  $p(S)(a) \subseteq [Q] \cup [R]$ -ből nem következik sem  $p(S)(a) \subseteq [Q]$ , sem  $p(S)(a) \subseteq [R]$ . Természetesen, ha  $p(S)$  determinisztikus, azaz  $\forall a \in A : p(S)(a)$  legfeljebb egy elemű halmaz, akkor az egyenlőség fennáll.

## 3.2. A feladat specifikációja

A következőkben bevezetjük a feladat megadásának egy másik módját, és kimondunk egy, a gyakorlat szempontjából nagyon fontos tételt.

Általában a feladat nem függ az állapotter összes komponensétől, azaz az állapotter több pontjához is ugyanazt rendeli. Ezeket a pontokat fogjuk össze egy ponttá a paraméterter segítségével.

### 3.2. DEFINÍCIÓ: PARAMÉTERTER

Legyen  $F \subseteq A \times A$  feladat. A  $B$  halmazt a feladat *paraméterterének* nevezzük, ha van olyan  $F_1$  és  $F_2$  reláció, hogy

$$\begin{aligned} F_1 &\subseteq A \times B, \\ F_2 &\subseteq B \times A, \\ F &= F_2 \circ F_1. \end{aligned}$$

Fontos észrevenni, hogy paraméterteret mindig lehet találni. Például maga a feladat állapottere minden esetben választható paraméterternek úgy, hogy a definícióban szereplő  $F_1$  relációnak az identikus leképezést,  $F_2$ -nek pedig magát az  $F$  feladatot választjuk. Ám az, hogy egy konkrét esetben mit is választunk paraméterternek, a feladattól függ. Általában úgy választjuk meg a paraméterteret, hogy a következő tételt kényelmesen tudjuk használni.

### 3.2. TÉTEL: SPECIFIKÁCIÓ TÉTELE

Legyen  $F \subseteq A \times A$  feladat,  $B$  az  $F$  egy paramétertere,  $F_1 \subseteq A \times B$ ,  $F_2 \subseteq B \times A$ ,  $F = F_2 \circ F_1$ . Legyen  $b \in B$ , és definiáljuk a következő állításokat:

$$\begin{aligned} [Q_b] &= \{a \in A \mid (a, b) \in F_1\} = F_1^{(-1)}(b), \\ [R_b] &= \{a \in A \mid (b, a) \in F_2\} = F_2(b). \end{aligned}$$

Ekkor ha  $\forall b \in B : Q_b \Rightarrow lf(S, R_b)$ , akkor az  $S$  program megoldja az  $F$  feladatot.

**Bizonyítás:** A megoldás definíciója két pontjának teljesülését kell belátnunk:

1.  $\mathcal{D}_F \subseteq \mathcal{D}_{p(S)}$ , ugyanis:

Legyen  $a \in \mathcal{D}_F$  tetszőleges. Ekkor az  $F_1$  és  $F_2$  relációk definíciója miatt  $a \in \mathcal{D}_{F_1}$  és

$$\exists b \in B : a \in [Q_b].$$

De ekkor a tétel feltétele alapján:

$$a \in [Q_b] \subseteq [lf(S, R_b)] \subseteq \mathcal{D}_{p(S)}.$$

2.  $\forall a \in \mathcal{D}_F : p(S)(a) \subseteq F(a)$ , ugyanis:

Legyen  $a \in \mathcal{D}_F$  tetszőlegesen rögzített,  $b \in B$  olyan, amelyre  $a \in [Q_b]$ . Ekkor a feltétel szerint:

$$p(S)(a) \subseteq [R_b] = F_2(b) \subseteq F_2(F_1(a)) = F(a).$$

□

A specifikáció tétele csak elégséges feltétel a megoldásra, azaz nem megfordítható: lehet adni olyan feladat-program párt, ahol a program megoldja a feladatot, de a specifikáció tétele nem teljesül. Ez természetesen attól is függ, hogy a feladatot hogyan specifikáljuk, azaz milyen paraméterteret választunk, és hogyan bontjuk a feladatot  $F_1$  és  $F_2$  relációk kompozíciójára.

A későbbiekben egy programozási feladat megoldásakor a specifikációból, tehát a specifikáció tételének megfelelő formában megadott feladatból indulunk ki, és a specifikáció tételének felhasználásával keressük a megoldó programot. Mivel a specifikáció tétele nem megfordítható, előfordulhat, hogy ezen az úton nem találunk megoldó programot. Elkerülhető-e ez a probléma? Azaz arra keressük a választ, hogy tetszőleges feladat estén megadható-e olyan specifikáció, amelyre a specifikáció tétele megfordítható. Nevezzük az ilyen specifikációt *jónak*.

**3.2. állítás:** Ha a feladat specifikációjának felírásakor úgy választjuk meg a paraméterteret és az elő-, utófeltételeket, hogy rájuk a következő két feltétel teljesüljön:

- (1)  $\forall b \in B : [Q_b] \neq \emptyset \Rightarrow [R_b] \neq \emptyset$ ,
- (2)  $\forall b_1, b_2 \in B : [Q_{b_1}] \cap [Q_{b_2}] = \emptyset$  vagy  
 $[Q_{b_1}] = [Q_{b_2}]$  és  $[R_{b_1}] = [R_{b_2}]$ ,

akkor a specifikáció tétele megfordítható.

Ez az állítás egyszerűen bizonyítható. Azt kell belátni, hogy ha  $S$  megoldása  $F$ -nek, akkor:

- a.  $\forall b \in B : [Q_b] \subseteq \mathcal{D}_{p(S)}$
- b.  $\forall b \in B : \forall a \in [Q_b] : p(S)(a) \subseteq [R_b]$ .

Az a. következik abból, hogy (1) teljesülése esetén  $\forall b \in B : [Q_b] \subseteq \mathcal{D}_F$ , a b. pedig abból, hogy ha  $a \in [Q_b]$ , akkor  $a \in \mathcal{D}_F$ , és a megoldás miatt

$$p(S)(a) \subseteq F(a) = \bigcup_{x \in B \text{ és } a \in [Q_x]} [R_x] = [R_b],$$

ugyanis (2) miatt az unió egy tagú, vagy minden tagja egyenlő.

Tehát a 3.2 állítás két feltételének megfelelő specifikáció *jó specifikáció*. A kérdés már csak az, hogy létezik-e minden feladat esetén ilyen? Ha a paraméterter az állapotter,  $F_1$  az identikus leképezés és  $F_2 = F$ , akkor a 3.2 állítás mindkét feltétele nyilvánvalóan teljesül.

### 3.3. A változó fogalma

Az eddig elmondottak alapján a specifikáció tétele még nem lenne hatékonyan használható, hiszen a paraméterter minden pontjára ellenőriznünk kellene a feltételek teljesülését. Ezért bevezetjük a változó fogalmát, amelynek a segítségével a feltételrendszer teljesülése egyszerűbben ellenőrizhetővé válik.

**3.3. DEFINÍCIÓ: VÁLTOZÓ**

Az  $A = \prod_{i \in I} A_i$  állapotter  $v_i : A \rightarrow A_i$  egydimenziós projekciós függvényeit *változóknak* nevezzük.

A változók használatával egyszerűsíthetjük az állapottéren értelmezett állítások (elő- és utófeltételek, leggyengébb előfeltétel) és relációk (programfüggvény) leírását.

Mivel minden változó értelmezési tartománya az állapotter és értékészlete egy típusértékhalmoz, egy változót jellemezhetünk egy típusal, azaz beszélhetünk a változó típusáról.

Ha a paraméterter is direktorzat alakú – márpedig ez gyakran így van, ugyanis általában az állapotter egy altere –, akkor a paraméterter egydimenziós projekciós függvényeit *paraméterváltozóknak* nevezzük.

Az állapotter, illetve a paraméterter egyes komponenseihez tartozó változókat, illetve paraméterváltozókat az adott komponens alá írjuk.

Most megvizsgáljuk, hogyan lehet a specifikáció tétele segítségével feladatokat megfogalmazni.

Tekintsünk egy már ismert feladatot: határozzuk meg két egész szám összegét!

Először felírjuk az állapotteret úgy, mint eddig, csak kiegészítjük a változónevek megadásával.

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$x \quad y \quad z$$

Az eddigi jelöléseink alkalmazásával a feladat

$$F = \{((u_1, u_2, u_3), (v_1, v_2, v_3)) \in A \times A \mid v_3 = u_1 + u_2\}.$$

A specifikáció tételének alkalmazásához írjuk föl a paraméterteret is:

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$x' \quad y'$$

Majd még visszatérünk arra, miért éppen így választottuk meg a paraméterteret.

Tehát az állapotter három egész komponensből áll, melyeknek változói rendre  $x$ ,  $y$  és  $z$ . A paraméterter két egész komponensből áll, az első komponens változója  $x'$ , a másodiké  $y'$ .

Legyen az  $F_1$  reláció a következő:

$$F_1 = \{((u_1, u_2, u_3), (b_1, b_2)) \in A \times B \mid u_1 = b_1 \text{ és } u_2 = b_2\},$$

$F_2$  pedig:

$$F_2 = \{((b_1, b_2), (v_1, v_2, v_3)) \in B \times A \mid v_3 = b_1 + b_2\}.$$

A fentiekből adódik, hogy  $F_2 \circ F_1 = F$ . A paraméterter egy tetszőleges  $b$  eleméhez tartozó elő- és utófeltételre:

$$[Q_b] = \{(a_1, a_2, a_3) \in A \mid a_1 = b_1 \text{ és } a_2 = b_2\},$$

$$[R_b] = \{(a_1, a_2, a_3) \in A \mid a_3 = b_1 + b_2\},$$



amit az állapotter és a paraméterter változóinak felhasználásával is fölírhatunk:

$$\begin{aligned} [Q_b] &= \{a \in A \mid x(a) = x'(b) \text{ és } y(a) = y'(b)\}, \\ [R_b] &= \{a \in A \mid z(a) = x'(b) + y'(b)\}. \end{aligned}$$

A függvénytereknél tárgyaltuk, hogy a függvényter elemein a relációk egy logikai függvényekből álló teret generálnak.  $x, y, z$  egy függvényter elemei,  $x'(b), y'(b)$  szintén annak tekinthetők,  $A$ -n értelmezett konstans függvények. Ezért  $x = x'(b)$  és  $y = y'(b)$  az állapotterén értelmezett logikai függvények, ahogy  $x = x'(b) \wedge y = y'(b)$  is az. Ebből következik, hogy

$$\begin{aligned} [Q_b] &= [x = x'(b) \wedge y = y'(b)], \\ [R_b] &= [z = x'(b) + y'(b)], \end{aligned}$$

és mivel mindegyik függvény,

$$\begin{aligned} Q_b &= (x = x'(b) \wedge y = y'(b)), \\ R_b &= (z = x'(b) + y'(b)). \end{aligned}$$

A jelölés egyszerűsíthető, mivel nyilvánvaló, hogy a paraméterváltozók argumentuma  $b$ , ezek el is hagyhatók, sőt általában az sem okoz félreértést, ha az elő- és utófeltételek indexeit elhagyjuk. Ezek a feltételek a paraméterter pontjaihoz tartoznak, és így a paraméterváltozók értékeitől függenek. A feladat specifikációja tehát:

$$\begin{aligned} A &= \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \\ &\quad x \quad y \quad z \\ B &= \mathbb{Z} \times \mathbb{Z} \\ &\quad x' \quad y' \\ Q &: (x = x' \wedge y = y') \\ R &: (z = x' + y') \end{aligned}$$

A továbbiakban a feladatot úgy definiáljuk, hogy megadjuk az állapotterét ( $A$ ), a paraméterterét ( $B$ ), valamint az elő- és utófeltételét ( $Q$ , illetve  $R$ ) a paraméterter minden pontjára, azaz paraméteresen. Ebben az esetben azt mondjuk, hogy a feladatot megadtuk a specifikáció tételének megfelelő formában, vagy ha nem okoz félreértést, specifikáltuk a feladatot.

Egy feladatot nagyon sokféleképpen lehet specifikálni, a sok lehetőség közül az egyik az, amit elő-, utófeltétellel történő specifikációnak szoktak nevezni, és nagyon hasonlít arra, amit most tárgyalunk. Felhívjuk a figyelmet arra, hogy a hasonlóság ellenére a kettő nem azonos.

Paraméterternek általában az állapotter egy alterét szoktuk választani. Azokat a komponenseket válogatjuk ki, amelyek értékétől függ, hogy a feladat mihez mit rendel, amelyek *paraméterezik* a feladatot. Lényegében azt fogalmazzuk meg, hogy az állapotter milyen tulajdonságú pontjaiból milyen tulajdonságú pontokba akarunk jutni.

A paraméterteret arra használjuk, hogy megadjuk, milyen összefüggés van az elérendő és a kiinduló állapotok között.

Ha egy programnak meg tudjuk határozni a (paraméteres) utófeltételhez tartozó leggyengébb előfeltételét, akkor a specifikáció tétele alapján könnyen eldönthetjük, hogy megoldása-e a specifikált feladatnak, más szóval *bebizonyíthatjuk a program helyességét*. Megjegyezzük azonban, hogy legtöbbször a „fordított” utat fogjuk követni, nem a program helyességét bizonyítjuk, hanem bizonyítottan helyes programot állítunk elő.

A későbbiekben bevezetünk majd olyan eszközöket, amelyek segítségével a feladat specifikációjából kiindulva olyan programokat készíthetünk, amelyek megoldják a feladatot.

A változókhoz kapcsolódóan bevezetünk még néhány egyszerű fogalmat, amelyeknek a szemléletes jelentése eléggé nyilvánvaló.

Azt mondjuk, hogy az  $S$  program  $p(S)$  programfüggvénye nem függ az állapotér  $a_i : A_i$  változójától, ha

$$\forall x, y \in \mathcal{D}_{p(S)} : (\forall k \in I \setminus \{i\} : x(k) = y(k) \Rightarrow p(S)(x) = p(S)(y)).$$

Azt mondjuk, hogy az  $S$  program  $a_i : A_i$  változója konstans, ha

$$\forall a \in A : \forall \alpha \in S(a) : \forall k \in \mathcal{D}_\alpha : \alpha_k(i) = a(i).$$

Azt mondjuk, hogy az  $S$  program végrehajtása nem változtatja meg az  $a_i : A_i$  változót, ha

$$\forall a \in \mathcal{D}_{p(S)} : \forall b \in p(S)(a) : b(i) = a(i).$$

## 3.4. Példák

**3.1. példa:** Legyen  $A = \{Bach, Bartók, Kodály, Liszt, Mozart, Vivaldi\}$ ,  $S \subseteq A \times A^{**}$  program.

$$S = \{ \begin{array}{ll} Vivaldi & \rightarrow \langle Vivaldi, Bach \rangle, & Bach & \rightarrow \langle Bach, Mozart \rangle, \\ Bach & \rightarrow \langle Bach, Liszt, Bartók \rangle, & Mozart & \rightarrow \langle Mozart, Vivaldi \rangle, \\ Liszt & \rightarrow \langle Liszt, Bartók \rangle, & Kodály & \rightarrow \langle Kodály, Mozart \rangle, \\ Bartók & \rightarrow \langle Bartók, Bach, Liszt \rangle \end{array} \}$$

Legyen továbbá az  $R : A \rightarrow \mathbb{L}$  állítás:

$$\forall x \in A : R(x) = (x \text{ magyar}).$$

Mi lesz a fenti program  $R$ -hez tartozó leggyengébb előfeltétele?

**Megoldás:** Írjuk fel először a program programfüggvényét:

$$p(S) = \{ \begin{array}{lll} (Vivaldi, Bach), & (Bach, Mozart), & (Bach, Bartók), \\ (Mozart, Vivaldi), & (Liszt, Bartók), & (Kodály, Mozart), \\ (Bartók, Liszt) & & \} \end{array} \}$$

Ezek után, a leggyengébb előfeltétel definícióját felhasználva:

$$\lceil lf(S, R) \rceil = \{Bartók, Liszt\}, \text{ ugyanis}$$

$$\begin{aligned} p(S)(Vivaldi) &= \{Bach\} \not\subseteq \lceil R \rceil \\ p(S)(Kodály) &= \{Mozart\} \not\subseteq \lceil R \rceil \\ p(S)(Bartók) &= \{Liszt\} \subseteq \lceil R \rceil \\ p(S)(Bach) &= \{Mozart, Bartók\} \not\subseteq \lceil R \rceil \\ p(S)(Mozart) &= \{Vivaldi\} \not\subseteq \lceil R \rceil \\ p(S)(Liszt) &= \{Bartók\} \subseteq \lceil R \rceil \end{aligned}$$

**3.2. példa:** Legyen  $H_1, H_2 : A \rightarrow \mathbb{L}$ . Igaz-e, hogy ha minden  $S \subseteq A \times A^{**}$  programra  $lf(S, H_1) = lf(S, H_2)$ , akkor  $\lceil H_1 \rceil = \lceil H_2 \rceil$ ?

**Megoldás:** Felhasználva, hogy a leggyengébb előfeltételek minden programra megegyeznek, egy alkalmas program választásával a válasz egyszerűen megadható: rendelje az  $S$  program az állapottér minden eleméhez az önmagából álló egy hosszúságú sorozatot. Ekkor könnyen belátható, hogy tetszőleges  $R$  utófeltétel esetén:

$$lf(S, R) = R.$$

Ekkor viszont

$$H_1 = lf(S, H_1) = lf(S, H_2) = H_2,$$

tehát a két feltétel megegyezik.

**3.3. példa:** Specifikáljuk a következő feladatot:  $A = \mathbb{L} \times \mathbb{L}$ ,  $F \subseteq A \times A$ ,

$$F = \{((l, k), (m, n)) \mid n = k \text{ és } m = (l \wedge k)\}.$$

**Megoldás:**

$$\begin{aligned}
 A &= \mathbb{L} \times \mathbb{L} \\
 &\quad x \quad y \\
 B &= \mathbb{L} \times \mathbb{L} \\
 &\quad x' \quad y' \\
 Q &: (x = x' \wedge y = y') \\
 R &: (x = (x' \wedge y') \wedge y = y')
 \end{aligned}$$

**3.4. példa:** Legyen  $F \subseteq A \times A$ ,  $S \subseteq A \times A^{**}$  program,  $B$  egy tetszőleges halmaz. Legyenek továbbá  $F_1 \subseteq A \times B$  és  $F_2 \subseteq B \times A$  olyan relációk, hogy  $F = F_2 \circ F_1$ , valamint  $\forall b \in B$ :

$$\begin{aligned}
 [\widehat{Q}_b] &= F_1^{-1}(b), \\
 [R_b] &= F_2(b).
 \end{aligned}$$

Igaz-e, hogy ha  $\forall b \in B : \widehat{Q}_b \Rightarrow lf(S, R_b)$ , akkor  $S$  megoldja  $F$ -et?

**Megoldás:** Próbáljuk meg a megoldás definíciójának két pontját belátni. Legyen  $a \in \mathcal{D}_F$ . Be kellene látnunk, hogy  $a \in \mathcal{D}_{p(S)}$ . Nézzük meg a specifikáció tételének bizonyítását: ott felhasználtuk, hogy ekkor van olyan  $b \in B$ , hogy  $a \in [Q_b]$ . Igaz ez a  $\widehat{Q}_b$ -re is? Sajnos – mivel  $\widehat{Q}_b$ -t ősképpel definiáltuk – ez nem feltétlenül van így. Próbáljunk a fenti gondolatmenet alapján ellenpéldát adni:

Legyen  $A = \{1\}$ ,  $B = \{1, 2\}$ ,  $F = \{(1, 1)\}$ ,  $F_1 = \{(1, 1), (1, 2)\}$ ,  $F_2 = \{(2, 1)\}$ . Ekkor  $\widehat{Q}_1 = Hamis$  és  $\widehat{Q}_2 = Hamis$ , tehát az állítás feltételei teljesülnek, függetlenül a programtól (ugyanis „hamisból minden következik”). Válasszuk most az alábbi programot:  $S = \{(1, < 1, 1, \dots >)\}$ . Ez a program nem megoldása a feladatnak, de teljesülnek rá is az állítás feltételei. Tehát az állítás nem igaz.

**3.5. Feladatok**

**3.1.** Legyen  $A = \{1, 2, 3, 4, 5\}$ ,  $S \subseteq A \times A^{**}$ .

$$\begin{aligned}
 S = \{ & (1, \langle 1251 \rangle), \quad (1, \langle 14352 \rangle), \quad (1, \langle 132 \dots \rangle), \quad (2, \langle 21 \rangle), \\
 & (2, \langle 24 \rangle), \quad (3, \langle 333333 \dots \rangle), \quad (4, \langle 41514 \rangle), \quad (4, \langle 431251 \rangle), \\
 & (4, \langle 41542 \rangle), \quad (5, \langle 524 \rangle), \quad (5, \langle 534 \rangle), \quad (5, \langle 5234 \rangle) \quad \} \\
 \text{és } [R] &= \{1, 2, 5\}. \text{ Írja fel az } [lf(S, R)] \text{ halmazt!}
 \end{aligned}$$

**3.2.** Mivel egyenlő  $lf(S, Igaz)$ ?

**3.3.** Legyen  $A$  tetszőleges állapottér,  $Q_i : A \rightarrow \mathbb{L} \quad (i \in \mathbb{N})$ . Igaz-e, hogy ha

$$\forall i \in \mathbb{N} : Q_i \Rightarrow Q_{i+1},$$

akkor

$$(\exists n \in \mathbb{N} : lf(S, Q_n)) = lf(S, (\exists n \in \mathbb{N} : Q_n))?$$

- 3.4.** Igaz-e, hogy ha  $lf(S_1, R) = lf(S_2, R)$ , akkor  $lf(S_1 \cup S_2, R) = lf(S_1, R) \vee lf(S_2, R)$ ?
- 3.5.** Igaz-e, hogy ha  $\forall x, y \in A : x \in [lf(S_1, \mathcal{P}(\{y\}))] \Leftrightarrow x \in [lf(S_2, \mathcal{P}(\{y\}))]$ , akkor  $\mathcal{D}_{p(S_1)} = \mathcal{D}_{p(S_2)}$ ?
- 3.6.**  $S_1, S_2 \subseteq A \times A^{**}$  programok. Igaz-e, hogy ha  $\forall H : A \rightarrow \mathbb{L}$  esetén  $lf(S_1, H) = lf(S_2, H)$ , akkor  $S_1$  ekvivalens  $S_2$ -vel?
- 3.7.**  $A = \mathbb{N}$ .  $S \subseteq \mathbb{N} \times \mathbb{N}^{**}$ .

$$S = \{(a, \langle a \dots \rangle) \mid a \equiv 1 \pmod{4}\} \\ \cup \{(b, \langle b \rangle), (b, \langle b, b/2 \rangle) \mid b \equiv 2 \pmod{4}\} \\ \cup \{(c, \langle c, 2c \rangle) \mid c \equiv 3 \pmod{4}\} \\ \cup \{(d, \langle d, d/2 \rangle) \mid d \equiv 0 \pmod{4}\},$$

$$\lceil H \rceil = \{x \in \mathbb{N} \mid 2 \mid x\}. \lceil lf(S, H) \rceil = ?$$

- 3.8.** Adott az  $A = V \times V \times \mathbb{L}$  állapotér ( $V = \{1, 2, 3\}$ ) és a  $B = V \times V$  paraméterter, továbbá az  $F_1$  és  $F_2$  feladatok.

$$F_1 = \{((a_1, a_2, l), (b_1, b_2, k)) \mid k = (a_1 > a_2)\},$$

$F_2$  specifikációja pedig:

$$A = V \times V \times \mathbb{L} \\ a_1 \quad a_2 \quad l$$

$$B = V \times V \\ a'_1 \quad a'_2$$

$$Q : (a_1 = a'_1 \wedge a_2 = a'_2)$$

$$R : (Q \wedge l = (a'_1 > a'_2))$$

Azonosak-e az  $F_1$  és  $F_2$  feladatok?

- 3.9.** Tekintsük az alábbi két feladatot.  $F_1$  specifikációja:

$$A = \mathbb{Z} \times \mathbb{Z} \\ x \quad y$$

$$B = \mathbb{Z} \\ x'$$

$$Q : (x = x')$$

$$R : (Q \wedge x = |y \cdot y|)$$

$$F_2 = \{((a, b), (c, d)) \mid c = a \wedge |d| \cdot d = c\}.$$

Megadható-e valamilyen összefüggés  $F_1$  és  $F_2$  között?

**3.10.** Írja le szövegesen az alábbi feladatot. Legyen  $f : \mathbb{Z} \rightarrow \mathbb{Z}$ ,

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{N}_0$$

$m \quad n \quad l$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$m' \quad n'$

$$Q : (m = m' \wedge n = n' \wedge m \leq n),$$

$$R : (Q \wedge l = \sum_{i=m}^n g(i)),$$

$$\text{ahol } g : \mathbb{Z} \rightarrow \{0, 1\},$$

$$g(i) = \begin{cases} 1, & \text{ha } \forall j \in [m..n] : f(j) \leq f(i); \\ 0 & \text{különben.} \end{cases}$$

**3.11.** Igaz-e a specifikáció tételének megfordítása? (Ha  $S$  megoldja  $F$ -et, akkor  $\forall b \in B : Q_b \Rightarrow lf(S, R_b)$ )

**3.12.** Tekintsük az alábbi feladatot:

$$A = \mathbb{N} \times \mathbb{N}$$

$k \quad p$

$$B = \mathbb{N}$$

$k'$

$$Q : (k = k' \wedge 0 < k)$$

$$R : (Q \wedge \text{prim}(p) \wedge \forall i > 1 : \text{prim}(i) \rightarrow |k - i| \geq |k - p|),$$

$$\text{ahol } \text{prim} : \mathbb{N} \rightarrow \mathbb{L} \text{ és } [\text{prim}] = \{x \in \mathbb{N} \mid x \text{ prímszám}\}.$$

Mit rendel a fent specifikált feladat az  $a = (10, 1)$  és a  $b = (9, 5)$  pontokhoz? Fogalmazza meg szavakban a feladatot!

**3.13.**  $A = \mathbb{N} \times \mathbb{N} \times \mathbb{N}$

$x \quad y \quad z$

$$B = \mathbb{N} \times \mathbb{N}$$

$x' \quad y'$

$$F_1, F_2 \subseteq A \times A$$

$F_1$  specifikációja:

$$Q : (x = x' \wedge y = y')$$

$$R : (x = x' \wedge y = y' \wedge x'|z \wedge y'|z \wedge \forall j \in \mathbb{N} : (x'|j \wedge y'|j) \rightarrow z|j)$$

$$F_2 = \{((a, b, c), (d, e, f)) \mid a = d \text{ és } b = e \text{ és } f \mid a \cdot b \text{ és } a \mid f \text{ és } b \mid f\}$$

Megadható-e valamilyen összefüggés  $F_1$  és  $F_2$  között?

**3.14.** Adott egy  $f : \mathbb{Z} \rightarrow \mathbb{Z}$  függvény.

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$\quad \quad \quad m \quad n \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$\quad \quad \quad m' \quad n'$$

$$F_1, F_2 \subseteq A \times A$$

$F_1$  specifikációja:

$$Q : (m = m' \wedge n = n')$$

$$R : (m = m' \wedge n = n' \wedge i \in [m..n] \wedge \forall j \in [m..i-1] : f(j) < f(i) \wedge \forall j \in [i..n] : f(j) \leq f(i))$$

$F_2$  specifikációja:

$$Q : (m = m' \wedge n = n')$$

$$R : (i \in [m'..n'] \wedge \forall j \in [m'..n'] : f(j) \leq f(i)).$$

Azonos-e a két feladat?

**3.15.** Specifikáljuk a következő feladatot:  $A = \mathbb{N}$  és  $v : \mathbb{N} \rightarrow \{0, 1\}$ .

$$F \subseteq A \times A, F = \{(s, s') \mid s' = \sum_{k=1}^n v(k)\}$$

**3.16.** Keressük meg egy természetes szám egy osztóját.

**3.17.** Keressük meg egy összetett természetes szám egy valódi osztóját.

**3.18.** Keressük meg egy természetes szám egy valódi osztóját.

**3.19.** Keressük meg egy természetes szám összes valódi osztóját.

**3.20.** Keressük meg egy természetes szám legnagyobb prímosztóját.

**3.21.** Állapítsuk meg, hány valódi osztója van egy természetes számnak.

**3.22.** Keressük az  $[m..n]$  intervallumban az első olyan számot, amelyiknek van valódi osztója.

**3.23.** Keressük az  $[m..n]$  intervallumban azt a számot, amelyiknek a legtöbb valódi osztója van, de nem osztható 6-tal.

**3.24.** Az  $[m..n]$  intervallumban melyik számnak van a legtöbb valódi osztója?





## 4. fejezet

# Kiterjesztések

Az előző fejezetekben bevezettük a program és a feladat fogalmát, és definiáltuk az azonos állapottéren levő feladat-program párok között a megoldás fogalmát. A gyakorlatban azonban általában a feladat és a program különböző állapottéren van értelmezve; példaként megemlíthetjük azt az esetet, amikor egy feladat megoldására a programban további változókat kell bevezetni, azaz a feladat állapotterét újabb komponensekkel kell bővíteni.

A továbbiakban megvizsgáljuk, hogy mit tudunk mondani a különböző állapottéren adott programok és feladatok viszonyáról a megoldás szempontjából, és ennek alapján általánosítjuk (kiterjesztjük) a megoldás fogalmát erre az esetre is.

### 4.1. A feladat kiterjesztése

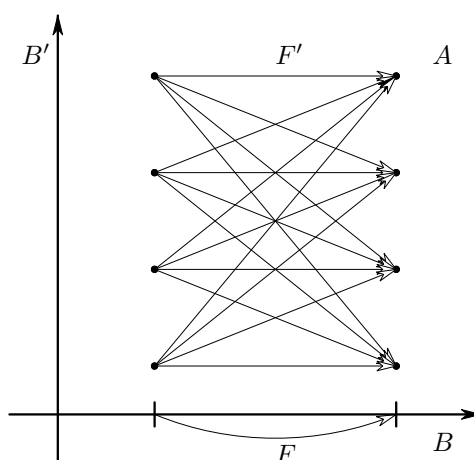
Ha egy feladat állapotterét kibővítjük újabb komponensekkel, mit jelentsen ez a feladat vonatkozásában? Elég kézenfekvő, hogy ebben az esetben a feladat ne tartalmazzon semmiféle kikötést az új komponensekre.

#### 4.1. DEFINÍCIÓ: FELADAT KITERJESZTÉSE

Legyen a  $B$  állapottér altere az  $A$  állapottérnek. Az  $F' \subseteq A \times A$  relációt az  $F \subseteq B \times B$  feladat *kiterjesztésének* nevezzük, ha

$$F' = \{(x, y) \in A \times A \mid (pr_B(x), pr_B(y)) \in F\}.$$

A definíciót úgy is megfogalmazhatjuk, hogy a feladat kiterjesztése az összes olyan  $A \times A$ -beli pontot tartalmazza, amelynek  $B$ -re vett projekciója benne van  $F$ -ben.



4.1. ábra. Feladat kiterjesztése

## 4.2. A program kiterjesztése

A program kiterjesztésének definíciójában az új komponensekre azt a kikötést tesszük, hogy azok nem változnak meg a kiterjesztett programban. Ezzel azt a gyakorlati követelményt írjuk le, hogy azok a változók, amelyeket a program nem használ, nem változnak meg a program futása során.

### 4.2. DEFINÍCIÓ: PROGRAM KITERJESZTÉSE

Legyen a  $B$  állapotter altere az  $A$  állapotternek, és jelölje  $B'$  a  $B$  kiegészítő alterét  $A$ -ra. Legyen továbbá  $S$  program a  $B$  állapotterén. Ekkor az  $S'$   $A$ -beli relációt az  $S$  program *kiterjesztésének* nevezzük, ha  $\forall a \in A$  :

$$S'(a) = \{\alpha \in A^{**} \mid pr_B(\alpha) \in S(pr_B(a)) \text{ és } \forall i \in D_\alpha : pr_{B'}(\alpha_i) = pr_{B'}(a)\}.$$

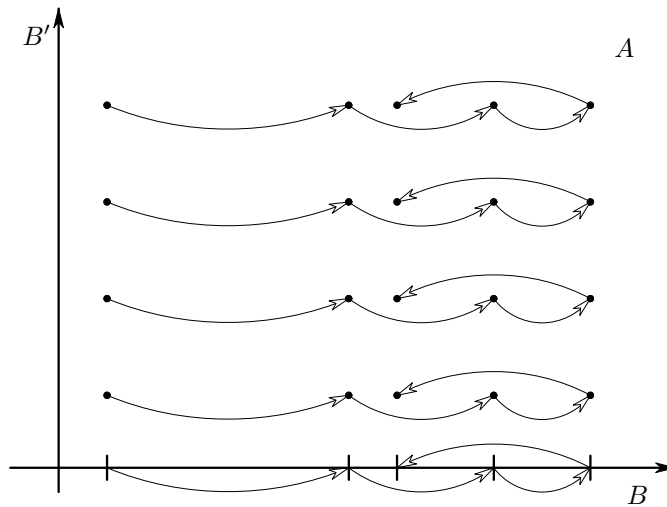
A fenti definíció alapján a kiterjesztett program értékészletében csak olyan sorozatok vannak, amelyek „párhuzamosak” valamely sorozattal az eredeti program értékészletéből.

Vajon a kiterjesztés megtartja-e a program-tulajdonságot? Erre a kérdésre válaszol az alábbi állítás.

**4.1. állítás:** Legyen a  $B$  állapotter altere az  $A$  állapotternek, és jelölje  $B'$  a  $B$  kiegészítő alterét  $A$ -ra. Legyen továbbá  $S$  program a  $B$  állapotterén, és  $S'$  az  $S$  kiterjesztése  $A$ -ra. Ekkor  $S'$  program.

A bizonyítás rendkívül egyszerű, a feladatok között szerepel.

Két azonos állapotterén definiált programot ekvivalensnek neveztünk, ha a programfüggvényük megegyezett. Most ezt a fogalmat is általánosítjuk.



4.2. ábra. Program kiterjesztése

**4.3. DEFINÍCIÓ: PROGRAMOK EKVIVALENCIÁJA**

Legyenek  $S_1 \subseteq A_1 \times A_1^{**}$ ,  $S_2 \subseteq A_2 \times A_2^{**}$  programok,  $B$  altere mind  $A_1$ -nek, mind  $A_2$ -nek. Azt mondjuk, hogy  $S_1$  ekvivalens  $S_2$ -vel  $B$ -n, ha

$$pr_B(p(S_1)) = pr_B(p(S_2)).$$

Nyilvánvaló, hogy a definíció valóban általánosítása az egyszerű esetnek. A definíciónak egyszerű következménye az is, hogy a két ekvivalens program a közös altéren pontosan ugyanazokat a feladatokat oldja meg.

Valójában attól, hogy két program ekvivalens – azaz megegyezik a programfüggvényük –, egyéb tulajdonságaik nagyon eltérők lehetnek. Ilyen – nem elhanyagolható – különbség lehet például a hatékonyságukban. Egyáltalán nem mindegy, hogy egy program mennyi ideig fut, és mekkora memóriára van szüksége. A program e jellemzőinek vizsgálatával azonban itt nem foglalkozunk.

A definíciókból közvetlenül adódik a következő állítás:

**4.2. állítás:** Egy program kiterjesztése és az eredeti program az eredeti állapottéren ekvivalens.

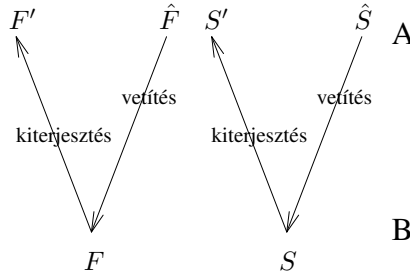
**4.3. Kiterjesztési tételek**

Az alábbiakban következő tételcsoport a megoldás és a kiterjesztések közötti kapcsolatot vizsgálja.

Legyen  $A$  egy állapottér, amelynek  $B$  altere. A  $B$ -n definiált feladatok és programok megfelelőinek tekinthetjük  $A$ -n a feladatok és programok kiterjesztéseit  $A$ -ra, az  $A$ -n definiált feladat megfelelőjének  $B$ -n pedig a feladat vetületét  $B$ -re. Az  $A$ -n definiált programok esetében a  $B$ -re való vetítés közvetlenül nem alkalmazható, mivel egy program vetülete nem biztos, hogy program, ugyanis nem biztos, hogy a sorozatok redukáltak. Természetesen, ha  $\hat{S} \subseteq A \times A^{**}$  program, akkor az

$$S = \{(b, \beta) \in B \times B^{**} \mid \exists (a, \alpha) \in \hat{S} : b = pr_B(a) \text{ és } \beta = red(pr_B(\alpha))\}$$

már program, és a  $B$  állapottéren  $S$  és  $\hat{S}$  ekvivalens. Tehát egy  $\hat{S} \subseteq A \times A^{**}$  programhoz mindig található olyan  $S \subseteq B \times B^{**}$  program, amely vele ekvivalens  $B$ -n.



4.3. ábra. Kapcsolat  $A$  és  $B$  között.

Ilyen módon, ahogy a 4.3. ábra is mutatja, a kiterjesztés és a vetítés segítségével kapcsolatot létesítünk az  $A$  és  $B$  állapottéren definiált programok között.

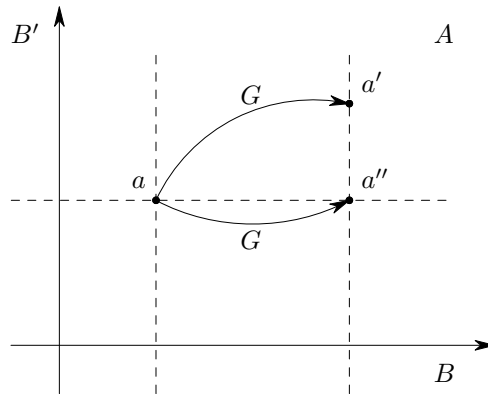
Természetesen általában sok olyan feladat van  $A$ -n, aminek a vetülete  $F$ , ilyen például az  $F$  kiterjesztése, de nem csak az. Tehát a 4.3. ábrán fölfelé mutató nyilak injektív megfeleltetések, a lefelé mutatók pedig szürjektívek.

Megjegyezzük még, hogy  $\hat{F}$ , vagyis egy olyan feladat, amelynek a vetülete  $F$ , mindig része  $F$  kiterjesztésének. Ugyanez a programok (programfüggvények) esetében nem igaz.

Megvizsgáljuk, milyen esetekben következtethetünk az  $A$  állapottéren fennálló megoldásból, ugyanerre a  $B$  állapottéren, és fordítva. Ahhoz, hogy a feltételeket megfogalmazzassuk, szükségünk lesz néhány definícióra.

#### 4.4. DEFINÍCIÓ: BŐVÍTETT IDENTITÁS

Legyen  $B$  altere  $A$ -nak,  $B'$  a  $B$  kiegészítő altere  $A$ -ra,  $G \subseteq A \times A$  reláció. A  $G$  bővített identitás  $B'$  felett, ha  $\forall (a, a') \in G : \exists a'' \in A$ , hogy  $(a, a'') \in G$  és  $pr_{B'}(a) = pr_{B'}(a'')$  és  $pr_B(a') = pr_B(a'')$ .

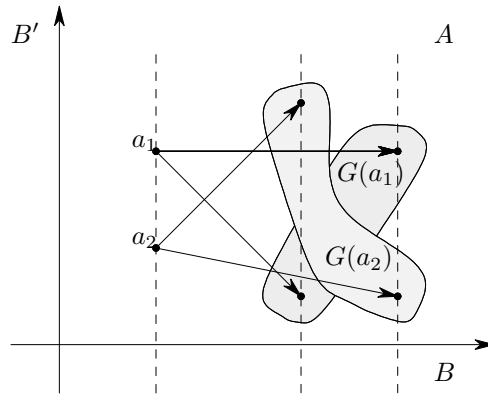


4.4. ábra. Bővített identitás

Ha egy feladat bővített identitás, az azt jelenti, hogy a feladat „megengedi”, hogy a kiegészítő altérbeli komponensek változatlanok maradjanak. Könnyű látni a definíciók alapján, hogy egy feladat kiterjesztése és egy program kiterjesztésének a programfüggvénye egyaránt bővített identitás.

#### 4.5. DEFINÍCIÓ: VETÍTÉSTARTÁS

Legyen  $B$  altere  $A$ -nak,  $G \subseteq A \times A$  feladat. A  $G$  vetítéstartó  $B$  felett, ha  $\forall a_1, a_2 \in \mathcal{D}_G : (pr_B(a_1) = pr_B(a_2)) \Rightarrow (pr_B(G(a_1)) = pr_B(G(a_2)))$ .



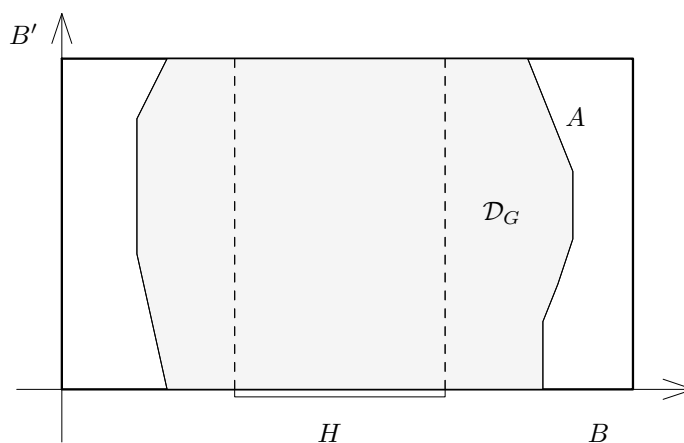
4.5. ábra. Vetítéstartás

A vetítéstartás nem jelenti azt, hogy a reláció nem függ a kiegészítő altér komponenseitől, hiszen mint a 4.5. ábra mutatja, két azonos vetületű pont képe lehet különböző, csak a vetületük azonos. Ebben az esetben is igaz, hogy egy feladat kiterjesztése

vetítéstartó (még a képek is megegyeznek), és a program kiterjesztése, így a kiterjesztés programfüggvénye is vetítéstartó.

#### 4.6. DEFINÍCIÓ: FÉLKITERJESZTÉS

Legyen  $B$  altere  $A$ -nak,  $G \subseteq A \times A$  feladat,  $H \subseteq B$ . Azt mondjuk, hogy a  $G$  félkiterjesztés  $H$  felett, ha  $pr_B^{-1}(H) \subseteq \mathcal{D}_G$ .



4.6. ábra. Félkiterjesztés

A félkiterjesztés szemléletes jelentése, hogy a kiegészítő altér felől nézve az értelmezési tartományban nincsenek „lyukak”. Most is igaz, hogy egy feladat kiterjesztése a feladat értelmezési tartománya fölött félkiterjesztés. Ugyancsak igaz, hogy a program kiterjesztésének programfüggvénye az eredeti programfüggvény értelmezési tartománya fölött félkiterjesztés.

Az imént bevezetett definíciók segítségével kimondhatók azok az állítások, amelyek a kiterjesztések és a projekció, valamint a megoldás közötti kapcsolatot vizsgáló tételcsoportot alkotják. A jelölések a 4.3. ábrának megfelelőek.

#### 4.1. TÉTEL: KITERJESZTÉSI TÉTELEK

Legyen  $B$  altere  $A$ -nak,  $B'$  a  $B$  kiegészítő altere  $A$ -ra,  $S$  program  $B$ -n,  $F \subseteq B \times B$  feladat,  $S'$ , illetve  $F'$   $S$ -nek, illetve  $F$ -nek a kiterjesztése  $A$ -ra. Legyen továbbá  $\hat{F} \subseteq A \times A$  olyan feladat, melyre  $pr_B(\hat{F}) = F$ ,  $\hat{S} \subseteq A \times A^{**}$  pedig olyan program, amely ekvivalens  $S$ -sel  $B$ -n. Ekkor az alábbi állítások teljesülnek:

- (1) ha  $S'$  megoldása  $F'$ -nek, akkor  $S$  megoldása  $F$ -nek;
- (2) ha  $S'$  megoldása  $\hat{F}$ -nek, akkor  $S$  megoldása  $F$ -nek;
- (3) ha  $\hat{S}$  megoldása  $F'$ -nek, akkor  $S$  megoldása  $F$ -nek;

- (4) a. ha  $\hat{S}$  megoldása  $\hat{F}$ -nek, és  $p(\hat{S})$  vetítéstartó  $B$  felett, akkor  $S$  megoldása  $F$ -nek;  
 b. ha  $\hat{S}$  megoldása  $\hat{F}$ -nek, és  $\hat{F}$  félkiterjesztés  $\mathcal{D}_F$  felett, akkor  $S$  megoldása  $F$ -nek;
- (5) ha  $S$  megoldása  $F$ -nek, akkor  $S'$  megoldása  $F'$ -nek;
- (6) ha  $S$  megoldása  $F$ -nek, és  $\hat{F}$  bővített identitás  $B'$  felett és vetítéstartó  $B$  felett, akkor  $S'$  megoldása  $\hat{F}$ -nek;
- (7) ha  $S$  megoldása  $F$ -nek, és  $p(\hat{S})$  félkiterjesztés  $\mathcal{D}_F$  felett, akkor  $\hat{S}$  megoldása  $F'$ -nek.

**Bizonyítás:** Mielőtt sorra bizonyítanánk az egyes tételeket, vegyük észre, hogy a (4) tételből következik az első három, hiszen  $S'$  ekvivalens  $S$ -sel  $B$ -n, és  $p(S')$  vetítéstartó, illetve  $pr_B(F') = F$ , és  $F'$  félkiterjesztés  $\mathcal{D}_F$ -en. Hasonló meggondolások alapján a (6) tételből is következik az (5) tétel, hiszen  $F'$  bővített identitás  $B'$  felett és vetítéstartó  $B$  felett. Elegendő tehát a (4), (6) és (7) tételeket bizonyítani.

Tekintsük először a (4) tétel bizonyítását: Legyen  $b \in \mathcal{D}_F$  tetszőleges. Ekkor

$$\begin{aligned} b \in \mathcal{D}_F &\Rightarrow \exists a \in \mathcal{D}_{\hat{F}} : pr_B(a) = b \\ &\stackrel{\text{megoldás}}{\Rightarrow} a \in \mathcal{D}_{p(\hat{S})} \\ &\stackrel{\hat{S} \text{ ekv. } S}{\Rightarrow} pr_B(a) \in \mathcal{D}_{p(S)}, \end{aligned}$$

tehát  $\mathcal{D}_F \subseteq \mathcal{D}_{p(S)}$ , s így a megoldás első kritériumának teljesülését bebizonyítottuk. Tekintsük most a második kritériumot; legyen  $b \in \mathcal{D}_F$  tetszőlegesen rögzített. Ekkor

$$\begin{aligned} p(S)(b) &= \bigcup_{a \in pr_B^{-1}(b) \cap \mathcal{D}_{p(\hat{S})}} pr_B(p(\hat{S})(a)), \\ F(b) &= \bigcup_{a \in pr_B^{-1}(b) \cap \mathcal{D}_{\hat{F}}} pr_B(\hat{F}(a)). \end{aligned}$$

Az a. esetben, azaz ha  $p(\hat{S})$  vetítéstartó, akkor  $\forall x, y \in pr_B^{-1}(b) \cap \mathcal{D}_{p(\hat{S})}$ -ra  $pr_B(p(\hat{S})(x)) = pr_B(p(\hat{S})(y))$ . Ekkor tetszőleges  $a \in pr_B^{-1}(b) \cap \mathcal{D}_{\hat{F}}$  esetén, mivel a megoldás definíciója miatt  $p(\hat{S}(a)) \subseteq \hat{F}(a)$ ,

$$p(S)(b) = pr_B(p(\hat{S})(a)) \subseteq pr_B(\hat{F}(a)) \subseteq F(b).$$

Tehát a megoldás második feltétele is teljesül.

A b. esetben, azaz ha  $\hat{F}$  félkiterjesztés, akkor  $pr_B^{-1}(b) \subseteq \mathcal{D}_{\hat{F}}$ , azaz  $a \in pr_B^{-1}(b) \cap \mathcal{D}_{\hat{F}} = pr_B^{-1}(b)$ , és a megoldás definíciója miatt

$$\forall a \in pr_B^{-1}(b) : p(\hat{S})(a) \subseteq \hat{F}(a),$$

tehát

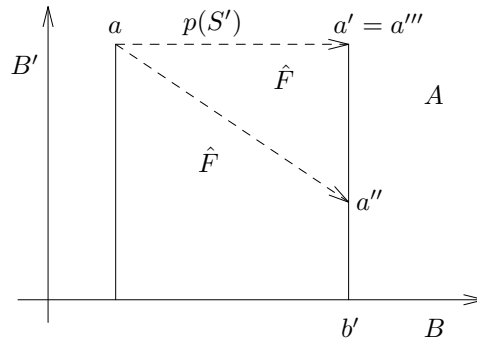
$$\begin{aligned} & \forall a \in pr_B^{-1}(b) : pr_B(p(\hat{S})(a)) \subseteq pr_B(\hat{F}(a)) \\ \Rightarrow & \bigcup_{a \in pr_B^{-1}(b)} pr_B(p(\hat{S})(a)) \subseteq \bigcup_{a \in pr_B^{-1}(b)} pr_B(\hat{F}(a)) \\ \Rightarrow & p(S)(b) \subseteq F(b), \end{aligned}$$

és ezzel ebben az esetben is beláttuk, hogy az  $S$  program megoldja az  $F$  feladatot. Nézzük most a (6) tétel bizonyítását.

1. Először belátjuk, hogy  $\mathcal{D}_{\hat{F}} \subseteq \mathcal{D}_{p(S')}$ .

Legyen  $a \in \mathcal{D}_{\hat{F}}$ . Ekkor  $pr_B(a) \in \mathcal{D}_F$ . Felhasználva, hogy  $S$  megoldása  $F$ -nek,  $pr_B(a) \in \mathcal{D}_{p(S)}$ . A program kiterjesztésének definíciójából következik, hogy ekkor  $a \in \mathcal{D}_{p(S')}$ .

2. Ezután megmutatjuk, hogy  $\forall a \in \mathcal{D}_{\hat{F}} : p(S')(a) \subseteq \hat{F}(a)$  is teljesül.



4.7. ábra.

A 4.7. ábrának megfelelően legyen  $a \in \mathcal{D}_{\hat{F}}$  és  $a' \in p(S')(a)$ . Ekkor – felhasználva, hogy  $S'$  az  $S$  kiterjesztése –  $a'$ -re fennáll az alábbi tulajdonság:

$$pr_{B'}(a') = pr_{B'}(a)$$

Legyen  $b' = pr_B(a')$ . Ekkor  $b' \in p(S)(pr_B(a))$ . Mivel  $S$  megoldja  $F$ -et, adódik, hogy  $b' \in F(pr_B(a))$ . Ekkor – mivel  $\hat{F}$  vetítéstartó  $B$  felett, és  $F$  az  $\hat{F}$  projekciója – adódik, hogy  $\exists a'' \in \hat{F}(a) : pr_B(a'') = b'$ . Felhasználva, hogy  $\hat{F}$  bővített identitás  $B'$  felett,  $\exists a''' \in \hat{F}(a)$ , amelyre

$$pr_{B'}(a''') = pr_{B'}(a) \text{ és } pr_B(a''') = b'.$$

Ekkor viszont  $a' = a'''$ , azaz  $a' \in \hat{F}(a)$ .



Most már csak a (7) állítás bizonyítása van hátra:

- (1) Legyen  $a \in \mathcal{D}_{F'}$ . Ekkor a feladat kiterjesztésének definíciója alapján  $pr_B(a) \in \mathcal{D}_F$ . Mivel  $p(\hat{S})$  félkiterjesztés  $\mathcal{D}_F$  felett,  $a \in \mathcal{D}_{p(\hat{S})}$ .
- (2) Legyen  $a \in \mathcal{D}_{F'}$ ,  $a' \in p(\hat{S})(a)$  és  $b' = pr_B(a')$ . Ekkor  $b' \in p(S)(pr_B(a))$ , hiszen  $p(S)$  a  $p(\hat{S})$  vetülete. Mivel  $S$  megoldja  $F$ -et, adódik, hogy  $b' \in F(pr_B(a))$ , de a feladat kiterjesztésének definíciója alapján  $\forall x \in pr_B^{-1}(b') : x \in F'(a)$ , így  $a' \in F'(a)$ . Tehát a megoldás második feltétele is teljesül.

Ezzel a (7) állítást is bebizonyítottuk.  $\square$

## 4.4. A feladat kiterjesztése és a specifikáció tétele

Emlékeztetünk a specifikáció tételének megfelelő formában megadott – állapotér, paramétertér, elő- és utófeltételek – feladatokra. Példaként felírtuk két szám összegét:

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$Q : (x = x' \wedge y = y')$$

$$R : (z = x' + y')$$

Mi lesz ennek a feladatnak a kiterjesztése egy  $A' = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{N} \times \mathbb{L}$  állapotérre? A válasz első pillantásra meglepő.

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{N} \times \mathbb{L}$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$Q : (x = x' \wedge y = y')$$

$$R : (z = x' + y')$$

Általánosságban is igaz, hogy a feladat kiterjesztésének specifikációja, a kibővített állapotértől eltekintve, megegyezik az eredeti feladat specifikációjával.

## 4.5. A paramétertér kiterjesztése

Természetesen nemcsak az állapotér, de a paramétertér is kiterjeszthető. A specifikáció tétele szempontjából két esetet különböztetünk meg.

Legyen  $B$  egy paramétertér és  $B'$  egy kiterjesztése. Ha egy feladat specifikálására  $B$  helyett  $B'$ -t használjuk úgy, hogy az elő- és utófeltételek nem függenek a csak a

$B'$ -ben szereplő paraméterváltozóktól, akkor a feladat nem változik, hiszen  $\forall b \in B : \forall b' \in pr_B^{-1}(b) : (\lceil Q_{b'} \rceil = \lceil Q_b \rceil \text{ és } \lceil R_{b'} \rceil = \lceil R_b \rceil)$ .

Ellenkező esetben a feladat is megváltozhat. Tegyük fel, hogy az új feladatra teljesülnek a következő feltételek:

$$\forall b \in B : \lceil Q_b \rceil = \bigcup_{b' \in pr_B^{-1}(b)} \lceil Q_{b'} \rceil$$

$$\forall b \in B : \forall b' \in pr_B^{-1}(b) : (\lceil R_{b'} \rceil \neq \emptyset \text{ és } \lceil R_{b'} \rceil \subseteq \lceil R_b \rceil),$$

akkor azt mondjuk, hogy az új feladat *finomítása* a réginek.

**4.3. állítás:** Ha egy  $F$  feladat finomítása egy  $G$  feladatnak, akkor  $F$  szigorúbb, mint  $G$ .

## 4.6. Példák

**4.1. példa:**  $B = \{1, 2, 3\}$ ,  $A = B \times \{1, 2, 3\}$ .  $F \subseteq B \times B$ .  $F = \{(1, 2), (1, 3)\}$ . Mi az  $F$  kiterjesztettje  $B$ -re?

**Megoldás:** A feladat kiterjesztésének definíciója alapján:

$$F = \left\{ \begin{array}{llll} ((1, 1), (2, 1)), & ((1, 1), (2, 2)), & ((1, 1), (2, 3)), & ((1, 2), (2, 1)), \\ ((1, 2), (2, 2)), & ((1, 2), (2, 3)), & ((1, 3), (2, 1)), & ((1, 3), (2, 2)), \\ ((1, 3), (2, 3)), & ((1, 1), (3, 1)), & ((1, 1), (3, 2)), & ((1, 1), (3, 3)), \\ ((1, 2), (3, 1)), & ((1, 2), (3, 2)), & ((1, 2), (3, 3)), & ((1, 3), (3, 1)), \\ ((1, 3), (3, 2)), & ((1, 3), (3, 3)) & & \end{array} \right\}$$

**4.2. példa:** Adott az  $\mathbb{L} \times \mathbb{L}$  állapottéren az  $F = \{(l, k), (m, n) \mid n = (l \wedge k)\}$  feladat, és az  $A' = \mathbb{L} \times \mathbb{L} \times V$  állapottéren ( $V = \{1, 2\}$ ) a következő program:

$$S = \left\{ \begin{array}{ll} (ii1, \langle ii1, ih2, hi2 \rangle), & (ii2, \langle ii2, hh1, ii1 \rangle), \\ (ii2, \langle ii2, ih2, hi1, hi2 \rangle), & (ih1, \langle ih1 \rangle), \\ (ih2, \langle ih2, ii1, hh1 \rangle), & (hi1, \langle hi1, hh2 \rangle), \\ (hi2, \langle hi2, hi1, ih1 \rangle), & (hi2, \langle hi2, hh1, hh2 \rangle), \\ (hh1, \langle hh1, ih1 \rangle), & (hh2, \langle hh2 \rangle) \end{array} \right\}$$

Megoldja-e  $S$  az  $F$   $A'$ -re való kiterjesztettjét?

**Megoldás:** Írjuk fel az  $F$   $A'$ -re való kiterjesztettjét:

$$F' = \left\{ \begin{array}{llll} (ii1, ii1), & (ii1, hi1), & (ii1, ii2), & (ii1, hi2), \\ (ii2, ii1), & (ii2, hi1), & (ii2, ii2), & (ii2, hi2), \\ (ih1, ih1), & (ih1, hh1), & (ih1, ih2), & (ih1, hh2), \\ (ih2, ih1), & (ih2, hh1), & (ih2, ih2), & (ih2, hh2), \\ (hi1, ih1), & (hi1, hh1), & (hi1, ih2), & (hi1, hh2), \\ (hi2, ih1), & (hi2, hh1), & (hi2, ih2), & (hi2, hh2), \\ (hh1, ih1), & (hh1, hh1), & (hh1, ih2), & (hh1, hh2), \\ (hh2, ih1), & (hh2, hh1), & (hh2, ih2), & (hh2, hh2) \end{array} \right\}$$

Az  $S$  program programfüggvénye:

$$p(S) = \left\{ \begin{array}{llll} (ii1, hi2), & (ii2, ii1), & (ii2, hi2), & (ih1, ih1), \\ (ih2, hh1), & (hi1, hh2), & (hi2, ih1), & (hi2, hh2), \\ (hh1, ih1), & (hh2, hh2) \end{array} \right\}$$

A megoldás definíciója két pontjának teljesülését kell belátnunk.  $\mathcal{D}_{F'} \subseteq \mathcal{D}_{p(S)}$  triviálisan teljesül, hiszen mindkét halmaz a teljes állapottér. Vizsgáljuk meg most, hogy  $\forall a \in \mathcal{D}_F : p(S)(a) \subseteq F'(a)$  teljesül-e!

$$\begin{aligned} p(S)(ii1) &= \{hi2\} \subseteq \{ii1, hi1, ii2, hi2\} = F(ii1) \\ p(S)(ii2) &= \{ii1, hi2\} \subseteq \{ii1, hi1, ii2, hi2\} = F(ii2) \\ p(S)(ih1) &= \{ih1\} \subseteq \{ih1, hh1, ih2, hh2\} = F(ih1) \\ p(S)(ih2) &= \{hh1\} \subseteq \{ih1, hh1, ih2, hh2\} = F(ih2) \\ p(S)(hi1) &= \{hh2\} \subseteq \{ih1, hh1, ih2, hh2\} = F(hi1) \\ p(S)(hi2) &= \{ih1, hh2\} \subseteq \{ih1, hh1, ih2, hh2\} = F(hi2) \\ p(S)(hh1) &= \{ih1\} \subseteq \{ih1, hh1, ih2, hh2\} = F(hh1) \\ p(S)(hh2) &= \{hh2\} \subseteq \{ih1, hh1, ih2, hh2\} = F(hh2) \end{aligned}$$

Tehát az  $S$  program megoldja az  $F$  feladat kiterjesztettjét.

**4.3. példa:** Igaz-e, hogy ha  $S \subseteq B \times B$ ,  $A$  altere  $B$ -nek, akkor

$$\mathcal{D}_{pr_A(p(S))} = pr_A(\mathcal{D}_{p(S)})?$$

**Megoldás:** Próbáljuk meg az állítást kétirányú tartalmazás belátásával bizonyítani.

$\mathcal{D}_{pr_A(p(S))} \subseteq pr_A(\mathcal{D}_{p(S)})$  : Legyen  $a \in \mathcal{D}_{pr_A(p(S))}$ . Ekkor

$$\begin{aligned} &\exists a' \in A : (a, a') \in pr_A(p(S)) \\ \Rightarrow &\exists (b, b') \in p(S) : pr_A((b, b')) = (a, a') \\ \Rightarrow &b \in \mathcal{D}_{p(S)} \Rightarrow pr_A(b) = a \in pr_A(\mathcal{D}_{p(S)}). \end{aligned}$$

$pr_A(\mathcal{D}_{p(S)}) \subseteq \mathcal{D}_{pr_A(p(S))}$  : Legyen  $a \in pr_A(\mathcal{D}_{p(S)})$ . Ekkor

$$\begin{aligned} & \exists b \in \mathcal{D}_{p(S)} : pr_A(b) = a \\ \Rightarrow & \exists b' \in B : (b, b') \in p(S) \\ \Rightarrow & (a, pr_A(b')) \in pr_A(p(S)) \\ \Rightarrow & a \in \mathcal{D}_{pr_A(p(S))}. \end{aligned}$$

és ezzel az állítást bebizonyítottuk.

## 4.7. Feladatok

- 4.1.  $B = \mathbb{N}, A = B \times \mathbb{N}, F \subseteq B \times B, F = \{(q, r) \mid r = q + 1\}$ . Mi az  $F$  kiterjesztettje  $A$ -ra?
- 4.2. Igaz-e, hogy ha  $S \subseteq A \times A^{**}$  program,  $B$  altere  $A$ -nak, akkor  $S$   $B$ -re történő projekciójának kiterjesztése  $A$ -ra azonos  $S$ -sel?
- 4.3. Bizonyítsuk be, hogy egy program kiterjesztettje valóban program!
- 4.4.  $A = A_1 \times A_2 \times \dots \times A_n$ . Mondjunk példát olyan programra, amelynek egyetlen valódi altérre vett projekciója sem program. ( $A_k = \mathbb{N}, k = 1, \dots, n$ ).
- 4.5. Legyen  $A$  altere  $B$ -nek,  $F \subseteq A \times A, F'' \subseteq B \times B, F'$  az  $F$  kiterjesztettje  $B$ -re. Igaz-e, hogy
  - a) ha  $F = pr_A(F'')$ , akkor  $F''$  az  $F$  kiterjesztettje?
  - b)  $F' = pr_A^{(-1)}(F)$ , illetve  $F' = pr_A^{-1}(F)$  ?
- 4.6. Legyen  $F \subseteq A \times A, F' \subseteq B \times B, F'' \subseteq C \times C, F''' \subseteq D \times D$ , ahol  $B = A \times A_1, C = A \times A_2, D = A \times A_1 \times A_2$ , és legyen  $F', F'', F'''$  az  $F$  kiterjesztése rendre  $B$ -re,  $C$ -re,  $D$ -re. Igaz-e, hogy  $F'''$  az  $F''$  kiterjesztése  $D$ -re? Adja meg az  $F'$  és az  $F''$  közötti kapcsolatot a projekció és a kiterjesztés fogalmának segítségével!
- 4.7.  $B$  és  $C$  altere  $A$ -nak.  $F \subseteq A \times A, F_1 \subseteq B \times B, F_2 \subseteq C \times C$ .  $F_1$  az  $F$  projekciója  $B$ -re.  $F$  az  $F_2$  kiterjesztése  $A$ -ra. Igaz-e, hogy az  $F_1$  feladat  $A$ -ra való kiterjesztettjének  $C$ -re vett projekciója megegyezik  $F_2$ -vel?

## 5. fejezet

# A megoldás fogalmának általánosítása

Ebben a fejezetben a megoldás fogalmát általánosítjuk. Eredetileg föltettük, hogy a program és a feladat állapottere „azonos”. Ezt a kikötést fogjuk két szempontból is gyengíteni.

### 5.1. A megoldás fogalmának kiterjesztése

Először a kiterjesztési tételek alapján általánosítjuk a megoldás fogalmát. Az állapotterek „azonossága” helyett csak azt követeljük meg, hogy egy közös állapottérre kiterjesztve a feladatot és a programot, teljesüljenek a megoldás feltételei.

#### 5.1. DEFINÍCIÓ: A MEGOLDÁS FOGALMÁNAK KITERJESZTÉSE

$A = \prod_{i \in I} A_i$  és  $B = \prod_{j \in J} A_j$ .  $F \subseteq A \times A$  feladat és  $S \subseteq B \times B^{**}$  program. Ha létezik  $C$  állapottér, amelynek  $A$  és  $B$  is altere, és  $S$  kiterjesztése  $C$ -re – eredeti értelemben – megoldása  $F$   $C$ -re való kiterjesztettjének, akkor azt mondjuk,  $S$  – kiterjesztett értelemben – megoldása  $F$ -nek.

Ez a definíció az eredeti definíció általánosítása, hiszen  $A = B = C$  esetén visszakapjuk azt.

A kiterjesztési tételekből, még hozzá az 1-ből és az 5-ből azonnal adódik, hogy a definícióban a „létezik” szót „minden”-re cserélhetnénk. Az is nyilvánvaló, hogy „közös többszörös” tulajdonságú állapottér, vagyis aminek  $A$  is és  $B$  is altere, mindig

létezik:  $\prod_{k \in I \cup J} A_k$ .

Ezért a definíciót úgy is megfogalmazhatjuk, hogy a  $C = \prod_{k \in I \cup J} A_k$  állapot-térre kiterjesztve a feladatot és a programot, teljesülnek a megoldás feltételei.

A kiterjesztett megoldás fogalmának ismeretében érdemes a kiterjesztési tételeket újra megvizsgálni. Az 1. és 5. tétel jelentőségét a definíciónál már tárgyaltuk. A 2., illetve 3. tétel azt jelenti, hogy ha egy program megoldása egy feladatnak, akkor akár a program, akár a feladat állapotterén is teljesülnek a megoldás feltételei. Ugyanez fordítva már csak bizonyos feltételek teljesülése esetén igaz (5., 6. tétel).

## 5.2. Megoldás ekvivalens állapotterén

Az ekvivalens direktoszorzat definíciójának megfelelően megmondjuk, hogy két állapotteret mikor tekintünk ekvivalensnek.

### 5.2. DEFINÍCIÓ: EKVIVALENS ÁLLAPOTTÉR

Azt mondjuk, hogy az  $A = \prod_{i \in I} A_i$  állapotter *ekvivalens* a  $B = \prod_{j \in J} B_j$  állapotterrel, ha létezik olyan  $f : I \rightarrow J$  kölcsönösen egyértelmű megfeleltetés (bijekció), hogy  $\forall i \in I : A_i = B_{f(i)}$ .

Azt, hogy  $A, B, f$ -re a fenti definíció teljesül,  $A \stackrel{f}{\sim} B$ -vel jelöljük.

Ha két állapotter ekvivalens, akkor nemcsak az értelmezési tartományok, hanem az állapotterek között is létezik kölcsönösen egyértelmű megfeleltetés:

$$\gamma_f : B \rightarrow A \text{ és } \forall b \in B : \forall i \in I : \gamma_f(b)(i) = b(f(i)).$$

### 5.3. DEFINÍCIÓ: MEGOLDÁS ÁTNEVEZÉSSSEL

Legyenek  $A \stackrel{f}{\sim} B$ ,  $F \subseteq A \times A$  feladat és  $S \subseteq B \times B^{**}$  program. Azt mondjuk, hogy  $S$  az  $f$  átnevezéssel megoldása  $F$ -nek, ha

1.  $\mathcal{D}_F \subseteq \mathcal{D}_{\gamma_f \circ p(S) \circ \gamma_f^{(-1)}}$ ,
2.  $\forall a \in \mathcal{D}_F : \gamma_f \circ p(S) \circ \gamma_f^{(-1)}(a) \subseteq F(a)$ .

A definícióban úgy is fogalmazhattunk volna, hogy  $\gamma_f \circ p(S) \circ \gamma_f^{(-1)}$  megoldása – az eredeti értelemben –  $F$ -nek. A  $\gamma_f \circ p(S) \circ \gamma_f^{(-1)}$  reláció jelentését úgy is megfogalmazhatjuk, hogy egy  $A$ -beli elemet „átnevezünk”  $B$ -belivé, alkalmazzuk rá a programfüggvényt, s a kapott elemeket „visszanevezzük”  $A$ -belivé.

Világos, hogy a definíció az eredeti általánosítása, hiszen ha  $A = B$  és  $f = id_A$ , akkor visszkapjuk az eredetit.

A kiterjesztés és az átnevezés segítségével megfogalmazhatjuk a megoldás általános definícióját:

**5.4. DEFINÍCIÓ: AZ ÁLTALÁNOSÍTOTT MEGOLDÁS**

Legyen  $F \subseteq A \times A$  feladat és  $S \subseteq B \times B^{**}$  program. Ha létezik olyan  $C$  és  $D$  állapotter, hogy  $C \stackrel{f}{\sim} D$ ,  $A$  altere  $C$ -nek,  $B$  altere  $D$ -nek, és  $S$  kiterjesztése  $D$ -re átnevezéssel megoldása  $F$   $C$ -re való kiterjesztettjének, akkor azt mondjuk,  $S$  – általános értelemben – megoldása  $F$ -nek.

**5.3. Reláció szerinti megoldás**

Nem csak átnevezéssel és nem csak ekvivalens állapottereket feleltethetünk meg egymásnak. A megoldást definiálhatjuk tetszőleges állapottereken definiált program és feladat esetén is, ha az állapottereket valahogy megfeleltetjük egymásnak.

**5.5. DEFINÍCIÓ: RELÁCIÓ SZERINTI MEGOLDÁS**

Legyen  $A$  és  $B$  tetszőleges állapotter,  $F \subseteq A \times A$ ,  $S \subseteq B \times B^{**}$  program és  $\gamma \subseteq B \times A$ . Azt mondjuk, hogy  $S$   $\gamma$  szerinti megoldása  $F$ -nek, ha

1.  $\mathcal{D}_F \subseteq \mathcal{D}_{\gamma \circ p(S) \circ \gamma^{(-1)}}$ , és
2.  $\forall a \in \mathcal{D}_F : \gamma \circ p(S) \circ \gamma^{(-1)}(a) \subseteq F(a)$ .

Emlékeztetünk arra, hogy a kompozíció és a szigorú kompozíció megegyezik, ha legalább az egyik reláció függvény. Ezért a megoldás átnevezéssel a reláció szerinti megoldás speciális esete. Így a következő tétel arra is alkalmazható.

**5.1. TÉTEL: RELÁCIÓ SZERINTI MEGOLDÁS TÉTELE**

Legyen  $F$  tetszőleges feladat, az állapottere  $A$ , egy paramétertere  $B$ , elő- és utófeltétele pedig  $Q_b$  és  $R_b$ . Legyen  $S \subseteq C \times C^{**}$  program és  $\gamma \subseteq C \times A$  tetszőleges olyan reláció, amelyre  $\mathcal{D}_F \subseteq \mathcal{R}_\gamma$ . Definiáljuk a következő függvényeket:

$$\begin{aligned} [Q_b^\gamma] &= [Q_b \circ \gamma], \\ [R_b^\gamma] &= [R_b \circ \gamma]. \end{aligned}$$

Ekkor ha  $\forall b \in B : Q_b^\gamma \Rightarrow lf(S, R_b^\gamma)$ , akkor az  $S$  program  $\gamma$  szerinti megoldja az  $F$  feladatot.

**Bizonyítás:** A  $\gamma$  szerinti megoldás definíciója két pontjának teljesülését kell belátnunk.

1.  $\mathcal{D}_F \subseteq \mathcal{D}_{\gamma \circ p(S) \circ \gamma^{(-1)}}$ .

Legyen  $a \in \mathcal{D}_F$ . Ekkor  $\exists b \in B : a \in [Q_b]$ . Mivel  $\mathcal{D}_F \subseteq \mathcal{R}_\gamma$ ,

$$\gamma^{(-1)}(a) \neq \emptyset \text{ és } \gamma^{(-1)}(a) \subseteq [Q_b^\gamma].$$

Felhasználva, hogy  $\lceil Q_b^\gamma \rceil \subseteq \lceil lf(S, R_b^\gamma) \rceil$ :

$$\gamma^{(-1)}(a) \subseteq \mathcal{D}_{p(S)} \text{ és } p(S)(\gamma^{(-1)}(a)) \subseteq \lceil R_b^\gamma \rceil.$$

Mivel  $\lceil R_b^\gamma \rceil = \lceil R_b \circ \gamma \rceil$ ,

$$p(S)(\gamma^{(-1)}(a)) \subseteq \mathcal{D}_\gamma,$$

tehát

$$a \in \mathcal{D}_{\gamma \circ p(S) \circ \gamma^{(-1)}}.$$

2.  $\forall a \in \mathcal{D}_F : \gamma \circ p(S) \circ \gamma^{(-1)}(a) \subseteq F(a)$ .

Legyen  $a \in \mathcal{D}_F$ . A bizonyítás első részében leírt lépéseket folytatva: mivel  $p(S)(\gamma^{(-1)}(a)) \subseteq \lceil R_b \circ \gamma \rceil$ ,

$$\gamma(p(S)(\gamma^{(-1)}(a))) \subseteq \lceil R_b \rceil \subseteq F(a).$$

□

A következő példában megmutatjuk, hogy ha  $Q_b^\gamma$ -t gyenge igazsághalmaz helyett igazsághalmazzal definiálnánk, akkor a tétel nem lenne igaz.

Legyen  $T = \{1, 2\}$ ,  $F$  állapottere  $A = T$ , és a paraméterter is legyen ugyanez:  $B = T$ . Legyen  $F$  specifikációja:

$$\begin{aligned} \lceil Q_1 \rceil &= \{1\}, & \lceil R_1 \rceil &= \{2\}; \\ \lceil Q_2 \rceil &= \emptyset, & \lceil R_2 \rceil &= \{1\}. \end{aligned}$$

Ekkor  $\mathcal{D}_F = \{1\}$  és  $F(1) = \{2\}$ . Legyen  $E = \{a, b\}$ ,  $C = E$  és

$$\gamma(a) = \gamma(b) = \{1, 2\}.$$

Tegyük fel, hogy  $S \subseteq C \times C^{**}$ , és rendelje  $S$  az állapottere minden pontjához az önmagából álló egy hosszúságú sorozatot. Ekkor

$$\lceil Q_1 \circ \gamma \rceil = \emptyset \text{ és } \lceil Q_2 \circ \gamma \rceil = \emptyset,$$

tehát

$$\begin{aligned} Q_1 \circ \gamma &\Rightarrow lf(S, R_1 \circ \gamma), \\ Q_2 \circ \gamma &\Rightarrow lf(S, R_2 \circ \gamma). \end{aligned}$$

Az viszont könnyen látható, hogy

$$\gamma \circ p(S) \circ \gamma^{(-1)}(1) = \{1, 2\} \not\subseteq F(1) = \{2\},$$

tehát  $S$  nem oldja meg  $F$ -et  $\gamma$  szerint.



## 6. fejezet

# Programkonstrukciók

Ebben a fejezetben azzal foglalkozunk, hogyan lehet meglévő programokból új programokat készíteni. Természetesen sokféleképpen konstruálhatunk meglévő programjainkból új programokat, de most csak háromféle konstrukciós műveletet fogunk megengedni: a szekvencia-, az elágazás- és a ciklusképzést. Később megmutatjuk, hogy ez a három konstrukciós művelet elegendő is a programozási feladatok megoldásához. Nemcsak definiáljuk ezeket a konstrukciókat, de megvizsgáljuk programfüggvényüket és leggyengébb előfeltételüket is.

### 6.1. Megengedett konstrukciók

Az első definíció arról szól, hogy egy programot közvetlenül egy másik után végezzünk el. A definícióban használni fogjuk a következő jelölést: legyen  $\alpha \in A^*$ ,  $\beta \in A^{**}$ ,

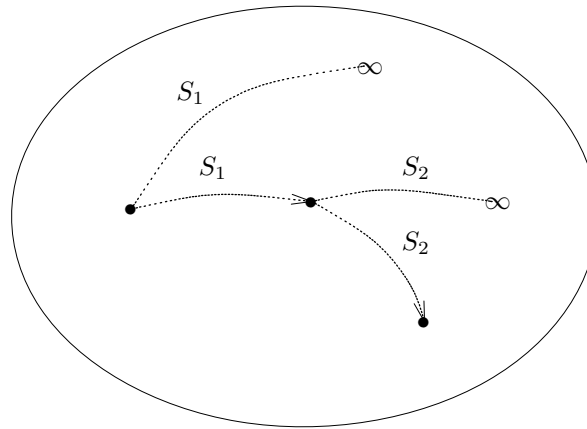
$$\chi_2(\alpha, \beta) ::= \text{red}(\text{kon}(\alpha, \beta)).$$

#### 6.1. DEFINÍCIÓ: SZEKVENCIA

Legyenek  $S_1, S_2 \subseteq A \times A^{**}$  programok. Az  $S \subseteq A \times A^{**}$  relációt az  $S_1$  és  $S_2$  *szekvenciájának* nevezzük, és  $(S_1; S_2)$ -vel jelöljük, ha  $\forall a \in A$ :

$$\begin{aligned} S(a) &= \{\alpha \in A^\infty \mid \alpha \in S_1(a)\} \cup \\ &\cup \{\chi_2(\alpha, \beta) \in A^{**} \mid \alpha \in S_1(a) \cap A^* \text{ és } \beta \in S_2(\tau(\alpha))\}. \end{aligned}$$

Vegyük észre, hogy ha két olyan program szekvenciáját képezzük, amelyek értékészlete csak véges sorozatokat tartalmaz, akkor a szekvencia is csak véges sorozatokat rendel az állapottér pontjaihoz.

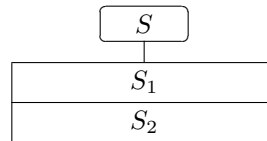


6.1. ábra. Szekvencia

Hasonlóan egyszerűen ellenőrizhető az is, hogy determinisztikus programok szekvenciája is determinisztikus reláció (függvény).

A programkonstrukciókat szerkezeti ábrákkal, úgynevezett *struktoqramokkal* szoktuk ábrázolni.

Legyenek  $S_1, S_2$  programok  $A$ -n. Ekkor az  $S = (S_1; S_2)$  szekvencia struktoqramja:



A második konstrukciós lehetőségünk az, hogy más-más meglevő programot hajtunk végre bizonyos feltételektől függően.

## 6.2. DEFINÍCIÓ: ELÁGAZÁS

Legyenek  $\pi_1, \dots, \pi_n : A \rightarrow \mathbb{L}$  feltételek,  $S_1, \dots, S_n$  programok  $A$ -n. Ekkor az  $IF \subseteq A \times A^{**}$  relációt az  $S_i$ -kből képzett,  $\pi_i$ -k által meghatározott elágazásnak nevezzük, és  $(\pi_1 : S_1, \dots, \pi_n : S_n)$ -nel jelöljük, ha  $\forall a \in A$ :

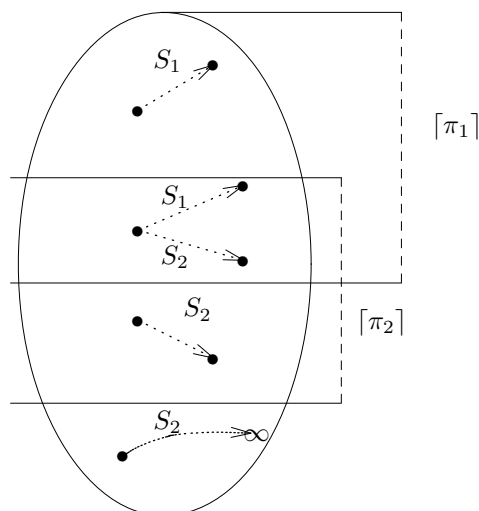
$$IF(a) = \bigcup_{i=1}^n w_i(a) \cup w_0(a),$$

ahol  $\forall i \in [1..n]$ :

$$w_i(a) = \begin{cases} S_i(a), & \text{ha } a \in [\pi_i]; \\ \emptyset & \text{különben;} \end{cases}$$

és

$$w_0(a) = \begin{cases} \{\langle a, a, a, \dots \rangle\}, & \text{ha } \forall i \in [1..n] : a \notin [\pi_i]; \\ \emptyset & \text{különben.} \end{cases}$$



6.2. ábra. Elágazás

Az elágazás definíciójában nem kötöttük ki, hogy a feltételek diszjunktak, tehát az állapottér egy pontjában több feltétel is igaz lehet. Ebben az esetben az elágazás az összes olyan sorozatot hozzárendeli ehhez a ponthoz, amelyet legalább egy olyan program hozzárendel, amelynek a feltételrészében ebben a pontban igaz. Ezért ha a feltételek nem diszjunktak, akkor a determinisztikus programokból képzett elágazás lehet nemdeterminisztikus is.

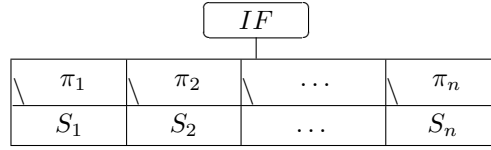
Az elágazás definícióját így is felírhattuk volna:

$$IF = \bigcup_{i=1}^n S_i|_{[\pi_i]} \cup \{\langle a, \dots \rangle \in A^{**} \mid a \in \bigcap_{i=1}^n [\neg\pi_i]\}.$$

Ha csak olyan programokból képzünk elágazást, amelyek csak véges sorozatokat rendelnek az állapottér minden pontjához, az elágazás akkor is rendelhet valamelyik ponthoz (azonos elemekből álló) végtelen sorozatot, ha a feltételek igazsághalmazai nem fedik le az egész állapottérrel.

Megjegyezzük, hogy a definíció szerint, ha egy pontban egyik feltétel sem teljesül, az elágazás „elszál”, ellentétben a legtöbb gyakorlatban használt programnyelvvvel.

Legyenek  $S_1, S_2, \dots, S_n$  programok,  $\pi_1, \pi_2, \dots, \pi_n$  feltételek  $A$ -n. Ekkor az  $IF = (\pi_1 : S_1, \pi_2 : S_2, \dots, \pi_n : S_n)$  elágazás struktogramja:



A harmadik konstrukciós lehetőségünk az, hogy egy meglevő programot egy feltételtől függően valahányszor egymás után végrehajtsunk. A ciklus definiálásához szükségünk van további két jelölésre: véges sok, illetve végtelen sok sorozat konkatenációjának redukáltjára.

Legyenek  $\alpha^1, \alpha^2, \dots, \alpha^{n-1} \in A^*$  és  $\alpha^n \in A^{**}$ ,

$$\chi_n(\alpha^1, \alpha^2, \dots, \alpha^n) ::= \text{red}(\text{kon}(\alpha^1, \alpha^2, \dots, \alpha^n)).$$

Legyenek  $\alpha^i \in A^*$  ( $i \in \mathbb{N}$ ),

$$\chi_\infty(\alpha^1, \alpha^2, \dots) ::= \text{red}(\text{kon}(\alpha^1, \alpha^2, \dots)).$$

### 6.3. DEFINÍCIÓ: CIKLUS

Legyen  $\pi : A \rightarrow \mathbb{L}$  feltétel és  $S_0$  program  $A$ -n. A  $DO \subseteq A \times A^{**}$  relációt az  $S_0$ -ból a  $\pi$  feltétellel képzett *ciklusnak* nevezzük, és  $(\pi, S_0)$ -lal jelöljük, ha

- $\forall a \notin \lceil \pi \rceil$ :

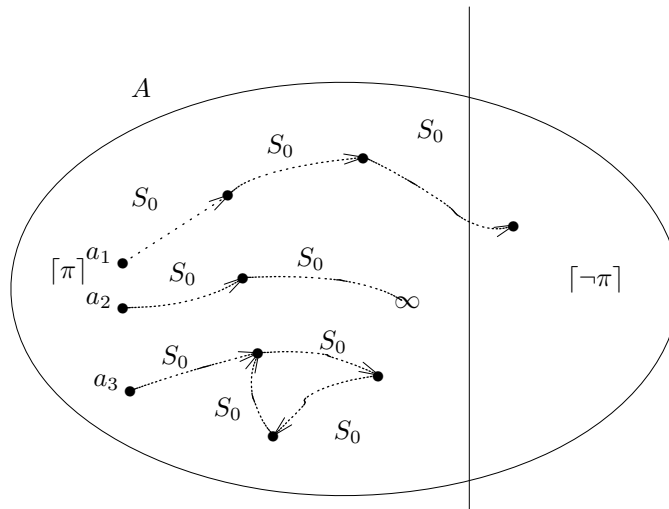
$$DO(a) = \{a\};$$

- $\forall a \in \lceil \pi \rceil$ :

$$\begin{aligned} DO(a) = & \{ \alpha \in A^{**} \mid \exists \alpha^1, \dots, \alpha^n \in A^{**} : \alpha = \chi_n(\alpha^1, \dots, \alpha^n) \text{ és} \\ & \alpha^1 \in S_0(a) \text{ és } \forall i \in [1..n-1] : (\alpha^i \in A^* \text{ és} \\ & \alpha^{i+1} \in S_0(\tau(\alpha^i)) \text{ és } \tau(\alpha^i) \in \lceil \pi \rceil) \text{ és } (\alpha^n \in A^\infty \text{ vagy} \\ & (\alpha^n \in A^* \text{ és } \tau(\alpha^n) \notin \lceil \pi \rceil)) \} \cup \\ & \cup \{ \alpha \in A^\infty \mid \forall i \in \mathbb{N} : \exists \alpha^i \in A^* : \alpha = \chi_\infty(\alpha^1, \alpha^2, \dots) \text{ és} \\ & \alpha^1 \in S_0(a) \text{ és } \forall i \in \mathbb{N} : (\alpha^i \in A^* \text{ és } \alpha^{i+1} \in S_0(\tau(\alpha^i)) \text{ és} \\ & \tau(\alpha^i) \in \lceil \pi \rceil) \}. \end{aligned}$$

Első ránézésre a definíció kissé bonyolult. Nézzük meg alaposabban! A ciklusmagot –  $S_0$  – véges sokszor alkalmazhatjuk egymás után, mert vagy olyan pontba jutunk, ahol a ciklusfeltétel –  $\pi$  – nem teljesül (a 6.3. ábrán az  $a_1$  pont), vagy az  $S_0$  végtelen sorozatot rendel egy ponthoz (a 6.3. ábrán az  $a_2$  pont). Ezeket az eseteket tartalmazza a definícióban az első halmaz. A harmadik lehetőség (a 6.3. ábrán az  $a_3$  pont) az, hogy a fentiek egyike sem teljesül, azaz az  $S_0$  végtelen sokszor alkalmazható egymás után. Ezt az esetet tartalmazza a definícióban a második halmaz.

Ezek alapján azt is megállapíthatjuk, hogy a determinisztikus programból képzett ciklus is determinisztikus lesz, ezzel ellentétben, ha egy, csak véges sorozatokat

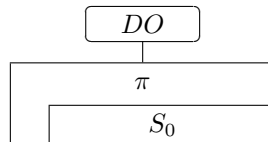


6.3. ábra. Ciklus

rendelő programot foglalunk ciklusba, akkor a ciklus értékészlete még tartalmazhat végtelen sorozatot (lásd harmadik lehetőség), azaz kétféle oka is lehet a „végtelen ciklusnak”.

Megjegyezzük még, hogy a különböző programnyelvekben többféle ciklus is létezik; amit most definiáltunk, az az úgynevezett „while típusú” ciklus.

Legyen  $S_0$  program,  $\pi$  feltétel  $A$ -n. Ekkor a  $DO = (\pi, S_0)$  ciklus struktogramja:



A fentiekben leírt konstrukciókat programokra lehet alkalmazni. De vajon programokat hoznak-e létre? Az alábbi tétel kimondja, hogy az előzőekben definiált három konstrukciós művelet meglévő programokból valóban programokat hoz létre.

**6.1. TÉTEL:** A SZEKVENCIA, AZ ELÁGAZÁS ÉS A CIKLUS PROGRAM.

Legyen  $A$  tetszőleges állapotter,  $S_0, S_1, S_2, \dots, S_n \subseteq A \times A^{**}$  programok, valamint  $\pi, \pi_1, \pi_2, \dots, \pi_n : A \rightarrow \mathbb{L}$  feltételek  $A$ -n. Ekkor

1.  $S = (S_1; S_2)$ ,
2.  $IF = (\pi_1 : S_1, \pi_2 : S_2, \dots, \pi_n : S_n)$ ,
3.  $DO = (\pi, S_0)$

programok  $A$ -n.

**Bizonyítás:** Mindhárom konstrukció definíciójában explicit szerepel, hogy része  $A \times A^{**}$ -nak. A továbbiakban a három kritérium teljesülését vizsgáljuk meg mindhárom konstrukció esetén.

1. Legyen  $a \in A$  tetszőleges, ekkor  $S_1(a)$ -nak van legalább egy eleme. Ha ez a sorozat végtelen, a definíció első halmaza nem üres, ha véges, létezik végpontja, amihez  $S_2$  hozzárendel legalább egy sorozatot, és ezért a definíció második halmaza nem üres, tehát  $S(a)$  nem üres.

Legyen  $\alpha \in S(a)$ . Ekkor két eset lehetséges:

- Ha  $\alpha \in S_1(a)$ , ebben az esetben  $\alpha_1 = a$  és  $\alpha = red(\alpha)$  triviálisan teljesül, hiszen  $S_1$  program.
- Ha  $\alpha = \chi_2(\alpha^1, \alpha^2)$  úgy, hogy  $\alpha^1 \in S_1(a)$  és  $\alpha^2 \in S_2(\tau(\alpha^1))$ . Ekkor a  $\chi_2$  definíciója miatt  $\alpha$  redukált. Másrészt  $\alpha_1 = \alpha_1^1$ , tehát  $\alpha_1 = a$ .

2. Legyen  $a \in A$  tetszőleges, ekkor vagy valamelyik  $\pi_i$  feltétel igaz  $a$ -ra, és így  $w_i(a)$  nem üres, vagy  $w_0(a)$  nem üres, tehát  $IF(a)$  nem üres.

Legyen  $\alpha \in IF(a)$ . Ekkor

$$\alpha \in \bigcup_{i=0}^n w_i(a).$$

- Tegyük fel, hogy  $\alpha \in w_0(a)$ . Ekkor  $\alpha = \langle a, a, \dots \rangle$ , tehát teljesíti a program definíciójának kritériumait.
- Tegyük fel, hogy  $\exists i \in [1..n] : \alpha \in w_i(a)$ . Ekkor  $\alpha \in S_i(a)$ , így mivel  $S_i$  program,  $\alpha$  teljesíti a definícióban megkívánt tulajdonságokat.

3. Legyen  $a \in A$  tetszőleges, ekkor vagy minden  $n \in \mathbb{N}$ -re teljesül a definíció második halmazának a feltétele, vagy ha nem, akkor teljesül az elsőé, azaz a két halmaz közül legalább az egyik nem üres, tehát  $DO(a)$  nem üres.

Legyen  $\alpha \in DO(a)$ .

- Tegyük fel, hogy  $a \notin \lceil \pi \rceil$ . Ekkor  $\alpha = \langle a \rangle$ , így az előírt tulajdonságok triviálisan teljesülnek.
- Tegyük fel, hogy  $a \in \lceil \pi \rceil$ . Ekkor három eset lehetséges:
  - (a)  $\alpha \in A^*$ :  
Ekkor  $\exists n \in \mathbb{N} : \alpha = \chi_n(\alpha^1, \dots, \alpha^n)$ , így  $\chi_n$  definíciója miatt  $\alpha$  redukált. Másrészt felhasználva, hogy  $\alpha^1 \in S_0(a)$  és  $\alpha_1 = \alpha_1^1$ ,  $\alpha_1 = a$  is teljesül.
  - (b)  $\exists n \in \mathbb{N} : \alpha = \chi_n(\alpha^1, \dots, \alpha^n)$  és  $\forall i \in [1..n-1] : \alpha^i \in A^*$  és  $\alpha_n \in A^\infty$ : Ekkor a kritériumok teljesülése az előző ponttal analóg módon ellenőrizhető.

(c)  $\alpha = \chi_\infty(\alpha^1, \alpha^2, \dots)$ :

Ekkor  $\alpha$  a  $\chi_\infty$  definíciója alapján redukált sorozat, és  $\alpha_1 = \alpha_1^1 = a$  is teljesül.

□

## 6.2. A programkonstrukciók programfüggvénye

Miután beláttuk, hogy meglevő programokból a programkonstrukciók segítségével új programokat készíthetünk, vizsgáljuk meg, milyen kapcsolat van a konstruált programok programfüggvénye és az eredeti programok programfüggvénye között.

A szekvencia a legegyszerűbb programkonstrukció, ennek megfelelően a programfüggvénye is egyszerűen felírható a két komponensprogram programfüggvényének segítségével. Mivel a szekvencia két program egymás utáni elvégzését jelenti, várható, hogy a programfüggvénye a két komponensprogram programfüggvényének kompozíciója. Azonban, mint látni fogjuk, kompozíció helyett szigorú kompozíciót kell alkalmazni.

### 6.2. TÉTEL: A SZEKVENCIA PROGRAMFÜGGVÉNYE

Legyen  $A$  tetszőleges állapottér,  $S_1, S_2$  programok  $A$ -n,  $S = (S_1; S_2)$ . Ekkor

$$p(S) = p(S_2) \odot p(S_1).$$

**Bizonyítás:** Legyen  $a \in A$  tetszőleges. Ekkor

$$\begin{aligned} a \in \mathcal{D}_{p(S)} &\iff S(a) \subseteq A^* \iff \\ S_1(a) \subseteq A^* \text{ és } \forall \alpha \in S_1(a) : S_2(\tau(\alpha)) \subseteq A^* &\iff \\ a \in \mathcal{D}_{p(S_1)} \text{ és } p(S_1)(a) \subseteq \mathcal{D}_{p(S_2)} &\iff \\ a \in \mathcal{D}_{p(S_2) \odot p(S_1)}, & \end{aligned}$$

tehát:

$$\mathcal{D}_{p(S)} = \mathcal{D}_{p(S_2) \odot p(S_1)}.$$

Legyen  $a \in \mathcal{D}_{p(S)}$ . Ekkor

$$\begin{aligned} (a, a') \in p(S_2) \odot p(S_1) &\iff \\ \exists b \in A : (a, b) \in p(S_1) \text{ és } (b, a') \in p(S_2) &\iff \\ \exists \alpha \in S_1(a) : \exists \beta \in S_2(b) : \tau(\alpha) = b \text{ és } \tau(\beta) = a' &\iff \\ (a, a') \in p(S). & \end{aligned}$$

□

Vegyük észre, hogy a nem szigorú kompozíció értelmezési tartományában olyan pont is lehet, amelyhez a szekvencia rendel végtelen sorozatot is. Nézzünk erre egy egyszerű példát: Legyen  $A = \{1, 2\}$ ,

$$\begin{aligned} S_1 &= \{(1, \langle 1 \rangle), (1, \langle 1, 2 \rangle), (2, \langle 2 \rangle)\}, \\ S_2 &= \{(1, \langle 1, 2 \rangle), (2, \langle 2, 2, \dots \rangle)\}. \end{aligned}$$

Ekkor  $1 \in \mathcal{D}_{p(S_2) \circ p(S_1)}$ , de  $\langle 1, 2, 2, 2, \dots \rangle \in S(1)$ .

Mivel az elágazást több programból képezzük, a programfüggvényét is csak kissé körülményesebben tudjuk megfogalmazni. Hiszen az, hogy egy ponthoz az elágazás rendel-e végtelen sorozatot, attól is függ, mely feltételek igazak az adott pontban. Sőt, ha egy pontban egyetlen feltétel sem igaz, akkor a komponensprogramok programfüggvényétől függetlenül abban a pontban az elágazás programfüggvénye nem lesz értelmezve. Az elágazás programfüggvényét adja meg a következő tétel.

### 6.3. TÉTEL: AZ ELÁGAZÁS PROGRAMFÜGGVÉNYE

Legyen  $A$  tetszőleges állapotter,  $S_1, S_2, \dots, S_n \subseteq A \times A^{**}$  programok, valamint  $\pi_1, \pi_2, \dots, \pi_n : A \rightarrow \mathbb{L}$  feltételek  $A$ -n,  $IF = (\pi_1 : S_1, \dots, \pi_n : S_n)$ . Ekkor

$$\bullet \mathcal{D}_{p(IF)} = \{a \in A \mid a \in \bigcup_{i=1}^n [\pi_i] \text{ és } \forall i \in [1..n] : a \in [\pi_i] \Rightarrow a \in \mathcal{D}_{p(S_i)}\}.$$

$$\bullet \forall a \in \mathcal{D}_{p(IF)}:$$

$$p(IF)(a) = \bigcup_{i=1}^n p(S_i)|_{[\pi_i]}(a).$$

**Bizonyítás:** Legyen  $a \in A$  tetszőleges. Ekkor

$$\begin{aligned} a \in \mathcal{D}_{p(IF)} &\iff IF(a) \subseteq A^* \iff \\ &\exists i \in [1..n] : a \in [\pi_i] \text{ és } \bigcup_{i=1}^n w_i(a) \subseteq A^* \iff \\ &\exists i \in [1..n] : a \in [\pi_i] \text{ és } \forall i \in [1..n] : a \in [\pi_i] \Rightarrow a \in \mathcal{D}_{p(S_i)}. \end{aligned}$$

Legyen továbbá  $a \in \mathcal{D}_{p(IF)}$ . Ekkor

$$p(IF)(a) = \bigcup_{i=1}^n \{\tau(\alpha) \mid \alpha \in w_i(a)\} = \bigcup_{i=1}^n p(S_i)|_{[\pi_i]}(a).$$

□

Természetesen, ha az elágazás-feltételek lefedik az egész állapotteret, akkor az a feltétel, hogy valamelyik  $\pi_i$ -nek igaznak kell lennie, nyilvánvalóan teljesül.

A ciklus programfüggvényét a feltételre vonatkozó lezárt segítségével fejezzük ki.

### 6.4. TÉTEL: A CIKLUS PROGRAMFÜGGVÉNYE

Legyen  $A$  tetszőleges állapotter,  $S$  program,  $\pi$  feltétel  $A$ -n,  $DO = (\pi, S)$ . Ekkor

$$p(DO) = \overline{p(S)|\pi}.$$



**Bizonyítás:** Legyen  $a \in A$  tetszőleges. Ekkor

$$\begin{array}{c}
 a \in \mathcal{D}_{p(DO)} \iff DO(a) \subseteq A^* \\
 \Downarrow \\
 DO(a) = \begin{cases} \{ \{a\} \}, & \text{ha } a \notin [\pi]; \\ \{ \alpha \in A^* \mid \exists n \in \mathbb{N} : \alpha = \chi_n(\alpha^1, \dots, \alpha^n) \text{ és} \\ \alpha^1 \in S(a) \text{ és } \forall i \in [1..n-1] : \\ (\alpha^{i+1} \in S(\tau(\alpha^i)) \text{ és } \tau(\alpha^i) \in [\pi]) \text{ és} \\ \tau(\alpha^n) \notin [\pi] \}, & \text{ha } a \in [\pi]. \end{cases} \\
 \Downarrow \\
 a \notin [\pi] \text{ vagy } (\exists \beta \in A^\infty : \beta_1 = a \text{ és } \forall i \in \mathbb{N} : (\beta_{i+1} \in p(S)(\beta_i) \text{ és } \beta_i \in [\pi])) \\
 \text{és } (\exists \beta \in A^* : (\beta_1 = a \text{ és } \forall i \in [1..|\beta|-1] : \\
 (\beta_{i+1} \in p(S)(\beta_i) \text{ és } \beta_i \in [\pi]) \text{ és } \tau(\beta) \in [\pi] \text{ és } \tau(\beta) \notin \mathcal{D}_{p(S)}). \\
 \Downarrow \\
 a \in \overline{\mathcal{D}_{p(S)|\pi}},
 \end{array}$$

tehát  $\mathcal{D}_{p(DO)} = \overline{\mathcal{D}_{p(S)|\pi}}$ . Másrészt legyen  $a \in \mathcal{D}_{p(DO)}$ .

- Ha  $a \notin [\pi]$ , akkor

$$p(DO)(a) = \{a\} = \overline{p(S)|\pi}(a).$$

- Ha  $a \in [\pi]$ , akkor  $p(DO)(a) =$

$$\begin{aligned}
 &= \{ \tau(\alpha) \mid \alpha \in A^* \text{ és } \exists n \in \mathbb{N} : \alpha = \chi_n(\alpha^1, \dots, \alpha^n) \text{ és } \alpha^1 \in S(a) \text{ és} \\
 &\quad \forall i \in [1..n-1] : (\alpha^{i+1} \in S(\tau(\alpha^i)) \text{ és } \tau(\alpha^i) \in [\pi]) \text{ és } \tau(\alpha^n) \notin [\pi] \} \\
 &= \{ \tau(\beta) \mid \beta \in A^* \text{ és } \beta_1 = a \text{ és } \forall i \in [1..|\beta|-1] : (\beta_{i+1} \in p(S)(\beta_i) \text{ és} \\
 &\quad \beta_i \in [\pi]) \text{ és } \tau(\beta) \notin [\pi] \} = \\
 &= \{ b \in A \mid \exists k \in \mathbb{N} : b \in (p(S)|\pi)^k(a) \text{ és } b \notin [\pi] \} \\
 &= \overline{p(S)|\pi}(a).
 \end{aligned}$$

□

### 6.3. Feladatok

**6.1.**  $A = \{1, 2, 3, 4, 5, 6\}$ .  $[\pi_1] = \{1, 2, 3, 4\}$ .  $[\pi_2] = \{1, 3, 4, 5\}$ .

$$\begin{array}{l}
 S_1 = \{ \begin{array}{cccc} 1 \rightarrow 14 & 1 \rightarrow 12 \dots & 2 \rightarrow 2132 & 3 \rightarrow 36 \\ 4 \rightarrow 463 & 4 \rightarrow 451 & 5 \rightarrow 563 & 6 \rightarrow 612 \end{array} \}, \\
 S_2 = \{ \begin{array}{cccc} 1 \rightarrow 134 & 1 \rightarrow 121 & 2 \rightarrow 2132 \dots & 3 \rightarrow 36 \\ 4 \rightarrow 463 & 4 \rightarrow 451 \dots & 5 \rightarrow 5632 & 6 \rightarrow 61 \dots \end{array} \}.
 \end{array}$$

Adja meg az  $(S_1; S_2)$ ,  $IF(\pi_1 : S_1, \pi_2 : S_2)$ ,  $DO(\pi_1, S_1)$  programokat és a programfüggvényeiket!

**6.2.** Legyen  $A = \{1, 2, 3, 4, 5, 6\}$ ,  $[\pi_1] = \{1, 2, 3, 4\}$ ,  $[\pi_2] = \{2, 3, 4\}$ ,  $[\pi_3] = \{1, 4, 6\}$  és

$$\begin{aligned} S_1 &= \{ \begin{array}{lll} 1 \rightarrow 12 \dots & 2 \rightarrow 23 & 3 \rightarrow 3456 \\ 4 \rightarrow 463 & 5 \rightarrow 53 & 6 \rightarrow 62 \end{array} \}, \\ S_2 &= \{ \begin{array}{lll} 1 \rightarrow 12 & 2 \rightarrow 24 & 3 \rightarrow 3 \dots \\ 4 \rightarrow 43 & 5 \rightarrow 5 & 6 \rightarrow 61 \end{array} \}, \\ S_3 &= \{ \begin{array}{lll} 1 \rightarrow 12 & 2 \rightarrow 2 \dots & 3 \rightarrow 31 \\ 4 \rightarrow 432 & 5 \rightarrow 5 \dots & 6 \rightarrow 63 \dots \end{array} \}. \end{aligned}$$

Mi lesz  $IF(\pi_1 : S_1, \pi_2 : S_2, \pi_3 : S_3)$ ,  $\mathcal{D}_{p(IF)}$ ,  $p(IF)$ ?

**6.3.** Legyen  $S_1$  és  $S_2$  egy-egy program az  $A$  állapottéren. Igaz-e, hogy  $S_2 \circ \tau \circ S_1$  megegyezik  $(S_1; S_2)$ -vel?

**6.4.**  $S = (S_1; S_2)$ . Igaz-e, hogy

- $\mathcal{D}_{p(S)} = [lf(S_1, \mathcal{P}(\mathcal{D}_{p(S_2)}))]$ ?
- tetszőleges  $R$  utófeltételre:  $lf((S_1; S_2), R) = lf(S_1, lf(S_2, R))$ ?

**6.5.** Van-e olyan program, amely felírható szekvenciaként is, elágazásként is, és felírható ciklusként is?

**6.6.** Igaz-e, hogy minden program felírható szekvenciaként is, elágazásként is, és felírható ciklusként is?

**6.7.**  $IF = (\pi_1 : S_1, \dots, \pi_n : S_n)$ . Igaz-e, hogy  $\mathcal{D}_{p(IF)} = \bigcup_{k=1}^n ([\pi_k] \cap \mathcal{D}_{p(S_k)})$ ?

**6.8.** Legyen  $S_1, S_2, \dots, S_n$  program  $A$ -n.  $IF = (\pi_1 : S_1, \dots, \pi_n : S_n)$ .  $S = S_1 \cup S_2 \cup \dots \cup S_n$ . Keressünk olyan  $\pi_k$  feltételeket és  $S_k$  programokat, hogy  $\mathcal{D}_{p(IF)} = A$  és  $\mathcal{D}_{p(S)} = \emptyset$ !

**6.9.**  $IF = (\pi_1 : S_1, \dots, \pi_n : S_n)$ .  $S = S_1 \cup S_2 \cup \dots \cup S_n$ . Igaz-e, hogy  $p(IF)$  része  $p(S)$ -nek?

**6.10.** Igaz-e? Ha  $IF = (\pi_1 : S_1, \pi_2 : S_2)$ , akkor  $\mathcal{D}_{p(IF)} = ([\pi_1] \cap [\pi_2] \cap \mathcal{D}_{p(S_1)} \cap \mathcal{D}_{p(S_2)}) \cup (\mathcal{D}_{p(S_1)} \cap ([\pi_1] \setminus [\pi_2])) \cup (\mathcal{D}_{p(S_2)} \cap ([\pi_2] \setminus [\pi_1]))$ ?

**6.11.**  $A = \{1, 2, 3, 4\}$ .

$IF$	
$i = 1$	$i \leq 2$
$i := 2i$	SKIP

Milyen sorozatokat rendel  $S_1, S_2, IF$  az állapottér egyes pontjaihoz?

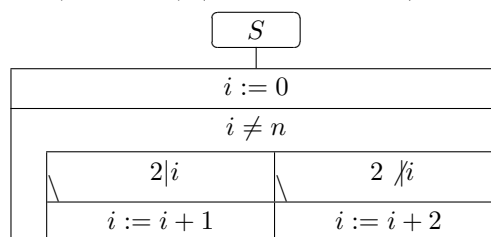
- 6.12.**  $S = (S_1; S_2)$ .  $S_1$  megoldja  $F_1$ -et, és  $S_2$  megoldja  $F_2$ -t. Megoldja-e  $S$  az
- $F = F_2 \circ F_1$  feladatot?
  - $F = F_2 \odot F_1$  feladatot?
- 6.13.**  $S = (S_1; S_2)$ .  $S$  megoldása az  $(F_2 \odot F_1)$  feladatnak. Megoldja-e  $S_1$  az  $F_1$ -et, illetve  $S_2$  az  $F_2$ -t?
- 6.14.**  $IF = (\pi_1 : S_1, \dots, \pi_n : S_n)$ .  $F \subseteq A \times A$  feladat.  $\forall k \in [1..n]$ :  $S_k$  megoldja az  $F|_{\lceil \pi_k \rceil}$  feladatot.
- $IF$  megoldja-e az  $F$  feladatot?
  - $IF$  megoldja-e az  $F$  feladatot, ha  $\pi_1 \vee \pi_2 \vee \dots \vee \pi_n = \text{Igaz}$ ?
  - $IF$  megoldja-e az  $F$  feladatot, ha  $\mathcal{D}_F \subseteq \lceil \pi_1 \vee \pi_2 \vee \dots \vee \pi_n \rceil$ ?
- 6.15.**  $IF = (\pi_1 : S_1, \dots, \pi_n : S_n)$ .  $F \subseteq A \times A$  feladat.  $IF$  megoldja az  $F$  feladatot. Igaz-e, hogy  $\forall k \in [1..n]$ :  $S_k$  megoldja az  $F|_{\lceil \pi_k \rceil}$  feladatot?
- 6.16.**  $IF = (\pi_1 : S_1, \dots, \pi_n : S_n)$ .  $F_1, \dots, F_k \subseteq A \times A$  feladat.  $\forall k \in [1..n]$ :  $S_k$  megoldja az  $F_k$  feladatot. Megoldja-e  $IF$  az  $F = F_1 \cup \dots \cup F_n$  feladatot?
- 6.17.**  $IF = (\pi_1 : S_1, \dots, \pi_n : S_n)$ .  $F \subseteq A \times A$  feladat.  $IF$  megoldja az  $F$  feladatot, és  $\lceil \pi_1 \rceil \cup \dots \cup \lceil \pi_n \rceil \subseteq \mathcal{D}_F$ . Igaz-e, hogy  $\forall k \in [1..n]$ :  $S_k$  megoldja az  $F|_{\lceil \pi_k \rceil}$  feladatot?
- 6.18.**  $IF = (\pi_1 : S_1, \dots, \pi_n : S_n)$ .  $F_1, \dots, F_n \subseteq A \times A$  feladat.  $\forall k \in [1..n]$ :  $\mathcal{D}_{F_k} \subseteq \lceil \pi_k \rceil$ , és  $S_k$  megoldja az  $F_k$  feladatot.  $F = F_1 \cup \dots \cup F_n$ . Megoldja-e  $IF$  az  $F$  feladatot?
- 6.19.** Igaz-e, hogy  $IF_1 = (\pi_1 : S_1, \pi_2 : S_2)$  és  $IF_2 = (\pi_1 : S_1, \pi_1 \wedge \pi_2 : S_1 \cup S_2, \pi_2 : S_2)$
- egyenlő?
  - ekvivalens?
- 6.20.** Legyen  $IF_{34} = (\pi_3 : S_3, \pi_4 : S_4)$ ,  $IF_1 = (\pi_1 : S_1, \pi_2 : IF_{34})$ ,  $IF_2 = (\pi_1 : S_1, \pi_2 \wedge \pi_3 : S_3, \pi_2 \wedge \pi_4 : S_4)$ . Igaz-e, hogy  $IF_1$  és  $IF_2$
- egyenlő?
  - ekvivalens?
- 6.21.**  $F \subseteq A \times A$  feladat.  $S_0$  program  $A$ -n.  $S_0$  megoldja  $F$ -et. Megoldja-e a  $DO(\pi, S_0)$  program az  $F$  feladat  $\pi$ -re vonatkozó lezártját?
- 6.22.** Legyen  $DO = (\pi, S)$ . Igaz-e, hogy

a)  $p(DO) \subseteq p(S)$ ?

b)  $p(S) \subseteq p(DO)$ ?

6.23.  $A = \mathbb{N}_0 \times \mathbb{N}_0$   
 $\quad \quad \quad i \quad \quad n$

$S = (i := 0; DO(i \neq n, IF(2|i : i := i + 1, 2 \nmid i : i := i + 2)))$



Milyen sorozatokat rendel  $S$  a (2, 4), illetve a (3, 7) ponthoz?

## 7. fejezet

# Levezetési szabályok

Ebben a fejezetben megvizsgáljuk a programkonstrukciók és a specifikáció kapcsolatát.

### 7.1. A szekvencia levezetési szabálya

Először azt fogjuk megvizsgálni, hogy a szekvencia adott utófeltételhez tartozó leggyengébb előfeltétele milyen kapcsolatban van az őt alkotó programok leggyengébb előfeltételével.

#### 7.1. TÉTEL: A SZEKVENCIA LEVEZETÉSI SZABÁLYA

Legyen  $S = (S_1; S_2)$ , és adott  $Q, R$  és  $Q'$  állítás  $A$ -n. Ha

$$(1) Q \Rightarrow lf(S_1, Q') \text{ és}$$

$$(2) Q' \Rightarrow lf(S_2, R),$$

akkor  $Q \Rightarrow lf(S, R)$ .

**Bizonyítás:** Legyen  $q \in [Q]$ . Ekkor (1) miatt  $q \in \mathcal{D}_{p(S_1)}$  és  $p(S_1)(q) \subseteq [Q']$ . Mivel (2) miatt  $[Q'] \subseteq \mathcal{D}_{p(S_2)}$ :  $q \in \mathcal{D}_{p(S_2) \circ p(S_1)} = \mathcal{D}_{p(S)}$ .

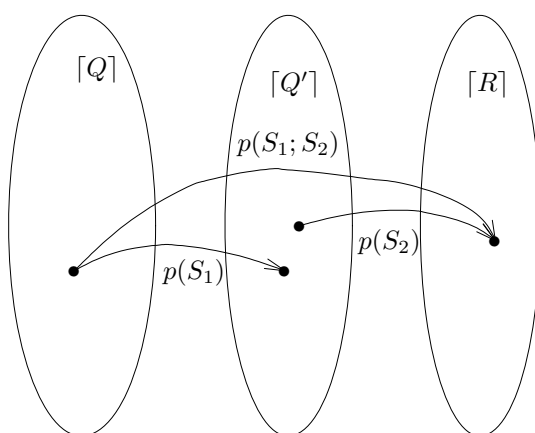
Továbbá (2) miatt  $p(S_2)(p(S_1)(q)) \subseteq [R]$ , tehát  $q \in lf(S, R)$ .  $\square$

A szekvencia levezetési szabálya és a specifikáció tétele alapján a következőt mondhatjuk: ha  $S_1$  és  $S_2$  olyan programok, amelyekre a paramétertér minden  $b$  pontjában  $Q_b \Rightarrow lf(S_1, Q'_b)$  és  $Q'_b \Rightarrow lf(S_2, R_b)$  teljesül, akkor  $(S_1; S_2)$  megoldja a  $Q_b, R_b$  párokkal adott feladatot.

#### 7.2. TÉTEL: A SZEKVENCIA LEVEZETÉSI SZABÁLYÁNAK MEGFORDÍTÁSA

Legyen  $S = (S_1; S_2)$ , és  $Q, R$  olyan állítások  $A$ -n, amelyekre  $Q \Rightarrow lf(S, R)$ .

Ekkor  $\exists Q' : A \rightarrow \mathbb{L}$  állítás, amelyre



7.1. ábra. A szekvencia levezetési szabálya

(1)  $Q \Rightarrow lf(S_1, Q')$  és

(2)  $Q' \Rightarrow lf(S_2, R)$ .

**Bizonyítás:** Legyen  $Q' = lf(S_2, R)$ . Ekkor (2) automatikusan teljesül. Csak (1) fennállását kell belátnunk. Ehhez indirekt módon feltesszük, hogy

$$\exists q \in [Q] : q \notin [lf(S_1, lf(S_2, R))].$$

Ez kétféleképpen fordulhat elő:

- $q \notin \mathcal{D}_{p(S_1)}$ , de ez ellentmond annak a feltételnek, mely szerint  $[Q] \subseteq \mathcal{D}_{p(S)} \subseteq \mathcal{D}_{p(S_1)}$ .
- $p(S_1)(q) \not\subseteq [lf(S_2, R)]$ . Ekkor legyen  $r \in p(S_1)(q) \setminus [lf(S_2, R)]$ . Ekkor két eset lehetséges:
  - $r \notin \mathcal{D}_{p(S_2)}$ . Ez ellentmond annak, hogy  $q \in \mathcal{D}_{p(S_2) \circ p(S_1)}$ .
  - $p(S_2)(r) \not\subseteq [R]$ : Legyen  $s \in p(S_2)(r) \setminus [R]$ . Ekkor  $s \in p(S)(q)$  és  $s \notin [R]$ , és ez ellentmond a  $p(S)(q) \subseteq [R]$  feltételnek.

□

## 7.2. Az elágazás levezetési szabálya

Vizsgáljuk meg, mi a kapcsolat az elágazás és az őt alkotó programok adott utófeltételhez tartozó leggyengébb előfeltétele között.

**7.3. TÉTEL:** AZ ELÁGAZÁS LEVEZETÉSI SZABÁLYA

Legyen  $IF = (\pi_1 : S_1, \dots, \pi_n : S_n)$ , és adott  $Q, R$  állítás  $A$ -n. Ha

- (1)  $Q \Rightarrow \bigvee_{i=1}^n \pi_i$ ,
- (2)  $\forall i \in [1..n] : Q \wedge \pi_i \Rightarrow lf(S_i, R)$ ,

akkor

$$Q \Rightarrow lf(IF, R).$$

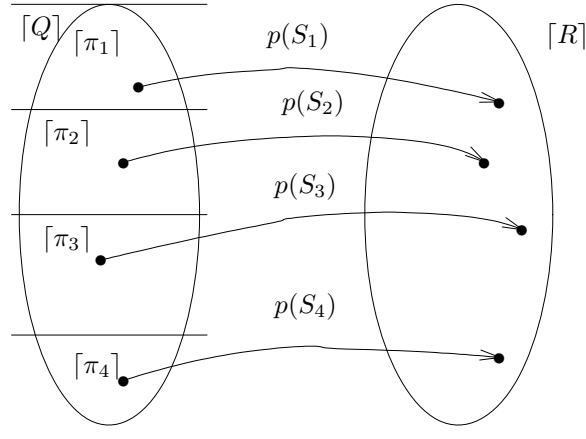
**Bizonyítás:** Legyen  $q \in [Q]$ , és tegyük fel, hogy valamelyik feltétel igaz  $q$ -ra, azaz  $\exists i \in [1..n] : q \in [\pi_i]$ . Ekkor  $q \in \mathcal{D}_{p(IF)}$ , ugyanis

$$\forall j \in [1..n] : q \in [\pi_j] \Rightarrow q \in [lf(S_j, R)] \Rightarrow q \in \mathcal{D}_{p(S_j)}.$$

Másrészt mivel  $\forall j \in [1..n] : q \in [\pi_j] \Rightarrow p(S_j)(q) \subseteq [R]$ :

$$p(IF)(q) = \bigcup_{\substack{j \in [1..n] \\ q \in [\pi_j]}} p(S_j)(q) \subseteq [R],$$

azaz  $q \in [lf(IF, R)]$ .  $\square$



7.2. ábra. Az elágazás levezetési szabálya

Felhasználva a specifikáció tételét és az elágazás levezetési szabályát azt mondhatjuk: Legyen adott az  $F$  feladat specifikációja  $(A, B, Q, R)$ . Ekkor ha minden  $b \in B$  paraméterre és minden  $S_i$  programra  $Q_b \wedge \pi_i \Rightarrow lf(S_i, R_b)$ , és minden  $b$  paraméterhez van olyan  $\pi_i$  feltétel, amelyre  $Q_b \Rightarrow \pi_i$ , akkor az elágazás megoldja a  $Q_b, R_b$  párokkal specifikált feladatot.

Hasonlóan a szekvencia levezetési szabályához az elágazás levezetési szabálya is megfordítható, tehát ha egy elágazás megold egy specifikált feladatot, akkor le is lehet vezetni.

**7.4. TÉTEL: AZ ELÁGAZÁS LEVEZETÉSI SZABÁLYÁNAK MEGFORDÍTÁSA**

Legyen  $IF = (\pi_1 : S_1, \dots, \pi_n : S_n)$ , és  $Q, R$  olyan állítások  $A$ -n, amelyekre

$$Q \Rightarrow lf(IF, R).$$

Ekkor

$$(1) Q \Rightarrow \bigvee_{i=1}^n \pi_i,$$

$$(2) \forall i \in [1..n] : Q \wedge \pi_i \Rightarrow lf(S_i, R).$$

**Bizonyítás:** Ha  $q \in [Q]$ , akkor  $q \in \mathcal{D}_{p(IF)}$ , vagyis  $\exists i \in [1..n] : q \in [\pi_i]$ . Tehát (1) teljesül.

Ezután belátjuk, hogy (2) is teljesül. Tegyük fel indirekt módon, hogy

$$\exists i \in [1..n] : [Q \wedge \pi_i] \not\subseteq [lf(S_i, R)].$$

Legyen  $q \in [Q \wedge \pi_i] \setminus [lf(S_i, R)]$ . Ekkor két eset lehetséges:

- $q \notin \mathcal{D}_{p(S_i)}$ . Ez ellentmond annak a feltevésnek, hogy  $q \in \mathcal{D}_{p(IF)}$ .
- $p(S_i)(q) \not\subseteq [R]$ . Ekkor

$$p(S_i)(q) \subseteq p(IF)(q) \subseteq [R],$$

tehát ellentmondásra jutottunk.

□

### 7.3. A ciklus levezetési szabálya

Az előző két konstrukcióhoz hasonlóan most megvizsgáljuk, milyen kapcsolat van a ciklus és a ciklusmag leggyengébb előfeltétele között.

**7.5. TÉTEL: A CIKLUS LEVEZETÉSI SZABÁLYA**

Adott  $P, Q, R$  állítás  $A$ -n,  $t : A \rightarrow \mathbb{Z}$  függvény, és legyen  $DO = (\pi, S_0)$ . Ha

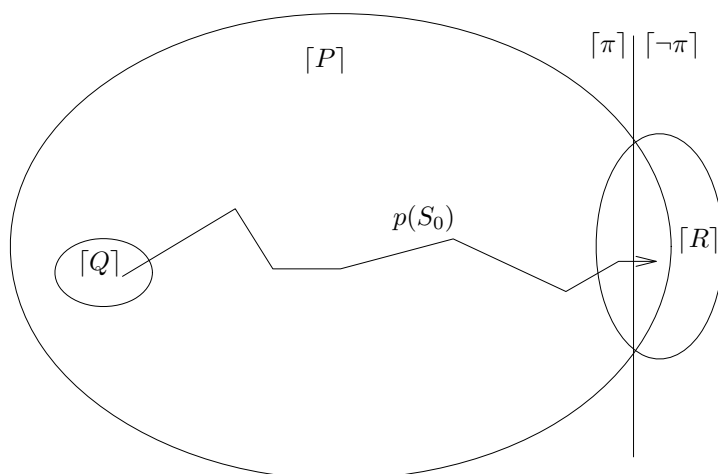
$$(1) Q \Rightarrow P,$$

$$(2) P \wedge \neg\pi \Rightarrow R,$$

$$(3) P \wedge \pi \Rightarrow t > 0,$$

$$(4) P \wedge \pi \Rightarrow lf(S_0, P),$$





7.3. ábra. A ciklus levezetési szabálya

$$(5) P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_0, t < t_0),$$

akkor

$$Q \Rightarrow lf(DO, R).$$

A tétel szerint azt kellene belátnunk, hogy:

$$(1) \forall q \in [Q] : q \in \mathcal{D}_{p(DO)} = \overline{\mathcal{D}_{p(S_0)|\pi}} \text{ és}$$

$$(2) \forall q \in [Q] : p(DO)(q) \subseteq [R], \text{ azaz } \overline{p(S_0)|\pi}(q) \subseteq [R].$$

Legyen a továbbiakban  $g = p(S_0)|\pi$ . Mielőtt magát a levezetési szabályt bizonyítanánk, ennek a  $g$  relációnak látjuk be két tulajdonságát.

**1. állítás:** Legyen  $q \in [P]$ . Ekkor

$$\forall k \in \mathbb{N}_0 : g^k(q) \subseteq [P].$$

**Bizonyítás:**  $k$  szerinti teljes indukcióval:

- $k = 0$ :  $g^0(q) = \{q\} \subseteq [P]$ .
- Tegyük fel, hogy  $g^k(q) \subseteq [P]$ . Legyen  $r \in g^k(q)$  tetszőleges. Ekkor két eset lehetséges:
  - $r \in [P \wedge \neg\pi]$ . Ekkor  $g(r) = \emptyset \subseteq [P]$ .
  - $r \in [P \wedge \pi]$ . Ekkor (4) miatt  $r \in \mathcal{D}_{p(S_0)}$  és  $g(r) = p(S_0)(r) \subseteq [P]$ .

Tehát:  $g^{k+1}(q) \subseteq \lceil P \rceil$ .

□

**2. állítás:** Legyen  $q \in \lceil P \rceil$ . Ekkor

$$\exists n \in \mathbb{N}_0 : g^n(q) = \emptyset.$$

**Bizonyítás:** Indirekt: tegyük fel, hogy  $\forall n \in \mathbb{N}_0 : g^n(q) \neq \emptyset$ . Ekkor  $\forall n \in \mathbb{N}_0$  esetén létezik a  $\max_{b \in g^n(q)} t(b)$ .

Ez  $k = 0$  esetén nyilván igaz. Ha

$$m = \max_{b \in g^k(q)} t(b),$$

akkor az előzőek szerint  $\forall r \in g^k(q)$  esetén két lehetőség van:

- $r \in \lceil P \wedge \neg \pi \rceil$ . Ekkor  $g(r) = \emptyset$ .
- $r \in \lceil P \wedge \pi \rceil$ . Ekkor (5) miatt  $r \in \mathcal{D}_{p(S_0)}$  és  $\forall x \in g(r) : t(x) < t(r) \leq m$ .

Mivel feltettük, hogy  $g^{k+1}(q) \neq \emptyset$ , a második lehetőség legalább egy  $r$  esetén teljesül. Tehát a maximum minden  $n$ -re létezne, sőt szigorúan monoton csökkenne.

Ez viszont ellentmond (3)-nak, hiszen  $t$  egész értékű függvény.

□

**Bizonyítás:** (Ciklus levezetési szabálya) Legyen  $q \in \lceil Q \rceil$  tetszőleges. Mivel (1) miatt  $\lceil Q \rceil \subseteq \lceil P \rceil$ , ezért  $q \in \lceil P \rceil$  teljesül.

Ekkor a 2. állítás alapján

$$\exists n \in \mathbb{N}_0 : (p(S_0)|\pi)^n(q) = \emptyset,$$

azaz

$$q \in \overline{\mathcal{D}_{p(S_0)|\pi}} = \mathcal{D}_{p(DO)}.$$

Ekkor a feltételre vonatkozó lezárt definíciója alapján:

$$p(DO)(q) \subseteq \lceil \neg \pi \rceil.$$

Másrészt az 1. állítás miatt

$$p(DO)(q) \subseteq \lceil P \rceil,$$

és ekkor (2) miatt

$$p(DO)(q) \subseteq \lceil P \wedge \neg \pi \rceil \subseteq \lceil R \rceil,$$

tehát

$$q \in \lceil lf(DO, R) \rceil.$$

□

A levezetési szabályban szereplő  $P$  állítást a *ciklus invariáns* tulajdonságának, a  $t$  függvényt *terminálófüggvénynek* nevezzük. A  $P$  invarianciáját az (1) és (4) feltételek biztosítják: garantálják, hogy az invariáns tulajdonság a ciklusmag minden lefutása előtt és után teljesül. A terminálófüggvény a ciklus befejeződését biztosítja: az (5) pont alapján a ciklusmag minden lefutása legalább eggyel csökkenti a terminálófüggvényt, másrészt a (3) miatt a terminálófüggvénynek pozitívnak kell lennie. A (2) feltétel garantálja, hogy ha a ciklus befejeződik, akkor az utófeltétel igazsághalmazába jut.

A ciklus levezetési szabályának és a specifikáció tételének felhasználásával elégséges feltételt adhatunk a megoldásra: ha adott a feladat specifikációja  $(A, B, Q, R)$ , és találunk olyan invariáns állítást és terminálófüggvényt, hogy a paramétertér minden elemére teljesül a ciklus levezetési szabályának öt feltétele, akkor a ciklus megoldja a  $(Q_b, R_b)$  párokkal definiált feladatot.

A ciklus levezetési szabálya visszafelé nem igaz, azaz van olyan ciklus, amelyet nem lehet levezetni. Ennek az az oka, hogy egy levezetett ciklus programfüggvénye mindig korlátos lezárt, hiszen az állapotter minden pontjában a terminálófüggvény értéke korlátozza a ciklusmag lefutásainak számát.

Azonban ha egy ciklus programfüggvénye megegyezik a ciklusmag ciklusfeltételre vonatkozó korlátos lezártjával, akkor a ciklus levezethető.

#### 7.6. TÉTEL: A CIKLUS LEVEZETÉSI SZABÁLYÁNAK MEGFORDÍTÁSA

Legyen  $DO = (\pi, S_0)$ , és  $Q, R$  olyan állítások  $A$ -n, amelyekre  $Q \Rightarrow lf(DO, R)$ , és tegyük fel, hogy  $p(DO) = \overline{p(S_0)}|\pi$ . Ekkor létezik  $P$  állítás és  $t : A \rightarrow \mathbb{Z}$  függvény, amelyekre

- (1)  $Q \Rightarrow P$ ,
- (2)  $P \wedge \neg\pi \Rightarrow R$ ,
- (3)  $P \wedge \pi \Rightarrow t > 0$ ,
- (4)  $P \wedge \pi \Rightarrow lf(S_0, P)$ ,
- (5)  $P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_0, t < t_0)$ .

**Bizonyítás:** Legyen  $P = lf(DO, R)$ , és válasszuk  $t$ -t úgy, hogy  $\forall a \in A$ :

$$t(a) = \begin{cases} 0, & \text{ha } a \notin \mathcal{D}_{p(DO)} \cap \lceil \pi \rceil; \\ \max\{i \in \mathbb{N} \mid p(S_0)^i(a) \cap \lceil \pi \rceil \neq \emptyset\}, & \text{ha } a \in \mathcal{D}_{p(DO)} \cap \lceil \pi \rceil. \end{cases}$$

Ekkor

- (1)  $Q \Rightarrow lf(DO, R)$  triviálisan teljesül.
- (2) Legyen  $a \in \lceil P \wedge \neg\pi \rceil$ . Ekkor mivel  $a \in \lceil lf(DO, R) \rceil$ ,  $p(DO)(a) \subseteq \lceil R \rceil$ . Másrészt mivel  $a \in \lceil \neg\pi \rceil$ ,  $p(DO)(a) = \{a\}$ . Tehát  $a \in \lceil R \rceil$ , azaz  $\lceil P \wedge \neg\pi \rceil \subseteq \lceil R \rceil$ .

(3)  $t$  definíciója miatt nyilvánvaló.

(4) Felhasználva a lezárt azon egyszerű tulajdonságát, hogy

$$a \in \mathcal{D}_{\bar{R}} \implies R(a) \subseteq \mathcal{D}_{\bar{R}},$$

a következő eredményt kapjuk. Legyen  $a \in [lf(DO, R) \wedge \pi]$ . Ekkor

$$a \in \mathcal{D}_{p(S_0)} \quad \text{és} \quad p(S_0)(a) \subseteq [lf(DO, R)],$$

tehát

$$a \in lf(S_0, P).$$

(5) A  $t$  definíciójából leolvasható, hogy a ciklusmag egyszeri végrehajtása csökkenti a terminálófüggvény értékét:

Legyen  $a \in [P \wedge \pi]$ ,  $t_0 = t(a)$ ,  $b \in p(S_0)(a)$ . Ekkor ha

$$t(a) = \max\{i \in \mathbb{N} \mid p(S_0)^i(a) \cap [\pi] \neq \emptyset\},$$

akkor

$$t(b) < t(a),$$

azaz

$$t(b) < t_0.$$

□

Azt, hogy a lezárt és a korlátos lezárt megegyezik, a  $t$  definíciójában használtuk ki: ez a feltétel garantálja, hogy a definícióban szereplő maximum véges.

Megjegyzés: mivel a ciklus levezetési szabálya nem megfordítható, nem minden ciklus vezethető le, ez azonban nem jelent érdemi korlátozást. Ha van egy ciklus, amely megoldása egy feladatnak, akkor biztosan van olyan ciklus is, amelynek a feltétele ugyanaz, szintén megoldása a feladatnak, és a programfüggvénye korlátos lezárt.

A ciklus definíciójából látszik, hogy ha  $S_1 \subseteq S_2$ , akkor  $DO_1(\pi, S_1) \subseteq DO_2(\pi, S_2)$ , és a 2.7. példa szerint  $DO_1$  megoldása minden olyan feladatnak, amelynek  $DO_2$  megoldása. Tehát  $S_1$  akár determinisztikus is lehet, és akkor a lezártja  $p(S_1)|\pi$ -nek biztosan korlátos.

## 7.4. Feladatok

**7.1.** Tegyük fel, hogy teljesül a ciklus levezetési szabályának mind az öt feltétele, és  $Q$  igazsághalmaza nem üres. Lehet-e üres a

a)  $[P \wedge R]$  halmaz?

b)  $[P \wedge \neg\pi \wedge R]$  halmaz?

**7.2.** Tegyük fel, hogy teljesül a ciklus levezetési szabályának mind az öt feltétele, és  $(Q \wedge \pi)$  igazsághalmaza nem üres. Lehet-e üres az  $lf(S_0, P)$  és  $lf(DO, R)$  igazsághalmazának metszete?

**7.3.** Tegyük fel, hogy teljesül a ciklus levezetési szabályának mind az öt feltétele. Legyen  $g = p(S_0)|_{\lceil \pi \rceil}$  és  $q \in \lceil P \rceil \cap \lceil \pi \rceil$ . Igaz-e, hogy

- $\forall k \in \mathbb{N}_0 : g^k(q) \subseteq \lceil P \rceil$ ?
- $b \in g^k(q) \cap \lceil \pi \rceil \cap \lceil P \rceil \Rightarrow t(b) \leq t(q) - k$ ?
- $g|_{\pi} = p(S_0)|_{\pi}$ ?
- $\exists k \in \mathbb{N}_0 : k \leq t(q)$  és  $g^k(q) \subseteq \lceil \neg \pi \rceil$ ?

**7.4.** Legyen  $S = (S_1; S_2)$  és  $Q, Q'$  és  $R$  olyan állítások, amelyekre

$$Q \Rightarrow lf(S_1, Q'), Q' \Rightarrow lf(S_2, R), Q \Rightarrow lf(S, R).$$

Lehetséges-e, hogy  $\lceil Q \rceil \cap \lceil R \rceil = \emptyset$  és  $\lceil Q \rceil \cap \lceil Q' \rceil \neq \emptyset$  és  $\lceil Q' \rceil \cap \lceil R \rceil \neq \emptyset$ ?

Indokolja, ha nem, és írjon rá példát, ha igen!

**7.5.**  $A = \mathbb{Z} \times \mathbb{N}_0$

$$x \quad y$$

$B = \mathbb{Z} \times \mathbb{N}_0$

$$x' \quad y'$$

$$Q : (x = x' \wedge y = y')$$

$$R : (x = x' - y' \wedge y = 0)$$

$$S_0 = \{((x, y), < (x, y), (x - 1, y), (x - 1, y - 1) >) \mid x \in \mathbb{Z} \text{ és } y \in \mathbb{N}\} \cup \{((x, 0), < (x, 0) >) \mid x \in \mathbb{Z}\}$$

$$DO = \{((x, y), < (x, y), (x - 1, y), (x - 1, y - 1), (x - 2, y - 1), (x - 2, y - 2), \dots, (x - y + 1, 1), (x - y, 1)(x - y, 0) >) \mid x \in \mathbb{Z} \text{ és } y \in \mathbb{N}_0\}.$$

Megjegyzés: Az  $(x, 0)$  párhoz 1 hosszúságú, az  $(x, 1)$  párhoz 3 hosszúságú, az  $(x, 2)$  párhoz 5 hosszúságú sorozatot rendel a program.

Tudjuk, hogy  $DO = (\pi, S_0)$  valamilyen  $\pi$ -re. Igaz-e, hogy található olyan  $P$  állítás és  $t : A \rightarrow \mathbb{Z}$  függvény, hogy a ciklus levezetési szabályának feltételei teljesülnek, és ha igen, adjon meg egy megfelelő  $\pi$ -t,  $P$ -t és  $t$ -t!

**7.6.**  $A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$

$$k \quad x \quad i \quad a \quad b$$

$B = \mathbb{Z} \times \mathbb{Z}$

$$a' \quad b'$$

$$S = (k := 5; (IF(a > b : x := a - b, a \leq b : x := b - a); i := i + 1))$$

$$Q : (a = a' \wedge b = b' \wedge i \in [0..1] \wedge |a - b| > 10)$$

$$R : (a = a' \wedge b = b' \wedge k \cdot i \leq x)$$

Bizonyítsuk be, hogy  $Q \Rightarrow lf(S, R)$ !



## 8. fejezet

# Elemi programok

A 2. fejezetben a programozási feladat kapcsán megfogalmaztuk, hogy mindig felteesszük, létezik a programoknak egy adott halmaza, amelyekből összerakhatjuk a programjainkat (2.7.). Ezeket a programokat szoktuk *primitív programoknak* nevezni.

Ebben a fejezetben bevezetünk néhány elméleti szempontból „egyszerű” programot. Ezeket a programokat a következő fejezetekben gyakorta fogjuk használni, ezért megvizsgáljuk a programfüggvényüket és egy adott utófeltételhez tartozó leggyengébb előfeltételüket.

### 8.1. Elemi programok

#### 8.1. DEFINÍCIÓ: ELEMI PROGRAM

*Eleminek* nevezünk egy  $A$  állapotterén egy  $S$  programot, ha

$$\forall a \in A : S(a) \subseteq \{\langle a \rangle, \langle a, a, a, \dots \rangle, \langle a, b \rangle \mid b \neq a\}.$$

A definíció szerint minden programhoz található vele ekvivalens elemi program, csak a sorozatok közbülső elemeit el kell hagyni, s így lényegében, egy adott szinten minden program elemi.

Az elemi programok közül kiválasztunk néhány speciális tulajdonsággal rendelkezőt, és a továbbiakban velük foglalkozunk.

Az első ilyen program az lesz, ami nem csinál semmit.

#### 8.2. DEFINÍCIÓ: SKIP

*SKIP*-nek nevezzük azt a programot, amelyre

$$\forall a \in A : SKIP(a) = \{\langle a \rangle\}.$$

A második program a törlődés, amelynek legfontosabb jellemzője, hogy soha sem terminál.

### 8.3. DEFINÍCIÓ: ABORT

*ABORT*-tal jelöljük azt a programot, amelyre

$$\forall a \in A : ABORT(a) = \{\langle a, a, a, \dots \rangle\}.$$

Az eddig felsorolt két elemi program valóban nagyon egyszerű, de – talán meglepő módon – mégis jelentős szerepük van. A harmadik speciális elemi program az értékadás. Az előzőeknél bonyolultabb, de sokkal fontosabb is.

Az értékadás definíciójához bevezetünk egy jelölést. Legyen  $A = A_1 \times \dots \times A_n$  és  $\forall i \in [1..n] : F_i \subseteq A \times A_i$ . Jelöljük  $F = (F_1, \dots, F_n)$ -nel a következő  $F \subseteq A \times A$  relációt:

$$\mathcal{D}_F = \bigcap_{i=1}^n \mathcal{D}_{F_i} \quad \text{és} \quad \forall a \in \mathcal{D}_F : F(a) = \prod_{i \in [1..n]} F_i(a).$$

### 8.4. DEFINÍCIÓ: ÉRTÉKADÁS

Legyen  $A = A_1 \times \dots \times A_n$  és  $F = (F_1, \dots, F_n)$ . Az  $S$  program *általános értékadás*, ha  $\forall a \in A$ :

$$S(a) = \begin{cases} \{red(\langle a, b \rangle) \mid b \in F(a)\}, & \text{ha } a \in \mathcal{D}_F; \\ \{\langle a, a, a, \dots \rangle\}, & \text{ha } a \notin \mathcal{D}_F. \end{cases}$$

A fenti  $F_i$  komponensrelációk pontosan azt írják le, hogy az adott értékadás miként változtatja meg az állapotteret egyes komponenseit.

Az általános értékadason belül speciális eseteket különböztetünk meg:

- Ha  $\mathcal{D}_F = A$ , akkor az  $S$  programot *értékkiválasztásnak* nevezzük.
- Ha az  $F$  reláció függvény, akkor az  $S$  programot *értékadásnak* nevezzük.
- Ha  $\mathcal{D}_F \subset A$ , akkor  $S$  *parciális értékkiválasztás*.
- Ha  $\mathcal{D}_F \subset A$  és  $F$  determinisztikus ( $F$  parciális függvény), akkor  $S$  *parciális értékadás*.

Az értékadást és a parciális értékadást  $a := F(a)$ -val, az értékkiválasztást és a parciális értékkiválasztást  $a \in F(a)$ -val jelöljük.

Ha egy kivételével az összes  $F_i$  projekció – azaz az értékadás az állapotteret csak egy komponensét (csak egy változó értékét) változtatja meg –, akkor  $S$ -et *egyszerű értékadásnak*, egyébként *szimultán értékadásnak* nevezzük.

Az értékadás egy kicsit bonyolultabb, mint előző két társa, de egy kicsit „értékesebb” is, hiszen értékadással minden feladat megoldható! A kérdés persze csupán az, hogy az éppen adott feladat által definiált értékadás megengedett művelet-e. Ezzel a kérdéssel a későbbiekben – a programozási feladat megoldása során – fogunk foglalkozni.

Vizsgáljuk meg a fent definiált speciális elemi programok programfüggvényeit!



**8.1. állítás:**

1.  $p(SKIP) = id_A$ ,
2.  $p(ABORT) = \emptyset$ ,
3.  $p(a := F(a)) = F$ ,
4.  $p(a : \in F(a)) = F$ .

A bizonyítás triviális (a feladatok között szerepel).

**8.2. Elemi programok leggyengébb előfeltétele**

Most, hogy megvizsgáltuk a programfüggvényeket, nézzük meg az elemi programok adott utófeltételhez tartozó leggyengébb előfeltételét.

Mivel a SKIP program programfüggvénye az identikus leképezés, egy tetszőleges  $R$  utófeltételhez tartozó leggyengébb előfeltétele:

$$lf(SKIP, R) = R.$$

Hasonlóan egyszerűen látható, hogy – mivel az ABORT program programfüggvénye üres – a leggyengébb előfeltétel definíciója alapján egy tetszőleges  $R$  utófeltétel esetén

$$lf(ABORT, R) = Hamis.$$

Az általános értékadás leggyengébb előfeltételét külön vizsgáljuk a determinisztikus és nondeterminisztikus, illetve a globális és a parciális esetben. Felhasználva a 3. és a 8. állításokat:

$$\lceil lf(a : \in F(a), R) \rceil = \lceil R \circ F \rceil.$$

Ha  $F : A \rightarrow A$  függvény, akkor  $R \circ F$  függvény, és ezért

$$lf(a := F(a), R) = R \circ F.$$

Ha  $F$  parciális függvény, tehát az értékadás parciális, akkor ennek leggyengébb előfeltétele:

$$\forall b \in A : lf(a := F(a), R)(b) = \begin{cases} R \circ F(b), & \text{ha } b \in \mathcal{D}_F; \\ hamis, & \text{ha } b \notin \mathcal{D}_F. \end{cases}$$

Most vizsgáljuk azt a két esetet, amikor  $F$  nondeterminisztikus. Feltéve, hogy  $\mathcal{D}_F = A$ , az érték kiválasztás leggyengébb előfeltétele

$$\forall b \in A : lf(a : \in F(a), R)(b) = \begin{cases} igaz, & \text{ha } F(b) \subseteq \lceil R \rceil; \\ hamis & \text{egyébként.} \end{cases}$$

Ugyanez parciális esetben:

$$\forall b \in A : lf(a : \in F(a), R)(b) = \begin{cases} igaz, & \text{ha } a \in \mathcal{D}_F \text{ és } F(b) \subseteq [R]; \\ hamis & \text{egyébként.} \end{cases}$$

Az értékadást általában változókkal írjuk le. Legyenek az állapotér változói rendre  $x_1, x_2, \dots, x_n$ . Ekkor az  $a := F(a)$  program jelölésére az alábbi formulát használhatjuk:

$$x_1, x_2, \dots, x_n := F_1(x_1, x_2, \dots, x_n), F_2(x_1, x_2, \dots, x_n), \dots, F_n(x_1, x_2, \dots, x_n).$$

A gyakorlatban az esetek többségében  $F$  komponenseinek nagy része projekció, azaz  $F_i(x_1, x_2, \dots, x_n) = x_i$ . Ekkor az értékadás jelöléséből a bal oldalról  $x_i$ -t, a jobb oldalról pedig  $F_i(x_1, x_2, \dots, x_n)$ -t elhagyjuk. Vegyük észre, hogy egyszerű értékadás esetén a bal oldalon csak egy változó, a jobb oldalon pedig csak egy kifejezés marad.

Jelölésünket abban az esetben még tovább egyszerűsíthetjük, ha az értékadás jobb oldala ( $F_i$ ) nem függ minden változótól. Ekkor a jobb oldalon csak azokat a változókat tüntetjük fel, amelyekről  $F_i$  függ.

Nézzük meg egy egyszerű példán a fent leírtakat. Legyen az állapotér

$$A = \mathbb{Z} \times \mathbb{L} \\ x \quad l$$

A továbbiakban a fentihez hasonlóan az állapotér egyes komponenseihez tartozó változókat a komponensek alá fogjuk írni. Legyenek az értékadás komponensei:  $\forall a = (a_1, a_2) \in A$ :

$$F_1(a_1, a_2) = a_1, \text{ azaz } F_1 = pr_{\mathbb{Z}}, \text{ és} \\ F_2(a_1, a_2) = (a_1 > 0).$$

Ekkor az  $a := F(a)$  értékadás változókkal felírva:

$$x, l := x, (x > 0).$$

A jelölés fent leírt egyszerűsítéseit elvégezve az

$$l := (x > 0)$$

egyszerű értékadást kapjuk.

Ha felhasználjuk, hogy az értékadás leggyengébb előfeltétele  $R \circ F$ , akkor egy változókkal felírt értékadás változókkal megadott előfeltételét egyszerűen kiszámíthatjuk a kompozíció képzésének megfelelően: helyettesítsük az utófeltételben az értékadásban szereplő változókat az új értékükkel. Erre bevezetünk egy új jelölést is: legyenek  $x_1, x_2, \dots, x_n$  rendre az állapotér változói, ekkor

$$lf(x_{i_1}, \dots, x_{i_m} := F_{i_1}(x_1, \dots, x_n), \dots, F_{i_m}(x_1, \dots, x_n), R) = \\ R^{x_{i_1} \leftarrow F_{i_1}(x_1, \dots, x_n), \dots, x_{i_m} \leftarrow F_{i_m}(x_1, \dots, x_n)}$$

Parciális értékadás esetén:

$$lf(x_{i_1}, \dots, x_{i_m} := F_{i_1}(x_1, \dots, x_n), \dots, F_{i_m}(x_1, \dots, x_n), R) = \\ (x_1, \dots, x_n) \in \mathcal{D}_F \wedge R^{x_{i_1} \leftarrow F_{i_1}(x_1, \dots, x_n), \dots, x_{i_m} \leftarrow F_{i_m}(x_1, \dots, x_n)}$$

Az érték kiválasztás kicsit bonyolultabb:

$$lf(x_{i_1}, \dots, x_{i_m} := F_{i_1}(x_1, \dots, x_n), \dots, F_{i_m}(x_1, \dots, x_n), R) = \\ \forall (e_{i_1}, \dots, e_{i_m}) \in \times_{j=1}^m F_{i_j}(x_1, \dots, x_n) : R^{x_{i_1} \leftarrow e_{i_1}, \dots, x_{i_m} \leftarrow e_{i_m}}$$

Végül a parciális érték kiválasztás:

$$lf(x_{i_1}, \dots, x_{i_m} := F_{i_1}(x_1, \dots, x_n), \dots, F_{i_m}(x_1, \dots, x_n), R) = \\ (x_1, \dots, x_n) \in \mathcal{D}_F \wedge \forall (e_{i_1}, \dots, e_{i_m}) \in \times_{j=1}^m F_{i_j}(x_1, \dots, x_n) : \\ R^{x_{i_1} \leftarrow e_{i_1}, \dots, x_{i_m} \leftarrow e_{i_m}}$$

A továbbiakban, ha mást nem mondunk, primitív programnak tekintjük azokat az – esetleg parciális – értékadásokat, amelyek jobb oldalán a szokásos aritmetikai, logikai kifejezésekkel felírt függvények állnak.

### 8.3. Az értékadás mint feladatspecifikáció

Az értékadás lényegében egy programfüggvényével definiált program. Kézenfekvő, hogy ezt a programfüggvényt feladatnak is tekinthetjük. Egy értékadás által definiált feladat könnyen átírható a specifikáció tételének megfelelő formára is.

Legyen  $x_1, \dots, x_n := F_1(x_1, \dots, x_n), \dots, F_n(x_1, \dots, x_n) \subseteq A \times A$  egy értékadás. Az  $A$  állapotér változói értelemszerűen  $x_1, \dots, x_n$ .

Tekintsük a következő feladatspecifikációt:

$$A = A_1 \times \dots \times A_n \\ x_1 \quad \dots \quad x_n$$

$$B = B_1 \times \dots \times B_n \\ x'_1 \quad \dots \quad x'_n$$

$$Q : (x_1 = x'_1 \wedge \dots \wedge x_n = x'_n)$$

$$R : (x_1 = F_1(x'_1, \dots, x'_n) \wedge \dots \wedge x_n = F_n(x'_1, \dots, x'_n))$$

Nyilvánvaló, hogy éppen az  $F$  függvényt specifikáltuk.

Ha nem értékadásról van szó, hanem érték kiválasztásról, akkor az utófeltételekben nem  $=$ , hanem  $\in$  szerepel.

A parciális esetekben az előfeltételeket célszerű leszűkíteni az  $F$  értelmezési tartományára az  $x_i \in \mathcal{D}_{F_i}$  feltételek hozzávételével.

## 8.4. Feladatok

**8.1.**  $A = \{1, 2, 3\}, B = \{a, b\}, C = A \times B$ . Legyen  $S$  program  $A$ -n,  $S = \{1 \rightarrow \langle 1 \rangle, 2 \rightarrow \langle 2222 \dots \rangle, 3 \rightarrow \langle 31 \rangle\}$ .

Legyen  $S_1$  az  $S$  kiterjesztése  $C$ -re,  $M$  pedig olyan program  $C$ -n, hogy  $M$  ekvivalens  $S$ -sel  $A$ -n.

- (a) Elemi program-e  $S$ ?
- (b) Elemi program-e  $S_1$ , és biztosan elemi program-e  $M$ ?

**8.2.** Tekintsük az alábbi állapotteret:

$$A = \mathbb{N} \times \mathbb{N}$$

$$x \quad y$$

Mi az  $(x, y) := F(x, y)$ ,  $F = (F_1, F_2)$ ,  $F_1(x, y) = y$ ,  $F_2(x, y) = x$ , azaz az  $F(p, q) = \{b \in A \mid x(b) = q \text{ és } y(b) = p\}$  értékadás  $R = (x < y)$  utófeltételhez tartozó leggyengébb előfeltétele?

- 8.3.** Legyen  $A$  tetszőleges állapotter. Melyek azok a feladatok az  $A$ -n, amelyeknek megoldása a *SKIP* program?
- 8.4.** Legyen  $A$  tetszőleges állapotter. Melyek azok a feladatok az  $A$ -n, amelyeknek megoldása az *ABORT* program?

## 9. fejezet

# A programkonstrukciók és a kiszámíthatóság

Ebben a fejezetben kis kitérőt teszünk a kiszámíthatóság-elmélet felé, és megmutatjuk, hogy az imént bevezetett három programkonstrukció segítségével minden – elméletileg megoldható – feladatot meg tudunk oldani. Ehhez kapcsolatot létesítünk a kiszámítható függvények és a „jól konstruált” programok között, és ezzel megmutatjuk azt is, hogy ha egy feladat megoldható, akkor megoldható levezetéssel is.

A kiszámíthatóság fogalmát általában a parciálisan rekurzív függvények bevezetésén keresztül szokás megadni. Más definíciós lehetőségekről, például a Turing-gépről, bizonyítható, hogy a kiszámíthatóság szempontjából egyenértékűek a parciálisan rekurzív függvényekkel.

Felhívjuk a figyelmet arra, hogy ebben a fejezetben függvény alatt mindig parciális függvényt értünk. A mindenütt definiált függvényekre a totális jelzőt használjuk.

### 9.1. Parciálisan rekurzív függvények

A kiszámíthatóság parciálisan rekurzív függvényekre alapozott modelljében csak  $f \in \mathbb{N}^m \rightarrow \mathbb{N}^n$  típusú függvények szerepelnek. Először az alapfüggvényeket definiáljuk:

- $suc : \mathbb{N} \rightarrow \mathbb{N}, \quad \forall x \in \mathbb{N}$ :

$$suc(x) = x + 1,$$

- $\forall n \in \mathbb{N}_0 : \quad c_1^{(n)} : \mathbb{N}^n \rightarrow \mathbb{N}, \quad \forall (x_1, \dots, x_n) \in \mathbb{N}^n$ :

$$c_1^{(n)}(x_1, \dots, x_n) = 1,$$

- $\forall n \in \mathbb{N} : \forall i \in [1..n] : pr_i^{(n)} : \mathbb{N}^n \rightarrow \mathbb{N}, \quad \forall (x_1, \dots, x_n) \in \mathbb{N}^n :$   

$$pr_i^{(n)}(x_1, \dots, x_n) = x_i.$$

A továbbiakban definiálunk néhány elemi függvény-operátort:

**Kompozíció.** Legyen  $f \in \mathbb{N}^m \rightarrow \mathbb{N}^n$  és  $g \in \mathbb{N}^n \rightarrow \mathbb{N}^k$ . Az  $f$  és  $g$  kompozíciója az alábbi függvény:

$$g \circ f \in \mathbb{N}^m \rightarrow \mathbb{N}^k, \quad \mathcal{D}_{g \circ f} = \{x \in \mathcal{D}_f \mid f(x) \in \mathcal{D}_g\} \text{ és } \forall x \in \mathcal{D}_{g \circ f} :$$

$$(g \circ f)(x) = g(f(x)).$$

Vegyük észre, hogy ez az operátor megegyezik az alapfogalmaknál bevezetett relációk közötti kompozícióval (tulajdonképpen a szigorú kompozícióval, de függvények esetén ez a kettő azonos).

**Direktszorzat.** Legyen  $k \in \mathbb{N}$  rögzített, és  $\forall i \in [1..k] : f_i \in \mathbb{N}^m \rightarrow \mathbb{N}^{n_i}$ . E függvények direktszorzata  $(f_1, f_2, \dots, f_k) \in \mathbb{N}^m \rightarrow \mathbb{N}^{n_1} \times \dots \times \mathbb{N}^{n_k}$ ,  $\mathcal{D}_{(f_1, f_2, \dots, f_k)} = \bigcap_{i=1}^k \mathcal{D}_{f_i}$  és  $\forall x \in \mathcal{D}_{(f_1, f_2, \dots, f_k)}$ :

$$(f_1, f_2, \dots, f_k)(x) = (f_1(x), \dots, f_k(x)).$$

**Rekurzió.** Legyen  $n$  rögzített,  $f \in \mathbb{N}^n \rightarrow \mathbb{N}$  és  $g \in \mathbb{N}^{n+2} \rightarrow \mathbb{N}$ . Az  $f$  függvény  $g$  szerinti rekurziója  $\varrho(f, g) \in \mathbb{N}^{n+1} \rightarrow \mathbb{N}$ , és

$$\begin{aligned} \varrho(f, g)(x_1, \dots, x_n, 1) &= f(x_1, \dots, x_n), \\ \varrho(f, g)(x_1, \dots, x_n, k+1) &= g(x_1, \dots, x_n, k, \varrho(f, g)(x_1, \dots, x_n, k)). \end{aligned}$$

$\varrho(f, g)$  értelmezési tartománya azon pontok halmaza, ahonnan kiindulva a fenti rekurzió elvégezhető.

**$\mu$ -operátor.** Legyen  $f \in \mathbb{N}^{n+1} \rightarrow \mathbb{N}$ . A  $\mu$ -operátort erre a függvényre alkalmazva azt a  $\mu(f) \in \mathbb{N}^n \rightarrow \mathbb{N}$  függvényt kapjuk, amelyre  $\mathcal{D}_{\mu(f)} = \{(x_1, \dots, x_n) \in \mathbb{N}^n \mid \exists y \in \mathbb{N} : f(x_1, \dots, x_n, y) = 1 \wedge \forall i \in [1..y-1] : (x_1, \dots, x_n, i) \in \mathcal{D}_f\}$ , és  $\forall (x_1, \dots, x_n) \in \mathcal{D}_{\mu(f)}$ :

$$\mu(f)(x_1, \dots, x_n) = \min\{y \mid f(x_1, \dots, x_n, y) = 1\}.$$

A fenti alapfüggvények és a bevezetett operátorok segítségével már definiálható a parciális rekurzív függvények halmaza.

### 9.1. DEFINÍCIÓ: PARCIÁLIS REKURZÍV FÜGGVÉNY

Az  $f : \mathbb{N}^m \rightarrow \mathbb{N}^n$  ( $n, m \in \mathbb{N}$ ) függvény akkor és csak akkor *parciális rekurzív*, ha az alábbiak egyike teljesül:

- $f$  az alapfüggvények valamelyike;
- $f$  kifejezhető a fenti operátoroknak parciális rekurzív függvényekre történő véges sokszori alkalmazásával.

## 9.2. A parciális rekurzív függvények kiszámítása

Ahhoz, hogy a fenti függvényeket kiszámító programokat tudjunk adni, definiálnunk kell az ilyen függvények által meghatározott feladatot. Legyen  $f : \mathbb{N}^m \rightarrow \mathbb{N}^n$  egy tetszőleges függvény ( $m, n$  rögzített). Az  $f$  által meghatározott feladat specifikációja:

$$\begin{aligned} A &= \mathbb{N} \times \dots \times \mathbb{N} \times \mathbb{N} \times \dots \times \mathbb{N} \\ &\quad x_1 \qquad \qquad x_m \quad y_1 \qquad \qquad y_m \\ B &= \mathbb{N} \times \dots \times \mathbb{N} \\ &\quad x'_1 \qquad \qquad x'_m \\ Q &: (x_1 = x'_1 \wedge \dots \wedge x_m = x'_m \wedge (x_1, \dots, x_m) \in \mathcal{D}_f) \\ R &: (Q \wedge (y_1, \dots, y_n) = f(x'_1, \dots, x'_m)) \end{aligned}$$

Feltesszük, hogy az alapfüggvényeket kiszámító programok benne vannak a megengedett programok halmazában. Ezt nyugodtan megtehetjük, hiszen minden programozási nyelv tartalmaz ilyen utasításokat.

Megmutatjuk, hogy az elemi függvény-operátorok (kompozíció, direktszorzat, rekurzió,  $\mu$ -operátor) kiszámíthatók jól konstruált programokkal.

**Kompozíció.** Legyen  $f \in \mathbb{N}^m \rightarrow \mathbb{N}^n$  és  $g \in \mathbb{N}^n \rightarrow \mathbb{N}^k$ . Ekkor a  $g \circ f$  által meghatározott feladat specifikációja:

$$\begin{aligned} A &= \mathbb{N} \times \dots \times \mathbb{N} \times \mathbb{N} \times \dots \times \mathbb{N} \\ &\quad x_1 \qquad \qquad x_m \quad y_1 \qquad \qquad y_k \\ B &= \mathbb{N} \times \dots \times \mathbb{N} \\ &\quad x'_1 \qquad \qquad x'_m \\ Q &: (x_1 = x'_1 \wedge \dots \wedge x_m = x'_m \wedge (x_1, \dots, x_m) \in \mathcal{D}_{g \circ f}) \\ R &: (Q \wedge (y_1, \dots, y_n) = (g \circ f)(x'_1, \dots, x'_m)) \end{aligned}$$

Jelöljük  $z_1, \dots, z_n := f(x_1, \dots, x_m)$ -mel azt a programot, amely kiszámítja  $f$ -et, és hasonlóan  $y_1, \dots, y_k := g(z_1, \dots, z_n)$ -nel azt, amelyik kiszámítja  $g$ -t. Tegyük fel, hogy ez a két program jól konstruált program. Ebben az esetben a két program szekvenciája kiszámítja  $g \circ f$ -et, azaz megoldja a fent specifikált feladatot. Legyen  $Q'$  a második program  $R$  utófeltételhez tartozó leggyengébb előfeltétele:

$$Q' : (Q \wedge g(z_1, \dots, z_n) = (g \circ f)(x'_1, \dots, x'_m) \wedge (z_1, \dots, z_n) \in \mathcal{D}_g)$$

Most vizsgáljuk meg az első program ezen  $Q'$  utófeltételhez tartozó leggyengébb előfeltételét:

$$\begin{aligned} \text{If}(z_1, \dots, z_n := f(x_1, \dots, x_m), Q') &= (Q \wedge g(f(x_1, \dots, x_m))) = \\ &= ((g \circ f)(x'_1, \dots, x'_m) \wedge f(x_1, \dots, x_m) \in \mathcal{D}_g \wedge (x_1, \dots, x_m) \in \mathcal{D}_f) \end{aligned}$$

Könnyen látható, hogy ez a leggyengébb előfeltétel következik  $Q$ -ből, és így a szekvencia levezetési szabályának és a specifikáció tételének alkalmazásával beláttuk, hogy a

$z_1, \dots, z_n := f(x_1, \dots, x_m)$
$y_1, \dots, y_k := g(z_1, \dots, z_n)$

program megoldja a fent specifikált feladatot, azaz kiszámítja  $f$  és  $g$  kompozícióját.

**Direktszorzat.** Legyen  $k \in \mathbb{N}$  rögzített, és  $\forall i \in [1..k] : f_i \in \mathbb{N}^m \rightarrow \mathbb{N}^{n_i}$ . Ekkor az  $e$  függvények direktszorzata által meghatározott feladat specifikációja:

$$A = \mathbb{N} \times \dots \times \mathbb{N} \times \mathbb{N} \times \dots \times \mathbb{N} \times \dots \times \mathbb{N} \times \dots \times \mathbb{N}$$

$$\begin{array}{ccccccc} x_1 & & x_m & y_{11} & & y_{1n_1} & & y_{k1} & & y_{kn_k} \end{array}$$

$$B = \mathbb{N} \times \dots \times \mathbb{N}$$

$$\begin{array}{cc} x'_1 & x'_m \end{array}$$

$$Q : (x_1 = x'_1 \wedge \dots \wedge x_m = x'_m \wedge (x_1, \dots, x_m) \in \mathcal{D}_{(f_1, \dots, f_k)})$$

$$R : (Q \wedge ((y_{11}, \dots, y_{1n_1}), \dots, (y_{k1}, \dots, y_{kn_k})) = (f_1, \dots, f_k)(x'_1, \dots, x'_m))$$

Tegyük fel, hogy a komponensfüggvények kiszámíthatók jól konstruált programmal. Jelölje  $y_{i1}, \dots, y_{in_i} := f_i(x_1, \dots, x_m)$  az  $i$ -edik függvényt ( $1 \leq i \leq k$ ) kiszámító programot. Ekkor ezeknek a programoknak a szekvenciája megoldja a fenti feladatot. Legyen  $Q_k$  a  $k$ -adik program  $R$  utófeltételhez tartozó leggyengébb előfeltétele:

$$Q_k : (Q \wedge ((y_{11}, \dots, y_{1n_1}), \dots, (y_{k-11}, \dots, y_{k-1n_{k-1}}), f_k(x_1, \dots, x_m)) = (f_1, \dots, f_k)(x'_1, \dots, x'_m) \wedge (x_1, \dots, x_m) \in \mathcal{D}_{f_k})$$

Továbbá minden  $i \in [1..k-1]$  esetén jelölje  $Q_i$  az  $i$ -edik program  $Q_{i+1}$ -hez tartozó leggyengébb előfeltételét. Könnyen látható, hogy az értékadás leggyengébb előfeltételére vonatkozó szabályt alkalmazva:

$$Q_1 : (Q \wedge (f_1(x_1, \dots, x_m), \dots, f_k(x_1, \dots, x_m)) = (f_1, \dots, f_k)(x'_1, \dots, x'_m) \wedge (x_1, \dots, x_m) \in \mathcal{D}_{f_1} \wedge \dots \wedge (x_1, \dots, x_m) \in \mathcal{D}_{f_k})$$

Ha most  $Q$ -t és  $Q_1$ -et összehasonlítjuk, észrevehetjük, hogy megegyeznek. A szekvencia levezetési szabálya és a specifikáció tétele alapján az

$y_{11}, \dots, y_{1n_1} := f_1(x_1, \dots, x_m)$
$\vdots$
$y_{k1}, \dots, y_{kn_k} := f_k(x_1, \dots, x_m)$

program megoldása a fent specifikált feladatnak, azaz kiszámítja  $(f_1, \dots, f_k)$ -t.

**Rekurzió.** Legyen  $n$  rögzített,  $f \in \mathbb{N}^n \rightarrow \mathbb{N}$  és  $g \in \mathbb{N}^{n+2} \rightarrow \mathbb{N}$ . Tegyük fel, hogy mindketten kiszámíthatók jól konstruált programokkal. Az  $f$  függvény  $g$  szerinti rekurziója által meghatározott feladat specifikációja:

$$A = \mathbb{N} \times \dots \times \mathbb{N} \times \mathbb{N}$$

$$\begin{array}{ccc} x_1 & x_{n+1} & y \end{array}$$



$$B = \mathbb{N} \times \dots \times \mathbb{N}$$

$$x'_1 \quad x'_{n+1}$$

$$Q : (x_1 = x'_1 \wedge \dots \wedge x_{n+1} = x'_{n+1} \wedge (x_1, \dots, x_{n+1}) \in \mathcal{D}_{\varrho(f,g)})$$

$$R : (Q \wedge y = \varrho(f,g)(x'_1, \dots, x'_{n+1}))$$

Jelölje az  $f$ -et és a  $g$ -t kiszámító programot  $y := f(x_1, \dots, x_n)$ , illetve  $y := g(x_1, \dots, x_{n+2})$ . Oldjuk meg a feladatot egy olyan ciklussal, amelynek invariáns tulajdonsága:

$$P : (Q \wedge k \in [1..x_{n+1}] \wedge y = \varrho(f,g)(x'_1, \dots, x'_n, k))$$

A ciklus levezetési szabályát vizsgálva azt találjuk, hogy  $Q$ -ból nem következik  $P$ . Ezért adunk egy olyan  $Q'$  feltételt, amelyből már következik  $P$ , és adunk egy programot, amely  $Q$ -ból  $Q'$ -be jut (így a megoldóprogram egy szekvencia lesz, amelynek második része egy ciklus). Legyen  $Q'$  az alábbi:

$$Q' : (Q \wedge k = 1 \wedge y = f(x_1, \dots, x_n))$$

Ez a  $k, y := 1, f(x_1, \dots, x_n)$  szimultán értékadással elérhető. Az értékadás leggyengébb előfeltételére vonatkozó szabály felhasználásával könnyen látható, hogy az következik  $Q$ -ből.

A ciklus levezetési szabályának második pontja alapján a ciklusfeltétel  $k \neq x_{n+1}$  lesz.

A harmadik pontnak megfelelően válasszuk a  $t = x_{n+1} - k$  kifejezést terminálófüggvénynek.

Az ötödik pont azt írja le, hogy az imént definiált terminálófüggvény értékének a ciklusmagban csökkennie kell. Ez elérhető a  $k$  eggyel való növelésével.

A négyes pont kielégítéséhez vizsgáljuk meg, mi lesz a leggyengébb előfeltétele a  $k$ -t növelő értékadásnak a  $P$ -re vonatkozóan.

$$Q'' = lf(k := k + 1, P) = (Q \wedge k + 1 \in [1..x_{n+1}] \wedge y = \varrho(f,g)(x'_1, \dots, x'_n, k + 1))$$

Most már – a szekvencia levezetési szabálya alapján – csak egy olyan programot kell találnunk, amelyre  $P \wedge (k \neq x_{n+1}) \Rightarrow lf(S, Q'')$ . Ez a program a rekurzió definíciójához illeszkedően épp  $y := g(x_1, \dots, x_n, k, y)$  lesz.

Így a szekvencia és a ciklus levezetési szabálya, valamint a specifikáció tétele garantálja, hogy a

$k, y := 1, f(x_1, \dots, x_n)$
$k \neq x_{n+1}$
$y := g(x_1, \dots, x_n, k, y)$
$k := k + 1$

program megoldja a  $\varrho(f,g)$  által meghatározott feladatot, azaz kiszámítja  $f$   $g$  szerinti rekurzióját.

**$\mu$ -operátor.** Legyen  $f \in \mathbb{N}^{n+1} \rightarrow \mathbb{N}$ , és tegyük fel, hogy  $f$  kiszámítható egyszerű értékadással vagy jól konstruált programmal. Tekintsük a  $\mu(f)$  által meghatározott feladatot:

$$A = \mathbb{N} \times \dots \times \mathbb{N} \times \mathbb{N}$$

$$\quad \quad \quad x_1 \quad \quad \quad x_n \quad \quad y$$

$$B = \mathbb{N} \times \dots \times \mathbb{N}$$

$$\quad \quad \quad x'_1 \quad \quad \quad x'_n$$

$$Q : (x_1 = x'_1 \wedge \dots \wedge x_n = x'_n \wedge (x_1, \dots, x_{n+1}) \in \mathcal{D}_{\mu(f)})$$

$$R : (Q \wedge y = \mu(f)(x'_1, \dots, x'_n))$$

Jelölje  $z := f(x_1, \dots, x_n, x_{n+1})$  az  $f$ -et kiszámító programot. Oldjuk meg a feladatot egy olyan ciklussal, melynek invariánsa:

$$P : (Q \wedge z = f(x_1, \dots, x_n, y) \wedge \forall i \in [1..y-1] : f(x_1, \dots, x_n, i) \neq 1)$$

Az invariáns a  $z, y := f(x_1, \dots, x_n, 1), 1$  szimultán értékadással teljesíthető. Egyszerűen belátható, hogy ennek a programnak a  $P$ -re vonatkozó leggyengébb előfeltétele,

$$lf(z, y := f(x_1, \dots, x_n, 1), 1; P) = (Q \wedge f(x_1, \dots, x_n, 1) = f(x_1, \dots, x_n, 1) \\ \wedge \forall i \in [1..1-1] : f(x_1, \dots, x_n, i) \neq 1)$$

következik  $Q$ -ból. A ciklus levezetési szabályának második pontja alapján a ciklusfeltétel  $z \neq 1$  lesz.

A feladat előfeltétele garantálja, hogy van olyan  $m \in \mathbb{N}$  szám, amelyre  $f(x_1, \dots, x_n, m) = 1$  fennáll. Legyen  $N$  egy ilyen tulajdonságú, rögzített természetes szám. Ennek az értéknek a segítségével definiálhatjuk a terminálófüggvényt:  $t = N - y$ . Ez kielégíti a levezetési szabály harmadik pontját.

Az ötödik pont megkívánja, hogy a terminálófüggvény a ciklusmag lefutása során csökkenjen. Ezt elérhetjük  $y$  eggyel való növelésével.

A negyedik pont teljesítéséhez legyen  $Q'$  ennek a növelésnek a  $P$ -re vonatkozó leggyengébb előfeltétele:

$$Q' : (Q \wedge z = f(x_1, \dots, x_n, y+1) \wedge \forall i \in [1..y-1] : f(x_1, \dots, x_n, i) \neq 1)$$

Most már csak találnunk kell egy programot a  $P \wedge (z \neq 1)$  és  $Q'$  állítások közé. A  $z := f(x_1, \dots, x_n, y+1)$  értékadásra teljesül, hogy

$$P \wedge (z \neq 1) \Rightarrow lf(z := f(x_1, \dots, x_n, y+1), Q').$$

A specifikáció tétele, valamint a ciklus és a szekvencia levezetési szabálya garantálja, hogy a

$z, y := f(x_1, \dots, x_n, 1), 1$
$z \neq 1$
$z := f(x_1, \dots, x_n, y + 1)$
$y := y + 1$

program megoldja a  $\mu(f)$  által meghatározott feladatot, azaz kiszámítja  $\mu(f)$ -et.

**Következmény.** Az előzőekben megmutattuk, hogy ha az alapfüggvények kiszámíthatók, akkor a belőlük – a parciális rekurzív függvényeknél megengedett – operátorokkal felépített függvények is kiszámíthatók jól konstruált programokkal. Ennek alapján kimondhatjuk az alábbi tételt:

**9.1. TÉTEL: STRUKTURÁLT PROGRAMOZÁS ÉS KISZÁMÍTHATÓSÁG**

Minden kiszámítható függvény kiszámítható egy jól konstruált programmal.

### 9.3. Relációk

Fordítsuk most figyelmünket a relációk felé. Ahhoz, hogy a relációk kiszámíthatóságát megvizsgálhassuk, definiálnunk kell a kiszámítható reláció fogalmát.

**9.2. DEFINÍCIÓ: REKURZÍVAN FELSOROLHATÓ RELÁCIÓ**

Legyen  $R \subseteq \mathbb{N}^k \times \mathbb{N}^k$  tetszőleges reláció.  $R$  akkor és csak akkor *rekurzívan felsorolható*, ha van olyan  $f \in \mathbb{N}^{2k} \rightarrow \mathbb{N}$  parciális rekurzív függvény, amelynek értelmezési tartományára:  $\mathcal{D}_f = R$ .

A kiszámíthatóság-elmélethez igazodva a továbbiakban csak rekurzívan felsorolható relációkkal fogunk foglalkozni. Ahhoz, hogy megmutassuk, minden kiszámítható (rekurzívan felsorolható) feladat – emlékezzünk, hogy minden feladat egy reláció – megoldható strukturált programmal, először megadjuk a rekurzívan felsorolható relációk egy másik jellemzését.

**9.2. TÉTEL: KLEENE [1936]**

Ha  $R$  egy tetszőleges reláció, akkor az alábbi három állítás ekvivalens:

- (1)  $R$  rekurzívan felsorolható.
- (2)  $R$  egy  $f$  parciális rekurzív függvény értékészlete.
- (3)  $R = \emptyset$  vagy  $R$  egy  $\varphi$  rekurzív függvény értékészlete.

Ennek a tételnek a bizonyítása konstruktív, azaz megadja mind  $f$ , mind pedig  $\varphi$  felépítését. Mi ezt a  $\varphi$  függvényt fogjuk használni a kiszámítható feladatunk megoldóprogramjában.

Legyen  $F \subseteq \mathbb{N}^k \times \mathbb{N}^k$  egy rekurzívan felsorolható reláció, és jelölje  $\varphi$  az előző tétel konstrukciójával kapott (totális) rekurzív függvényt. Specifikáljuk a feladatot az alábbi módon:

$$A = \mathbb{N}^k \times \mathbb{N}^k$$

$$\quad \quad \quad x \quad y$$

$$B = \mathbb{N}^k$$

$$\quad \quad \quad x'$$

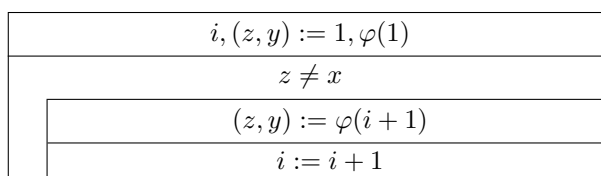
$$Q : (x = x' \wedge x \in \mathcal{D}_F)$$

$$R : (Q \wedge (x, y) \in F)$$

Ez a feladat megoldható egy olyan ciklussal, amelynek invariáns tulajdonsága:

$$P : (Q \wedge i \in \mathbb{N} \wedge (z, y) = \varphi(i))$$

A ciklus levezetési szabályának felhasználásával könnyen belátható, hogy az alábbi program megoldása a fenti feladatnak:



A bizonyításban a terminálófüggvény megadásánál ugyanazt a technikát alkalmazzuk, mint a  $\mu$ -operátornál.

Ezzel az előzőleg függvényekre vonatkozó tételünket általánosíthatjuk relációkra:

### 9.3. TÉTEL: STRUKTURÁLT PROGRAMOZÁS ÉS KISZÁMÍTHATÓ RELÁCIÓK

Minden kiszámítható reláció kiszámítható egy jól konstruált programmal.

Megjegyezzük, hogy ezek az eredmények kiterjeszthetők a relatíve kiszámítható függvények (ugyanezen operátorokkal egy tetszőleges alaphalmazból képzett függvények) vagy a parciális rekurzív funkcionálok halmazára is [Odi 98].

## 10. fejezet

# A típus

Az állapotér definíciójában szereplő halmazokat típusértékalmazoknak neveztük, és csak annyit mondtunk róluk, hogy legfeljebb megszámlálhatóak.

A továbbiakban arról lesz szó, hogy ezek a halmazok hogyan jönnek létre, milyen közös tulajdonság jellemző az elemeikre.

### 10.1. A típusspecifikáció

Először bevezetünk egy olyan fogalmat, amelyet arra használhatunk, hogy pontosan leírjuk a követelményeinket egy típusértékalmazzal és a rajta végezhető műveletekkel szemben.

#### 10.1. DEFINÍCIÓ: TÍPUSSPECIFIKÁCIÓ

A  $\mathcal{T}_s = (H, I_s, \mathbb{F})$  hármast *típus-specifikációnak* nevezzük, ha teljesülnek rá a következő feltételek:

1.  $H$  az alaphalmaz,
2.  $I_s : H \rightarrow \mathbb{L}$  a specifikációs invariáns,
3.  $T_{\mathcal{T}} = \{(T, x) \mid x \in [I_s]\}$  a *típusértékalmaz*,
4.  $\mathbb{F} = \{F_1, F_2, \dots, F_n\}$  a típusműveletek specifikációja, ahol  
 $\forall i \in [1..n] : F_i \subseteq A_i \times A_i, A_i = A_{i_1} \times \dots \times A_{i_{n_i}}$  úgy,  
hogy  $\exists j \in [1..n_i] : A_{i_j} = T_{\mathcal{T}}$ .

Az alaphalmaz és az invariáns tulajdonság segítségével azt fogalmazzuk meg, hogy mi az a halmaz,  $T_{\mathcal{T}}$ , amelynek elemeivel foglalkozni akarunk, míg a feladatok halmazával azt írjuk le, hogy ezekkel az elemekkel milyen műveletek végezhetőek el.

Az állapotér definíciójában szereplő típusértékhalmozok mind ilyen típus-specifikációban vannak definiálva. Az állapotér egy komponensét egy program csak a típusműveleteken keresztül változtathatja meg.

## 10.2. A típus

Vizsgáljuk meg, hogy a típus-specifikációban leírt követelményeket hogyan valósítjuk meg.

### 10.2. DEFINÍCIÓ: TÍPUS

A  $T = (\varrho, I, \mathbb{S})$  hármast *típusnak* nevezzük, ha az alábbi feltételek teljesülnek rá:

1.  $\varrho \subseteq E^* \times T$  a reprezentációs függvény (reláció),  
 $T$  a típusértékhalmoz,  
 $E$  az elemi típusértékhalmoz,
2.  $I : E^* \rightarrow \mathbb{L}$  típusinvariáns,
3.  $\mathbb{S} = \{S_1, S_2, \dots, S_m\}$ , ahol  
 $\forall i \in [1..m] : S_i \subseteq B_i \times B_i^{**}$  program,  $B_i = B_{i_1} \times \dots \times B_{i_{m_i}}$  úgy,  
 hogy  $\exists j \in [1..m_i] : B_{i_j} = E^*$  és  $\nexists j \in [1..m_i] : B_{i_j} = T$ .

A típus első két komponense az absztrakt típusértékek reprezentációját írja le, míg a programhalmoz a típusműveletek implementációját tartalmazza. Az elemi típusértékhalmoz lehet egy tetszőleges másik típus típusértékhalmoza vagy egy, valamilyen módon definiált, legfeljebb megszámlálható halmoz.

*Elemi típusnak* nevezünk egy  $T = (\varrho, I, \mathbb{S})$  típust, ha  $T = E$ , és  $\varrho|_{\Gamma I} = Id_E$ .

Meg kell még adnunk, hogy egy típus mikor felel meg a típus-specifikációnak, azaz mikor teljesíti a specifikációban leírt követelményeket.

Legyen a továbbiakban  $\mathcal{T}_s = (H, I_s, \mathbb{F})$ ,  $T = (\varrho, I, \mathbb{S})$ ,  $F \in \mathbb{F}$ , és  $S \in \mathbb{S}$ ,  
 $F \subseteq A \times A$ ,  $A = A_1 \times \dots \times A_p$ ,  $S \subseteq B \times B^{**}$ ,  $B = B_1 \times \dots \times B_q$ .

Legyen  $A$  altere  $C$ -nek és  $B$  altere  $D$ -nek.

Azt mondjuk, hogy a  $C = C_1 \times \dots \times C_r$  és  $D = D_1 \times \dots \times D_r$  állapotterek *illeszkednek*, ha

$$\forall i \in [1..r] : D_i = \begin{cases} E^*, & \text{ha } C_i = T_{\mathcal{T}}; \\ C_i & \text{különben.} \end{cases}$$

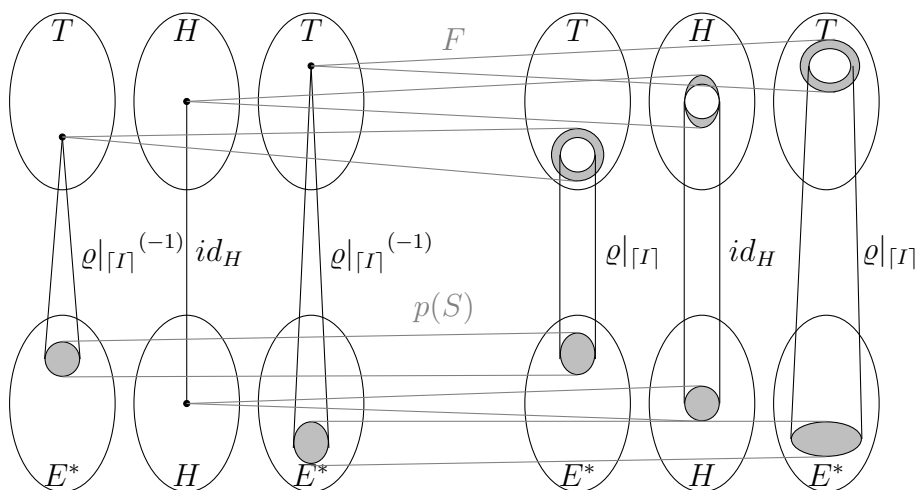
Ebben az esetben legyen a  $\gamma \subseteq D \times C$  reláció a következő:

$$\forall d \in D : \gamma(d) = \gamma_1(d_1) \times \gamma_2(d_2) \times \dots \times \gamma_r(d_r),$$

ahol  $\forall i \in [1..r] : \gamma_i \subseteq D_i \times C_i$  és

$$\gamma_i = \begin{cases} \varrho|_{\Gamma I}, & \text{ha } C_i = T_{\mathcal{T}}; \\ id_{D_i} & \text{különben.} \end{cases}$$

Vegyük észre, hogy a  $\gamma$  tulajdonképpen a  $\varrho$  egyfajta kiterjesztése több komponensű, de egymáshoz illeszkedő állapotterek esetére.



10.1. ábra. A  $\varrho$ -n keresztüli megoldás illeszkedő állapotterek között

### 10.3. DEFINÍCIÓ: MEGOLDÁS $\varrho$ -N KERESZTÜL

Azt mondjuk, hogy az  $S \subseteq B \times B^{**}$  program a  $\varrho$ -n keresztül *megoldja* az  $F \subseteq A \times A$  feladatot, ha vannak olyan  $C$  és  $D$  illeszkedő terek, amelynek altere  $A$ , illetve  $B$ , hogy  $S'$   $\gamma$  szerint megoldása  $F'$ -nek, ahol  $\gamma \subseteq D \times C$  a fenti értelemben definiált leképezés,  $S'$  az  $S$  kiterjesztése  $D$ -re,  $F'$  pedig az  $F$  kiterjesztése  $C$ -re.

A kiterjesztési tételek szerint, ha vannak a definíciónak megfelelő illeszkedő terek, akkor mindegyik ilyen.

### 10.4. DEFINÍCIÓ: MEGFELELÉS

Egy  $\mathcal{T} = (\varrho, I, \mathbb{S})$  típus *megfelel* a  $\mathcal{T}_s = (H, I_s, \mathbb{F})$  típusspecifikációnak, ha

1.  $\varrho(\lceil I \rceil) = T_{\mathcal{T}}$ ,
2.  $\forall F \in \mathbb{F} : \exists S \in \mathbb{S} : S$  a  $\varrho$ -n keresztül megoldja  $F$ -et.

Természetesen egy típusspecifikációnak több különböző típus is megfelelhet. Ekkor az, hogy melyiket választjuk, a reprezentáció és a műveletek implementációinak további tulajdonságaitól – ilyen például a reprezentáció memóriaiigénye vagy az implementációk műveletigénye – függ. Ez a döntés mindig a megoldandó programozási feladat függvénye.

### 10.3. A típusspecifikáció tétele

A típusspecifikáció és a típus fogalmának bevezetésével tulajdonképpen a feladat fogalmát általánosítottuk, míg a megfeleltetés a megoldás fogalmának volt egyfajta általánosítása. A specifikáció tétele a megoldásra adott elégséges feltételt. Most a  $\varrho$ -n keresztüli megoldásra adunk egy hasonló feltételt.

#### 10.1. TÉTEL: TÍPUSSPECIFIKÁCIÓ TÉTELE

Legyen  $\mathcal{T}_s = (H, I_s, \mathbb{F})$  és  $\mathcal{T} = (\varrho, I, \mathbb{S})$  adott típusspecifikáció és típus, és tegyük fel, hogy a reprezentáció helyes, azaz  $\varrho(\llbracket I \rrbracket) = T_{\mathcal{T}}$ . Legyen továbbá  $F \in \mathbb{F}$ , az  $F$  állapotterere  $A$ , egy paraméterterere  $B$ , elő- és utófeltétele pedig  $Q_b$  és  $R_b$ . Legyen  $S \in \mathbb{S}$ , és tegyük fel, hogy  $S$  állapotterere illeszkedik  $F$  állapotteréhez. Definiáljuk a következő állításokat:

$$\begin{aligned} \llbracket Q_b^\gamma \rrbracket &= \llbracket Q_b \circ \gamma \rrbracket, \\ \llbracket R_b^\gamma \rrbracket &= \llbracket R_b \circ \gamma \rrbracket, \end{aligned}$$

ahol  $\gamma$  a program és a feladat állapotterere közötti, a  $\varrho$ -n keresztüli megoldás definíciójában szereplő leképezés. Ekkor ha  $\forall b \in B : Q_b^\gamma \Rightarrow lf(S, R_b^\gamma)$ , akkor az  $S$  program a  $\varrho$ -n keresztül megoldja az  $F$  feladatot.

**Bizonyítás:** A reláció szerinti megoldás tétele miatt csak azt kell belátnunk hogy  $\mathcal{D}_F \subseteq \mathcal{R}_\gamma$  teljesül. Ez pedig nyilvánvaló, mivel föltettük, hogy  $\varrho(\llbracket I \rrbracket) = T_{\mathcal{T}}$ .

□

Az, hogy a fenti tétel feltételei között kikötöttük, hogy a program állapotterere illeszkedik a feladat állapotteréhez, tulajdonképpen elhagyható. Ekkor a tétel a feladat és a program olyan kiterjesztéseire mondható ki, amelyek állapotterei illeszkednek egymáshoz (pontosan úgy, ahogy a megfeleltetést definiáltuk nem illeszkedő állapotterek között).

### 10.4. Absztrakt típus

Ebben a részben a megfelelés fogalmának általánosításával foglalkozunk. Először definiáljuk az ekvivalens feladat fogalmát. Ez az 5. fejezet alapján eléggé kézenfekvő. Az 5.2.-ben bevezetett jelöléseket használjuk.

#### 10.5. DEFINÍCIÓ: FELADATOK EKVIVALENCIÁJA

Az  $F_1 \subseteq A \times A$  és az  $F_2 \subseteq B \times B$  feladatot *ekvivalensnek* nevezzük, és  $F_1 F_2$ -vel jelöljük, ha  $A \stackrel{f}{\sim} B$  és  $\gamma_f(F_2) = F_1$ .

Most megadjuk két típusspecifikáció ekvivalenciájának definícióját. Két típusspecifikációt ekvivalensnek nevezünk, ha lényegében csak a nevükben különböznek, azaz:



**10.6. DEFINÍCIÓ: TÍPUSSPECIFIKÁCIÓK EKVIVALENCIÁJA**

A  $\mathcal{T}_{s_1} = (H_1, I_{s_1}, \mathbb{F}_1)$  és a  $\mathcal{T}_{s_2} = (H_2, I_{s_2}, \mathbb{F}_2)$  *típus-specifikáció ekvivalens*, ha  $\lceil I_{s_1} \rceil = \lceil I_{s_2} \rceil$ , és létezik  $\xi : \mathbb{F}_2 \rightarrow \mathbb{F}_1$  bijekció, hogy  $\forall F \in \mathbb{F}_2 : \xi(F) \sim F^{\mathcal{T}_2 \leftarrow \mathcal{T}_1}$ , ahol

$$F^{\mathcal{T}_2 \leftarrow \mathcal{T}_1} = \{(x^{\mathcal{T}_2 \leftarrow \mathcal{T}_1}, y^{\mathcal{T}_2 \leftarrow \mathcal{T}_1}) \mid (x, y) \in F\}, \text{ ahol}$$

$$x^{\mathcal{T}_2 \leftarrow \mathcal{T}_1} = \{(i, a^{\mathcal{T}_2 \leftarrow \mathcal{T}_1}) \mid (i, a) \in x\}, \text{ ahol}$$

$$a^{\mathcal{T}_2 \leftarrow \mathcal{T}_1} = \begin{cases} (\mathcal{T}_1, e), & \text{ha } a = (\mathcal{T}_2, e); \\ a & \text{különben.} \end{cases}$$

Ezután a megfelelés fogalmát általánosítjuk.

**10.7. DEFINÍCIÓ: MEGFELELÉS ÁLTALÁNOSÍTÁSA**

Azt mondjuk, hogy a  $\mathcal{T} = (\varrho, I, \mathbb{S})$  típus – általános értelemben – *megfelel* a  $\mathcal{T}_s = (H, I_s, \mathbb{F})$  típus-specifikációnak, ha létezik olyan ekvivalens típus-specifikáció, amelynek az eredeti értelemben megfelel.

A típusérték-halmazokat, mint párok halmazait definiáltuk. A fenti definíció értelmében azonban, a megfelelés szempontjából, az első komponensnek – a névnek – nincs jelentősége, ezért azzal a továbbiakban nem is foglalkozunk.

A programfüggvény általánosításaként definiálhatjuk az absztrakt típust:

**10.8. DEFINÍCIÓ: ABSZTRAKT TÍPUS**

Legyen  $\mathcal{T} = (\varrho, I, \mathbb{S})$  egy típus. A  $p(\mathcal{T}) = (p(\varrho), p(\mathbb{S}))$ -t  *$\mathcal{T}$  absztrakt típusának* nevezzük, ha

1.  $\forall \varepsilon \in E^* : p(\varrho)(\varepsilon) = \varrho|_{\lceil I \rceil}(\varepsilon)_2$ ,
2.  $p(\mathbb{S}) = \{p(S) \mid S \in \mathbb{S}\}$ .

A programokhoz hasonlóan a  $\mathcal{T}_1 = (\varrho_1, I_1, \mathbb{S}_1)$  és a  $\mathcal{T}_2 = (\varrho_2, I_2, \mathbb{S}_2)$  típust *ekvivalensnek* nevezzük, ha  $p(\mathcal{T}_1) = p(\mathcal{T}_2)$ . Az elnevezés azért indokolt, mert a megfelelés szempontjából az ekvivalens típusok egyformák.

**10.5. Példák**

**10.1. példa:** A típusértékek halmaza legyen a magyar ábécé magánhangzói: { a, á, e, é, i, í, o, ó, ö, ő, u, ú, ü, ű }. Szeretnénk tudni, hogy egy adott magánhangzónak melyik a (rövid, illetve hosszú) párja. Legyen egy olyan típusműveletünk, amely erre a kérdésre választ tud adni. Az elemi típusértékek halmaza legyen a { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14 } halmaz. Adjunk típus-specifikációt, és készítsünk el egy olyan típust, amely megfelel a specifikációnak!

**Megoldás:** Írjuk fel először a típus-specifikációt! Legyen a magánhangzók halmaza *MGH*. Ekkor

$$\mathcal{T}_s = (MGH, I_{gaz}, \{F\}), \text{ ahol } F \subseteq T_{\mathcal{T}} \times T_{\mathcal{T}},$$

$$F = \{ (a, \acute{a}), (\acute{a}, a), (e, \acute{e}), (\acute{e}, e), (i, \acute{i}), (\acute{i}, i), (o, \acute{o}), (\acute{o}, o), (\acute{o}, \delta), (\delta, \acute{o}), (u, \acute{u}), (\acute{u}, u), (\acute{u}, \acute{u}), (\acute{u}, \acute{u}) \}.$$

Felhívjuk a figyelmet arra, hogy  $T_{\mathcal{T}}$  elemeit az egyszerűség kedvéért  $(\mathcal{T}, x)$  helyett  $x$ -el jelöltük. Továbbiakban is ezt a megoldást fogjuk alkalmazni. Adjuk meg a típust!

$$\mathcal{T} = (\varrho, I, \{S\}), \text{ ahol } \varrho \subseteq E^* \times MGH,$$

$$\varrho = \{ (\langle 0 \rangle, a), (\langle 14 \rangle, \acute{a}), (\langle 1 \rangle, e), (\langle 13 \rangle, \acute{e}), (\langle 2 \rangle, i), (\langle 12 \rangle, \acute{i}), (\langle 3 \rangle, o), (\langle 11 \rangle, \acute{o}), (\langle 4 \rangle, \delta), (\langle 10 \rangle, \acute{\delta}), (\langle 5 \rangle, u), (\langle 9 \rangle, \acute{u}), (\langle 6 \rangle, \acute{u}), (\langle 8 \rangle, \acute{u}) \};$$

$\forall \alpha \in E^*$  :

$$I(\alpha) = \begin{cases} igaz, & \text{ha } |\alpha| = 1 \text{ és } \alpha_1 \neq 7; \\ hamis & \text{egyébként.} \end{cases}$$

$$S \subseteq E^* \times (E^*)^{**},$$

$$S = \{ (\langle i \rangle, red(\langle i \rangle, \langle 14 - i \rangle)) \mid i \in E \} \cup \{ (\alpha, \langle \alpha, \alpha, \dots \rangle) \mid |\alpha| \neq 1 \}.$$

Az, hogy a most megadott típus megfelel a fenti típuspecifikációnak, könnyen látható: a reprezentáció helyessége a  $\varrho$  és az  $I$  definíciójából leolvasható, míg az, hogy az  $S$  program a  $\varrho$ -n keresztül megoldja az  $F$  feladatot, a program egyszerű hozzárendeléséből és a  $\varrho$  „trükkös” megválasztásából látszik.

Természetesen másmilyen reprezentációs függvényt is meg lehet adni, de ekkor meg kell változtatnunk a típusinvariánst és a programot is.

**10.2. példa:** Specifikálja azt a típust, melynek értékei a  $[0..127]$  halmaz részhalmazai, típusműveletei pedig két részhalmaz metszetének és uniójának képzése, illetve annak megállapítása, hogy egy elem eleme-e egy részhalmaznak. Adjon meg egy típust, amely megfelel a specifikációnak! (Az elemi értékek halmaza:  $\{0, 1\}$ , a programokat elég a programfüggvényükkel megadni.)

**Megoldás:**  $\mathcal{T}_s = (H, I_s, \mathbb{F})$ , ahol

$$\begin{aligned} H &= \wp([0..127]), \\ I_s &= Igaz, \\ \mathbb{F} &= \{F_m, F_u, F_e\}, \end{aligned}$$

$$\text{és } A_m = T \times T \times T, F_m \subseteq A_m \times A_m,$$

$$F_m = \{((a, b, c), (p, q, r)) \mid p = a \text{ és } q = b \text{ és } r = a \cap b\};$$

$$A_u = T \times T \times T, F_u \subseteq A_u \times A_u,$$

$$F_u = \{((a, b, c), (p, q, r)) \mid p = a \text{ és } q = b \text{ és } r = a \cup b\};$$

$$A_e = T \times [0..127] \times \mathbb{L}, F_e \subseteq A_e \times A_e,$$

$$F_e = \{((h, e, l), (h', e', l')) \mid h = h' \text{ és } e = e' \text{ és } l' = (e \in h)\}.$$

Adjunk a fenti specifikációnak megfelelő típust!  $T = (\varrho, I, \mathbb{S})$ , és  $\varrho \subseteq E^* \times \wp(\mathbb{N})$ ,  $\forall \alpha \in E^*$  :

$$\varrho(\alpha) = \{\{i \mid \alpha_{i+1} = 1\}\};$$

$$I : E^* \rightarrow \mathbb{L}, \forall \alpha \in E^* :$$

$$I(\alpha) = \begin{cases} igaz, & \text{ha } |\alpha| = 128; \\ hamis & \text{egyébként.} \end{cases}$$

$\mathbb{S} = \{S_m, S_u, S_e\}$ , és  $B_m = E^* \times E^* \times E^*$ ,  $S_m \subseteq B_m \times B_m$  program:

$$p(S_m) = \{((\alpha, \beta, \gamma), (\alpha', \beta', \gamma')) \mid \alpha \in [I] \text{ és } \beta \in [I] \text{ és } \gamma' \in [I] \text{ és } \alpha = \alpha' \text{ és } \beta = \beta' \text{ és } \forall i \in [1..128] : \gamma'_i = \alpha_i \cdot \beta_i\};$$

$B_u = E^* \times E^* \times E^*$ ,  $S_u \subseteq B_u \times B_u$  program:

$$p(S_u) = \{((\alpha, \beta, \gamma), (\alpha', \beta', \gamma')) \mid \alpha \in [I] \text{ és } \beta \in [I] \text{ és } \gamma' \in [I] \text{ és } \alpha = \alpha' \text{ és } \beta = \beta' \text{ és } \forall i \in [1..128] : \gamma'_i = \alpha_i + \beta_i - \alpha_i \cdot \beta_i\};$$

$B_e = E^* \times [0..127] \times \mathbb{L}$ ,  $S_e \subseteq B_e \times B_e$  program:

$$p(S_e) = \{((\alpha, x, l), (\alpha', x', l')) \mid \alpha \in [I] \text{ és } \alpha = \alpha' \text{ és } x = x' \text{ és } l' = (\alpha_{x+1} = 1)\}.$$

Vajon megfelel a most leírt típus a fenti típusspecifikációnak? A reprezentáció helyes, ugyanis a pontosan 128 hosszú sorozatokat a reprezentációs függvény éppen a kívánt részhalmazokba képezi le, azaz

$$\varrho([I]) = [I_s].$$

Vizsgáljuk meg a programok és a feladatok viszonyát. Vegyük észre, hogy a programok állapotterei illeszkednek a megfelelő feladat állapotteréhez, tehát felírható közöttük a  $\gamma$  reláció.

$$\begin{aligned} \gamma_m &= (\varrho|_{[I]}; \varrho|_{[I]}; \varrho|_{[I]}), \\ \gamma_u &= (\varrho|_{[I]}; \varrho|_{[I]}; \varrho|_{[I]}), \\ \gamma_e &= (\varrho|_{[I]}; id_{[0..127]}; id_{\mathbb{L}}). \end{aligned}$$

Ezek felhasználásával a  $\varrho$ -n keresztüli megoldás egyszerűen adódik a reprezentációs függvény és a programfüggvények szemantikájából.

**10.3. példa:** Legyen  $\mathcal{T}_s = (H, I_s, \mathbb{F})$  egy típusspecifikáció,  $\mathbb{F} = \{F\}$ . Legyenek  $\mathcal{T}_1 = (\varrho_1, I_1, \mathbb{S}_1)$  és  $\mathcal{T}_2 = (\varrho_2, I_2, \mathbb{S}_2)$  típusok, melyekre:  $\mathbb{S}_1 = \{S_1\}$ ,  $\mathbb{S}_2 = \{S_2\}$ ,  $\varrho_1 = \varrho_2$ ,  $[I_1] = [I_2]$ , és  $S_2 \subseteq S_1$ .

Igaz-e, hogy ha  $\mathcal{T}_1$  megfelel  $\mathcal{T}_s$ -nek, akkor  $\mathcal{T}_2$  is?

**Megoldás:** A reprezentáció helyessége  $\varrho_1 = \varrho_2$  és  $[I_1] = [I_2]$  miatt triviálisan teljesül, hiszen ekkor:

$$\varrho_2([I_2]) = \varrho_1([I_1]) = [I_s].$$

Azt kell tehát megvizsgálnunk, hogy vajon az  $S_2$  program megoldja-e az  $F$  feladatot a  $\varrho_2$ -n keresztül. Mivel a programok állapottere közös, feltehetjük, hogy a programok állapottere és a feladat állapottere egymásnak megfeleltethető, hiszen ellenkező esetben mindkét megoldás-vizsgálatnál a feladatnak ugyanazt a kiterjesztését kellene használnunk, és így az eredeti feladatot ezzel a kiterjesztéssel helyettesítve az alábbi gondolatmenet végigvihető.

Mivel  $S_2 \subseteq S_1$ , a két program programfüggvényére teljesül a következő:

- i.*  $\mathcal{D}_{p(S_1)} \subseteq \mathcal{D}_{p(S_2)}$ ,
- ii.*  $\forall a \in \mathcal{D}_{p(S_1)} : p(S_2)(a) \subseteq p(S_1)(a)$ .

Jelöljük most is  $\gamma$ -val a program és a feladat állapottere közötti, a megfeleltetésben definiált leképezést. Könnyen látható, hogy az *i.* tulajdonság miatt

$$\mathcal{D}_{\gamma \circ p(S_1) \circ \gamma^{(-1)}} \subseteq \mathcal{D}_{\gamma \circ p(S_2) \circ \gamma^{(-1)}}.$$

Másrészt mivel az  $S_1$  program megoldja  $F$ -et a  $\varrho_1$ -n keresztül,

$$\mathcal{D}_F \subseteq \mathcal{D}_{\gamma \circ p(S_1) \circ \gamma^{(-1)}}$$

is teljesül. A fenti két állítás alapján

$$\mathcal{D}_F \subseteq \mathcal{D}_{\gamma \circ p(S_2) \circ \gamma^{(-1)}}.$$

Használjuk fel a második tulajdonságot is! Az *ii.* tulajdonság miatt igaz az alábbi állítás is:

$$\forall a \in \mathcal{D}_{\gamma \circ p(S_1) \circ \gamma^{(-1)}} : \gamma \circ p(S_2) \circ \gamma^{(-1)}(a) \subseteq \gamma \circ p(S_1) \circ \gamma^{(-1)}(a).$$

Ekkor viszont, mivel az  $S_1$  program – a  $\varrho_1$ -n keresztül – megoldása a feladatnak, teljesül, hogy

$$\forall a \in \mathcal{D}_F : \gamma \circ p(S_1) \circ \gamma^{(-1)}(a) \subseteq F(a),$$

és ezért

$$\forall a \in \mathcal{D}_F : \gamma \circ p(S_2) \circ \gamma^{(-1)}(a) \subseteq F(a),$$

azaz az  $S_2$  program is megoldja az  $F$  feladatot a  $\varrho_2$ -n keresztül, tehát a  $\mathcal{T}_2$  típus is megfelel a specifikációnak.

## 10.6. Feladatok

**10.1.** Adjunk típusspecifikációt, reprezentációs függvényt, típust (amely megfelel a specifikációnak). A lehetséges értékek:  $[0..99999]$ . A műveletek: a következő és az előző 100000 szerinti maradékkal. Az elemi értékek a decimális számjegyek:  $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ . Mutassa meg, hogy a típus megfelel a típusspecifikációnak!

**10.2.**  $E = \{0, 1, 2\}, T = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 0\}, \mathbb{F} = \{F\}$ .

$$F = \{((a, b, c), (d, e, f)) \mid \exists k \in \mathbb{Z} : f + k \cdot 10 = a + b\}$$

Készítsen egy olyan típust, amely megfelel a specifikációnak!

**10.3.** Legyen  $\mathcal{T}_{s_1} = (H_1, I_{s_1}, \mathbb{F}_1), \mathcal{T}_{s_2} = (H_2, I_{s_2}, \mathbb{F}_2)$  két típusspecifikáció.

1. állítás: Minden  $\mathcal{T}$  típusra:  $\mathcal{T}$  megfelel  $\mathcal{T}_{s_1}$ -nek  $\Leftrightarrow \mathcal{T}$  megfelel  $\mathcal{T}_{s_2}$ -nek.
2. állítás:  $[I_{s_1}] = [I_{s_2}]$  és  $\mathbb{F}_1 = \mathbb{F}_2$ .

Ekvivalens-e a két állítás?

**10.4.** Adott a  $\mathcal{T}_s = (H, I_s, \mathbb{F})$  típusspecifikáció, továbbá adottak a  $\mathcal{T}_1 = (\varrho_1, I_1, \mathbb{S}_1), \mathcal{T}_2 = (\varrho_2, I_2, \mathbb{S}_2)$  típusok. Tegyük fel, hogy  $[I_1] = [I_2], \mathbb{S}_1 = \mathbb{S}_2$  és  $\varrho_1([I_1]) = \varrho_2([I_2])$ , és  $\forall \alpha \in E^* : \varrho_2(\alpha) \subseteq \varrho_1(\alpha)$ , valamint  $\mathcal{T}_1$  feleljen meg  $\mathcal{T}_s$ -nek!

Igaz-e, hogy  $\mathcal{T}_2$  is megfelel  $\mathcal{T}_s$ -nek?

**10.5.** Legyen  $\mathcal{T}_s = (H, I_s, \mathbb{F})$  egy típusspecifikáció,  $\mathcal{T}_1 = (\varrho_1, I_1, \mathbb{S}_1), \mathcal{T}_2 = (\varrho_2, I_2, \mathbb{S}_2)$ . Legyen  $[I_2] \subseteq [I_1], \mathbb{S}_1 = \mathbb{S}_2$  és  $\varrho_1([I_1]) = \varrho_2([I_2])$ , és  $\forall \alpha \in E^* : \varrho_2(\alpha) = \varrho_1(\alpha)$ , valamint  $\mathcal{T}_1$  feleljen meg  $\mathcal{T}_s$ -nek.

Igaz-e, hogy  $\mathcal{T}_2$  is megfelel  $\mathcal{T}_s$ -nek?

**10.6.** Legyen  $\mathcal{T}_S = (T, I_S, \mathbb{F})$  a következő típusspecifikáció:

$I_S = \text{Igaz}, T_2 = \mathbb{N}_0, \mathbb{F} = \{F_1, F_2\}$ .

$F_1$  specifikációja:

$$A = T$$

$$x$$

$$B = T$$

$$x'$$

$$Q : (x = x')$$

$$R : (\exists z \in \mathbb{Z} : x'_2 = 8 \cdot z + x_2 \text{ és } 0 \leq x_2 < 8)$$

$F_2$  specifikációja:

$$A = \begin{matrix} T & \times & T & \times & \mathbb{L} \\ x & & y & & l \end{matrix}$$

$$B = \begin{matrix} T & \times & T \\ x' & & y' \end{matrix}$$

$$Q : (x = x' \wedge y = y')$$

$$R : (l = (x' = y') \wedge x = x' \wedge y = y')$$

$$T = (\varrho, I, \S), E = \{0, 1, 2, 3, 4, 5, 6, 7\},$$

$$\forall e \in E^* : \varrho(e) = \left\{ (T, \sum_{i=1}^{|e|} (e_i \cdot 8^{|e|-i})) \right\}$$

a)

$$I(e) = \begin{cases} igaz, & \text{ha } |e| \geq 1 \text{ és } (e_1 = 0 \Rightarrow |e| = 1); \\ hamis & \text{egyébként.} \end{cases}$$

b)

$$I(e) = \begin{cases} igaz, & \text{ha } |e| \geq 1; \\ hamis & \text{egyébként.} \end{cases}$$

$$S_1 \subseteq (E^*) \times (E^*)^{**}$$

$$\forall e \in E^* : S_1(e) = \left\{ \alpha \in (E^*)^* \mid \begin{array}{l} |\alpha| = |e| \text{ és} \\ \forall i \in [1..|\alpha|] : |\alpha_i| = |\alpha| - i + 1 \text{ és} \\ \forall i \in [2..|\alpha|] : \forall j \in [1..|\alpha_i|] : \alpha_{i,j} = \alpha_{i-1,j+1} \end{array} \right\}$$

$$S_2 \subseteq (E^* \times E^* \times \mathbb{L}) \times (E^* \times E^* \times \mathbb{L})^{**}$$

$$\forall e, d \in E^* : \forall l \in \mathbb{L} :$$

$$S_2(e, d, l) = \left\{ \beta \in (E^* \times E^* \times \mathbb{L}) \times (E^* \times E^* \times \mathbb{L})^{**} \mid \begin{array}{l} |\beta| = \min(|e|, |d|) + 1 \text{ és} \\ \forall i \in [2..|\beta|] : \beta_i = (ee, dd, ll) \text{ és} \\ ll = (\forall j \in [1..i-1] : ee_j = dd_j) \text{ és} \\ |ee| = i-1 \text{ és } |dd| = i-1 \text{ és} \\ \forall j \in [1..i-1] : (ee_{i-j} = e_{|e|-j+1} \text{ és } dd_{i-j} = d_{|d|-j+1}) \end{array} \right\}$$

Írja le szavakkal az  $F_1, F_2$  feladatot, a  $\varrho$  relációt és az  $S_1, S_2$  programfüggvényét! Megfelel-e a típus a specifikációnak az a), illetve a b) esetben?

**10.7.** Adjunk típusspecifikációt, reprezentációs függvényt, absztrakt típust (amely megfelel a specifikációnak)! A típusértékek: a síkvektorok halmaza, a műveletek: két vektor összeadása, valamint annak eldöntése, hogy két vektor számszorosa-e egymásnak.

**10.8.** Adjunk típusspecifikációt, reprezentációs függvényt, absztrakt típust (amely megfelel a specifikációnak)! A típusértékek: a térvektorok halmaza, a műveletek: két vektor kivonása, valamint egy vektornak egy számmal való szorzása.

- 10.9.** Adjunk típusspecifikációt, reprezentációs függvényt, absztrakt típust (amely megfelel a specifikációnak) a komplex számok típusára, ahol a műveletek két komplex szám összeadása és egy komplex szám képzetes részének meghatározása.
- 10.10.** Adjunk típusspecifikációt, reprezentációs függvényt, absztrakt típust (amely megfelel a specifikációnak) a komplex számok típusára, ahol a műveletek két komplex szám összeszorozása és egy komplex szám  $n$ -dik ( $n \in \mathbb{N}$ ) hatványának meghatározása.
- 10.11.** Adjunk típusspecifikációt, reprezentációs függvényt, absztrakt típust (amely megfelel a specifikációnak). A típusértékek a körlemez halmaza, a műveletek: egy körlemez eltolása és annak eldöntése, hogy egy síkbeli pont rajta van-e a körlemezen.
- 10.12.** Adjunk típusspecifikációt, reprezentációs függvényt, absztrakt típust (amely megfelel a specifikációnak). A típusértékek: a gömbök halmaza, a műveletek: egy gömb eltolása és annak eldöntése, hogy egy térbeli pont benne van-e a gömbben.
- 10.13.** Adjunk típusspecifikációt, reprezentációs függvényt, absztrakt típust (amely megfelel a specifikációnak). A típusértékek: a négyzetek halmaza, a műveletek: egy négyzet eltolása, egy négyzet méretének megváltoztatása, egy négyzet területének kiszámítása és annak eldöntése, hogy egy síkbeli pont rajta van-e a négyzeten.
- 10.14.** Adjunk típusspecifikációt, reprezentációs függvényt, absztrakt típust (amely megfelel a specifikációnak). A típusértékek: a kockák halmaza, a műveletek: egy kocka eltolása, egy kocka méretének megváltoztatása, egy kocka térfogatának kiszámítása és annak eldöntése, hogy egy térbeli pont benne van-e a kockában.





## 11. fejezet

# Típuskonstrukciók

Azzal már foglalkoztunk, hogy milyen lehetőségeink vannak meglévő programokból újak készítésére. A továbbiakban azt fogjuk megvizsgálni, hogyan használhatunk fel meglévő típusokat új típusok létrehozására. Ezeket a módszereket típuskonstrukciós módszereknek, az általuk megkapható típusokat típuskonstrukcióknak nevezzük.

A legkézenfekvőbb lehetőség új típus létrehozására, hogy egy adott típus műveleteinek halmazát megváltoztatjuk, új műveletet veszünk hozzá, elhagyunk belőle, megváltoztatunk meglévő műveletet. Ez egy egyszerű és nagyon gyakran alkalmazott eljárás.

A másik, egy kicsit bonyolultabb lehetőség az invariáns tulajdonság megváltoztatása. Az invariáns szűkítése könnyen kezelhető és jól használható lehetőség.

A harmadik lehetőség a reprezentációs függvény megváltoztatása. Ez is egy fontos lehetőség. A továbbiakban ennek általánosabb esetével fogunk foglalkozni, meglévő reprezentációs függvényekből állítunk elő újakat.

### 11.1. A megengedett konstrukciók

Természetesen sokféle lehetőségünk van meglévő reprezentációs függvényekből újat csinálni, de mi a továbbiakban csak három speciális konstrukcióval foglalkozunk: a direktszorzattal, az unióval és az iterálttal. Ezeket fogjuk megengedett típuskonstrukcióknak nevezni.

Az első típuskonstrukciós módszer, amivel megismerkedünk, a direktszorzat. Legyenek  $T_i = (\rho_i, I_i, \mathbb{S}_i)$  ( $i = 1, 2, \dots, n$ ) típusok, és jelöljük  $T_1, T_2, \dots, T_n$ -nel a nekik megfelelő típusérték-halmazokat,  $E_1, E_2, \dots, E_n$ -nel pedig a hozzájuk tartozó elemi típusérték-halmazokat, és vezessük be az  $E = E_1 \cup E_2 \cup \dots \cup E_n$  és  $B = T_1 \times T_2 \times \dots \times T_n$  jelölést.

**11.1. DEFINÍCIÓ: DIREKTSZORZAT**

A  $\mathcal{T} = (\varrho, I, \mathbb{S})$  típus *direktszorzata* a  $\mathcal{T}_1, \mathcal{T}_2, \dots, \mathcal{T}_n$  típusoknak, ha

$$\varrho = \varphi_{\mathcal{D}} \circ \psi_{\mathcal{D}}$$

ahol  $\varphi_{\mathcal{D}} \subseteq B \times T$ ,  $\psi_{\mathcal{D}} \subseteq E^* \times B$  és

$$\psi_{\mathcal{D}} = \{(\varepsilon, b) \in E^* \times B \mid \forall i \in [1..n] : \exists \varepsilon_i \in E_i^* : (\varepsilon_i, b_i) \in \varrho_i \text{ és } \varepsilon = \text{kon}(\varepsilon_1, \dots, \varepsilon_n)\}.$$

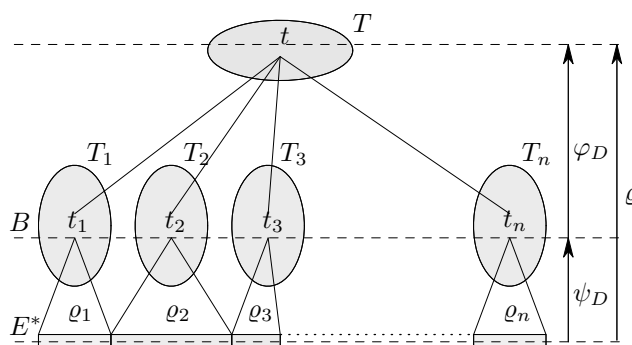
A direktszorzat értékalmazára bevezetjük a  $T = (T_1, T_2, \dots, T_n)$  jelölést.

Ha  $\varphi_{\mathcal{D}}$  kölcsönösen egyértelmű leképezés, akkor a direktszorzatot *rekord* típusnak nevezzük. A direktszorzat-típusok általában rekordok, de nem mindig. Például tekintsük a racionális számok halmazának egy lehetséges reprezentációját:

$$B = \mathbb{Z} \times \mathbb{Z}, \varphi_{\mathcal{D}} \subseteq B \times \mathbb{Q}:$$

$$((x, y), t) \in \varphi_{\mathcal{D}} \iff y \neq 0 \text{ és } t = x/y$$

Egyszerűen látható, hogy a fent definiált  $\varphi_{\mathcal{D}}$  reláció a racionális számok halmazát reprezentálja, de nem kölcsönösen egyértelmű.



11.1. ábra. Direktszorzat konstrukció

Nagyon fontos továbbá, hogy az új típusértékalmazt ( $T$ ) ne keverjük össze a közbülső direktszorzattal ( $B$ ), hiszen egy adott  $B$  és  $T$  között nagyon sokféle  $\varphi_{\mathcal{D}}$  leképezés megadható, és az új típus szempontjából egyáltalán nem mindegy, hogy ezek közül melyiket választjuk.

Tekintsük például a komplex egészek ( $a + bi$ ,  $a, b \in \mathbb{Z}$  alakú számok) halmazát. Legyen továbbá  $B = \mathbb{Z} \times \mathbb{Z}$ ,  $x, y \in \mathbb{Z}$ , és

$$\varphi_{D_1}((x, y)) = x + yi,$$

$$\varphi_{D_2}((x, y)) = y + xi.$$

A két  $\varphi_{\mathcal{D}}$  közötti különbség elsősorban akkor válik szignifikánssá, ha például a komplex egészek közötti szorzásműveletet kell implementálnunk a számpárok szintjén, hiszen ekkor az első és a második komponens értékét az

$$(a + bi)(c + di) = (ac - bd) + (ad + bc)i$$

formula alapján különböző módon kell kiszámítani.

A következő módszer, amivel régi típusokból újakat hozhatunk létre, az unió. Legyenek  $\mathcal{T}_i = (\varrho_i, I_i, \mathbb{S}_i)$  ( $i = 1, 2, \dots, n$ ) típusok, és jelölje  $T_1, T_2, \dots, T_n$  a hozzájuk tartozó típusértékhalmozokat, illetve  $E_1, E_2, \dots, E_n$  a nekik megfelelő elemi típusértékhalmozokat. Vezessük be továbbá az  $E = E_1 \cup E_2 \cup \dots \cup E_n$  és  $B = T_1 \cup T_2 \cup \dots \cup T_n$  jelöléseket.

### 11.2. DEFINÍCIÓ: UNIÓ

Azt mondjuk, hogy a  $\mathcal{T} = (\varrho, I, \mathbb{S})$  típus *uniója* a  $\mathcal{T}_1, \mathcal{T}_2, \dots, \mathcal{T}_n$  típusoknak, ha

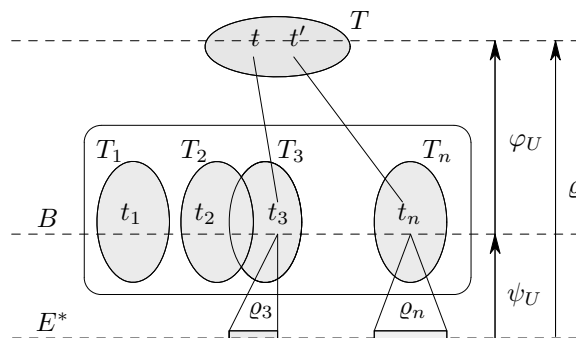
$$\varrho = \varphi_U \circ \psi_U,$$

ahol  $\varphi_U \subseteq B \times T$ ,  $\psi_U \subseteq E^* \times B$  és

$$\psi_U = \{(\varepsilon, b) \in E^* \times B \mid \exists i \in [1..n] : (\varepsilon, b) \in \varrho_i\}.$$

Az unió értékhalmozának jelölése:  $T = (T_1; T_2; \dots; T_n)$ .

Itt is külön elnevezést adtunk annak az esetnek, amikor a  $\varphi_U$  leképezés kölcsönösen egyértelmű, ekkor az uniót *egyesítésnek* nevezzük.



11.2. ábra. Unió konstrukció

Ebben az esetben is nagyon fontos, hogy mindig megkülönböztessük a konstrukció közbülső szintjén levő uniót ( $B$ ) az új típusértékhalmoztól ( $T$ ).

A harmadik megengedett típuskonstrukciós művelet az iterált, amellyel egy meglévő típusból alkothatunk új típust. Legyen  $\mathcal{T}_0 = (\varrho_0, I_0, \mathbb{S}_0)$  típus,  $T_0$  a neki megfelelő típusértékhalmoz,  $E$  a  $\mathcal{T}_0$  típus elemi típusértékhalmoza és  $B = T_0^*$ .

**11.3. DEFINÍCIÓ: ITERÁLT**

Azt mondjuk, hogy a  $\mathcal{T} = (\varrho, I, \mathbb{S})$  típus *iteráltja* a  $\mathcal{T}_0$  típusnak, ha

$$\varrho = \varphi_{\mathcal{T}} \circ \psi_{\mathcal{T}},$$

ahol  $\varphi_{\mathcal{T}} \subseteq B \times T$ ,  $\psi_{\mathcal{T}} \subseteq E^* \times T_0^*$  és

$$\psi_{\mathcal{T}} = \{(\varepsilon, b) \in E^* \times B \mid \exists \varepsilon_1, \dots, \varepsilon_{|b|} \in E^* : (\varepsilon_i, b_i) \in \varrho_0 \text{ és} \\ \varepsilon = \text{kon}(\varepsilon_1, \dots, \varepsilon_{|b|})\}.$$

Az iterált típusértékhalmozának jelölése:  $T = it(T_0)$ .

Az iterált típuskonstrukciónak három speciális esetét különböztetjük meg aszerint, hogy a  $\varphi_{\mathcal{T}}$  leképezésre milyen feltételek teljesülnek.

- Ha a  $\varphi_{\mathcal{T}}$  leképezés kölcsönösen egyértelmű, akkor *sorozat* típuskonstrukcióról beszélünk, és típusértékhalmozát  $T = seq(T_0)$ -lal jelöljük.

- Ha

$$(\alpha, t), (\beta, t) \in \varphi_{\mathcal{T}} \Leftrightarrow \alpha \in perm(\beta),$$

akkor az iterált konstrukciót *kombináció* típusnak nevezzük. A kombináció értékhalmozának jelölése:  $T = com(T_0)$ .

- Ha

$$(\alpha, t), (\beta, t) \in \varphi_{\mathcal{T}} \Leftrightarrow \bigcup_{i=1}^{|\alpha|} \{\alpha_i\} = \bigcup_{i=1}^{|\beta|} \{\beta_i\},$$

akkor *halmaz* típuskonstrukcióról beszélünk. A halmaz típus értékhalmozának jelölése:  $T = set(T_0)$ .

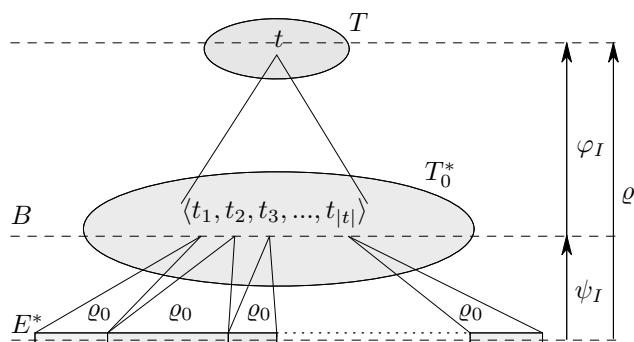
Természetesen az imént felsorolt három eset csak speciális formája az iteráltképezésnek; létezik olyan iterált is, amely egyik fenti kritériumot sem teljesíti.

A programokhoz hasonlóan, ha mást nem mondunk, *primitív típusnak* fogjuk tekinteni azokat az elemi típusokat, amiknek a típusértékhalmozata a természetes számok, az egész számok, a logikai értékek vagy a karakterek.

**11.2. Szelektorfüggvények**

Az előzőekben definiált típuskonstrukciókra most bevezetünk néhány olyan függvényt és jelölést, amelyek leegyszerűsítik a rájuk vonatkozó állítások, programok megfogalmazását.

Legyen  $T = (T_1, T_2, \dots, T_n)$  rekord. A  $\varphi_{\mathcal{T}}^{(-1)}$  függvény komponenseit a  $T$  rekord *szelektorfüggvényeinek* vagy röviden *szelektorainak* nevezzük. ha  $A$  fenti rekordnak



11.3. ábra. Iterált konstrukció

tehát pontosan  $n$  darab szelektora van, és ha  $s_i$ -vel jelöljük az  $i$ -edik szelektort, akkor  $s_i : T \rightarrow T_i$ , és

$$\forall t \in T : \varphi_{\mathfrak{D}}(s_1(t), s_2(t), \dots, s_n(t)) = t.$$

Tehát a szelektorfüggvényeket arra használhatjuk, hogy lekérdezzük a rekord egyes mezőinek (komponenseinek) az értékét. A szelektorokat bele szoktuk írni a típusértékhalmoz jelölésébe; a fenti esetben a szelektorokkal felírt jelölés:

$$T = (s_1 : T_1, s_2 : T_2, \dots, s_n : T_n).$$

A rekordtípushoz hasonlóan az egyesítéshez is bevezetünk szelektorfüggvényeket. Mivel az unió esetében a közbülső szinten a típusértékhalmozok uniója szerepel, így nincs értelme komponensről beszélni. Hogyan definiáljuk ez esetben a szelektorokat? Azt fogják visszaadni, hogy egy adott  $T$ -beli elemet melyik eredeti típusértékhalmoz egy eleméhez rendelte hozzá a  $\varphi_{\mathfrak{U}}$  függvény.

Legyen  $T = (T_1; T_2; \dots; T_n)$  egyesítés típus,  $s_i : T \rightarrow \mathbb{L}$  ( $i = 1, \dots, n$ ). Azt mondjuk, hogy az  $s_i$  logikai függvények a  $T$  egyesítés szelektorai, ha  $\forall i \in [1..n] : \forall t \in T$ :

$$s_i(t) = \left( \varphi_{\mathfrak{U}}^{(-1)}(t) \in T_i \right).$$

A rekordtípushoz hasonló módon az egyesítés szelektorait is bele szoktuk írni az új típusértékhalmoz jelölésébe. A szelektorokkal felírt típusértékhalmoz jelölése:

$$T = (s_1 : T_1; s_2 : T_2; \dots; s_n : T_n).$$

Az iterált típuskonstrukciók közül a sorozathoz definiálunk szelektorfüggvényt. A sorozattípusban a közbülső szinten  $T_0$ -beli sorozat szerepel, a szelektor ennek a sorozatnak a tagjait adja vissza.

Formálisan: Legyen  $T = seq(T_0)$ . Az  $s : T \times \mathbb{N} \rightarrow T_0$  parciális függvény a  $T$  szelektorfüggvénye, ha  $\forall t \in T : \forall i \in [1..|\varphi_{\mathfrak{S}}^{(-1)}(t)|]$ :

$$s(t, i) = \varphi_{\mathfrak{S}}^{(-1)}(t)_i.$$

A sorozat szelektorát nem szoktuk külön elnevezni, helyette indexelést alkalmazunk, azaz a  $t_i = s(t, i)$  jelölést használjuk.

### 11.3. Az iterált specifikációs függvényei

Ha az iterált típus az előzőekben bevezetett három speciális osztály valamelyikébe tartozik, akkor további függvényeket definiálunk hozzá.

Legyen  $T = it(T_0)$ ,  $(\alpha, t) \in \varphi_{\mathcal{T}}$ , és tegyük fel, hogy az iterált sorozat, kombináció vagy halmaz. Ekkor  $dom : T \rightarrow \mathbb{N}_0$ ,

$$dom(t) = \begin{cases} |\alpha|, & \text{ha } T = seq(T_0) \text{ vagy } T = com(T_0); \\ |\bigcup_{i=1}^{|\alpha|} \{\alpha_i\}|, & \text{ha } T = set(T_0). \end{cases}$$

A  $dom$  függvény tehát a  $t$  elemeinek számát adja meg. A függvény jól definiált, ugyanis felhasználva a sorozat, kombináció és halmaz típus definícióját, könnyen látható, hogy a függvényérték független az  $\alpha$  választásától.

A továbbiakban a sorozattípussal fogunk foglalkozni. Ahol külön nem jelöljük, ott  $T = seq(T_0)$ ,  $(\alpha, t) \in \varphi_{\mathcal{T}}$ ,  $\alpha = \langle \alpha_1, \alpha_2, \dots, \alpha_{|\alpha|} \rangle$ .

- Nem üres sorozat első és utolsó eleme:  $lov \in T \rightarrow T_0$ ,  $hiv \in T \rightarrow T_0$ ,

$$\begin{aligned} lov(t) &= \alpha_1; \\ hiv(t) &= \alpha_{|\alpha|}. \end{aligned}$$

- Sorozat kiterjesztése a sorozat elején vagy végén (legyen  $e \in T_0$ ):  $loext : T \times T_0 \rightarrow T$ ,  $hiext : T \times T_0 \rightarrow T$ ,

$$\begin{aligned} loext(t, e) &= \varphi_{\mathcal{T}}(kon(\langle e \rangle, \alpha)); \\ hiext(t, e) &= \varphi_{\mathcal{T}}(kon(\alpha, \langle e \rangle)). \end{aligned}$$

- Nem üres sorozat első vagy utolsó elemének elhagyásával kapott sorozat:  $lorem \in T \rightarrow T$ ,  $hirem \in T \rightarrow T$ ,

$$\begin{aligned} lorem(t) &= \varphi_{\mathcal{T}}(\langle \alpha_2, \dots, \alpha_{|\alpha|} \rangle); \\ hirem(t) &= \varphi_{\mathcal{T}}(\langle \alpha_1, \dots, \alpha_{|\alpha|-1} \rangle). \end{aligned}$$

### 11.4. A függvénytípus

A gyakorlatban nagyon fontos szerepet játszik egy speciális rekordtípus. Legyen  $H$  egy tetszőleges (megszámlálható) halmaz, amelyen van egy rákövetkezési reláció. Jelöljük ezt a rákövetkezési relációt  $succ$ -cal, és inverzére vezessük be a  $pred$  jelölést.

**11.4. DEFINÍCIÓ: FÜGGVÉNYTÍPUS**

Legyen  $E$  egy tetszőleges típus értékhalmaza. Ekkor az  $F = (H, seq(E))$  rekordot *függvénytípusnak* nevezzük, és  $F = fun(H, E)$ -vel jelöljük.

A függvénytípusra is bevezetünk néhány fontos specifikációs függvényt. A továbbiakban legyen  $F = fun(H, E)$ ,  $((h, t), f) \in \varphi_{\mathfrak{D}}$ . Ekkor

- $dom : F \rightarrow \mathbb{N}_0$ ,  
 $dom(f) = dom(t)$ .
- $lob : F \rightarrow H$ ,  
 $lob(f) = h$ .
- $hib : F \rightarrow H$ ,  
 $hib(f) = succ^{dom(f)-1}(h)$ .
- $lov \in F \rightarrow E$ ,  
 $lov(f) = lov(t)$ .
- $hiv \in F \rightarrow E$ ,  
 $hiv(f) = hiv(t)$ .
- $loext, hiext : F \times E \rightarrow F$ ,  
 $loext(f, e) = \varphi_{\mathfrak{D}}(pred(h), loext(t, e));$   
 $hiext(f, e) = \varphi_{\mathfrak{D}}(h, hiext(t, e)).$
- $lorem, hirem \in F \rightarrow F$ ,  
 $lorem(f) = \varphi_{\mathfrak{D}}(succ(h), lorem(t));$   
 $hirem(f) = \varphi_{\mathfrak{D}}(h, hirem(t)).$

A sorozathoz hasonlóan a függvénytípusra is bevezetünk egy szelekciós parciális függvényt. Tekintsük a fentiekben használt  $f$ -et. Ekkor  $s_f \in H \rightarrow E$ ,  $\mathcal{D}_{s_f} = \{succ^i(lob(f)) \mid 0 \leq i < dom(f)\}$ , és ha  $g \in \mathcal{D}_{s_f}$ ,  $g = succ^k(lob(f))$ , akkor

$$s_f(g) = t_{k+1}.$$

A függvénytípus szelektorfüggvényét nem szoktuk külön elnevezni, helyette a matematikában – a függvény helyettesítési értékének jelölésére – használt zárójeles hivatkozást használjuk, vagy egyszerűen indexelünk, azaz

$$f_g = f(g) = s_f(g).$$

A függvénytípus elnevezés azt a szemléletes képet tükrözi, hogy egy függvénytípusú érték felfogható egy  $H \rightarrow E$  típusú parciális függvénynek, amelynek értelmezési tartománya a „*lob*-tól a *hib*-ig tart”, értékeit pedig a sorozatkomponens tartalmazza.

Az előbbieken bevezetett *dom*, *lov*, *hiv*, *lob*, *hib* függvényeket kiterjeszthetjük az egész állapottérre is: komponáljuk a megfelelő változóval. Tehát ha például  $x$  egy sorozat típusú változó, akkor  $dom \circ x$  egy, az egész állapottéren értelmezett függvény. Az ilyenfajta függvénykompozíciókra bevezetünk egy újabb jelölést: ha  $t$  a fenti függvények valamelyike, és  $x$  a neki megfelelő típusú változó, akkor a  $t \circ x$  helyett  $x.t$ -t írunk.

## 11.5. A típuskonstrukciók típusműveletei

A típuskonstrukciók eddigi tárgyalásából még hiányzik valami: nem beszéltünk még a konstruált típusok műveleteiről. Az előbb felsorolt speciális esetekhez – az imént definiált függvények segítségével – bevezetünk néhány típusműveletet.

A továbbiakban megengedett feltételnek fogjuk nevezni azokat az  $A \rightarrow \mathbb{L}$  állításokat, amelyek lehetnek elágazás vagy ciklus feltételei.

Legyen  $T = (s_1 : T_1, \dots, s_n : T_n)$  rekord,  $t : T$ ,  $t_i : T_i$  ( $i \in [1..n]$ ). Mivel  $t$  az állapottér változója,  $t$  komponálható a szelektorfüggvényekkel, és így az állapottéren értelmezett függvényeket kapunk. Az  $s_i \circ t$  függvénykompozíciót a továbbiakban  $t.s_i$ -vel fogjuk jelölni. Egy rekord típusnál a szelektorfüggvény használatát megengedettnek tekintjük.

Ezenkívül bevezetjük a  $t.s_i := t_i$  jelölést is. Ezen azt a  $t := t'$  értékadást értjük, amelyben  $t'.s_i = t_i$ , és  $t'$  minden más komponense megegyezik  $t$  megfelelő komponensével.

A fenti típusműveletek arra adnak lehetőséget, hogy egy rekord „mezőinek” az értékét lekérdezhessük, illetve megváltoztathassuk. A fent definiált műveletben zavaró lehet, hogy egy függvénynek ( $t.s_i$ ) „adunk értéket”. Ezért fontos megjegyezni, hogy ez csupán egy jelölése az értékadásnak.

Legyen  $T = (s_1 : T_1; \dots; s_n : T_n)$  egyesítés,  $t : T$ ,  $t_i : T_i$  ( $i \in [1..n]$ ). Ekkor a rekord típusnál bevezetett jelölést az egyesítés esetén is bevezetjük,  $t.s_i$ -n az  $s_i \circ t$  kompozíciót értjük, és megengedett függvénynek tekintjük.

Ezen kívül megengedett művelet a  $t := \varphi_{\mathbb{L}}(t_i)$  értékadás. Ennek az értékadásnak a jelölését leegyszerűsítjük, a továbbiakban  $t := t_i$  alakban fogjuk használni.

A fenti értékadást bizonyos ésszerű korlátozások bevezetésével „megfordíthatjuk”. Így kapjuk a következő parciális értékadást:  $t_i := t$ . Ez az értékadás csak akkor végezhető el, ha  $t.s_i$  igaz.

A sorozat típuskonstrukció nagyon gyakori, és sokféle művelet definiálható vele kapcsolatban. Attól függően, hogy melyeket tekintjük megvalósítottak, különböző konstrukciókról beszélünk. Most előbb megadunk néhány lehetséges műveletet, majd a sorozattípusokat osztályokba soroljuk a megengedett műveleteik alapján.



Legyen a továbbiakban  $T = seq(E)$ ,  $t : T$ ,  $e : E$ . Ekkor az iménti szelektorokhoz hasonlóan bevezetjük az alábbi jelöléseket:

$$\begin{aligned} dom \circ t &\rightarrow t.dom \\ lov \circ t &\rightarrow t.lov \\ hiv \circ t &\rightarrow t.hiv \end{aligned}$$

Természetesen  $t.lov$  és  $t.hiv$  csak parciális függvények. Ezenkívül az alábbi (esetleg parciális) értékadásokra a következő jelöléseket fogjuk használni:

$$\begin{aligned} t := lorem(t) &\rightarrow t : lorem \\ t := hirem(t) &\rightarrow t : hirem \\ t := loext(t, e) &\rightarrow t : loext(e) \\ t := hiext(t, e) &\rightarrow t : hiext(e) \\ e, t := lov(t), lorem(t) &\rightarrow e, t : lopop \\ e, t := hiv(t), hirem(t) &\rightarrow e, t : hipop \end{aligned}$$

A bevezetett jelölések első látásra zavarba ejtőnek tűnhetnek, hiszen ugyanazt a kulcsszót a bal oldalon függvényként, a jobb oldalon pedig a művelet nevéként használjuk. Lényeges ezért megjegyezni, hogy a jobb oldalon található műveletek csak a bal oldali értékadás egyszerűsítő *jelölései*.

Attól függően, hogy a fent definiált műveletek közül melyeket tekintjük megengedettnek, különböző konstrukciókról beszélünk.

Legyen  $T = seq(E)$ . Ekkor a  $T$

- *szekvenciális input fájl*, ha csak a *lopop* a megengedett művelet;
- *szekvenciális output fájl*, ha csak a *hiext* a megengedett művelet;
- *verem*, ha a megengedett műveletek a *loext* és a *lopop*, vagy a *hiext* és a *hipop*;
- *sor*, ha a megengedett műveletek a *hiext* és a *lopop*, vagy a *loext* és a *hipop*.

Ahhoz, hogy a szekvenciális input fájl a *lopop* művelettel használható legyen, tudnunk kell, hogy mikor olvastuk ki az utolsó elemet a fájlból. Ezt a problémát úgy szoktuk megoldani, hogy bevezetünk egy extrémális elemet, és kikötjük, hogy a fájlban ez az utolsó elem (tehát még az üres fájl is tartalmazza). Ez a technika valósul meg azokban az operációs rendszerekben, ahol a szövegfájlok végét fájlvége (EOF) karakter jelzi.

Mivel a *lopop* művelet bizonyos esetekben kényelmetlen lehet – gondoljunk arra, amikor nehézkes extrémális elemet találni –, bevezetünk egy másik olvasóműveletet is. Használjuk az olvasás sikerességének jelzésére a  $\{norm, abnorm\}$  halmaz elemeit. Ekkor az  $sx, dx, x : read$  műveleten az alábbi szimultán értékadást értjük:

$$sx, dx, x : read = \begin{cases} norm, lov(x), lorem(x), & \text{ha } dom(x) \neq 0; \\ abnorm, dx, x, & \text{ha } dom(x) = 0. \end{cases}$$

Ha egy szekvenciális fájlra a *read* művelet van megengedve, akkor nincs szükség extrémális elemre, helyette az *sx* változó értéke alapján lehet eldönteni, hogy végére értünk-e a fájlnek.

Legyen  $F = fun(H, E)$ ,  $f : F$ ,  $e : E$ ,  $i : H$ . Ekkor a sorozat típushoz hasonlóan bevezetjük az alábbi jelöléseket:

$$\begin{aligned} dom \circ f &\rightarrow f.dom \\ lov \circ f &\rightarrow f.lov \\ hiv \circ f &\rightarrow f.hiv \\ lob \circ f &\rightarrow f.lob \\ hib \circ f &\rightarrow f.hib \end{aligned}$$

A fenti függvényeken kívül a függvénytípus szelektorfüggvényét,  $f(i)$ -t is megengedtettnek tekintjük. A rekord típusnál bevezetett szelektorra (mezőre) vonatkozó értékadásnak jelen esetben is van megfelelője: az  $f(i) := e$  parciális értékadás. Az értékadás azért parciális, mert csak akkor végezhető el, ha  $f.lob \leq i \leq f.hib$ . Ekkor a fenti jelölésen azt az  $f := f'$  értékadást értjük, amelyre:

$$\begin{aligned} f'.lob &= f.lob \text{ és } f'.hib = f.hib \text{ és } f(i) = e \text{ és} \\ &\forall j \in [f.lob..f.hib] : j \neq i \rightarrow f'(j) = f(j). \end{aligned}$$

A sorozatokra bevezetett kiterjesztő és elhagyó műveleteket függvény típusra is definiáljuk:

$$\begin{aligned} f := lorem(f) &\rightarrow f : lorem \\ f := hirem(f) &\rightarrow f : hirem \\ f := loext(f, e) &\rightarrow f : loext(e) \\ f := hieft(f, e) &\rightarrow f : hieft(e) \end{aligned}$$

Ha ezen utolsó csoportban felsorolt műveleteket egy függvénytípusra nem engedjük meg, akkor egy speciális függvénytípushoz, a vektorhoz jutunk. Az általános függvénytípustól megkülönböztetendő a vektortípusra külön jelölést vezetünk be:  $V = vekt(H, E)$ .

## 12. fejezet

# Programozási tételek (Levezetés)

Először néhány egyszerű feladatra keresünk megoldóprogramot. Ezek a feladatok két szempontból is fontosak számunkra. Egyrészt sok olyan konkrét feladat van, amelyeknek ezek általánosításai, így megoldásuk lehetőséget ad sok konkrét (konkrétabb) feladat megoldására is; ezért nevezzük őket programozási tételeknek. Másrészt megoldásukon keresztül megmutatjuk, hogyan lehet megoldóprogramot levezetni. Ezután néhány összetettebb tételt vezetünk le. Végül a tételek egy fontos csoportjával, speciális tulajdonságú függvények helyettesítési értékének kiszámításával foglalkozunk.

Ebben a fejezetben a feladatok megfogalmazásában használjuk a következő kijelentést: adott az  $f : X \rightarrow Y$  függvény. Ezen azt fogjuk érteni, hogy a feladat  $A$  állapotterének van egy olyan  $H$  altere, amin az  $X \rightarrow Y$  függvényeket definiáljuk, és a  $H$  altér változóit egy-egy megfelelő paraméterváltozóval az elő- és utófeltételben rögzítjük.

### 12.1. Programozási tételek intervallumon

A következő tételeknek sok közös vonása van. Mindegyik arról szól, hogy egy  $[m..n]$  intervallumon teljesül egy  $\varphi$  tulajdonság. A  $\varphi$  az intervallumon kívül az állapotter számos más komponensétől is függhet.

$$A = \mathbb{Z} \times \mathbb{Z} \times \dots$$
$$m \quad n \quad \dots$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \dots$$
$$m' \quad n' \quad \dots$$

$$Q : (m = m' \wedge n = n' \wedge m^* \leq n)$$

$$R : (Q \wedge \varphi(m, n, \dots))$$

Az előfeltételben  $m^*m-1$  vagy  $m$ , attól függően, hogy a feladatnak van-e értelme üres intervallum esetén, vagy nincs.

Az ilyen feladatokat ciklussal oldhatjuk meg. Invariáns tulajdonságnak azt választjuk, hogy  $\varphi$  nem az egész  $[m..n]$  intervallumra, hanem csak egy  $[m..k]$  részére teljesül, ezért az állapotteret kibővítjük egy új egész komponenssel ( $k$ ), és kiterjesztjük a feladatot az új állapottérre. Természetesen a kiterjesztésnek csak elvi jelentősége van, hiszen a kiterjesztett feladat specifikációja formálisan csak az állapottérben különbözik az eredetitől. Tehát az invariáns tulajdonság:

$$P = (Q \wedge k \in [m^*..n] \wedge \varphi(m, k, \dots))$$

Ez az invariáns tulajdonság azért megfelelő mert:

1. A ciklusfeltételt  $k \neq n$ -nek választva  $P \wedge \neg\pi \Rightarrow R$ , azaz teljesül a ciklus levezetési szabályának a 2. feltétele.
2. A levezetési szabály első feltétele általában nem teljesül, de könnyen tudunk olyan  $Q'$ -t választani, amiből már következik  $P$ , és könnyű eljutni  $Q$ -ból  $Q'$ -be. Ez a  $Q'$  rendszerint az üres, illetve néha az egy hosszúságú intervallum esete, azaz

$$Q' = (Q \wedge k = m^* \wedge \varphi(m, k, \dots)).$$

3. Még terminálófüggvényt kell választani, ilyen esetekben kézenfekvő választás:

$$t = (n - k), \text{ hiszen így } P \wedge \pi \Rightarrow t > 0.$$

4. Most már a csak következő két feltételnek kell teljesülnie:

$$Q \Rightarrow lf(S_1, Q') \text{ és } P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_0, P \wedge t < t_0).$$

Általánosságban még annyit mondhatunk, hogy a második esetben kézenfekvő, hogy  $S_0$ -t szekvencia formában keressük. Egyszerűen belátható, hogy a szekvencia második tagjaként alkalmazott

$$k := k + 1$$

értékadás csökkenti a terminálófüggvény értékét. Azért, hogy az invariáns tulajdonság is teljesüljön, a szekvencia közbülső tulajdonsága

$$Q'' = (lf(k := k + 1, P) \wedge t = t_0) = (P^{k \leftarrow k+1} \wedge t = t_0)$$

lesz, mivel így

$$Q'' \Rightarrow (lf(k := k + 1, P \wedge t < t_0)) = (P^{k \leftarrow k+1} \wedge n - k - 1 < t_0)$$

valóban teljesül.

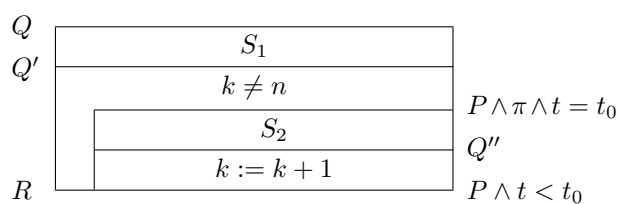
A specifikáció tétele értelmében, ha  $S_1$  és  $S_2$  olyan programok, hogy  $\forall b \in B$ -re, vagy másképpen fogalmazva, a paraméterváltozók értékétől függetlenül

$$Q \Rightarrow lf(S_1, Q')$$

és

$$P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_2, Q'' \wedge t = t_0)$$

teljesül, akkor a megoldóprogram sémája:



### 12.1.1. Összegzés

Legyen adott az  $f : \mathbb{Z} \rightarrow \mathbb{Z}$  függvény. Feladatunk az, hogy egy adott  $[m..n] \subset \mathbb{Z}$  intervallumban összegezzük az  $f$  függvény értékeit. Specifikáljuk először a feladatot.

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$m \quad n \quad s$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$m' \quad n'$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R : (Q \wedge s = \sum_{i=m}^n f(i))$$

Ebben az esetben

$$\varphi(m, n, s) = \left( s = \sum_{j=m}^n f(j) \right),$$

így az invariáns tulajdonság:

$$P = \left( Q \wedge k \in [m - 1..n] \wedge s = \sum_{j=m}^k f(j) \right),$$

a  $Q'$  állítás:

$$Q' = (Q \wedge k = m - 1 \wedge s = 0).$$

Most még keresnünk kell egy olyan programot, ami  $Q$ -ból  $Q'$ -be jut. A  $k, s := m - 1, 0$  értékadás megfelel ennek a kritériumnak, hiszen

$$Q \Rightarrow lf(k, s := m - 1, 0, Q') = (Q \wedge m - 1 = m - 1 \wedge 0 = 0) = Q.$$

Már nincs más dolgunk, mint találni egy olyan programot, ami  $P \wedge \pi \wedge t = t_0$ -ból  $Q''$ -be jut, ahol

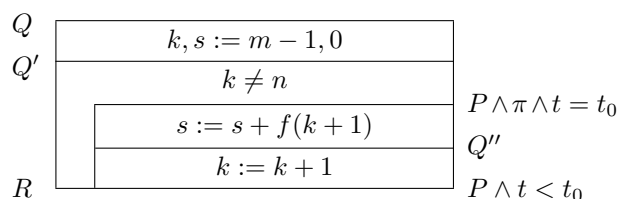
$$Q'' = (P^{k \leftarrow k+1} \wedge t = t_0) = \left( Q \wedge k + 1 \in [m - 1..n] \wedge s = \sum_{j=m}^{k+1} f(j) \wedge t = t_0 \right).$$

Nézzük meg, hogy mi nem teljesül  $Q''$ -ben: mivel  $k \in [m - 1..n]$  ( $P$ ) és  $k \neq n$  ( $\pi$ ),  $k+1 \in [m-1..n]$  fennáll.  $s$  értéke viszont nem jó, mert  $P$  szerint csak  $k$ -ig tartalmazza  $f$  értékeinek összegét,  $Q''$  szerint pedig már  $k+1$ -ig kell. A fenti megfontolás alapján tehát  $s$  növelése  $f(k+1)$ -gyel jó lesz, azaz:

$$\begin{aligned} P \wedge \pi \wedge t = t_0 \Rightarrow lf(s := s + f(k + 1), Q'') = \\ = \left( Q \wedge k + 1 \in [m - 1..n] \wedge s + f(k + 1) = \sum_{i=m}^{k+1} f(i) \wedge t = t_0 \right). \end{aligned}$$

A fenti levezetés alapján kimondható az alábbi tétel:

**Tétel:** Az alábbi, struktogram formában megadott program megoldása a fent specifikált feladatnak:



### 12.1.2. Számlálás

Legyen  $\beta$  egy, az egész számokon értelmezett logikai függvény. A feladat az, hogy számoljuk meg, hány helyen igaz  $\beta$  az  $[m..n] \subset \mathbb{Z}$  intervallumban.

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{N}_0$$

$m \quad n \quad d$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$m' \quad n'$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R : (Q \wedge d = \sum_{i=m}^n \chi(\beta(i)))$$

A fenti specifikációban  $\chi : \mathbb{L} \rightarrow \{0, 1\}$ , amelyre  $\chi(\text{igaz}) = 1$  és  $\chi(\text{hamis}) = 0$ . A feladat megoldása analóg az összegzés tételénél leírtakkal:

$$\varphi(m, n, d) = \left( d = \sum_{j=m}^n \chi(\beta(j)) \right),$$

az invariáns tulajdonság:

$$P = \left( Q \wedge k \in [m-1..n] \wedge d = \sum_{j=m}^k \chi(\beta(j)) \right),$$

a  $Q'$  állítás:

$$Q' = (Q \wedge k = m - 1 \wedge d = 0).$$

A  $Q$ -ből  $Q'$ -be jutó program a  $k, d := m - 1, 0$  értékadás, mivel  
 $Q \Rightarrow lf(k, d := m - 1, 0, Q') = (Q \wedge m - 1 = m - 1 \wedge 0 = 0) = Q$ .

$$Q'' = \left( Q \wedge k + 1 \in [m-1..n] \wedge d = \sum_{i=m}^{k+1} \chi(\beta(i)) \wedge t = t_0 \right)$$

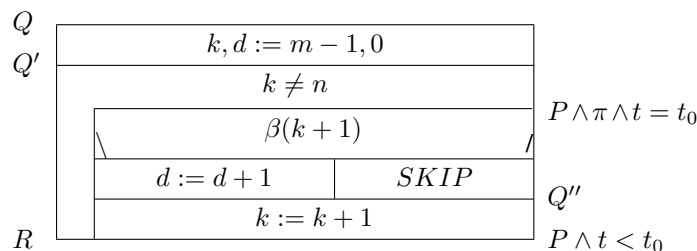
Most is azt kell megvizsgálni, hogy  $P \wedge \pi \wedge t = t_0$ -ből következik-e  $Q''$ . Ha  $\neg\beta(k+1)$ , akkor következik, míg  $\beta(k+1)$  esetén – az összegzéshez hasonlóan – meg kell növelnünk  $d$  értékét. Ezért a  $P \wedge \pi \wedge t = t_0$ -ből  $Q''$ -be jutó program az  $IF(\beta(k+1) : d := d + 1, \neg\beta(k+1) : SKIP)$  elágazás lesz, ugyanis az elágazás levezetési szabályát alkalmazva:

$$\begin{aligned} P \wedge \pi \wedge t = t_0 \wedge \beta(k+1) &\Rightarrow lf(d := d + 1, Q''), \\ P \wedge \pi \wedge t = t_0 \wedge \neg\beta(k+1) &\Rightarrow lf(SKIP, Q'') \end{aligned}$$

miatt  $P \wedge \pi \wedge t = t_0 \Rightarrow lf(IF, Q'')$  teljesül.

A fenti megfontolások alapján nyilvánvaló az alábbi tétel:

**Tétel:** Az alábbi, struktogram formában megadott program megoldása a fent specifikált feladatnak:



### 12.1.3. Maximumkeresés

Legyen  $\mathcal{H}$  egy tetszőlegesen rendezett halmaz és  $f : \mathbb{Z} \rightarrow \mathcal{H}$  egy adott függvény. Feladatunk az, hogy egy adott  $[m..n] \subset \mathbb{Z}$  intervallumban keressük meg az  $f$  függvény maximumát és egy olyan helyét, ahol ezt a maximumértéket felveszi.

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathcal{H}$$

$m \quad n \quad i \quad max$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$m' \quad n'$

$$Q : (m = m' \wedge n = n' \wedge m \leq n)$$

$$R : (Q \wedge i \in [m..n] \wedge max = f(i) \wedge \forall j \in [m..n] : f(j) \leq f(i))$$

Az előbbiekkal ellentétben ebben a specifikációban nem engedjük meg az üres intervallumot. Ennek oka rendkívül egyszerű: üres intervallumon nincs értelme megkérdezni, hogy hol van a maximum. Most

$$\varphi(m, n, i, max) = (i \in [m..n] \wedge max = f(i) \wedge \forall j \in [m..n] : f(j) \leq f(i)).$$

Tehát az invariáns tulajdonság:

$$P = (Q \wedge k \in [m..n] \wedge i \in [m..k] \wedge max = f(i) \wedge \forall j \in [m..k] : f(j) \leq f(i)),$$

$$Q' = (Q \wedge k = m \wedge i = m \wedge max = f(m)),$$

és  $S_1$  a  $i, k, max := m, m, f(m)$  értékadás. A

$$Q'' = (Q \wedge k + 1 \in [m..n] \wedge i \in [m..k + 1] \wedge max = f(i) \wedge \forall j \in [m..k + 1] : f(j) \leq f(i) \wedge t = t_0).$$

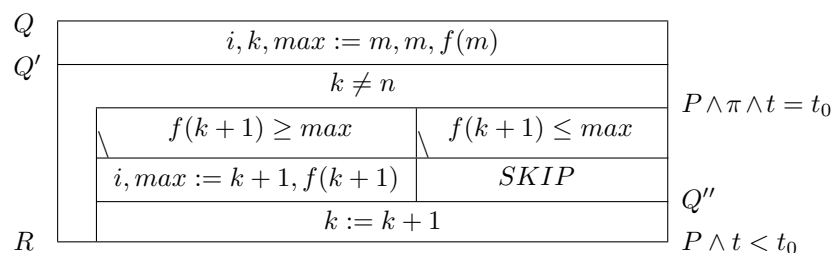
A  $P \wedge \pi \wedge t = t_0$ -ból  $Q''$ -be jutó program itt is egy elágazás lesz:  $IF(f(k + 1) \geq max : i, max := k + 1, f(k + 1); f(k + 1) \leq max : SKIP)$ , ugyanis

$$P \wedge \pi \wedge t = t_0 \wedge f(k + 1) \geq max \Rightarrow lf(i, max := k + 1, f(k + 1), Q''),$$

$$P \wedge \pi \wedge t = t_0 \wedge f(k + 1) \leq max \Rightarrow lf(SKIP, Q'')$$

miatt  $P \wedge \pi \wedge t = t_0 \Rightarrow lf(IF, Q'')$  teljesül.

**Tétel:** Az alábbi, struktogram formában megadott program megoldása a fent specifikált feladatnak:





### 12.1.4. Feltételes maximumkeresés

Legyen  $\mathcal{H}$  egy tetszőleges rendezett halmaz és  $f : \mathbb{Z} \rightarrow \mathcal{H}$  egy adott függvény. Legyen  $\beta$  egy, az egész számokon értelmezett logikai függvény. Határozzuk meg a  $[\beta] \cap [m..n]$  halmaz felett az  $f$  függvény maximumát és a halmaz egy olyan elemét, amelyen  $f$  a maximumértékét felveszi.

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathcal{H} \times \mathbb{L}$$

$m \quad n \quad i \quad max \quad l$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$m' \quad n'$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R : (Q \wedge l = (\exists i \in [m..n] : \beta(i)) \wedge l \rightarrow (i \in [m..n] \wedge \beta(i) \wedge max = f(i) \wedge \forall j \in [m..n] : \beta(j) \rightarrow f(j) \leq f(i)))$$

Itt újra megengedhető az üres intervallum, és ekkor az a válasz, hogy az intervallumban nincs  $\beta$  tulajdonságú elem.

A levezetés az előzőekhez hasonlóan:

$$P = (Q \wedge k \in [m - 1..n] \wedge l = (\exists i \in [m..k] : \beta(i)) \wedge l \rightarrow (i \in [m..k] \wedge \beta(i) \wedge max = f(i) \wedge \forall j \in [m..k] : \beta(j) \rightarrow f(j) \leq f(i)))$$

$$Q' = (Q \wedge k = m - 1 \wedge l = hamis)$$

$$Q'' = (Q \wedge k + 1 \in [m - 1..n] \wedge l = (\exists i \in [m..k + 1] : \beta(i)) \wedge l \rightarrow (i \in [m..k + 1] \wedge \beta(i) \wedge max = f(i) \wedge \forall j \in [m..k + 1] : \beta(j) \rightarrow f(j) \leq f(i)))$$

$P \wedge \pi$  és  $Q''$  összehasonlításával látható, hogy három fő lehetőség van:

- $\neg\beta(k + 1)$ : ekkor **SKIP**;
- $\beta(k + 1) \wedge \neg l$ : ez az első  $\beta$  tulajdonságú elem, tehát  $l, i, max := igaz, k + 1, f(k + 1)$ ;
- $\beta(k + 1) \wedge l$ : ekkor a maximumkeresésnél megismert két eset lehetséges:
  - $f(k + 1) \geq max$ : ekkor  $i, max := k + 1, f(k + 1)$ ,
  - $f(k + 1) \leq max$ : ekkor **SKIP**.

**Tétel:** Az alábbi, struktogram formában megadott program megoldása a fent specifikált feladatnak:

$k, l := m - 1, hamis$			
$k \neq n$			
$\neg\beta(k+1)$	$\beta(k+1) \wedge \neg l$	$\beta(k+1) \wedge l$	
$SKIP$	$l, i, max := igaz, k+1, f(k+1)$	$f(k+1) \geq max$	$f(k+1) \leq max$
		$i, max := k+1, f(k+1)$	$SKIP$
$k := k+1$			

### 12.1.5. Lineáris keresés

Legyen  $\beta : \mathbb{Z} \rightarrow \mathbb{L}$  adott függvény és  $[m..n]$  egy intervallum. Keressük meg az első olyan helyet az  $[m..n]$  intervallumban, ahol  $\beta$  igaz értéket vesz fel, ha egyáltalán van ilyen hely.

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{L}$$

$m \quad n \quad i \quad l$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$m' \quad n'$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R : (Q \wedge l = (\exists j \in [m..n] : \beta(j)) \wedge l \rightarrow (i \in [m..n] \wedge \beta(i) \wedge \forall j \in [m..i-1] : \neg\beta(j)))$$

A ciklus invariáns tulajdonsága:

$$P = (Q \wedge i \in [m-1..n] \wedge l = (\exists j \in [m..i] : \beta(j)) \wedge \forall j \in [m..i-1] : \neg\beta(j)),$$

$$Q' = (Q \wedge i = m - 1 \wedge l = hamis),$$

$$Q'' = (Q \wedge i + 1 \in [m-1..n] \wedge l = \exists j \in [m..i+1] : \beta(j) \wedge \forall j \in [m..i] : \neg\beta(j) \wedge t = t_0),$$

és így a ciklusmag első fele az  $l := \beta(i+1)$  értékadás lesz.

**Tétel:** Az alábbi, struktogram formában megadott program megoldása a fent specifikált feladatnak:

$Q$	$i, l := m - 1, hamis$	
$Q'$	$\neg l \wedge i \neq n$	
	$l := \beta(i+1)$	$P \wedge \pi \wedge t = t_0$
$R$	$i := i+1$	$Q''$ $P \wedge t < t_0$

Megjegyzés: ezt a programozási tételt *lineáris keresés 2.8* változatnak is nevezzük.

## 12.2. Tételek „feltételig” változata

A továbbiakban használjuk a következő jelölést. Legyen  $\delta : \mathbb{Z} \rightarrow \mathbb{L}$  és  $m \in \mathbb{Z}$ . Tegyük fel, hogy  $\exists i \geq m : \delta(i)$ . Jelöljük  $\alpha(m, \delta)$ -val az első olyan egész számot, amely nem kisebb, mint  $m$ , és igaz rá a  $\delta$ , azaz

$$\alpha(m, \delta) ::= \min \{x \in \mathbb{Z} \mid x \geq m \text{ és } \delta(x)\}.$$

Az előző rész tétéleit fogalmazzuk meg most kicsit más formában. A  $\varphi$  teljesülését most nem egy  $[m..n]$  intervallumon, hanem  $m$ -től az első  $\delta$  tulajdonságú helyig követeljük meg.

$$A = \mathbb{Z} \times \dots$$

$$m \quad \dots$$

$$B = \mathbb{Z} \times \dots$$

$$m' \quad \dots$$

$$Q : (m = m' \wedge \exists i \geq m : \delta(i))$$

$$R : (Q \wedge \varphi(m, \alpha(m, \delta)) \dots)$$

A megoldás most is egy ciklus lesz. Az állapotot egy egész és egy logikai komponenssel terjesztjük ki, legyenek a megfelelő változók  $k$  és  $v$ . A ciklus invariáns tulajdonsága pedig az, hogy  $k \geq m - 1$ , a  $\varphi$   $m$ -től  $k$ -ig teljesül,  $v$  annak megfelelően igaz vagy hamis, hogy  $k$ -ra  $\delta$  igaz-e, és végül még azt is megköveteljük, hogy  $k$  előtt nincs  $\delta$  tulajdonságú hely, azaz

$$P = (Q \wedge k \geq m - 1 \wedge \varphi(m, k, \dots)) \wedge$$

$$v = \exists i \in [m..k] : \delta(i) \wedge \forall j \in [m..k - 1] : \neg \delta(j).$$

A ciklusfeltétel:  $\neg v$ , mivel  $v \wedge P$  esetén  $k = \alpha(m, \delta)$  ezért  $v \wedge P \Rightarrow R$ .

Ebben az esetben  $Q'$ -nek választható:

$$Q' = (Q \wedge k = m - 1 \wedge \varphi(m, k, \dots)) \wedge \neg v.$$

A terminálófüggvény a következő: az előfeltétel miatt létezik olyan  $N$  egész szám, ami nagyobb  $m$ -nél, és igaz rá  $\delta$ , ezért legyen

$$t = (N - k),$$

ami  $P \wedge \neg v$  esetén biztosan pozitív.

Most is szekvenciaként határozzuk meg a ciklusmagot. A szekvencia második fele legyen

$$k, v := k + 1, \delta(k + 1)$$

és

$$Q'' = (lf(k, v := k + 1, \delta(k + 1), P) \wedge t = t_0) = P^{k \leftarrow k+1, v \leftarrow \delta(k+1)} \wedge t = t_0.$$

Mivel

$$P^{k \leftarrow k+1, v \leftarrow \delta(k+1)} = (Q \wedge k + 1 \geq m - 1 \wedge \varphi(m, k + 1, \dots) \wedge \delta(k + 1) = \exists i \in [m..k + 1] : \delta(i) \wedge \forall j \in [m..k] : \neg \delta(j)),$$

ha  $P \wedge \pi \wedge t = t_0$  teljesül, akkor  $Q''$  teljesüléséhez már csak az kell, hogy  $\varphi(m, k + 1, \dots)$  is igaz legyen.

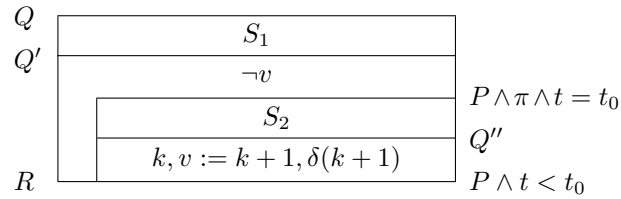
Tehát, kibővítve a paraméterteret is egy  $k'$  egész és egy  $v'$  logikai komponenssel, már csak a következő két programot keressük:

$$Q \Rightarrow lf(S_1, Q')$$

és

$$P \wedge \pi \wedge k = k' \wedge v = v' \Rightarrow lf(S_2, \varphi(m, k + 1, \dots) \wedge k = k' \wedge v = v').$$

Tehát a megoldóprogram sémája:



Megjegyzés: az invariáns tulajdonságban  $v = \exists i \in [m..k] : \delta(i)$  helyett azért nem írható  $v = \delta(k)$ , mert  $k = m - 1$  is lehet, és  $\delta(k - 1)$ -ről nem tudunk semmit.

### 12.2.1. Összezés feltételig

Legyenek adottak az  $f : \mathbb{Z} \rightarrow \mathbb{Z}$  és  $\delta : \mathbb{Z} \rightarrow \mathbb{L}$  függvények. Tegyük fel, hogy létezik  $i \geq m$ , amire  $\delta$  igaz. Feladatunk az, hogy az adott  $m$ -től az első olyan  $i$ -ig, amelyre  $\delta$  igaz, összegezzük az  $f$  függvény értékeit. Specifikáljuk először a feladatot.

$$A = \mathbb{Z} \times \mathbb{Z}$$

$$m \quad s$$

$$B = \mathbb{Z}$$

$$m'$$

$$Q : (m = m' \wedge \exists j \geq m : \delta(j))$$

$$R : (Q \wedge s = \sum_{j=m}^{\alpha(m, \delta)} f(j))$$

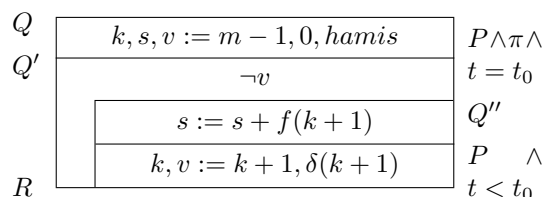
Ebben az esetben tehát

$$\varphi(m, i, s) = \left( s = \sum_{j=m}^i f(j) \right),$$

vagyis ugyanaz mint az intervallumos összegzésnél volt, ezért  $S_1$ -nek választhatjuk a  $k, s, v := m - 1, 0, hamis$  értékadást és  $S_2$ -nek ugyanazt, mint amit az intervallumos összegzésnél:  $s := s + f(k + 1)$ .

Tehát kimondható a következő tétel:

**Tétel:** Az alábbi, struktogram formában megadott program megoldása a fent specifikált feladatnak:



### 12.2.2. Számlálás feltételig

Legyenek  $\beta$  és  $\delta$  az egész számokon értelmezett logikai függvények. Tegyük fel, hogy létezik  $i \geq m$ , amelyre  $\delta$  igaz. A feladat az, hogy számoljuk meg, hány helyen igaz  $\beta$  az  $m$ -től az első olyan  $i$ -ig, amire  $\delta$  igaz.

$$A = \mathbb{Z} \times \mathbb{N}_0$$

$m \quad d$

$$B = \mathbb{Z}$$

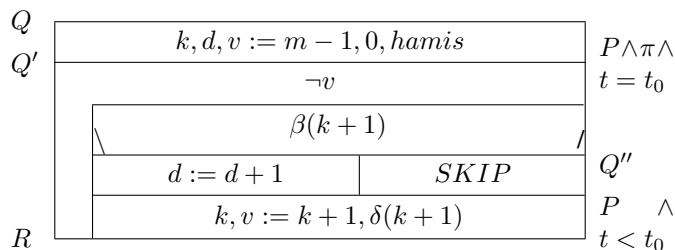
$m'$

$$Q : (m = m' \wedge \exists i \geq m : \delta(i))$$

$$R : (Q \wedge d = \sum_{i=m}^{\alpha(m, \delta)} \chi(\beta(i)))$$

A feladat megoldása analóg az összegzés tételénél leírtakkal.

**Tétel:** Az alábbi, struktogram formában megadott program megoldása a fent specifikált feladatnak:



### 12.2.3. Maximumkeresés feltételig

Legyen  $\mathcal{H}$  egy tetszőleges rendezett halmaz és  $f : \mathbb{Z} \rightarrow \mathcal{H}$  egy adott függvény és  $\delta$  az egész számokon értelmezett logikai függvény. Tegyük fel, hogy létezik  $i \geq m$ , amire  $\delta$  igaz. Feladatunk az, hogy az  $m$ -től az első olyan  $i$ -ig, amelyre  $\delta$  igaz, keressük meg az  $f$  függvény maximumát és egy olyan helyét, ahol ezt a maximumértéket felveszi.

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathcal{H}$$

$m \quad i \quad max$

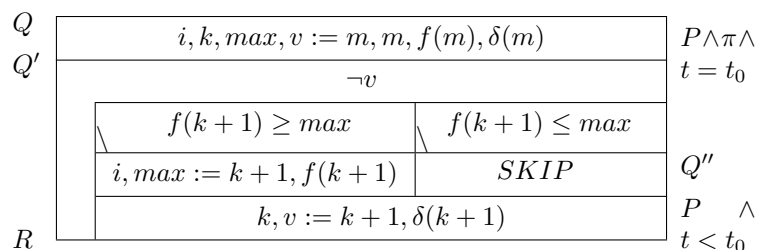
$$B = \mathbb{Z}$$

$m'$

$$Q : (m = m' \wedge \exists j \geq m : \delta(j))$$

$$R : (Q \wedge i \in [m.. \alpha(m, \delta)] \wedge max = f(i) \wedge \forall j \in [m.. \alpha(m, \delta)] : f(j) \leq f(i))$$

**Tétel:** Az alábbi, struktogram formában megadott program megoldása a fent specifikált feladatnak:



### 12.2.4. Feltételes maximumkeresés feltételig

Legyen  $\mathcal{H}$  egy tetszőleges rendezett halmaz és  $f : \mathbb{Z} \rightarrow \mathcal{H}$  egy adott függvény. Legyenek  $\beta$  és  $\delta$  az egész számokon értelmezett logikai függvények. Határozzuk meg a  $[\beta] \cap [m.. \alpha(m, \delta)]$  halmaz felett, ha az nem üres, az  $f$  függvény maximumát és a halmaz egy olyan elemét, amelyen  $f$  a maximumértékét felveszi.

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathcal{H} \times \mathbb{L}$$

$m \quad i \quad max \quad l$

$$B = \mathbb{Z}$$

$m'$

$$Q : (m = m' \wedge \exists j \geq m : \delta(j))$$

$$R : (Q \wedge l = (\exists i \in [m.. \alpha(m, \delta)] : \beta(i)) \wedge l \rightarrow (i \in [m.. \alpha(m, \delta)] \wedge \beta(i) \wedge max = f(i) \wedge \forall j \in [m.. \alpha(m, \delta)] : \beta(j) \rightarrow (f(j) \leq f(i))))$$

**Tétel:** Az alábbi, struktogram formában megadott program megoldása a fent specifikált feladatnak:

$k, l, v := m - 1, \textit{hamis}, \textit{hamis}$			
$\neg v$			
$\neg\beta(k + 1)$	$\beta(k + 1) \wedge \neg l$	$\beta(k + 1) \wedge l$	
<i>SKIP</i>	$l, i, \textit{max} :=$ $\textit{igaz}, k + 1, f(k + 1)$	$f(k + 1) \geq \textit{max}$ $i, \textit{max} :=$ $k + 1, f(k + 1)$	$f(k + 1) \leq \textit{max}$ <i>SKIP</i>
$k, v := k + 1, \delta(k + 1)$			

### 12.2.5. Lineáris keresés

Most egy kicsit eltérünk az általános sémától, és több speciális esetet vizsgálunk meg. Legyen  $\beta : \mathbb{Z} \rightarrow \mathbb{L}$  adott tulajdonság. Az első feladat az, hogy keressük meg azt a legkisebb  $\beta$  tulajdonságú egész számot, amely nem kisebb, mint az adott  $m \in \mathbb{Z}$  szám, feltéve, hogy van ilyen.

$$A = \mathbb{Z} \times \mathbb{Z}$$

$$m \quad i$$

$$B = \mathbb{Z}$$

$$m'$$

$$Q : (m = m' \wedge \exists j \geq m : \beta(j))$$

$$R : (Q \wedge i \geq m \wedge \beta(i) \wedge \forall j \in [m..i - 1] : \neg\beta(j))$$

A feladatot ciklussal oldjuk meg. Az invariánshoz gyengítjük az utófeltételt, elhagyjuk belőle  $\beta(i)$ -t:

$$P = (Q \wedge i \geq m \wedge \forall j \in [m..i - 1] : \neg\beta(j))$$

$$1) Q' = (Q \wedge i = m)$$

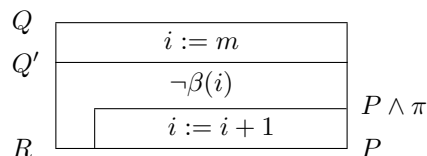
$$2) \pi = \neg\beta(i)$$

3) Legyen  $N \geq m$  tetszőlegesen rögzített olyan szám, amelyre  $\beta(N)$  igaz (ilyen az előfeltétel miatt létezik). Ekkor  $t = N - i$ .

5) Az  $i := i + 1$  értékadás csökkenti a terminálófüggvény értékét.

$$4) P \wedge \pi \Rightarrow lf(i := i + 1, P)$$

**Tétel:** Az alábbi, struktogram formában megadott program megoldása a fent specifikált feladatnak:



A fenti program csak akkor megoldása a feladatnak, ha a  $\beta$  tulajdonság megengedett feltétel. Ha ez nem így van, akkor másik megoldást kell keresnünk. Válasszuk az alábbi invariánst (a feladat marad ugyanaz!):

$$P = (Q \wedge i \geq m - 1 \wedge (\forall j \in [m..i - 1] : \neg\beta(j)) \wedge l = \exists j \in [m..i] : \beta(j))$$

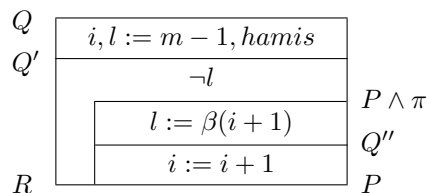
Ekkor:

- 1)  $Q' = (Q \wedge i = m - 1 \wedge l = \text{hamis})$
- 2)  $\pi = \neg l$
- 3) Legyen  $N \geq m$  tetszőlegesen rögzített olyan szám, amelyre  $\beta(N)$  igaz (ilyen az előfeltétel miatt létezik). Ekkor  $t = N - i$ .
- 5) Az  $i := i + 1$  értékadás csökkenti a terminálófüggvény értékét.
- 4) A ciklusmag szekvencia lesz, melynek közbülső feltétele:

$$Q'' = (Q \wedge i + 1 \geq m - 1 \wedge (\forall j \in [m..i] : \neg\beta(j)) \wedge l = \exists j \in [m..i + 1] : \beta(j)),$$

és így a ciklusmag első fele az  $l := \beta(i + 1)$  értékadás lesz.

**Tétel:** Ekkor az alábbi program is megoldása a specifikált feladatnak.



Legyenek adottak a  $\gamma, \delta : \mathbb{Z} \rightarrow \mathbb{L}$  és az  $m$  egész szám. Jelöljük  $\beta$ -val a  $\gamma \vee \delta$ -t, és tegyük fel, hogy létezik  $j \geq m$ , hogy  $\beta(j)$ . Keressük meg a legelső  $i \geq m : \gamma(i)$  elemet (ha van olyan), amely előtt ( $m$ -től kezdve) nem volt igaz  $\delta$ ! Ennek a változatnak már a specifikációja is más lesz, hiszen a feladat is megváltozott.

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{L}$$

$$B = \mathbb{Z}$$



$$\begin{aligned}
Q &: (m = m' \wedge \exists j \geq m : \beta(j)) \\
R &: (Q \wedge u = (\exists j \geq m : \gamma(j) \wedge \forall k \in [m..j-1] : \neg\delta(k)) \wedge \\
&\quad u \rightarrow (i \geq m \wedge \gamma(i) \wedge \forall j \in [m..i-1] : \neg\beta(j)))
\end{aligned}$$

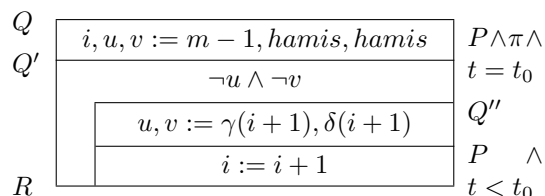
A feladatot megoldó ciklus invariánsa:

$$\begin{aligned}
P &= (Q \wedge i \geq m - 1 \wedge u = (\exists j \in [m..i] : \gamma(j)) \wedge v = (\exists j \in [m..i] : \delta(j)) \wedge \\
&\quad \forall j \in [m..i-1] : \neg\beta(j))
\end{aligned}$$

Ekkor:

- 1)  $Q' = (Q \wedge i = m - 1 \wedge u = \text{hamis} \wedge v = \text{hamis})$
- 2)  $\pi = \neg u \wedge \neg v$
- 3) Legyen  $N \geq m$  tetszőlegesen rögzített olyan szám, amelyre  $\beta(N)$  igaz (ilyen az előfeltétel miatt létezik). Ekkor  $t = N - i$ .
- 5) Az  $i := i + 1$  értékadás csökkenti a terminálófüggvény értékét.
- 4) A ciklusmag szekvencia lesz, melynek közbülső feltétele ( $lf(i := i + 1, P)$ ):
$$\begin{aligned}
Q'' &= (Q \wedge i + 1 \geq m - 1 \wedge u = (\exists j \in [m..i+1] : \gamma(j)) \wedge \\
&\quad v = (\exists j \in [m..i+1] : \delta(j)) \wedge \forall j \in [m..i] : \neg\beta(j) \wedge t = t_0),
\end{aligned}$$
és így a ciklusmag első fele az  $u, v := \gamma(i + 1), \delta(i + 1)$  értékadás lesz.

**Tétel:** Az alábbi, struktogram formában megadott program megoldása a fent specifikált feladatnak:



Megjegyzés: A lineáris keresés fenti három változatát rendre a *lineáris keresés 1.*, *2.*, *3.* változatának is szoktuk nevezni.

### 12.2.6. Tételek másképpen

A fejezetben szereplő tételeket kimondhattuk volna kicsit más formában, formákban is. Ezzel a tételek különféle változatát kaphatjuk. Nézzünk néhány példát!

Eddig a „feltételes” tételeinket úgy fogalmaztuk meg, hogy  $\varphi$  teljesülését  $m$ -től az első  $\delta$  tulajdonságú helyig követeljük meg, ezt a  $\delta$  tulajdonságú helyet is beleértve. Kimondhatnánk őket úgy is, hogy  $\varphi$  teljesülését  $m$ -től csak a  $\neg\delta$  tulajdonságú helyekre követeljük meg.

Legyen  $m^*$  most  $m + 1$  vagy  $m$ , attól függően, hogy van, vagy nincs értelme a feladatnak az üres intervallumon. Tegyük fel, hogy  $\exists i \geq m^* : \delta(i)$ , és legyen

$$\alpha'(m, \delta) ::= \min \{x \in \mathbb{Z} \mid x \geq m^* \text{ és } \delta(x + 1)\}.$$

$$A = \mathbb{Z} \times \dots$$

$$m \quad \dots$$

$$B = \mathbb{Z} \times \dots$$

$$m' \quad \dots$$

$$Q : (m = m' \wedge \exists i \geq m^* : \delta(i))$$

$$R : (Q \wedge \varphi(m, \alpha'(m, \delta)) \dots)$$

A megoldás nagyon hasonló lesz az előzőhöz, csak az invariáns tulajdonságban  $[m..k]$  helyett  $[m..k + 1]$  szerepel,

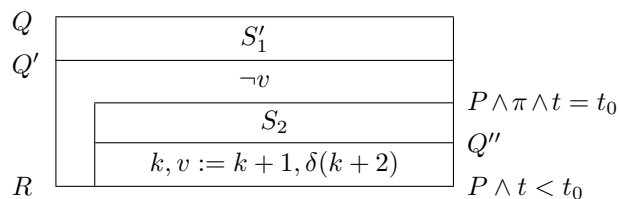
$$P_1 = (Q \wedge k \geq m^* - 1 \wedge \varphi(m, k, \dots)) \wedge$$

$$v = \exists i \in [m^*..k + 1] : \delta(i) \wedge \forall j \in [m^*..k] : \neg \delta(j),$$

és  $Q'$ -ben pedig  $\neg v$  helyett  $v = \delta(k)$ ,

$$Q'_1 = (Q \wedge k = m^* - 1 \wedge \varphi(m, k, \dots) \wedge v = \delta(k + 1)).$$

Innen kezdve a levezetés teljesen azonos módon megy, és a megoldóprogram sémája:



ahol  $S'_1$  specifikációja

$$Q \Rightarrow lf(S'_1, Q'_1),$$

ami azt jelenti, hogy a konkrét tételekbe  $\dots, v := \dots$ , *hamis* helyett  $\dots, v := \dots, \delta(m)$  kerül.

Egy másik lehetőség változatokra: az ebben és az előző részben szereplő programozási tételeket megfogalmazhattuk volna általánosabban is.

A függvényeket értelmezhetjük az egészek helyett egy tetszőleges, olyan halmazon, amelynek minden elemének van rákövetkezője (*succ*) és megelőzője (*pred*). Ez semmi különös problémát nem okoz, csak  $x + 1$  helyett *succ*( $x$ )-et,  $x - 1$  helyett *pred*( $x$ )-et kell írni.

Kicsit bonyolultabb a helyzet, ha csak azt tesszük fel, hogy a halmaz bármelyik eleméből megkaphatjuk bármelyik elemét a *succ* vagy a *pred* véges sokszori alkalmazásával, ami egyébként sokszor előforduló eset a gyakorlatban. A megoldás ebben az esetben hasonló, a  $P$  és a  $Q'$  módosításával új  $S'_1$  programot specifikálunk.

### 12.3. Programozási tételek halmazon

Az intervallumon értelmezett függvényekre vonatkozó tételeket megfogalmazhatjuk általánosabban is.

Legyenek a függvények tetszőleges véges halmazon értelmezve, és tegyük fel, hogy el tudjuk dönteni erről a halmazról, hogy üres-e, ha nem üres, ki tudjuk választani egy elemét, és végül egy elemet, amely benne van, el tudunk hagyni belőle.

Azaz legyen az állapottérnek egy olyan  $H$  komponense, amelynek az elemei az  $E$  halmaz véges részhalmazai. Az  $E$  szintén komponense az állapottérnek. A megfelelő változók legyenek  $h$  és  $e$ ; a  $h \neq \emptyset$  megengedett feltétel és az  $e : \in h$  illetve  $h := h \setminus \{e\}$  megengedett programok. Ezekkel a programokkal kapcsolatban emlékeztetünk arra, hogy ha  $R$  feltétel nem függ  $e$ -től, akkor  $lf(e : \in h, e \in h \wedge R) = h \neq \emptyset \wedge R$ , és tetszőleges  $R$  feltétel esetén  $lf(h := h \setminus \{e\}, R) = R^{h \leftarrow h \setminus \{e\}}$ .

A feladatok sémája ebben az esetben:

$$A = H \times E \times \dots$$

$$B = H \times \dots$$

$$Q : (h = h' (\wedge h \neq \emptyset))$$

$$R : (\varphi(h', \dots))$$

Megjegyzés: a  $(\wedge h \neq \emptyset)$  részre akkor van szükség, ha a feladatnak nincs értelme üres halmaz esetén (maximumkeresés).

A feladatot most is ciklussal oldjuk meg. Az invariáns tulajdonság

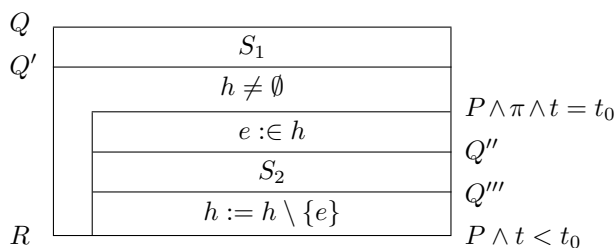
$$P = (h \subseteq h' \wedge \varphi(h' \setminus h, \dots)),$$

a ciklusfeltétel:  $h \neq \emptyset$ ,

a termináló függvény:  $|h|$ ,

$$Q' = (h = h' \wedge \varphi(\emptyset, \dots)).$$

A megoldó program sémája:

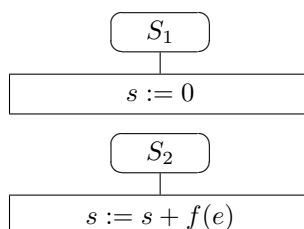


A  $Q'' = (e \in h \wedge P \wedge \pi \wedge t = t_0)$   $e : \in h$  után teljesül,  $h := h \setminus \{e\}$  csökkenti a termináló függvény értékét, ezért a szokásos módon  $Q''' = (P^{h \leftarrow h \setminus \{e\}} \wedge t = t_0)$ . Tehát a  $Q, Q'$  párral specifikált  $S_1$  és a  $Q'', Q'''$  párral specifikált  $S_2$  programok meghatározása van hátra.

Nézzük a konkrét tételeket!

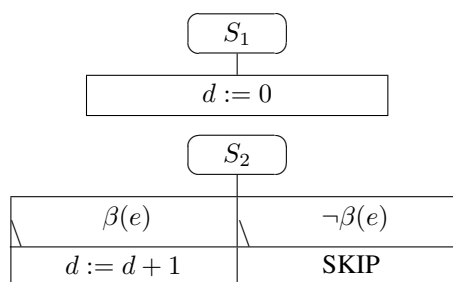
## 1. Összegzés

$$\varphi(g, s) = \left( s = \sum_{x \in g} f(x) \right)$$



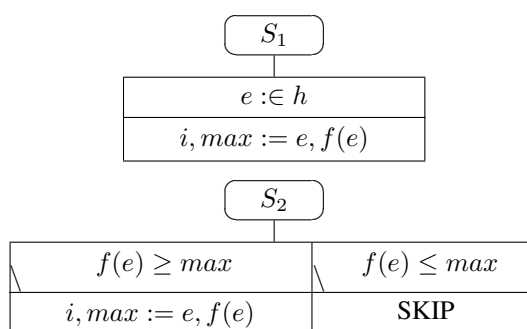
## 2. Számlálás

$$\varphi(g, d) = \left( d = \sum_{x \in g} \chi(\beta(x)) \right)$$



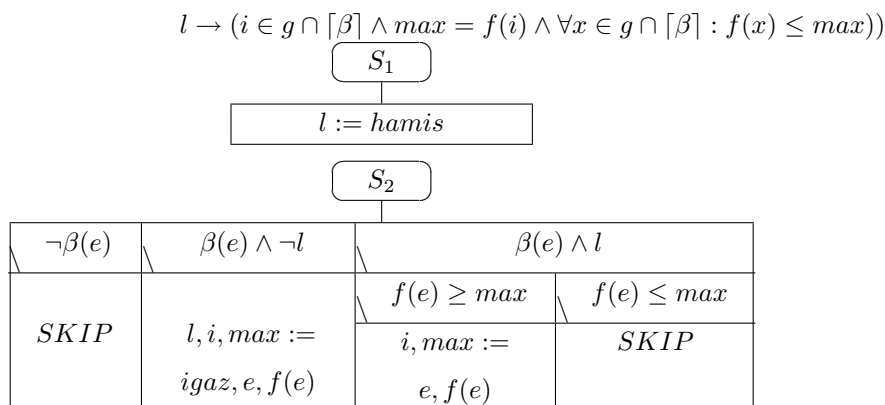
## 3. Maximumkeresés

$$\varphi(g, i, max) = (i \in h' \wedge max = f(i) \wedge \forall x \in g : f(x) \leq max)$$



## 4. Feltételes maximumkeresés

$$\varphi(g, l, i, max) = (l = \exists x \in g : \beta(x) \wedge$$



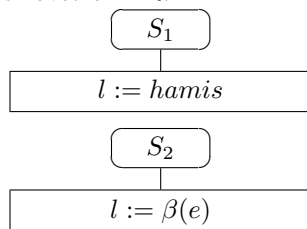
## 5. „Lineáris keresés”

Ebben az esetben

$$\varphi(g, l, e) = (l = \exists x \in g : \beta(x) \wedge$$

$$l \rightarrow (e \in g \wedge \beta(e)))$$

és a ciklusfeltétel ismódosul. Most a  $h \neq \emptyset \wedge \neg l$  a célszerű választás, hiszen  $P \wedge (h = \emptyset \vee l)$ -ből is következik  $R$ .



Megjegyzés: Ez a tétel a lineáris keresés 2.8 megfelelőjének tekinthető, azzal az eltéréssel, hogy nem beszélhetünk az első  $\beta$  tulajdonságú elemről, mivel nem tételeztük fel a halmazról a rendezettségét.

## 12.4. Bonyolultabb feladatok

Ebben a részben néhány kevésbé egyszerű tételt vezetünk le elsősorban azért, hogy megmutassuk, összetettebb feladatok esetén is használható a levezetés, másrészt az így kapott tételek is elég fontosak.

### 12.4.1. Logaritmikus keresés

Legyen  $\mathcal{H}$  egy olyan halmaz, amin értelmezve van egy rendezési reláció. Legyen  $f : \mathbb{Z} \rightarrow \mathcal{H}$  monoton növekedő függvény. A feladat az, hogy döntsük el az  $f$  függvényről,

az adott  $[m..n] \subset \mathbb{Z}$  intervallumon felveszi-e a  $h \in \mathcal{H}$  adott értéket, és ha igen, akkor adjuk meg az intervallum egy olyan pontját, ahol a függvényérték  $h$ .

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathcal{H} \times \mathbb{Z} \times \mathbb{L}$$

$$\quad m \quad n \quad h \quad i \quad l$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \mathcal{H}$$

$$\quad m' \quad n' \quad h'$$

$$Q : (m = m' \wedge n = n' \wedge h = h' \wedge m \leq n + 1 \wedge \forall k, j \in [m..n] : \\ (k < j) \rightarrow (f(k) \leq f(j)))$$

$$R : (Q \wedge l = (\exists j \in [m..n] : f(j) = h) \wedge l \rightarrow (i \in [m..n] \wedge f(i) = h))$$

A monotonitást felhasználva az intervallumot mindkét végéről szűkítjük az invariánsban:

$$P = (Q \wedge [u..v] \subseteq [m..n] \wedge \forall j \in [m..n] \setminus [u..v] : f(j) \neq h \wedge \\ l \rightarrow (i \in [u..v] \wedge f(i) = h))$$

Ekkor:

- 1)  $Q' = (Q \wedge u = m \wedge v = n \wedge l = \text{hamis})$
- 2)  $\pi = \neg l \wedge u \leq v$
- 3) Informálisan fogalmazva jelölje  $t$  a még megvizsgálandó elemek számát. Ezt egy esetszétválasztással adhatjuk meg:

$$t = \begin{cases} v - u + 1, & \text{ha } \neg l; \\ 0, & \text{ha } l. \end{cases}$$

- 4) A ciklusmag legyen egy szekvencia, melynek közbülső feltétele:

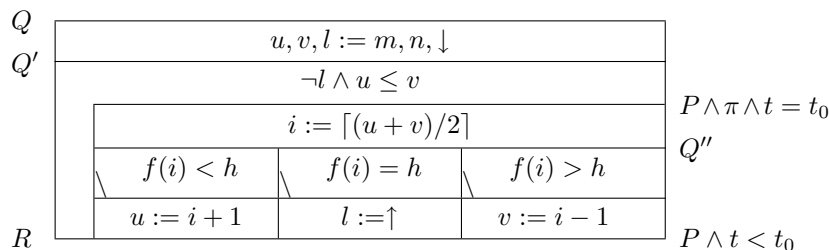
$$Q'' = (Q \wedge [u..v] \subseteq [m..n] \wedge \forall j \in [m..n] \setminus [u..v] : f(j) \neq h \wedge \\ \neg l \wedge (i \in [u..v]))$$

Ekkor a szekvencia első fele lehetne az  $i \in [u..v]$  értékkiválasztás. Hatékony-sági szempontokat is figyelembe véve azonban válasszuk az intervallum köz-zépső elemét:  $i := \lceil (u + v)/2 \rceil$ . A ciklusmag második felében három eset lehetséges:

- $f(i) < h$ : ekkor az  $u := i + 1$  értékadás az invariánst megtartja;
- $f(i) = h$ : ekkor megtaláltuk a keresett elemet, tehát  $l := igaz$ ;
- $f(i) > h$ : ekkor a  $v := i - 1$  értékadás az invariánst megtartja.

- 5) Egyszerűen ellenőrizhető, hogy a fenti elágazás mindhárom ága csökkenti a terminálófüggvényt.

**Tétel:** Az alábbi, struktogram formában megadott program megoldása a fent specifikált feladatnak:



### 12.4.2. Visszalépéses keresés

Legyen  $N \in \mathbb{N}$ , és  $N > 1$ . Legyenek  $U_i$  ( $i \in [1..N]$ ) tetszőleges véges, legalább kételemű halmazok ( $1 < \sigma_i = |U_i| < \infty$ ).  $U = U_1 \times \dots \times U_N$ .

Legyen  $\varrho : U \rightarrow \mathbb{L}$ , amely felbontható  $\varrho_i : U \rightarrow \mathbb{L}$  ( $i \in [0..N]$ ) tulajdonságok sorozatára az alábbi módon:

1.  $\varrho_0 = igaz$ ;
2.  $\forall i \in [0..N - 1] : \forall u \in U : \varrho_{i+1}(u) \rightarrow \varrho_i(u)$ ;
3.  $\forall i \in [1..N] : \forall u, v \in U : (\forall j \in [1..i] : u_j = v_j) \rightarrow \varrho_i(u) = \varrho_i(v)$ ;
4.  $\varrho = \varrho_N$ .

A feladat annak eldöntése, hogy létezik-e olyan elem  $U$ -ban, amelyre teljesül a  $\varrho$  feltétel, és ha igen, adjunk meg egy ilyen elemet.

$$A = U \times \mathbb{L}$$

$$\quad \quad \quad u \quad l$$

$$B = \{\mathcal{X}\}$$

$$Q : Igaz$$

$$R : (l = \exists v \in U : \varrho(v) \wedge l \rightarrow (u \in U \wedge \varrho(u)))$$

Számozzuk meg  $U$  elemeit a következő módon. Minden  $U_i$  halmaz elemeit számozzuk meg nullától  $\sigma_i - 1$ -ig. Ezután  $U$  minden  $u$  eleméhez van egy  $(i_1, \dots, i_N)$  rendezett  $N$ -es, amelyre  $u = (u_{i_1}, \dots, u_{i_N})$ , ahol  $0 \leq i_1 < \sigma_1, \dots, 0 \leq i_N < \sigma_N$ . E megszámozás egy lexikografikus rendezést definiál  $U$ -n.

Legyen  $\mathcal{N} = [0..\sigma_1 - 1] \times \dots \times [0..\sigma_N - 1]$ . Ekkor a fenti megszámozás egy bijekciót létesít  $\mathcal{N}$  és  $U$  között. Jelölje ezt az  $\mathcal{N} \rightarrow U$  leképezést  $\varphi$ .

Vegyük észre, hogy az  $\mathcal{N}$  halmaz elemei felfoghatók vegyes alapú számrendszerben felírt számként is. Ez alapján egy  $\nu \in \mathcal{N}$   $N$ -es számértéke:

$$f(\nu) = \sum_{i=1}^N \nu_i \cdot \Omega_i, \quad \text{ahol:}$$

$$\Omega_i = \prod_{j=i+1}^N \sigma_j \quad (i \in [1..N]).$$

A bevezetett jelölésekkel a feladatot újra specifikáljuk, és most már megkövetelhetjük azt is, hogy ha létezik keresett tulajdonságú elem, akkor az első ilyen adjuk meg:

$$A = \mathcal{N} \times \mathbb{L}$$

$$B = \{\mathcal{X}\}$$

$$Q: \text{Igaz}$$

$$R: (l = \exists \mu \in \mathcal{N} : \varrho(\varphi(\mu)) \wedge$$

$$l \rightarrow (\varrho(\varphi(\nu)) \wedge \forall \mu \in \mathcal{N} : f(\mu) < f(\nu) \rightarrow \neg \varrho(\varphi(\mu))))$$

Ha nem használjuk ki a  $\varrho$  speciális tulajdonságait, akkor a fenti feladat megoldható lineáris kereséssel, a  $[0..|\mathcal{N}| - 1]$  intervallumon. Vizsgáljuk meg, hogyan használhatnánk ki  $\varrho$  specialitását! Nevezetesen azt, hogy a  $\varrho$  3. tulajdonsága miatt, ha  $\varrho_i(\varphi(\nu))$  igaz,  $\varrho_{i+1}(\varphi(\nu))$  pedig hamis, akkor minden olyan  $\nu' \in \mathcal{N}$ -re, amelyek első  $i + 1$  komponense megegyezik  $\nu$  első  $i + 1$  komponensével,  $\varrho_{i+1}(\varphi(\nu'))$  is hamis lesz.

Legyen  $\varepsilon_0 = (0, \dots, 0) \in \mathcal{N}$  és  $\forall i \in [1..N] : \varepsilon_i \in \mathcal{N}$  olyan, hogy  $\forall j \in [1..N] \setminus \{i\} : \varepsilon_{ij} = 0$  és  $\varepsilon_{ii} = 1$ . Nyilvánvaló, hogy  $f(\varepsilon_0) = 0$ ,  $f(\varepsilon_N) = 1$  és  $\forall j \in [1..N] : f(\varepsilon_j) = \Omega_j$ .

Egészítsük ki a  $\nu$ -t egy „túlcsoordulásbittel”, amelynek 1 értéke azt jelzi, hogy  $\nu$  értéke már nem növelhető.

Terjesszük ki az  $f$  függvényt az alábbi módon:

$$f : \{0, 1\} \times \mathcal{N} \rightarrow \mathbb{N}_0,$$

$$f(\nu_0, \nu) = \nu_0 * \Omega_0 + \sum_{i=1}^n \nu_i \cdot \Omega_i, \quad \text{ahol:}$$

$$\Omega_0 = \prod_{j=1}^N \sigma_j.$$

Ezeket a jelöléseket használva definiáljuk az összeadás műveletét két  $\mathcal{N}$ -beli elem között.

$$\nu' + \nu'' = \nu''' \iff f(\nu') + f(\nu'') = f(\nu_0, \nu''')$$

Most már megfogalmazhatunk egy egyszerű, de fontos állítást, ami alapján kihasználhatjuk a megoldás során  $\varrho$  speciális voltát.



**12.1. állítás:** Legyen  $\nu \in \mathcal{N}$ , valamilyen  $i \in [1..N]$ -re  $\neg \varrho_i(\varphi(\nu))$ , és  $\forall j \in [i + 1..N] : \nu_j = 0$ . Ekkor  $\forall \mu$ -re, ami nagyobb, mint  $\nu$ , és kisebb, mint  $\nu + \varepsilon_i$ ,  $\neg \varrho_i(\varphi(\mu))$ .

A következőkben a vegyes alapú számrendszerbeli számot típusnak tekintve definiálunk két típusműveletet is, a fenti állítást is figyelembe véve.

Az első művelet  $\nu$  megnövelése  $\varepsilon_m$ -mel.

$$A_{növel} = \mathcal{N} \times \mathbb{N}_0 \times \{0, 1\}$$

$$\nu \quad m \quad \nu_0$$

$$B_{növel} = \mathcal{N} \times \mathbb{N}_0$$

$$\nu' \quad m'$$

$$Q_{növel} : (\nu = \nu' \wedge m = m' \wedge m' \in [1..N] \wedge \forall i \in [m' + 1..N] : \nu_i = 0)$$

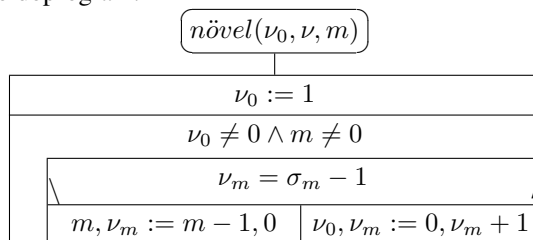
$$R_{növel} : ((\nu_0, \nu) = \nu' + \varepsilon_{m'} \wedge m \in [0..m'] \wedge \forall i \in [m + 1..N] : \nu_i = 0 \wedge \nu_m \neq 0)$$

A megoldás egy ciklus, amelynek invariáns tulajdonsága:

$$P_{növel} : (f(\nu) + \nu_0 * Q_m = f(\nu') + Q_{m'} \wedge m \in [0..m'] \wedge \forall i \in [m + 1..N] : \nu_i = 0 \wedge \forall i \in [1..m - 1] : \nu_i = \nu'_i)$$

Ciklusfeltétel:  $\nu_0 \neq 0 \wedge m \neq 0$ , és a terminálófüggvény:  $\nu_0 + m$ .

Ekkor a megoldóprogram:



Definiáljuk a másik műveletet is, amely amellet, hogy  $\varrho(\varphi(\nu))$ -t eldönti, azt a legkisebb indexet is megadja, amelyre  $\varrho_i(\varphi(\nu))$  hamis.

$$A_{keres} = \mathcal{N} \times \mathbb{N}_0 \times \mathbb{L}$$

$$\nu \quad m \quad l$$

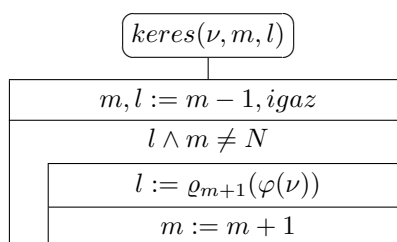
$$B_{keres} = \mathcal{N} \times \mathbb{N}_0$$

$$\nu' \quad m$$

$$Q_{keres} : (\nu = \nu' \wedge m = m' \wedge m' \in [1..N] \wedge \varrho_{m'-1}(\varphi(\nu)))$$

$$R_{keres} : (\nu = \nu' \wedge l = \varrho(\varphi(\nu)) \wedge \neg l \rightarrow (m \in [m'..N] \wedge \varrho_{m-1}(\varphi(\nu)) \wedge \neg \varrho_m(\varphi(\nu)) \wedge l \rightarrow m = N))$$

Ennek a feladatnak a levezetése majdnem azonos a lineáris keresésével.



Megjegyezzük, hogy az utófeltétel utolsó sora esetünkben fölösleges, de a visszalépéses számlálásnál majd építünk rá.

Mindezek birtokában már egyszerű lesz a megoldóprogram levezetése. Legyen az invariáns tulajdonság:

$$\begin{aligned}
 P = & (\forall \mu \in \mathcal{N} : 0 \leq f(0, \mu) < f(\nu_0, \nu) \rightarrow \neg \varrho(\varphi(\mu)) \wedge l = \varrho(\varphi(\nu)) \wedge \\
 & \neg l \rightarrow (\nu_0 = 1 \vee m \in [1..N] \wedge \neg \varrho_m(\varphi(\nu)) \wedge \varrho_{m-1}(\varphi(\nu)) \wedge \\
 & \forall i \in [m+1..N] : \nu_i = 0))
 \end{aligned}$$

A ciklusfeltétel és a terminálófüggvény kézenfekvő módon adódik:  $\neg l \wedge \nu_0 = 0$  és  $\prod_{j=1}^N \sigma_j - f(\nu_0, \nu)$ .

Az invariáns tulajdonság teljesülését a

$$\begin{aligned}
 Q' = & (\nu = \varepsilon_0 \wedge \nu_0 = 0 \wedge \\
 & l = \varrho(\varphi(\varepsilon_0)) \wedge \neg l \rightarrow (m \in [1..N] \wedge \neg \varrho_m(\varphi(\nu)) \wedge \varrho_{m-1}(\varphi(\nu))))
 \end{aligned}$$

garantálja.

$Q'$ -t egy szekvenciával érjük el, amelynek a közbülső tulajdonsága

$$Q'' = (\nu = \varepsilon_0 \wedge \nu_0 = 0 \wedge m = 1).$$

A  $Q''$ -ből következik  $Q_{keres}$ , ezért a szekvencia második tagjának a  $keres(\nu, m, l)$ -t választva teljesül  $R_{keres}$ , amiből pedig következik  $Q'$ . Természetesen a szekvencia első tagjának a  $\nu, \nu_0, m := \varepsilon_0, 0, 1$  értékadást választjuk.

A ciklusmag levezetése is egyszerű.  $P \wedge \pi$ -ből következik  $Q_{növel}$ , ezért a ciklusmag első felének  $növel(\nu_0, \nu, m)$ -t választva igaz lesz  $R_{növel}$ , sőt  $P$ -t és a 12.1. állítást figyelembe véve az is igaz, hogy  $\forall \mu \in \mathcal{N} : 0 \leq f(0, \mu) < f(\nu_0, \nu) \rightarrow \neg \varrho(\varphi(\mu))$ . Abban az esetben, ha  $\nu_0 = 1$  is igaz,  $P$  is teljesül. Ha  $\nu_0 \neq 1$ , akkor még

$$\begin{aligned}
 \neg l = & \varrho(\varphi(\nu)) \wedge \\
 & \neg l \rightarrow (\nu_0 = 1 \vee m \in [1..N] \wedge \neg \varrho_m(\varphi(\nu)) \wedge \varrho_{m-1}(\varphi(\nu)) \wedge \\
 & \forall i \in [m+1..N] : \nu_i = 0)
 \end{aligned}$$

teljesüléséről kell gondoskodni, de ehhez  $R_{keres}$  már elég lenne, az pedig igaz lesz  $keres(\nu, m, l)$  után, mert  $Q_{keres}$  igaz volt.

Mivel  $növel(\nu_0, \nu, m)$  nyilván csökkenti a terminálófüggvény értékét, a következő program megoldása a feladatnak:

$\nu, \nu_0, m := \varepsilon_0, 0, 1$									
$keres(\nu, m, l)$									
$\neg l \wedge \nu_0 = 0$									
<table border="1" style="border-collapse: collapse; width: 100%; text-align: center;"> <tr> <td colspan="2"><math>növel(\nu_0, \nu, m)</math></td> </tr> <tr> <td colspan="2" style="border: 1px solid black;"> <table border="1" style="border-collapse: collapse; width: 100%; text-align: center;"> <tr> <td colspan="2"><math>\nu_0 = 0</math></td> </tr> <tr> <td style="width: 50%;"><math>keres(\nu, m, l)</math></td> <td style="width: 50%;"><math>SKIP</math></td> </tr> </table> </td> </tr> </table>		$növel(\nu_0, \nu, m)$		<table border="1" style="border-collapse: collapse; width: 100%; text-align: center;"> <tr> <td colspan="2"><math>\nu_0 = 0</math></td> </tr> <tr> <td style="width: 50%;"><math>keres(\nu, m, l)</math></td> <td style="width: 50%;"><math>SKIP</math></td> </tr> </table>		$\nu_0 = 0$		$keres(\nu, m, l)$	$SKIP$
$növel(\nu_0, \nu, m)$									
<table border="1" style="border-collapse: collapse; width: 100%; text-align: center;"> <tr> <td colspan="2"><math>\nu_0 = 0</math></td> </tr> <tr> <td style="width: 50%;"><math>keres(\nu, m, l)</math></td> <td style="width: 50%;"><math>SKIP</math></td> </tr> </table>		$\nu_0 = 0$		$keres(\nu, m, l)$	$SKIP$				
$\nu_0 = 0$									
$keres(\nu, m, l)$	$SKIP$								

### 12.4.3. Visszalépéses számlálás

A visszalépéses kereséshez hasonlóan több visszalépéses technikát alkalmazó algoritmus is levezethető, például a visszalépéses számlálás.

$$A = \mathcal{N} \times \mathbb{N}_0$$

$\nu \quad d$

$$B = \{\mathcal{X}\}$$

$$Q : Igaz$$

$$R : \left( d = \sum_{\mu \in \mathcal{N}} \varrho(\varphi(\mu)) \right)$$

A feladat megoldása most is egy ciklus lesz, amelynek az invariáns tulajdonsága

$$P = \left( d = \sum_{\mu < (\nu_0, \nu)} \varrho(\varphi(\mu)) \wedge \right. \\ \left. (\nu_0 = 1 \vee m \in [1..N] \wedge \varrho_{m-1}(\varphi(\nu)) \wedge \forall i \in [m+1..N] : \nu_i = 0) \right).$$

Az invariáns második sora garantálja a 12.1. állítás alkalmazhatóságát. Most az invariáns teljesülését egy szimultán értékadással érhetjük el:  $\nu, \nu_0, m, d := \varepsilon_0, 0, 1, 0$ .

A ciklusfeltétel  $\nu_0 = 0$  lesz. A terminálófüggvény ugyanaz, mint a visszalépéses keresésnél.

Az invariáns tulajdonságból következik  $Q_{keres}$ , ezért  $keres(\nu, m, l)$  után  $R_{keres}$  igaz lesz. Ha  $\neg l$ , akkor  $\left( \sum_{\mu < (\nu_0, \nu)} \varrho(\varphi(\mu)) = \sum_{\mu \leq (\nu_0, \nu)} \varrho(\varphi(\mu)) \right)$ , egyébként  $\left( \sum_{\mu < (\nu_0, \nu)} \varrho(\varphi(\mu)) + 1 = \sum_{\mu \leq (\nu_0, \nu)} \varrho(\varphi(\mu)) \right)$ .

Végül, a  $növel(\nu_0, \nu, m)$  megtartja az invariáns tulajdonságot a 12.1. állítás miatt, és csökkenti a terminálófüggvény értékét. Megjegyezzük, hogy itt használtuk ki azt, hogy a keresési feladatban  $\varrho(\varphi(\nu))$  esetén is meghatároztuk, mi legyen  $m$  értéke.

$\nu, \nu_0, m, d := \varepsilon_0, 0, 1, 0$	
$\nu_0 = 0$	
$keres(\nu, m, l)$	
$l$	
$d := d + 1$	$SKIP$
$növel(\nu_0, \nu, m)$	

## 12.5. Függvényérték kiszámítása

A továbbiakban bizonyos speciális függvények helyettesítési értékének kiszámításával fogunk foglalkozni. Tegyük fel, hogy van egy  $f : X \rightarrow Y$  függvényünk, ahol  $X$  és  $Y$  tetszőleges halmazok. A feladat specifikációja tehát:

$$A = X \times Y$$

$$B = X$$

$$Q : (x = x')$$

$$R : (y = f(x'))$$

Természetesen az  $y := f(x)$  értékadás megoldása a feladatnak. Ha ez az értékadás nem megengedett program, és semmi mást nem tudunk a függvényről, akkor a megoldóprogram előállításáról sem tudunk mondani semmit. Ezért az elkövetkezőkben további feltételezésekkel fogunk élni.

### 12.5.1. Függvénykompozícióval adott függvény kiszámítása

Tegyük fel, hogy  $f = h \circ g$ , ahol  $g : X \rightarrow Z$  és  $h : Z \rightarrow Y$  függvények.

Ekkor a feladat megoldása egy szekvencia lesz. Kibővítjük az állapotteret egy újabb ( $Z$  típusú) komponenssel, melynek változója legyen  $z$ . A szekvencia közbülső feltétele legyen

$$Q' : (z = g(x')).$$

**Tétel:** A kompozíció helyettesítési értékét kiszámító program:

$z := g(x)$
$y := h(z)$

### 12.5.2. Esetsztékválasztással adott függvény kiszámítása

Legyenek  $\pi_1, \pi_2, \dots, \pi_n : X \rightarrow \mathbb{L}$  feltételek és  $g_1, g_2, \dots, g_n : X \rightarrow Y$  függvények, és tegyük fel, hogy a  $\pi_i$  feltételek lefedik az  $X$  halmazt. Legyen  $f : X \rightarrow Y$  a következő:

$$f(x) = \begin{cases} g_1(x), & \text{ha } \pi_1(x); \\ g_2(x), & \text{ha } \pi_2(x); \\ \vdots & \vdots \\ g_n(x), & \text{ha } \pi_n(x). \end{cases}$$

Az elágazás első feltétele  $f$  definíciója miatt teljesül. Az  $y := g_i(x)$  értékadások garantálják a második teljesülését is.

**Tétel:** Az esetsztékválasztással adott függvény értékét kiszámoló program:

$\pi_1(x)$	$\pi_2(x)$	$\dots$	$\pi_n(x)$
$y := g_1(x)$	$y := g_2(x)$	$\dots$	$y := g_n(x)$

### 12.5.3. Rekurzív formulával adott függvény kiszámítása

Legyen  $H$  egy tetszőleges halmaz,  $k > 0$  egy egész szám, továbbá  $F : \mathbb{Z} \times H^k \rightarrow H$  függvény,  $t_0, t_{-1}, \dots, t_{-k+1} \in H$  rögzített, és definiáljuk az  $f \in \mathbb{Z} \rightarrow H$  parciális függvényt az alábbi módon:

$$\begin{aligned} f(m) &= t_0, \\ f(m-1) &= t_{-1}, \\ &\vdots \\ f(m-k+1) &= t_{-k+1}; \end{aligned}$$

továbbá  $\forall i \geq m$ :

$$f(i+1) = F(i+1, f(i), \dots, f(i-k+1)).$$

Feladatunk az, hogy meghatározzuk az  $f$  függvény  $n \geq m$  helyen felvett értékét.

$$A = \mathbb{Z} \times \mathbb{Z} \times H \times H \times \dots \times H$$

$m \quad n \quad y \quad t_0 \quad \dots \quad t_{-k+1}$

$$B = \mathbb{Z} \times \mathbb{Z} \times H \times \dots \times H$$

$m' \quad n' \quad t'_0 \quad \dots \quad t'_{-k+1}$

$$Q : (m = m' \wedge n = n' \wedge n \geq m \wedge t_0 = t'_0 \wedge \dots \wedge t_{-k+1} = t'_{-k+1})$$

$$R : (Q \wedge y = f(n))$$

Ez a feladat is a fejezet elején tárgyalt intervallumos feladatok közé tartozik. Levezetése ennek megfelelően ugyanúgy történik. A ciklus invariáns tulajdonsága, a  $Q'$  és a  $Q''$ :

$$P = (Q \wedge i \in [m..n] \wedge y = f(i), y_{-1} = f(i-1) \wedge \dots \wedge y_{-k+1} = f(i-k+1)),$$

$$Q' = (Q \wedge i = m \wedge y = t_0 \wedge y_{-1} = t_{-1} \wedge \dots \wedge y_{-k+1} = t_{-k+1}),$$

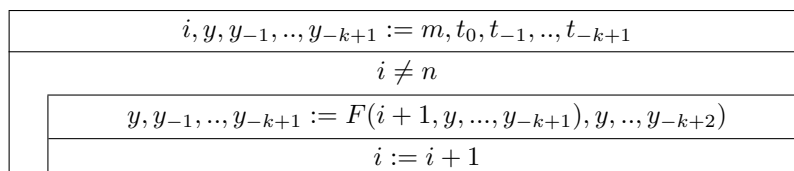
$$Q'' = (Q \wedge i + 1 \in [m..n] \wedge y = f(i+1) \wedge y_{-1} = f(i) \wedge \dots \wedge y_{-k+1} = f(i-k+2)).$$

Az  $i, y, y_{-1}, \dots, y_{-k+1} := 0, t_0, t_{-1}, \dots, t_{-k+1}$  szimultán értékadás  $Q$ -ból  $Q'$ -be jut, és  $P \wedge \pi$ -ből következik

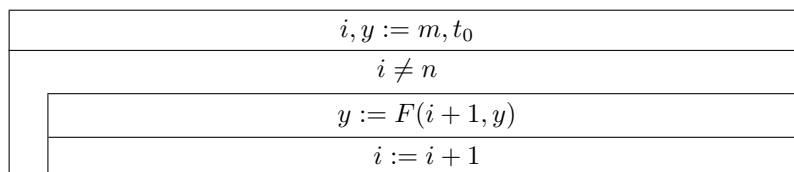
$$lf(y, y_{-1}, \dots, y_{-k+1} := F(i+1, y, \dots, y_{-k+1}), y, \dots, y_{-k+2}), Q'').$$

Variáns függvénynek  $n - i$ -t választva az  $i := i + 1$  értékadás csökkenti azt.

**Tétel:** Az alábbi struktogrammal adott program megoldása a specifikált feladatnak:



Megjegyezzük, hogy a gyakorlatban  $k$  nagyon sokszor egyenlő eggyel, ezért erre az esetre külön is felírjuk a megoldóprogramot.



#### 12.5.4. Elemenként feldolgozható függvény

A továbbiakban legyenek  $H_1$  és  $H_2$  tetszőleges halmazok,  $X$  és  $Y$  pedig az alábbi formában felírható halmazok:

$$\begin{aligned} X &= X_1 \times \dots \times X_n, \\ Y &= Y_1 \times \dots \times Y_m, \end{aligned}$$

ahol  $\forall i \in [1..n] : X_i = \{x \in \wp(H_1) : |x| < \infty\}$ , azaz  $\mathfrak{F}(H_1)$  és  $\forall i \in [1..m] : Y_i = \{y \in \wp(H_2) : |y| < \infty\}$ , azaz  $\mathfrak{F}(H_2)$ . Amint az a fenti leírásból kiderül, az  $X$  az összes olyan halmaz  $n$ -est tartalmazza, amelyeknek minden komponense az adott  $H_1$  halmaz véges részhalmaza. Hasonlóan, az  $Y$  elemei pedig az olyan halmaz  $m$ -esek, amelyek  $H_2$ -beli véges részhalmazokból állnak.

**12.1. DEFINÍCIÓ: TELJESEN DISZJUNKT FELBONTÁS**

Azt mondjuk, hogy  $\bar{x}, \bar{\bar{x}} \in X$  teljesen diszjunkt felbontása  $x \in X$ -nek, ha

- i)  $\forall i \in [1..n] : x_i = \bar{x}_i \cup \bar{\bar{x}}_i$  és
- ii)  $\forall i, j \in [1..n] : \bar{x}_i \cap \bar{\bar{x}}_j = \emptyset$ .

Vegyük észre, hogy ha  $X$  egydimenziós, akkor a teljesen diszjunkt felbontás megegyezik a diszjunkt felbontással, de többdimenziós esetben a teljesen diszjunkt felbontás egy jóval erősebb feltételt jelent.

**12.2. DEFINÍCIÓ: ELEMENKÉNT FELDOLGOZHATÓ FÜGGVÉNY**

Legyen  $f : X \rightarrow Y$ . Ha minden  $x \in X$  minden  $\bar{x}, \bar{\bar{x}}$  teljesen diszjunkt felbontására

- i)  $\forall i \in [1..m] : f_i(\bar{x}) \cup f_i(\bar{\bar{x}}) = f_i(x)$  és
- ii)  $\forall i \in [1..m] : f_i(\bar{x}) \cap f_i(\bar{\bar{x}}) = \emptyset$ ,

akkor  $f$ -et *elemenként feldolgozhatónak* nevezzük.

**Példa:** Legyen  $H$  egy tetszőleges halmaz,  $X_1 = X_2 = Y = \{x \in \wp(H) : |x| < \infty\}$ ,  $f : X_1 \times X_2 \rightarrow Y$ ,  $f((x_1, x_2)) = x_1 \cup x_2$ . Ekkor  $f$  elemenként feldolgozható, ugyanis tekintsük az  $(x_1, x_2)$  halmazpár egy tetszőleges  $\bar{(x_1, x_2)}, \bar{\bar{(x_1, x_2)}}$  teljesen diszjunkt felbontását. Ekkor a teljesen diszjunkt felbontás definíciója alapján:

$$\begin{aligned} \bar{x}_1 \cup \bar{\bar{x}}_1 &= x_1, & \bar{x}_2 \cup \bar{\bar{x}}_2 &= x_2 \\ \bar{x}_1 \cap \bar{\bar{x}}_1 &= \emptyset, & \bar{x}_2 \cap \bar{\bar{x}}_2 &= \emptyset \\ \bar{x}_1 \cap \bar{\bar{x}}_2 &= \emptyset, & \bar{x}_2 \cap \bar{\bar{x}}_1 &= \emptyset \end{aligned}$$

Vizsgáljuk most meg az elemenként feldolgozhatóság két kritériumát:

1.  $f(\overline{(x_1, x_2)}) \cup f(\overline{\bar{\bar{(x_1, x_2)}}}) = (\bar{x}_1 \cup \bar{x}_2) \cup (\bar{\bar{x}}_1 \cup \bar{\bar{x}}_2) = (\bar{x}_1 \cup \bar{\bar{x}}_1) \cup (\bar{x}_2 \cup \bar{\bar{x}}_2) = x_1 \cup x_2 = f((x_1, x_2))$ ,
2.  $f(\overline{(x_1, x_2)}) \cap f(\overline{\bar{\bar{(x_1, x_2)}}}) = (\bar{x}_1 \cup \bar{x}_2) \cap (\bar{\bar{x}}_1 \cup \bar{\bar{x}}_2) = (\bar{x}_1 \cap \bar{\bar{x}}_1) \cup (\bar{x}_1 \cap \bar{\bar{x}}_2) \cup (\bar{x}_2 \cap \bar{\bar{x}}_1) \cup (\bar{x}_2 \cap \bar{\bar{x}}_2) = \emptyset$ .

Tehát a – kétváltozós – unió elemenként feldolgozható függvény.

A következő tételt gyakran fogjuk használni arra, hogy elemenként feldolgozható függvényt definiáljunk.

**12.1. TÉTEL: ELÉGSÉGES FELTÉTEL ELEMENKÉNTI FELDOLGOZHATÓSÁGRA**

Legyen  $X = X_1 \times \dots \times X_n, Y = Y_1 \times \dots \times Y_m, X_i = \mathfrak{F}(H_1) \quad i \in [1..n]$  és  $Y_i = \mathfrak{F}(H_2) \quad i \in [1..m]$ .

Az  $f : X \rightarrow Y$  függvény elemenként feldolgozható ha

$$\forall j \in [1..m] : f_j(x_1, \dots, x_n) = \bigcup_{e \in x_1 \cup \dots \cup x_n} f_j(s_1(e), \dots, s_n(e))$$

és

$$\forall a, b \in x_1 \cup \dots \cup x_n, a \neq b :$$

$$\forall j \in [1..m] : f_j(s_1(a), \dots, s_n(a)) \cap f_j(s_1(b), \dots, s_n(b)) = \emptyset,$$

ahol

$$\forall i \in [1..n] : s_i(e) = \begin{cases} \emptyset & \text{ha } e \notin x_i, \\ \{e\} & \text{ha } e \in x_i. \end{cases}$$

**Bizonyítás:** A tétel egyszerűen következik abból, hogy ha  $\bar{x}, \bar{\bar{x}}$  teljesen diszjunkt felbontása  $x$ -nek, akkor

$$(\bar{x}_1 \cup \dots \cup \bar{x}_n) \cup (\bar{\bar{x}}_1 \cup \dots \cup \bar{\bar{x}}_n) = x_1 \cup \dots \cup x_n$$

és

$$(\bar{x}_1 \cup \dots \cup \bar{x}_n) \cap (\bar{\bar{x}}_1 \cup \dots \cup \bar{\bar{x}}_n) = \emptyset.$$

□

A továbbiakban az elemenként feldolgozható függvények helyettesítési értékének kiszámításával fogunk foglalkozni.

Mielőtt belekezdenénk a feladat specifikálásába és megoldásába, bevezetünk két olyan, halmazokra vonatkozó parciális értékadást, amelyeket aztán a megoldóprogramokban primitív műveletnek tekintünk.

- Legyen  $H$  egy tetszőleges halmaz, és definiáljuk az  $f_{\cup} : \mathfrak{F}(H) \times H \rightarrow \mathfrak{F}(H)$  parciális függvényt:

$$f_{\cup}(h, e) = h \cup \{e\}, \text{ ha } e \notin h.$$

- Ugyanezt a jelölést használjuk akkor is, ha  $f_{\cup} \in \mathfrak{F}(H) \times \mathfrak{F}(H) \rightarrow \mathfrak{F}(H)$  és

$$f_{\cup}(h, g) = h \cup g, \text{ ha } h \cap g = \emptyset.$$

- Hasonlóan, legyen  $H$  egy tetszőleges halmaz, és definiáljuk az  $f_{\setminus} \in \mathfrak{F}(H) \times H \rightarrow \mathfrak{F}(H)$  parciális függvényt:

$$f_{\setminus}(h, e) = h \setminus \{e\}, \text{ ha } e \in h.$$

- Ebben az esetben is, ha  $f_{\setminus} \in \mathfrak{F}(H) \times \mathfrak{F}(H) \rightarrow \mathfrak{F}(H)$  parciális függvény:

$$f_{\setminus}(h, g) = h \setminus g, \text{ ha } g \subseteq h.$$

A fenti függvényeket kiszámító  $h := f_{\cup}(h, x)$ , illetve  $h := f_{\setminus}(h, x)$  parciális értékadásokat a továbbiakban  $h := h \tilde{\cup} x$ -szel és  $h := h \simeq x$ -szel fogjuk jelölni.



**Egyváltozós egyértékű eset**

Először vizsgáljuk meg azt az esetet, amikor mind  $X$ , mind  $Y$  egykomponensű, azaz  $m = 1$  és  $n = 1$ . Ekkor az  $f$  függvény egy halmazhoz egy másik halmazt rendel.

$$A = X \times Y$$

$$B = X$$

$$Q : (x = x')$$

$$R : (y = f(x'))$$

Oldjuk meg a feladatot ciklussal: az invariánsban azt fogalmazzuk meg, hogy az  $x$  halmaz a még feldolgozandó elemeket, az  $y$  halmaz pedig a már feldolgozott elemek  $f$  szerinti képeinek unióját tartalmazza, azaz

$$P = (y \cup f(x) = f(x') \wedge y \cap f(x) = \emptyset).$$

Vizsgáljuk meg a ciklus levezetési szabályának feltételeit:

- 1)  $Q$ -ból az  $y = \emptyset$  fennállása esetén következik  $P$ , ezért a ciklus elé az  $y := \emptyset$  értékadás kerül.
- 2) Az invariánsból  $f(x) = \emptyset$  esetén következik az utófeltétel, ám ez jó eséllyel nem egy megengedett feltétel (hiszen éppen  $f$ -et akarjuk kiszámítani). Vegyük észre azonban, hogy  $f$  elemenkénti feldolgozhatósága miatt  $x = \emptyset$  esetén  $f(x) = \emptyset$  is teljesül (az üres halmaznak két üres halmaz egy teljesen diszjunkt felbontása). Tehát a ciklusfeltétel:  $\pi = (x \neq \emptyset)$ .
- 3) Ha (a ciklusfeltétel szerint)  $x$  nem üres, akkor terminálófüggvénynek választható  $x$  számossága, azaz  $t = |x|$ .
- 5)  $x$  számosságát úgy tudjuk csökkenteni, ha elhagyunk belőle egy – benne levő – elemet. Ezt megtehetjük az imént bevezetett parciális értékadással:  $x := x \simeq e$ .
- 4) Írjuk fel a fenti parciális értékadás  $P$ -re vonatkozó leggyengébb előfeltételét:

$$Q'' : (y \cup f(x \setminus \{e\}) = f(x') \wedge y \cap f(x \setminus \{e\}) = \emptyset \wedge e \in x)$$

Jól látható, hogy ez  $P \wedge \pi$ -ből nem következik. Vegyük azonban észre, hogy ha  $e$  egy  $x$ -beli elem, akkor  $\{e\}$  és  $x \setminus \{e\}$   $x$ -nek egy teljesen diszjunkt felbontása, tehát  $f$  elemenkénti feldolgozhatósága miatt:

$$\begin{aligned} f(\{e\}) \cup f(x \setminus \{e\}) &= f(x) \\ f(\{e\}) \cap f(x \setminus \{e\}) &= \emptyset \end{aligned}$$

Így az  $y := y \tilde{\cup} f(\{e\})$  értékadás már majdnem elegendő, hiszen

$$lf(y := y \tilde{\cup} f(\{e\}), Q'') = (y \cup f(\{e\}) \cup f(x \setminus \{e\}) = f(x') \wedge (y \cup f(\{e\})) \cap f(x \setminus \{e\}) = \emptyset \wedge e \in x).$$

Ezt a feltételt összevetve  $P \wedge \pi$ -vel látható, hogy már csak az  $e \in x$  állítást kell teljesítenünk. Ezt viszont megtehetjük az  $e : \in x$  érték kiválasztással, amelynek a fenti állításra vonatkozó leggyengébb előfeltétele  $P \wedge \pi$ .

**Tétel:** Ekkor a következő program megoldása a specifikált feladatnak:

$Q$	$y := \emptyset$	$P \wedge \pi \wedge$
$Q'$	$x \neq \emptyset$	$t = t_0$
	$e : \in x$	$Q'''$
	$y := y \tilde{\cup} f(\{e\})$	$Q''$
$R$	$x := x \simeq e$	$P \wedge t < t_0$

**Bizonyítás:** A tétel a fenti levezetésből következik.

#### Kétféltérő egyértékű eset

Legyen  $f : X \times Y \rightarrow Z$  ( $X, Y, Z \subseteq \mathfrak{F}(H)$ ) elemenként feldolgozható függvény.

$$A = X \times Y \times Z$$

$$x \quad y \quad z$$

$$B = X \times Y$$

$$x' \quad y'$$

$$Q : (x = x' \wedge y = y')$$

$$R : (z = f(x', y'))$$

**Tétel:** Ekkor a következő program megoldása a specifikált feladatnak:

$z := \emptyset$		
$x \neq \emptyset \vee y \neq \emptyset$		
$e : \in (x \cup y)$		
$e \in x \wedge e \notin y$	$e \in x \wedge e \in y$	$e \notin x \wedge e \in y$
$z := z \tilde{\cup} f(\{e\}, \emptyset)$	$z := z \tilde{\cup} f(\{e\}, \{e\})$	$z := z \tilde{\cup} f(\emptyset, \{e\})$
$x := x \simeq e$	$x := x \simeq e$	$y := y \simeq e$
	$y := y \simeq e$	

**Bizonyítás:** A tétel az egyváltozós esettel analóg módon levezethető, ha invariáns tulajdonságnak az alábbi állítást:

$$P = (z \cup f(x, y) = f(x', y') \wedge z \cap f(x, y) = \emptyset \wedge \\ (x' \setminus x) \cap y = \emptyset \wedge (y' \setminus y) \cap x = \emptyset),$$

terminálófüggvénynek pedig  $t = |x \cup y|$ -t választjuk.  $\square$

### Egyváltozós kétértékű eset

Legyen  $f : X \rightarrow Y \times Z$  ( $X, Y, Z \subseteq \mathfrak{F}(H)$ ),  $f_1 : X \rightarrow Y$ ,  $f_2 : X \rightarrow Z$ ,  $f = (f_1, f_2)$  elemenként feldolgozható függvény.

$$A = X \times Y \times Z \\ \quad \quad \quad x \quad y \quad z$$

$$B = X \\ \quad \quad \quad x'$$

$$Q : (x = x')$$

$$R : (y = f_1(x') \wedge z = f_2(x'))$$

**Tétel:** Ekkor az alábbi program megoldása a specifikált feladatnak:

$y, z := \emptyset, \emptyset$
$x \neq \emptyset$
$e := x$
$y, z := y \tilde{\cup} f_1(\{e\}), z \tilde{\cup} f_2(\{e\})$
$x := x \simeq e$

**Bizonyítás:** A tétel levezetése az egyértékű esettől csak az invariáns tulajdonság megválasztásában tér el:

$$P : (y \cup f_1(x) = f_1(x') \wedge y \cap f_1(x) = \emptyset \wedge \\ z \cup f_2(x) = f_2(x') \wedge z \cap f_2(x) = \emptyset)$$

A terminálófüggvény marad, és a levezetés lépései is megegyeznek.  $\square$

### Általános változat

Legyenek  $n, m$  rögzített természetes számok,  $f : X_1 \times \dots \times X_n \rightarrow Y_1 \times \dots \times Y_m$  ( $X_i, Y_j \in \mathfrak{F}(H)$ , ( $i \in [1..n], j \in [1..m]$ )) elemenként feldolgozható függvény, és legyenek az  $f_j : X_1 \times \dots \times X_n \rightarrow Y_j$  ( $j \in [1..m]$ ) függvények az  $f$  komponensfüggvényei, azaz  $f = (f_1, \dots, f_m)$ .

$$A = X_1 \times \dots \times X_n \times Y_1 \times \dots \times Y_m \\ \quad \quad \quad x_1 \quad \quad \quad x_n \quad y_1 \quad \quad \quad y_m$$

$$B = X_1 \times \dots \times X_n \\ \quad \quad \quad x'_1 \quad \quad \quad x'_n$$

$$Q : (x_1 = x'_1 \wedge \dots \wedge x_n = x'_n)$$

$$R : (y_1 = f_1(x'_1, \dots, x'_n) \wedge \dots \wedge y_m = f_m(x'_1, \dots, x'_n))$$

**Tétel:** Ekkor az alábbi program megoldása a specifikált feladatnak:

$y_1, \dots, y_m := \emptyset, \dots, \emptyset$		
$\bigcup_{i=1}^n x_i \neq \emptyset$		
$e := \bigcup_{i=1}^n x_i$		
\ ... \	$\forall i \in I : e \in x_i \wedge \forall i \in [1..n] \setminus I : e \notin x_i$	\ ... \
...	$y_1, \dots, y_m :=$ $y_1 \tilde{\cup} f_1(s_1(e), \dots, s_n(e)), \dots, y_m \tilde{\cup} f_m(s_1(e), \dots, s_n(e))$ $\forall i \in I : x_i := x_i \simeq e$	...

ahol  $I \subseteq [1..n]$  és  $I \neq \emptyset$ ,

$$\forall i \in [1..n] : s_i(e) = \begin{cases} \{e\}, & \text{ha } i \in I; \\ \emptyset, & \text{ha } i \notin I. \end{cases}$$

Az elágazás „ágainak” száma  $2^n - 1$ .

**Bizonyítás:** A tétel az alábbi invariáns tulajdonsággal és terminálófüggvénnyel könnyen levezethető.

$$P : (\forall j \in [1..m] : (y_j \cup f_j(x_1, \dots, x_n) = f_j(x'_1, \dots, x'_n) \wedge y_j \cap f_j(x_1, \dots, x_n) = \emptyset \wedge \forall i, k \in [1..n] : (x'_i \setminus x_i) \cap x_k = \emptyset))$$

$$t = \left| \bigcup_{i=1}^n x_i \right|$$

□

Megjegyzés: a fenti programozási tételből következik, hogy a 12.1 tétel feltétele nemcsak elégséges, de szükséges feltétele is az elemenkénti feldolgozhatóságnak.

## 12.6. Feladatok

**12.1.** Adott egy  $f : \mathbb{Z} \rightarrow \mathbb{Z}$  függvény. Határozzuk meg, hogy a függvény melyik két pontban veszi fel a maximumát és a minimumát az  $[m..n]$  intervallumban!

**12.2.** Határozzuk meg az  $x$  és az  $y$  természetes számok legnagyobb közös osztóját!

**12.3.** Határozzuk meg az  $x$  és az  $y$  természetes számok legkisebb közös többszörösét!

- 12.4.** Határozzuk meg  $-a$  hatványozás műveletének használata nélkül – az  $x$  szám  $n$ -edik hatványát!
- 12.5.** Döntsük el, hogy a  $k$  természetes szám osztja-e az  $x$  természetes számot!
- 12.6.** Döntsük el, hogy az  $x$  természetes szám prímszám-e!
- 12.7.** Adottak az  $x$  és  $y$  vektorok ( $x.dom = y.dom$ ). Képezzük az  $x + y$  és az  $x - y$  vektorok skaláris szorzatát!
- 12.8.** Határozzuk meg az  $x$  vektor elemeinek összegét úgy, hogy a páratlan indexű elemek a negáltjukkal szerepeljenek az összegésben!
- 12.9.** Adott egy egész számokból álló vektor és két egész szám. Állapítsuk meg, hogy a két szám előfordul-e a vektorban, és ha igen, akkor melyik előbb!
- 12.10.** Adott az egész számokat tartalmazó  $x$  vektor. Permutáljuk a vektor elemeit (helyben!) úgy, hogy a vektor egy eleme a monoton rendezés szerinti helyére kerüljön, azaz ne előzze meg őt nála nagyobb elem, és utána ne legyen nála kisebb!
- 12.11.** Adott az  $x$  négyzetes mátrix. Határozzuk meg az alsó háromszög elemeinek összegét!
- 12.12.** Adott az  $x$  négyzetes mátrix. Tükrözzük (transzponáljuk) a mellékátlójára helyben (azaz az eredmény  $x$ -ben keletkezzen)!
- 12.13.** Adott az  $x$  négyzetes mátrix. Tükrözzük (transzponáljuk) a főátlójára helyben (azaz az eredmény  $x$ -ben keletkezzen)!
- 12.14.** Adott az  $x$  vektor. Számítsuk ki a  $b$  vektor ( $b.dom \leq x.dom$ ) elemeinek értékét úgy, hogy  $b$   $i$ -edik eleme az első  $i$  darab  $x$ -beli elem összege legyen!
- 12.15.** Adottak az  $n$  és  $k$  számok. Számítsuk ki  $\binom{n}{k}$  értékét!
- 12.16.** Az  $x$  egész számokból álló vektor egy decimális szám számjegyeit tartalmazza helyi érték szerint csökkenő sorrendben. Számítsuk ki az ábrázolt szám értékét!
- 12.17.** Adott egy természetes szám. Az  $x$  egészértékű vektorban állítsuk elő a szám számjegyeit helyi érték szerint csökkenő sorrendben, és adjuk meg azt is, hogy a szám hány számjegyből áll!
- 12.18.** Az  $x$  egész számokból álló vektor egy decimális szám számjegyeit tartalmazza helyi érték szerint csökkenő sorrendben. Állítsuk elő  $x$ -ben az eredetinél eggyel nagyobb szám ugyanilyen ábrázolását, illetve mondjuk meg, ha túlcsoportulás volt!

- 12.19.** Az  $x$  egész számokból álló vektor egy decimális szám számjegyeit tartalmazza helyiérték szerint csökkenő sorrendben. Állítsuk elő  $x$ -ben az eredetnél eggyel kisebb szám ugyanilyen ábrázolását, illetve mondjuk meg, ha alulcsordulás volt!
- 12.20.** Az azonos értelmezési tartományú  $x$  és  $y$  vektorok egy-egy  $x.dom$  jegyű decimális szám számjegyeit tartalmazzák. A kisebb indexeken vannak 10 magasabb hatványainak együtthatói. Képezzük a  $z$  vektorban a számok összegét, illetve állapítsuk meg, hogy keletkezett-e túlcsoordulás!
- 12.21.** Adott az  $x$  vektor, melynek elemei  $k^2$ -es számrendszerbeli számjegyek. Állítsuk elő az így reprezentált szám  $k$ -s számrendszerbeli jegyeit az  $y$  vektorban (a szám magasabb helyi értékeit a vektor alacsonyabb indexű helyein találjuk)!
- 12.22.** Adott az  $x$  vektor, melynek elemei  $k$ -s számrendszerbeli számjegyek. Állítsuk elő az így reprezentált szám  $k^2$ -es számrendszerbeli jegyeit az  $y$  vektorban (a szám magasabb helyi értékeit a vektor alacsonyabb indexű helyein találjuk)!
- 12.23.** Egy vektor egy egész számot reprezentál úgy, hogy a vektor minden eleme a szám egy decimális számjegyét tartalmazza. Csökkentsük ezt a számot egy adott helyi értéken egy  $0 - -9$  közötti értékkel!
- 12.24.** Határozzuk meg az  $x$  természetes szám decimális alakja számjegyeinek számát!
- 12.25.** Határozzuk meg az  $x$  természetes szám decimális alakja számjegyeinek összegét!
- 12.26.** Adott egy egész számokból álló vektor. Rendezzük a vektor elemeit (helyben) csökkenő sorrendbe!
- 12.27.** Adott az  $x$  és  $b$  vektor úgy, hogy  $b$  az  $x$  indexeiből veszi fel elemeit. Az  $x$  vektor minden  $b_j$ -edik elemének helyére írjunk nullát!

## 13. fejezet

# Transzformációk

Ebben a fejezetben azzal fogunk foglalkozni, hogyan lehet a programozási tételeket konkrét feladatok esetében alkalmazni.

Először egy olyan feladatot nézünk meg, amelynek látszólag semmi köze a programozáshoz: oldjuk meg az  $x^2 - 5x + 6 = 0$  egyenletet. Az egyik lehetőség az, hogy valamilyen gondolatmenettel (gyökök és együtthatók közötti összefüggés vagy teljes négyzetté alakítás)  $(x - 3)(x - 2) = 0$  alakra hozzuk, és azután megoldjuk az  $x - 3 = 0$  és az  $x - 2 = 0$  egyenleteket. Vagyis az eredeti feladat helyett két másik, de egyszerűbb feladatot kell megoldani. Pontosan ez a *levezetés* lényege is. Ezt a módszert alkalmaztuk az előző fejezetben.

A másik lehetőség az, hogy megoldjuk a következő általános feladatot:  $ax^2 + bx + c = 0$ . Belátjuk azt a tételt, hogy ha  $a \neq 0$ , akkor a gyököket a közismert gyökképlettel kapjuk meg:

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Ezután alkalmazzuk a tételt: a képletben  $a$  helyébe 1-et,  $b$  helyébe  $-5$ -öt és  $c$  helyébe 6-ot helyettesítve megkapjuk a gyököket. A programozásban ezt a módszert nevezzük *visszavezetésnek*.

A másodfokú egyenlet gyökképletét gyakran alkalmazhatjuk nem másodfokú egyenletek megoldása esetén is, például legyen a megoldandó egyenlet:  $x^4 - 5x^2 + 4 = 0$ . Az  $x^2 = y$  helyettesítést alkalmazva  $y$ -ra kapunk egy másodfokú egyenletet, amit megoldva már csak az  $x^2 = 1$  és  $x^2 = 4$  egyenleteket kell megoldanunk. Azt mondhatjuk, hogy egy *transzformáció* segítségével oldottuk meg a feladatot.

A továbbiakban először a visszavezetéssel foglalkozunk.

### 13.1. Visszavezetés

A legegyszerűbb esetben a megoldandó feladat és a programozási tételek állapottere azonos, legfeljebb a jelölésben különbözik, és átjelölés után az elő- és utófeltételek is megegyeznek. Ekkor a megoldóprogramot is egyszerű átjelöléssel kapjuk meg. Ez az eset azonban elég ritka, erre szorítkozva a visszavezetés használhatósága nagyon korlátozott lenne.

Emlékeztetünk azonban arra, hogy a megoldás 5.1. definíciója szerint a feladat és a program állapottere különböző is lehet; másrészt felhasználhatjuk azt az összefüggést, hogy egy szigorúbb feladat megoldása egyben megoldása a gyengébb feladatnak is, lásd a 2.1. állítást.

Ebből következik, hogy ha a tétel utófeltétele szigorúbb, mint a feladaté, vagy ha a tétel állapotterében szerepelnek olyan komponensek, amelyek a feladatában nem (a kiterjesztés 4.1. definíciója szerint a tétel ebben az esetben is szigorúbb), akkor a tétel programja, esetleges átjelölés után, megoldása lesz a feladatnak. Ilyen esetekben mondjuk azt, hogy *szigorítással* vezetünk vissza.

Ha a feladat állapottere tartalmaz olyan komponenseket, amelyek nincsenek benne a tétel állapotterében, így a 4.2. definíció szerint a tétel megoldóprogramja nem változtatja meg őket, az ilyen változóknak a feladat elő- és utófeltételében egy-egy paraméterváltozóval rögzítettnek kell lenniük. Az ilyen változókat a *visszavezetés paramétereinek* nevezzük, és rendszerint a tétel „adott függvényeit” határozzák meg.

Előfordul, hogy a tétel feladatának értelmezési tartománya szűkebb, mint a megoldandó feladaté, ebben az esetben egy *feltétel* segítségével leszűkítjük az értelmezési tartományt. Valójában itt levezetési lépést alkalmazunk, a feladat megoldása egy elágazás, ha a feltétel teljesül, a megoldóprogram a tétel programja, egyébként vagy más tételt, vagy levezetést alkalmazunk.

### 13.2. Egyszerű programtranszformációk

A következő transzformációk segítségével olyan, az eredetivel ekvivalens programot hozunk létre, amely valamilyen szempontból előnyösebb számunkra, mint az eredeti. A szempont alapvetően kétféle lehet. Az egyik: a transzformált program megengedett programkonstrukció, míg az eredeti nem az. Gyakran fordul elő, hogy visszavezetéssel nem megengedett programot kapunk. A másik szempont is a visszavezetéshez kapcsolódik: sokszor a visszavezetéssel kapott program nagyon ügyetlen, és a transzformációval ezen javítunk.

**Nem megengedett feltétel kitranszformálása elágazásból.** Legyen  $S \subseteq A \times A^{**}$ ,  $S = IF(\pi_1 : S_1, \dots, \pi_n : S_n)$ ,  $A = A_1 \times \dots \times A_m$ . Konstruáljuk az  $S' \subseteq A' \times A'^{**}$  programot az alábbi módon:

$$A' = A_1 \times \dots \times A_m \times \mathbb{L} \times \dots \times \mathbb{L}$$

$$a_1 \qquad \qquad a_m \qquad l_1 \qquad \qquad l_n$$



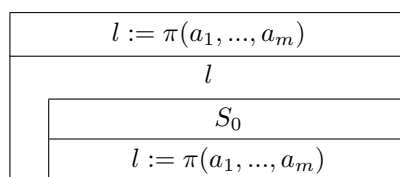
$l_1, \dots, l_n := \pi_1(a_1, \dots, a_m), \dots, \pi_n(a_1, \dots, a_m)$
$IF(l_1 : S_1, \dots, l_n : S_n)$

Ekkor az  $S'$  program ekvivalens  $S$ -sel  $A$ -n.

**Nem megengedett ciklusfeltétel kitranszformálása.** Legyen  $S \subseteq A \times A^{**}$ ,  $S = DO(\pi : S_0)$ ,  $A = A_1 \times \dots \times A_m$ . Konstruáljuk az  $S' \subseteq A' \times A'^{**}$  programot az alábbi módon:

$$A' = A_1 \times \dots \times A_m \times \mathbb{L}$$

$a_1 \qquad \qquad \qquad a_m \qquad \qquad \qquad l$

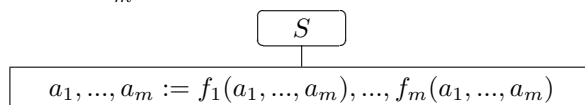


Ekkor az  $S'$  program ekvivalens  $S$ -sel  $A$ -n.

**Szimultán értékadás helyettesítése egyszerű értékadásokkal.** Legyen  $S \subseteq A \times A^{**}$  a következő szimultán értékadás:

$$A = A_1 \times \dots \times A_m$$

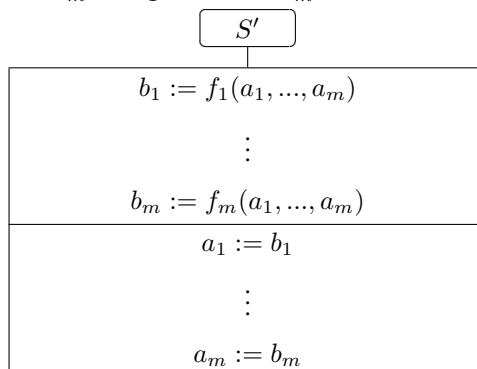
$a_1 \qquad \qquad \qquad a_m$



Konstruáljuk az  $S' \subseteq A' \times A'^{**}$  programot az alábbi módon:

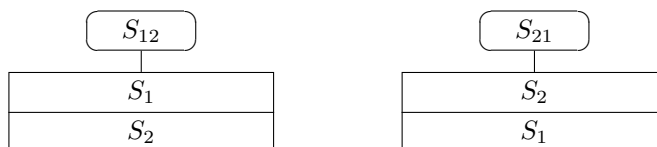
$$A' = A_1 \times \dots \times A_m \times A_1 \times \dots \times A_m$$

$a_1 \qquad \qquad \qquad a_m \qquad \qquad \qquad b_1 \qquad \qquad \qquad b_m$

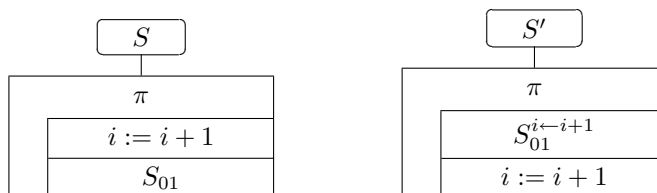


Ekkor az  $S'$  program ekvivalens  $S$ -sel  $A$ -n.

**Szekvencia sorrendjének felcserélése.** Ha  $S_{12} = (S_1; S_2)$ ,  $S_{21} = (S_2; S_1)$ , és az állapotér azon komponenseit, amelyektől az  $S_1$  függ,  $S_2$  nem változtatja meg, és viszont, akkor  $S_{12}$  ekvivalens  $S_{21}$ -gyel.



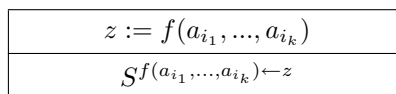
**Ciklusmagbéli szekvencia sorrendjének felcserélése.** Legyen  $S = DO(\pi : S_0)$ ,  $S_0 = (i := i + 1; S_{01})$ , és tegyük fel, hogy az  $i$  konstans  $S_{01}$ -ben. Legyen továbbá  $S' = (S_{01}^{i \leftarrow i+1}; i := i + 1)$ .



Ekkor  $S$  ekvivalens  $S'$ -vel.

**Függvény helyettesítése változóval.** Legyen  $S \subseteq A \times A^{**}$ ,  $A = A_1 \times \dots \times A_m$ , és legyen  $f$  egy, az  $A_{i_1} \times \dots \times A_{i_k}$  altér felett definiált függvény, melynek értékkészlete  $H$ . Tegyük fel továbbá, hogy az  $a_{i_1}, \dots, a_{i_k}$  változók konstansok  $S$ -ben, és készítsük el az  $S' \subseteq A' \times A'^{**}$  programot az alábbi módon:

$$A' = \begin{matrix} A_1 & \times & \dots & \times & A_m & \times & H \\ a_1 & & & & a_m & & z \end{matrix}$$



Ekkor  $S'$  ekvivalens  $S$ -sel  $A$ -n.

**Rekurzívan definiált függvény helyettesítése.** Legyen  $H$  egy tetszőleges halmaz,  $k > 0$  egy egész szám, továbbá  $F : \mathbb{Z} \times H^k \rightarrow H$  függvény,  $t_0, t_{-1}, \dots, t_{-k+1} \in H$

rögzített, és definiáljuk az  $f \in \mathbb{Z} \rightarrow H$  parciális függvényt az alábbi módon:

$$\begin{aligned} f(m) &= t_0, \\ f(m-1) &= t_{-1}, \\ &\vdots \\ f(m-k+1) &= t_{-k+1}, \end{aligned}$$

továbbá  $\forall i \geq m$ :

$$f(i+1) = F(i+1, f(i), \dots, f(i-k+1)).$$

Tekintsük az alábbi  $S$  programot:

$i := m$
$S_1$
$\pi$
$S_0$
$i := i + 1$

ahol  $i$  mind  $S_0$ -ban, mind  $S_1$ -ben konstans,  $S_0$ -ban hivatkozás történik  $f(i+1)$  értékre,  $S_1$ -ban legfeljebb  $f(i)$ -re és  $\pi = i \neq n(\wedge \dots)$ . Ekkor  $S$  ekvivalens az alábbi programmal:

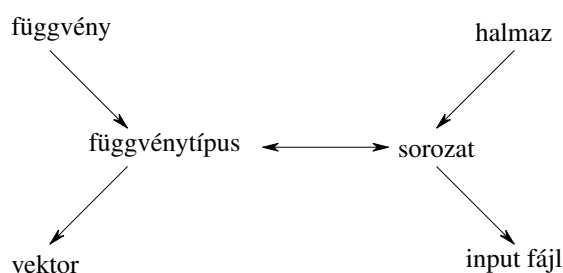
$i, z, z_{-1}, \dots, z_{-k+1} := m, t_0, t_{-1}, \dots, t_{-k+1}$
$S_1$
$\pi$
$z, z_{-1}, \dots, z_{-k+1} := F(i+1, f(i), \dots, f(i-k+1)), z, \dots, z_{-k+2}$
$S_0^{f(i+1) \leftarrow z}$
$i := i + 1$

Mindegyik esetben egyszerűen belátható a programfüggvények azonossága.

### 13.3. Típustranzformációk

*Típustranzformációról* akkor beszélhetünk, ha az állapottér bizonyos komponenseit valami kapcsolódó típusra cseréljük.

A programozási tételek – és általában véve a (feladat, megoldóprogram) párok – az alábbiakban bemutatásra kerülő tranzformációkon keresztül általánosíthatók, ha az állapotterek közötti tranzformáció tulajdonképpen típustranzformáció. Ekkor az ábrán látható valamely típuson megoldott program egyszerű szabályok segítségével átvihető egy kapcsolódó típusra.



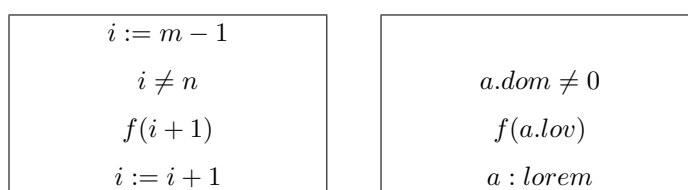
13.1. ábra. Típusok közötti kapcsolatok

**A programozási tételek átírása más típusra.** Az alábbi sémák a programozási tételek típusok közötti transzformációjakor elvégzendő helyettesítéseket definiálják. A transzformáció úgy történik, hogy az első típus adott műveletét a második típus megfelelő (azonos sorban levő) műveletére cseréljük.

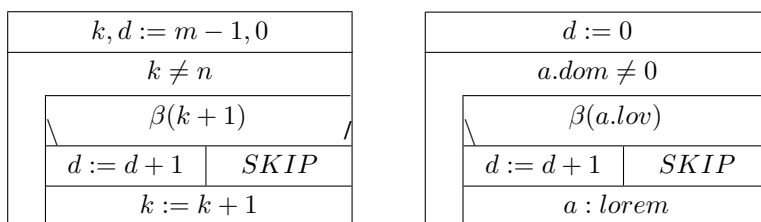
Induljunk el az intervallumon értelmezett függvényektől. A függvénytípusra való áttérés lényegében nem igényel transzformációt, hiszen az  $f : func(\mathbb{Z}, H)$  értéke egy  $[lob(f)..hib(f)]$  intervallumon értelmezett  $\mathbb{Z} \rightarrow H$  függvény. Ha olyan programról van szó, ahol a függvény rögzített – a tételeink ilyenek –, akkor egyúttal vektorra is transzformáltunk.

- **Intervallum felett definiált függvényről sorozatra**

Az  $s : seq(H)$  sorozat is tekinthető az  $[1..dom(s)]$  intervallumon értelmezett  $\mathbb{Z} \rightarrow H$  függvénynek, de a műveletek között nem szerepel az indexelés. Ezért nem is minden programunk írható át sorozatra. Tegyük föl, hogy egy programban mindig csak az  $f(i+1)$ -re hivatkozunk, ahol  $f$  az  $[m..n]$  intervallumon értelmezett függvény. Az első hivatkozást megelőzően  $i := m - 1$ , utána pedig az  $i := i + 1$  programok kivételével minden más programban  $i$  konstans. Ha ezek a feltételek teljesülnek, akkor garantálható a következő invariáns tulajdonság:  $s = \langle f(i+1), \dots, f(n) \rangle$ . Azaz a következő helyettesítéssel transzformálhatjuk a programot:

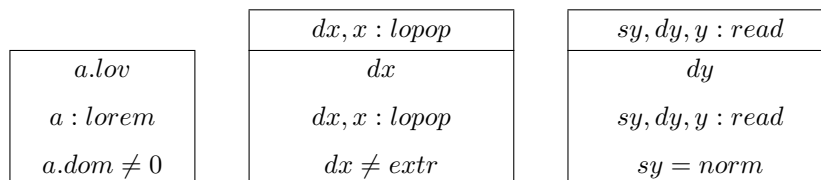


Példa (számlálás tétele):



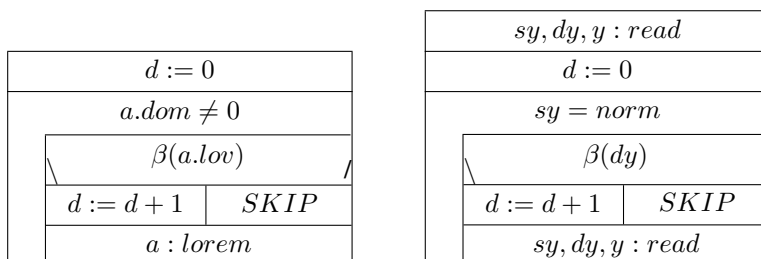
- **Sorozatról szekvenciális input fájlra** ( $dx, x : lopop$  vagy  $sy, dy, y : read$ )

A két típus közötti megfeleltetés: ha az  $a$  sorozat nem üres, akkor megegyezik a  $loext(x, dx)$  és  $loext(y, dy)$  sorozatokkal, míg üres sorozat esetén az  $x$  és  $y$  sorozatokkal.



A transzformált program egy plusz *lopop* (vagy *read*) művelettel kezdődik, ezért ezt a transzformációt *előreolvasási technikának* nevezzük.

Példa (számlálás tétele):



- **Halmazról sorozatra** (egy input halmaz/sorozat esetén)

A két típus közötti megfeleltetés: az  $a$  és  $b$  sorozatok tagjai felsorolják az  $x$  és  $y$  halmazok elemeit.

$y := \emptyset$
$x \neq \emptyset$
$e \in x$
$e$
$y \tilde{\cup}$
$x \simeq$

$b := \langle \rangle$
$a.dom \neq 0$
$a.lov$
$b : hiext$
$a : lorem$

A fenti megfeleltetést alkalmazhatjuk az elemenkénti feldolgozás esetén is, feltéve az  $f$  függvényről, hogy egy egyelemű halmazhoz egyelemű halmazt rendel hozzá. Ez viszonylag ritkán teljesül, az azonban már nagyon gyakori, hogy legfeljebb egyelemű a kép. Ebben az esetben  $b : hiext$  helyébe egy elágazás,  $IF(d = \emptyset : SKIP, d \neq \emptyset : (b : hiext(d)))$  kerül.

Megjegyezzük, hogy az  $e \in x$  művelet helyett az  $e := a.lov$  értékadást is írhatuk volna, ami szintén szigorítást jelent az eredeti érték kiválasztáshoz képest.

Példa (egyváltozós egyértékű elemenkénti feldolgozás):

$y := \emptyset$
$x \neq \emptyset$
$e \in x$
$y := y \tilde{\cup} f(\{e\})$
$x := x \simeq e$

$b := \langle \rangle$
$a.dom \neq 0$
$b : hiext(f(\{a.lov\}))$
$a : lorem$

Az előzőekben tárgyalt előreolvasási technika segítségével megkaphatjuk a szekvenciális fájlra vonatkozó változatot is.

Példa (egyváltozós egyértékű elemenkénti feldolgozás):

$b := \langle \rangle$
$a.dom \neq 0$
$e := a.lov$
$b : hiext(f(\{e\}))$
$a : lorem$

$b := \langle \rangle$
$sx, dx, x : read$
$sx = norm$
$e := dx$
$b : hiext(f(\{e\}))$
$sx, dx, x : read$

#### • Sorozatról vektorra

A transzformációt az akadályozza, hogy a vektor értelmezési tartományát nem változtathatjuk meg. Ezt a problémát könnyen kezelhetjük input sorozat esetén,

de az output sorozat esetében a transzformáció csak akkor hajtható végre, ha föltesszük, hogy a sorozat elemeinek száma nem haladja meg a vektor értelmezési tartományának számosságát.

Feleltessük meg az  $x$  (és  $y$ ) sorozatnak az  $(u, i)$  (illetve  $(v, j)$ ) párt, ahol  $u, v$  vektorok és  $i, j$  nemnegatív egészek, oly módon, hogy  $i$  az  $x$  (illetve  $j$  az  $y$ ) sorozat elemeinek száma, és az elemei az  $u$  vektor  $u_{u.lob}, \dots, u_{u.lob+i-1}$  (illetve a  $v$  vektor  $v_{v.lob}, \dots, v_{v.lob+i-1}$ ) elemeivel egyeznek meg.

$x.dom \neq 0$ $x.lov$ $x : lorem$ $y : hiext(d)$ $y := \langle \rangle$	$i \neq 0$ $u_{u.lob+i-1}$ $i := i - 1$ $v_{v.lob+j}, j := d, j + 1$ $j := 0$
--	---

Példa (egyváltozós egyértékű elemenkénti feldolgozás):

$y := \langle \rangle$ $x.dom \neq 0$ $y : hiext(f(\{x.lov\}))$ $x : lorem$	$j := 0$ $i \neq 0$ $v_{v.lob+j}, j := f(\{u_{u.lob+i-1}\}), j + 1$ $i := i - 1$
--	---

#### • Halmazokról sorozatokra

Kettő (több) input halmaz a tárgyalt tételek között a két(több)változós elemenkénti feldolgozás esetében fordul elő. A kétváltozós elemenkénti feldolgozásnál az okozhat gondot, hogy el kell tudnunk dönteni, egy adott elem melyik sorozatban van benne. Ezért feltesszük, hogy a sorozatok növekvően rendezettek, mert így mindkét sorozat legelső eleme a legkisebb, és ha ezek közül a kisebbiket választjuk, akkor a tartalmazás egyszerűen eldönthető. Egyébként a transzformáció többi része megegyezik az egyváltozós esetnél leírtakkal (itt is szükséges, hogy a függvényérték legfeljebb egyelemű legyen). Természetesen az elem kiválasztása itt is elhagyható, hiszen a  $lov$  művelet a sorozatnak mindig ugyanazt az elemét adja vissza.

$c := \langle \rangle$		
$a.dom \neq 0 \vee b.dom \neq 0$		
$(a.dom \neq 0 \wedge b.dom \neq 0 \wedge a.lov < b.lov) \vee b.dom = 0$	$(a.dom \neq 0 \wedge b.dom \neq 0 \wedge a.lov > b.lov) \vee a.dom = 0$	$a.dom \neq 0 \wedge b.dom \neq 0 \wedge a.lov = b.lov$
$c : \text{hiext}(f(\{a.lov\}, \emptyset))$ $a : \text{lorem}$	$c : \text{hiext}(f(\emptyset, \{b.lov\}))$ $b : \text{lorem}$	$c : \text{hiext}(f(\{a.lov\}, \{b.lov\}))$ $a : \text{lorem}$ $b : \text{lorem}$

A teljesség kedvéért bemutatjuk a szekvenciális fájlokra vonatkozó változatot is, alkalmazva az előreolvasási technikát:

$sx, dx, x : \text{read}$		
$sy, dy, y : \text{read}$		
$c := \langle \rangle$		
$sx = \text{norm} \vee sy = \text{norm}$		
$sy = \text{abnorm} \vee sx = sy \wedge dx < dy$	$sx = \text{abnorm} \vee sx = sy \wedge dx > dy$	$sx = sy \wedge dx = dy$
$c : \text{hiext}(f(\{dx\}, \emptyset))$ $sx, dx, x : \text{read}$	$c : \text{hiext}(f(\emptyset, \{dy\}))$ $sy, dy, y : \text{read}$	$c : \text{hiext}(f(\{dx\}, \{dy\}))$ $sx, dx, x : \text{read}$ $sy, dy, y : \text{read}$

Megjegyezzük, hogy a feltételekben azért írhattunk  $sx = \text{norm} \wedge sy = \text{norm}$  helyett  $sx = sy$ -t, mert a ciklusfeltétel szerint nem lehet mindkettő *abnorm*.

## 13.4. Állapottér-transzformáció

Az alábbiakban egy olyan példát mutatunk be, amelyben az állapottér-transzformáció nem egyszerű típustranszformáció. A feladatot úgy fogjuk megoldani, hogy eredeti állapottér helyett egy új (absztrakt) állapotteret választunk, azon megoldjuk a feladatot, majd a megoldásban szereplő műveleteket megvalósítjuk az eredeti téren.

### 13.4.1. Szekvenciális megfelelő

Egy feladat megoldása során sokszor szembesülünk azzal a problémával, hogy különböző típusérték-halmazokba eső értékek közötti kapcsolatot kell leírnunk, vagy ilyen értékeket kell összehasonlítani. Erre leggyakrabban akkor kerül sor, ha valamilyen állapottér-transzformációt végzünk, és meg kell adnunk az eredeti és az absztrakt tér



közötti kapcsolatot. Az ilyen megfeleltetések formális megadása általában elég nehézkes. A *szekvenciális megfelelő* fogalmának felhasználásával azonban ezek a kapcsolatok is egyszerűbben írhatók fel.

Legyenek  $E_1, E_2, \dots, E_n$  elemi típusok. Legyen

$$E = \bigcup_{i=1}^n E_i \text{ és } F = \{E_1, E_2, \dots, E_n\}.$$

Tegyük fel, hogy a  $T$  típus reprezentációs függvényére igazak az alábbi megszorítások:

- $\varrho \subseteq E^* \times T$ ,
- $\varrho$  kölcsönösen egyértelmű (bijektív),
- $\varrho$  megengedett típuskonstrukció (azaz direktszorzat, unió és iterált konstrukciókból épül fel).

Legyen továbbá  $B = \{B_1, B_2, \dots, B_m\}$  az úgynevezett *bázishalmaz*, ahol  $B_i$  ( $i = 1, \dots, m$ ) tetszőleges típusérték-halmaz. Ekkor a  $t \in T$  elem  $B$  bázisra vonatkoztatott szekvenciális megfelelője:

$$\text{seq}(t|B) = \begin{cases} \langle t \rangle, & \text{ha } T \in B; \\ \langle \rangle, & \text{ha } T \notin B \wedge T \in F; \\ \mathcal{R}(t|B), & \text{ha } T \notin B \wedge T \notin F \text{ és } T \text{ rekord}; \\ \mathcal{E}(t|B), & \text{ha } T \notin B \wedge T \notin F \text{ és } T \text{ egyesítés}; \\ \mathcal{S}(t|B), & \text{ha } T \notin B \wedge T \notin F \text{ és } T \text{ sorozat}, \end{cases}$$

ahol

$$\begin{aligned} \mathcal{R}(t|B) &= \text{kon}(\text{seq}(t.s_1|B), \dots, \text{seq}(t.s_k|B)); \\ \mathcal{E}(t|B) &= \text{seq}(\varphi_u^{-1}(t)|B); \\ \mathcal{S}(t|B) &= \text{kon}(\text{seq}(t_1|B), \dots, \text{seq}(t_{\text{dom}(t)}|B)). \end{aligned}$$

A jelölés egyszerűsítése végett, ha a bázis egyelemű, akkor a bázishalmaz helyett az elem is írható, azaz  $B = \{B_1\}$  esetén

$$\text{seq}(t|B_1) ::= \text{seq}(t|B).$$

Tekintsük a következő példát:

$$\begin{aligned}
 T &= seq(R), \\
 R &= (nev : NEV, szul : SZUL), \\
 NEV &= seq(CHAR), \\
 SZUL &= (hely : HELY, ido : DATUM), \\
 HELY &= seq(CHAR), \\
 DATUM &= (ev : EV, ho : HO, nap : NAP), \\
 EV &= \mathbb{N}_0, \\
 HO &= \mathbb{N}_0, \\
 NAP &= \mathbb{N}_0.
 \end{aligned}$$

Legyen  $F = \{CHAR, \mathbb{N}_0\}$ ,  $t \in T$ . Ekkor

$seq(t NEV)$	a $t$ sorozatbeli rekordok név részeinek sorozata ( $\in NEV^*$ );
$seq(t CHAR)$	a $t$ sorozatbeli rekordok név részének és születési hely részének egymás után fűzésével kapott karaktersorozat;
$seq(t \{HELY, EV\})$	a $t$ sorozatbeli rekordok születési hely és év részének egymás után fűzésével kapott sorozat ( $\in (HELY \cup EV)^*$ );
$seq(seq(t NEV) CHAR)$	a $t$ sorozatbeli rekordok név részeiből képzett karaktersorozat;
$seq(t \mathbb{Z})$	üres sorozat.

### 13.4.2. Példa állapotér-transzformációra

Tegyük fel, hogy van egy karakterekből álló szekvenciális fájlunk, ami egy szöveget tartalmaz. A feladat az, hogy számoljuk meg, hány olyan csupa betűből álló rész van a szövegben, amelynek hossza nagyobb, mint egy megadott  $k$  érték!

Így első ránézésre a feladat nem vezethető vissza egyetlen ismert programozási tételre sem. A feladatot ezért úgy transzformáljuk, hogy bevezetünk egy új állapotot. Tegyük fel, hogy a szövegfájl helyett egy olyan fájlunk van, amelyben szavak – betűkből álló sorozat – és elválasztó részek – csupa nem betű karakterből álló sorozat – vannak. Azaz az eredeti állapotér:

$$A = \begin{matrix} X \\ x \end{matrix} \times \begin{matrix} \mathbb{N} \\ d \end{matrix}, \quad X = file(CH),$$

és az új állapottér:

$$A' = \begin{matrix} F & \times & \mathbb{N} \\ f & & d \end{matrix}, \quad F = \text{file}(SZÓ;ELVR),$$

$$SZÓ = \text{seq}^+(BETŰ), \quad ELVR = \text{seq}^+(NBETŰ),$$

ahol  $BETŰ$  jelenti a betű, és  $NBETŰ$  a nem betű karakterek halmazát. Az  $F$ -re még egy invariáns tulajdonságot is fölírunk. Két egymást követő elem típusa nem lehet azonos, vagyis

$$I_F(z) = (\forall i \in [1..dom(z) - 1] : z_i.SZÓ \neq z_{i+1}.SZÓ).$$

Mi a kapcsolat a két állapottér között? Természetesen az, hogy  $x$  és  $f$  ugyanazokból a karakterekből áll, és a sorrendjük is azonos. Ezt fejezi ki az, hogy

$$\text{seq}(x|\{BETŰ,NBETŰ\}) = \text{seq}(f|\{BETŰ,NBETŰ\}).$$

Most már meg tudnánk fogalmazni, hogy a szavak hosszait akarjuk meghatározni, de ehhez az elválasztó részekre nincs is szükség, ezért azokat el is hagyjuk.

$$A'' = \begin{matrix} G & \times & \mathbb{N} \\ g & & d \end{matrix}, \quad G = \text{file}(SZÓ),$$

és az  $f$  és  $g$  közötti kapcsolat

$$\text{seq}(f|\{SZÓ\}) = \text{seq}(g|\{SZÓ\}).$$

Mivel nekünk nem a szavakra, hanem csak a hosszukra van szükségünk, tovább transzformáljuk az állapotteret.

$$A''' = \begin{matrix} H & \times & \mathbb{N} \\ h & & d \end{matrix} \quad H = \text{file}(\mathbb{N})$$

és a  $g$  és  $h$  közötti kapcsolat

$$\text{dom}(h) = \text{dom}(g) \text{ és } \forall i \in [1..dom(g)] : h_i = \text{dom}(g_i).$$

Tehát a feladat specifikációja:

$$A = \begin{matrix} H & \times & \mathbb{N} \\ h & & d \end{matrix}$$

$$B = \begin{matrix} H \\ h' \end{matrix}$$

$$Q : (h = h')$$

$$R : (d = \sum_{i=1}^{dom(h)} \chi(h_i > k))$$

Ez a feladat visszavezethető a számlálás tételének szekvenciális fájlra felírt változatára:

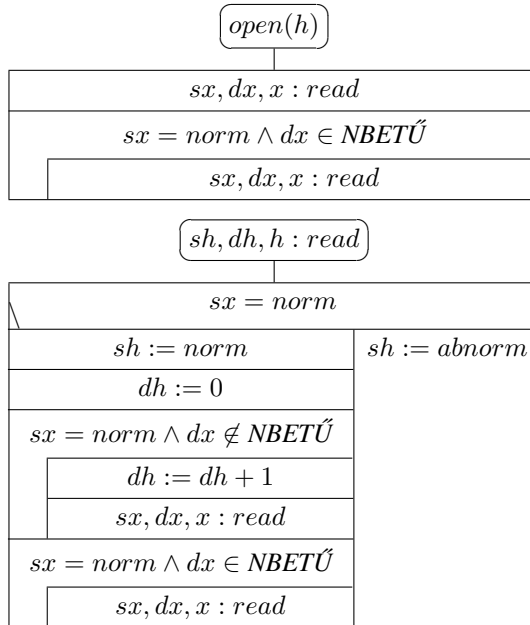
$open(h)$
$sh, dh, h : read$
$d := 0$
$sh = norm$
$dh > k$
$d := d + 1$
$SKIP$
$sh, dh, h : read$

Hátra van még az absztrakt  $open$  és  $read$  műveletek megvalósítása.

Az absztrakt  $h$  fájlak pontosan akkor van még eleme, ha az eredeti  $x$  fájlban van még karakter. Ezért van szükség az  $open$  műveletre, amely arra hivatott, hogy biztosítsa a  $read$  művelet elején megkívánt invariáns tulajdonságot: vagy  $sx = abnorm$  vagy  $dx$  a következő szó első karakterét tartalmazza. Ezen invariáns tulajdonság segítségével a  $read$  műveletben könnyen el lehet dönteni, hogy lesz-e visszaadott érték, vagy már az absztrakt fájl is kiürült.

Az absztrakt fájlak kezelésének általában is ez a technikája: egy  $open$  művelettel biztosítjuk a  $read$  elején megkívánt invariáns tulajdonságot, és gondoskodunk róla, hogy a  $read$  művelet ezt az invariáns tulajdonságot megtartsa. Az absztrakt  $read$  mindig egy elágazás a  $read$  definíciójának megfelelően.

Nézzük tehát az absztrakt műveletek megvalósítását az eredeti állapottéren: ezek, mint látható, szintén programozási tételek egyszerű alkalmazásai:



## 13.5. Programinverzió

### 13.5.1. Egyváltozós eset

Legyenek  $C$ ,  $D$ ,  $E$  és  $F$  tetszőleges típusok. Legyen továbbá  $X = seq(E)$  és  $Y = seq(F)$ , valamint  $f_1 : C \rightarrow X$ ,  $f_2 : X \rightarrow Y$  és  $f_3 : Y \rightarrow D$  függvények.

Tekintsük az alábbi specifikációval adott feladatot:

$$A = C \times D$$

$c \quad d$

$$B = C$$

$c'$

$$Q : (c = c')$$

$$R : (d = f_3 \circ f_2 \circ f_1(c'))$$

Tegyük fel, hogy az  $f_1$  függvény értékét kiszámító program az alábbi alakban írható fel:

$$A = C \times X$$

$c \quad x$

$$B = C$$

$c'$

$$Q : (c = c')$$

$$R : (x = f_1(c'))$$

$S_{11}(c)$
$x := \langle \rangle$
$\pi$
$S_{12}(c, e)$
$x : \text{hiext}(e)$
$S_{13}(c)$

Ha az  $S_{11}$ ,  $S_{12}$  és  $S_{13}$  programok végrehajtása nem változtatja meg az  $x$  változó értékét, akkor a fenti programot *elemenként előállító* programnak nevezzük.

Hasonlóan, tegyük fel, hogy az  $f_3$  függvény értékét kiszámító program pedig az alábbi formában írható fel:

$$A = Y \times D$$

$y \quad d$

$$B = Y$$

$y'$

$$Q : (y = y')$$

$$R : (d = f_3(y'))$$

$S_{31}(d)$
$y.dom \neq 0$
$S_{32}(d, y.lov)$
$y : lorem$
$S_{33}(d)$

Ha az  $S_{31}$ ,  $S_{32}$  és  $S_{33}$  programok végrehajtása nem változtatja meg az  $y$  változó értékét, akkor a fenti programot *elemenként felhasználó* programnak nevezzük.

Tegyük fel továbbá, hogy az  $f_2$  függvény elemenként feldolgozható, és minden egyelemű halmazra a függvényérték egyelemű. Ekkor a függvénykompozíció helyettesítési értékének kiszámítására vonatkozó programozási tétel alapján a feladat megoldható a fenti elemenként előállító, egy elemenként feldolgozó és az imént bemutatott elemenként felhasználó program szekvenciájaként.

A feladatra azonban egy hatékonyabb megoldást is adhatunk a következő program-transzformációval, melyet *programinverzió*nak hívunk. A közbülső két sorozat típusot kihagyjuk, és a három ciklust egybeírjuk az alábbi módon:

$S_{11}(a)$
$S_{31}(d)$
$\pi$
$S_{12}(a, e)$
$\varepsilon := f_2(\{e\})$
$S_{32}(d, \varepsilon)$
$S_{13}(a)$
$S_{33}(d)$

### 13.5.2. Kétváltozós eset

Legyenek  $A^1$ ,  $A^2$ ,  $B$ ,  $C$  és  $D$  tetszőleges típusok. Legyen továbbá  $X = seq(B)$  és  $Y = seq(C)$ , valamint  $f_1^1 : A^1 \rightarrow X$ ,  $f_1^2 : A^2 \rightarrow X$ ,  $f_2 : X \times X \rightarrow Y$  és  $f_3 : Y \rightarrow D$  függvények.

Tekintsük az alábbi specifikációval adott feladatot:

$$A = \begin{matrix} A^1 & \times & A^2 & \times & D \\ a^1 & & a^2 & & d \end{matrix}$$

$$B = \begin{matrix} A^1 \times A^2 \\ a^{1'} \quad a^{2'} \end{matrix}$$

$$Q : (a^1 = a^{1'} \wedge a^2 = a^{2'})$$

$$R : (d = f_3 \circ f_2(f_1^1(a^{1'}), f_1^2(a^{2'})))$$

Legyenek az  $f_1^1$  és  $f_1^2$  függvényeket kiszámító elemenként előállító programok az alábbiak:

$S_{11}^1(a^1)$	$S_{11}^2(a^2)$
$x^1 := \langle \rangle$	$x^2 := \langle \rangle$
$\pi^1$	$\pi^2$
$S_{12}^1(a^1, e^1)$	$S_{12}^2(a^2, e^2)$
$x^1 : \text{hiext}(e^1)$	$x^2 : \text{hiext}(e^2)$
$S_{13}^1(a^1)$	$S_{13}^2(a^2)$

és tegyük fel, hogy az előállított  $x^1$  és  $x^2$  sorozatok rendezettek.

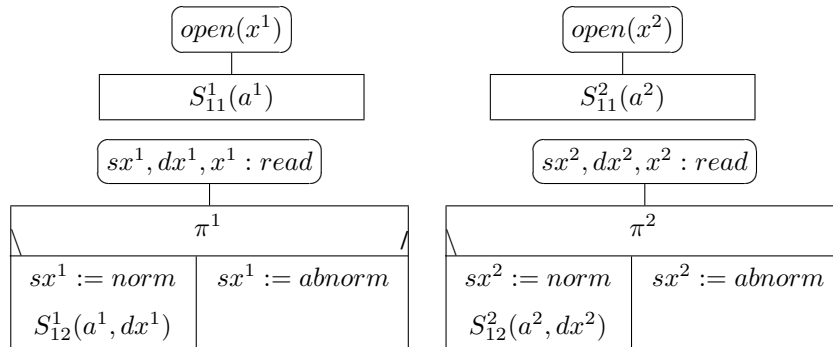
Legyen  $f_3$ , mint korábban, elemenként felhasználó programmal kiszámított függvény. Tegyük fel, hogy  $f_2$  egy olyan kétváltozós, egyértékű elemenként feldolgozható függvény, amely minden olyan halmazpárhoz, amelynek tagjai legfeljebb egy elemet tartalmaznak, egy egyelemű halmazt rendel hozzá.

Ekkor a függvénykompozíció helyettesítési értékének kiszámítására vonatkozó programozási tétel alapján a feladat megoldható a fenti két elemenként előállító, a kétváltozós, egyértékű elemenként feldolgozó és az  $f_3$ -at kiszámító elemenként felhasználó program szekvenciájaként. Vajon ebben az esetben is összeinvertálhatóak a fenti programok?

A válasz természetesen: igen, de a megoldás itt nem olyan egyszerű, mint az egyváltozós esetben volt. A probléma a szekvenciális fájlknál felmerülttel analóg: az elemet előállító program ( $S_1^2$ , illetve  $S_1^1$ ) az egyes elemeket ugyanúgy szolgáltatja, mint ha fájlból olvasnánk őket: csak akkor nézhetjük meg a következő elemet, ha legenerálhatjuk. Ez sugallja azt a megoldást, hogy az  $x^1$  és  $x^2$  sorozatokat tekintsük absztrakt fájlknak az elemenként feldolgozásban. Az elemenként felhasználó program beinvertálása nem okoz gondot, ugyanúgy történik, mint az egyváltozós esetben.

$open(x^1), open(x^2)$		
$sx^1, dx^1, x^1 : read, sx^2, dx^2, x^2 : read$		
$S_{31}(d)$		
$sx^1 = norm \vee sx^2 = norm$		
$sx^2 = abnorm \vee (sx^1 = sx^2 \wedge dx^1 < dx^2)$	$sx^1 = sx^2 \wedge dx^1 = dx^2$	$sx^1 = abnorm \vee (sx^1 = sx^2 \wedge dx^1 > dx^2)$
$e := f_2(\{dx^1\}, \emptyset)$ $sx^1, dx^1, x^1 : read$	$e := f_2(\{dx^1\}, \{dx^2\})$ $sx^1, dx^1, x^1 : read$ $sx^2, dx^2, x^2 : read$	$e := f_2(\emptyset, \{dx^2\})$ $sx^2, dx^2, x^2 : read$
$S_{32}(d, e)$		
$S_{13}^1(a^1)$		
$S_{13}^2(a^2)$		
$S_{33}(d)$		

ahol az absztrakt fájl műveleteinek megvalósításai:



### 13.6. Példák

**13.1. példa:** Keressük meg az  $f : \mathbb{Z} \rightarrow \mathbb{Z}$  függvény olyan értékeinek a maximumát az  $[a, b]$  intervallumon (és egy hely indexét), ahol az argumentum és a hozzá tartozó érték paritása azonos!

**Megoldás:**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{L}$$

$a \quad b \quad i \quad max \quad l$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$a' \quad b'$

$$Q : (a = a' \wedge b = b' \wedge a \leq b + 1)$$



$R : (Q \wedge l = (\exists i \in [a..b] : 2|i + f(i)) \wedge l \rightarrow (i \in [a..b] \wedge 2|i + f(i) \wedge \max = f(i) \wedge \forall j \in [a..b] : (2|j + f(j)) \rightarrow (f(j) \leq f(i))))$

A specifikáció nagyon hasonló a feltételes maximumkeresés programozási tételéhez. Az eltéréseket az alábbi táblázattal foglalhatjuk össze:

feladat		feltételes maximumkeresés
$a$	$\leftrightarrow$	$m$
$b$	$\leftrightarrow$	$n$
$2 i + f(i)$	$\leftrightarrow$	$\beta(i)$

Az olyan feladatokat, amelyek specifikációja csak átnevezéseket tesz egy már ismert programozási tétel specifikációjához képest, *természetes visszavezetéssel* oldhatjuk meg. Az említett átnevezés érintheti a tétel változóinak neveit, az általános kifejezéseket (pl.  $\beta(i)$ -t) vagy az utófeltétel állandó értékű kifejezéseit. Ismert programozási tételeknek tekintjük a programozási tételek különböző írásait (pl. vektorra, sorozatra, input fájlra).

Amennyiben a specifikációkat ily módon sikerül egy alakra hoznunk, akkor az átnevezéseket a programozási tétel már ismert programján elvégezve megkapjuk a kitzűött feladatot az 5.3 definíció értelmében megoldó programot.

$k, l := a - 1, \text{hamis}$			
$k \neq b$			
$2 \nmid (k + 1 + f(k + 1))$	$2 (k + 1 + f(k + 1)) \wedge \neg l$	$2 (k + 1 + f(k + 1)) \wedge l$	
<i>SKIP</i>	$l, i, \max := \text{igaz}, k + 1, f(k + 1)$	$f(k + 1) \geq \max$	$f(k + 1) \leq \max$
		$i, \max := k + 1, f(k + 1)$	<i>SKIP</i>
$k := k + 1$			

**13.2. példa:** Adjunk meg egy, az  $n$  és  $2n$  természetes számok közé eső prímszámot!

**Megoldás:**

$$A = \mathbb{N} \times \mathbb{N}$$

$$n \quad p$$

$$B = \mathbb{N}$$

$$n'$$

$$Q : (n = n' \wedge \exists j \in [n..2n] : \text{prím}(j))$$

$$R : (Q \wedge p \in [n..2n] \wedge \text{prím}(i))$$

A lineáris keresés 1. változatának specifikációja hasonló. Lássuk, hogy az alábbi átnevezések után milyen különbségek maradnak!

feladat		lineáris keresés 1.0
$n$	$\leftrightarrow$	$m$
$p$	$\leftrightarrow$	$i$
$\text{prím}(i)$	$\leftrightarrow$	$\beta(i)$

Különbség, hogy az előfeltételünk szigorúbb, hiszen a  $j$  futóindexnek nem csupán  $n$  fölött, hanem  $2n$  alatt kell lennie. (Megjegyezzük, hogy a specifikáció előfeltétele az általánosságot nem szorítja meg, ugyanis minden  $n$  természetes szám esetén létezik egy prím  $n$  és  $2n$  között (Csebisev-tétel).

A lineáris keresés 1. utófeltételének behelyettesítés utáni alakja:

$$R' = (Q \wedge p \geq n \wedge \text{prím}(p) \wedge \forall j \in [n..p-1] : \neg \text{prím}(j))$$

Ez az utófeltétel biztosítja az  $R$  feltételt a Csebisev-tétel miatt. Hiszen, ha  $p$ -nek olyanak kell lennie, hogy  $p-1$ -ig ne legyen prím  $n$ -től, és  $p$  prím, akkor  $p$  biztosan nem nagyobb mint  $2n$ . Ugyanakkor az  $R$  nem követeli meg, hogy  $p-1$ -ig minden szám  $n$ -től kezdve összetett legyen, ezért a két feltétel között egyenlőség nem, csupán következés áll fenn ( $R' \Rightarrow R$ ).

Tehát a tétel szigorúbb, mint a feladatunk, ezért szigorítással vezetünk vissza.

$p := n$
$\neg \text{prím}(p)$
$p := p + 1$

**13.3. példa:** Állapítsuk meg, hogy van-e az  $f : \mathbb{Z} \rightarrow \mathbb{Z}$  függvény értékei között páros szám az  $[m..n]$  intervallumban!

**Megoldás:**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{L}$$

$m \quad n \quad l$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$m' \quad n'$

$$Q : (n = n' \wedge m = m' \wedge m \leq n + 1)$$

$$R : (Q \wedge l = (\exists j \in [m..n] : 2|f(j)))$$

A lineáris keresés 2.8 specifikációja hasonló. Lássuk, hogy az alábbi megfeleltetés után milyen különbségek maradnak:

feladat		lineáris keresés 2.8
$2 f(i)$	$\leftrightarrow$	$\beta(i)$

A különbség az, hogy az  $i$ -t, azaz azt a komponenst, ami mutatná a megtalált páros elem helyét, az állapotterünk nem is tartalmazza. Ez a különbség persze indukálja az utófeltétel összes olyan tagjának kiesését is, amely  $i$ -re tett kikötést.

Ebben az esetben a feladatunk kiterjesztésénél a tétel nyilván szigorúbb, így szigorítással vezetünk vissza. A tétel programja az 5.4 definíció értelmében megoldása a feladatnak.

Ezt a visszavezetést nevezhetjük *alteres visszavezetésnek* is, hiszen a feladatunk állapottere altere a programozási tétel állapotterének. Amikor a programozási tétel állapotterének egy eredménykomponensét (tehát az előfeltételben nem szerepel, de az utófeltétel tesz rá kikötést) nem találjuk meg a feladat állapotterében (és így természetesen az utófeltételében sem), akkor az alteres visszavezetés *általánosított* esetéről beszélünk.

$i, l := m - 1, \text{hamis}$
$\neg l \wedge i \neq n$
$l := 2 f(i + 1))$
$i := i + 1$

**13.4. példa:** Határozzuk meg az  $n$  természetes szám osztóinak a számát!

**Megoldás:**

$$A = \mathbb{N} \times \mathbb{N}_0$$

$n \quad d$

$$B = \mathbb{N}$$

$n'$

$$Q : (n = n')$$

$$R : (Q \wedge d = \sum_{i=1}^n \chi(i|n))$$

A specifikáció nagyon hasonló a számlálás programozási tételéhez. Az eltéréseket az alábbi táblázattal foglalhatjuk össze:

feladat	számlálás
1	$\leftrightarrow$ $m$
$i n$	$\leftrightarrow$ $\beta(i)$

Első ránézésre itt is csak az a probléma, hogy bővebb a tétel állapotere. Most azonban nem alkalmazhatjuk a „kiterjesztés, szigorítás” receptet, mert nem csak az állapotterek és az utófeltételek különböznek, hanem a paraméterterek és az előfeltételek is.

A specifikáció tétele, illetve annak megfordíthatósága alapján, ha egy „jól” specifikált feladat paraméterterét szűkítjük, akkor kapunk egy új feladatot, aminek megoldása lesz az eredeti feladatot megoldó program. Esetünkben a megszámlálás tételéből a következő feladatot kapjuk:

$$A = \mathbb{N} \times \mathbb{N} \times \mathbb{N}_0$$

$m \quad n \quad d$

$$B = \{1\} \times \mathbb{N}$$

$m' \quad n'$

$$Q : (m = 1 \wedge n = n' \wedge 1 \leq n + 1)$$

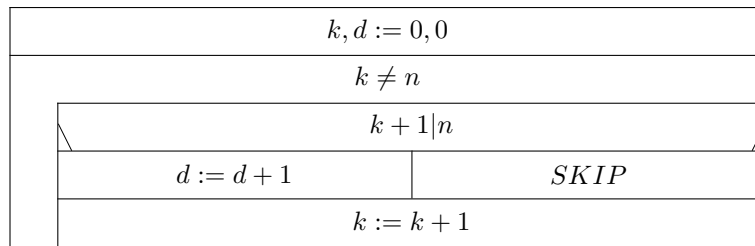
$$R : (Q \wedge d = \sum_{i=1}^n \chi(i|n))$$

Ha a megoldandó feladatunkat kiterjesztjük az  $A = \{1\} \times \mathbb{N} \times \mathbb{N}_0$  állapot-  
 $m \quad n \quad d$   
 térre alkalmazhatjuk a 2.2 állítást.

A fenti tulajdonsággal rendelkező visszavezetést is *alteres* visszavezetésnek, azon belül *konstanssal helyettesítésnek* nevezzük.

Ellenőrizni szükséges, hogy a konstans érték, amellyel  $m$ -et helyettesítettük, teljesíti-e a számlálás előfeltételének rá vonatkozó nem triviális részét ( $(m \leq n+1) = (1 \leq n+1)(n \in \mathbb{N})\checkmark$ ).

Ezekkel a megjegyzésekkel most már felírhatjuk a megoldóprogramot:



**13.5. példa:** Keressük meg az  $[m..n]$  intervallumban azt a legkisebb  $k$  számot, amelyre  $p$  és  $k$  relatív prímek!

**Megoldás:**

$$A = \mathbb{N} \times \mathbb{N} \times \mathbb{N} \times \mathbb{L} \times \mathbb{N}$$

$$m \quad n \quad k \quad l \quad p$$

$$B = \mathbb{N} \times \mathbb{N} \times \mathbb{N}$$

$$m' \quad n' \quad p'$$

$$Q : (n = n' \wedge m = m' \wedge m \leq n + 1 \wedge p = p')$$

$$R : (Q \wedge l = (\exists j \in [m..n] : \text{lnko}(p, j) = 1) \wedge l \rightarrow (k \in [a..b] \wedge \text{lnko}(p, k) = 1 \wedge \forall j \in [m..k-1] : \text{lnko}(p, j) \neq 1))$$

A specifikáció nagyon hasonló a lineáris keresés 2.8 programozási tételéhez. Az eltéréseket az alábbi táblázattal foglalhatjuk össze:

feladat	lineáris keresés 2.8
$k$	$i$
$\text{lnko}(p, i) = 1$	$\beta(i)$

Ez a visszavezetés *paraméteres visszavezetés*, mert az állapottér bővebb, és a helyettesítő táblázatban a  $\beta(i)$  helyettesítésekor fel is használjuk ezt a plusz komponenset. Ugyanakkor a bevezetett  $p$  az előfeltétel szerint adott értékű és a program során nem változik (az utófeltételben is szerepel rejtve a  $p = p'$ ).

$k, l := m - 1, \text{hamis}$	
$\neg l \wedge k \neq n$	
$l := (\text{lnko}(p, k + 1) = 1)$	
$k := k + 1$	

**13.6. példa:** Állapítsuk meg, hogy van-e az  $f$  függvény értékei között olyan szám, amely  $k$ -hoz relatív prím!

**Megoldás:**

$$f : [m, n] \rightarrow \mathbb{N}$$

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{L} \times \mathbb{N}$$

$$a \quad b \quad l \quad k$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$a' \quad b' \quad k'$$

$$Q : (a = a' \wedge b = b' \wedge a \leq b + 1 \wedge k = k')$$

$$R : (Q \wedge l = (\exists j \in [a..b] : \text{lnko}(k, j) = 1))$$

Ezt a lineáris keresés 2.8 tételre vezethetjük vissza, azonban a visszavezetés általánosított alteres, hiszen nem vagyunk kíváncsiak a keresés által visszaadott függvényargumentumra, amellyel a függvényérték relatív prím  $k$ -hoz. Ugyanakkor a visszavezetés paraméteres is  $k$  paraméter szerint.

A helyettesítések:

feladat	$\leftrightarrow$	lineáris keresés 2.8
$a$	$\leftrightarrow$	$m$
$b$	$\leftrightarrow$	$n$
$\text{lnko}(k, i) = 1$	$\leftrightarrow$	$\beta(i)$

$i, l := a - 1, \text{hamis}$	
$\neg l \wedge i \neq b$	
$l := (\text{lnko}(k, i + 1) = 1)$	
$i := i + 1$	

**13.7. példa:** Adott a kezdőpontja szerint növekvő sorrendben a számegyenes  $N$  darab intervalluma. Állapítsuk meg, hogy a  $k$  szám hány intervallumba esik bele.

**Megoldás:**

$$Iv = (ah : \mathbb{Z}, fh : \mathbb{Z})$$

$$I_{Iv}(i) = (i.ah \leq i.fh + 1)$$

$$V = \text{vekt}([1..N], Iv)$$

$$A = V \times \mathbb{Z} \times \mathbb{N}_0$$

$$v \quad k \quad d$$

$$B = V \times \mathbb{Z}$$

$$\begin{matrix} v' & k' \end{matrix}$$

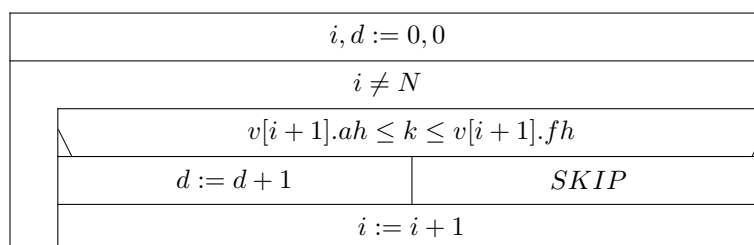
$$Q : (v = v' \wedge k = k' \wedge \forall i \in [1..N - 1] : v[i].ah \leq v[i + 1].ah)$$

$$R : (Q \wedge d = \sum_{i=1}^N \chi(v[i].ah \leq k \leq v[i].fh))$$

A specifikáció nagyon hasonló a számlálás programozási tételéhez. Az eltéréseket az alábbi táblázattal foglalhatjuk össze:

feladat		számlálás
1	$\leftrightarrow$	$m$
$N$	$\leftrightarrow$	$n$
$v[x].ah \leq k \leq v[x].fh$	$\leftrightarrow$	$\beta(x)$
$i$	$\leftrightarrow$	$k$

A visszavezetés paraméteres a  $k$  szerint, továbbá a számlálás programozási tétel nem követeli meg a bemenő vektor valamilyen rendezettségét, tehát a feladat előfeltétele erősebb, mint a tétel, így szigorítunk.



**13.8. példa:** Döntsük el a monoton növekedő  $f$  függvényről a szigorú értelemben vett monotonitást!

**Megoldás:**

$$f : [m..n] \rightarrow \mathbb{Z}$$

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$\begin{matrix} m & n & l \end{matrix}$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$\begin{matrix} m' & n' \end{matrix}$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1 \wedge \forall i \in [m..n - 1] : f(i) \leq f(i + 1))$$

$$R : (Q \wedge l = (\forall i \in [m..n - 1] : f(i) < f(i + 1))) = (Q \wedge \neg l = (\exists i \in [m..n - 1] : f(i) \geq f(i + 1)))$$

Ha az utófeltétel második alakját vesszük figyelembe, akkor a specifikáció hasonlít a lineáris keresés 2.8 specifikációjához, pár apró eltéréstől eltekintve. Ilyen eltérés (a táblázatban összefoglalhatókon kívül), hogy a tétel  $i$  állapotér-komponense a mi specifikációnkban nem szerepel, illetve az, hogy a tétel nem követelné meg a vizsgált függvény monotonitását. Tehát egyrésztől  $i$  szerint általánosított alteres, másrésztől – az előfeltételek különbözősége miatt – szigorításos ez a visszavezetés.

feladat		lineáris keresés 2.8
$f(i) \geq f(i+1)$	$\leftrightarrow$	$\beta(i)$
$\neg l$	$\leftrightarrow$	$l$
$n-1$	$\leftrightarrow$	$n$

Az  $n$ -nek  $n-1$ -gyel való helyettesítése esetén az előfeltétel csak akkor teljesül, ha  $m \leq n$ , ezért a visszavezetéshez erre a kiegészítő feltételre is szükség van.

A visszavezetés paraméteres a  $k$  szerint, továbbá a számlálás programozási tétele nem követeli meg a bemenő vektor valamilyen rendezettségét, tehát a feladat előfeltétele erősebb, mint a tételé, így a visszavezetés egyben szigorított is.

$i, \neg l := m-1, \text{hamis}$
$\neg \neg l \wedge i \neq n-1$
$\neg l := f(i+1) \geq f(i+2)$
$i := i+1$

Ezzel a struktogrammal több probléma is van. A legnagyobb az, hogy szerepelnek benne nem definiált értékadások. Hiszen az értékadás definíciójakor a jelölésben csak azt engedjük meg, hogy a bal oldalon változónevek álljanak. Itt azonban egy kifejezés, nevezetesen a  $\neg l$  áll, ahol  $l$  már változónév. Egyszerű megfontolások alapján azonban láthatjuk, hogy az ilyen típusú értékadások átírhatók érvényessé, ha „mindkét oldalt még egyszer tagadjuk”, majd a kialakult  $\neg \neg l$  formulák helyére egyszerűen csak  $l$ -et írunk. Ezzel az utolsó lépéssel a struktogram másik fölösleges alakú kifejezését, a benne szereplő  $\neg \neg l$  részt is kiküszöböltük.

Így:

$i, l := m-1, \text{igaz}$
$l \wedge i \neq n-1$
$l := f(i+1) < f(i+2)$
$i := i+1$

**13.9. példa:** Keressünk az  $[a..b]$  intervallumban ikerprímeket! Feltehetjük, hogy  $a > 2$ .

**Megoldás:**

$$A = \mathbb{N} \times \mathbb{N} \times \mathbb{L} \times \mathbb{N}$$

$$a \quad b \quad l \quad p$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$a' \quad b'$$

$$Q : (a = a' \wedge b = b' \wedge a \leq b+1 \wedge a > 2)$$

$$R : (Q \wedge l = (\exists i \in [a..b-2] : (\text{prím}(i) \wedge \text{prím}(i+2)))) \wedge l \rightarrow (p \in [a..b-2] \wedge \text{prím}(p) \wedge \text{prím}(p+2))$$

A specifikáció hasonlít a lineáris keresés 2.8 specifikációjához, pár apró eltéréstől eltekintve. Ezeket foglalja össze a táblázat:

feladat		lineáris keresés 2.8
$\text{prím}(i) \wedge \text{prím}(i+2)$	$\leftrightarrow$	$\beta(i)$
$p$	$\leftrightarrow$	$i$
$a$	$\leftrightarrow$	$m$
$b-2$	$\leftrightarrow$	$n$

Az utolsó helyettesítés esetében az előfeltétel csak akkor igaz, ha  $a \leq b-1$ , ezért ezt fel kell tennünk.

A visszavezetés szigorított, hiszen az előfeltétel az intervallumok tekintetében szigorúbb a feladatban, míg az utófeltétel gyengébb a feladatban, mint a tételben (mi nem követeljük meg, hogy az első ikerprímet találja meg a program).

$p, l := a-1, \text{hamis}$
$\neg l \wedge p \neq b-2$
$l := \text{prím}(p+1) \wedge \text{prím}(p+3)$
$p := p+1$

A megoldásban szereplő  $l := \text{prím}(p+1) \wedge \text{prím}(p+3)$  értékadást egy új feladatként is megfogalmazhatjuk: állapítsuk meg egy  $p$  számról, hogy ő maga és a nála kettővel nagyobb szám prím-e! Ez a feladat az eredeti állapottér egy alterén specifikálható:

$$A' = \mathbb{L} \times \mathbb{N}$$

$l \quad p$

$$B' = \mathbb{N}$$

$p'$

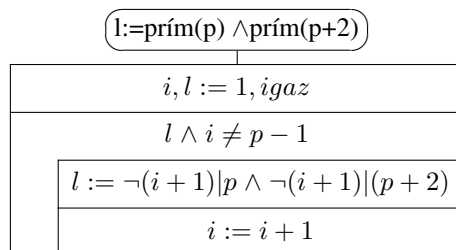
$$Q' : (p = p' \wedge p > 2)$$

$$R' : (Q \wedge l = (\forall i \in [2..p-1] : (\neg(i|p) \wedge \neg(i|(p+2))))) = (Q \wedge \neg l = (\exists i \in [2..p-1] : (i|p \vee i|(p+2))))$$

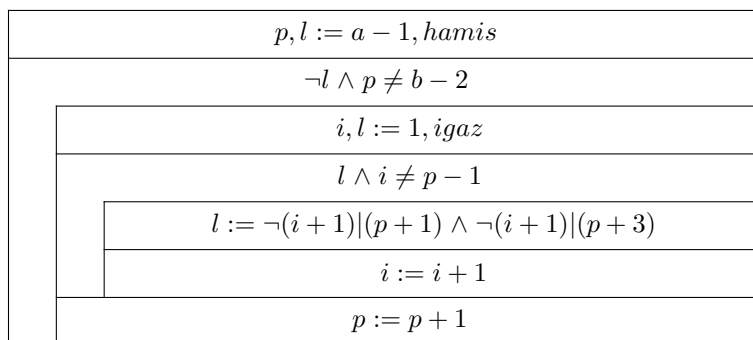
Ez visszavezethető a lineáris keresés 2.8-ra. A visszavezetés az alteres visszavezetés mindkét esetével összeegyeztethető (hiszen az intervallum kezdetét konstanssal helyettesítettük, ugyanakkor a tétel  $i$  eredménykomponensét nem használtuk a specifikációban). A visszavezetés általános is, mivel mi a 2 konstanssal mint az intervallum elejével és a  $p-1 \geq 2$  kifejezéssel mint az intervallum végével az üres intervallum feldolgozását kizártuk az előfeltételben, így az szigorúbb a tétel előfeltételénél.



feladat		lineáris keresés 2.8
$i p \vee i (p+2)$	$\leftrightarrow$	$\beta(i)$
$\neg l$	$\leftrightarrow$	$l$
$2$	$\leftrightarrow$	$m$
$p-1$	$\leftrightarrow$	$n$



Tehát a megoldást így is írhatjuk:



A későbbiekben ezt a helyettesítést nem fogjuk elvégezni, hiszen semmi új információt nem ad, sőt sok esetben a konkrét programnyelven történő kódolást is megnehezítheti.

**13.10. példa:** Keressük meg az  $f : \mathbb{Z} \rightarrow \mathbb{Z}$  függvény értékei között a  $k$  szám  $p$ -edik előfordulását az  $[a..b]$  intervallumban!

**Megoldás:**

A feladat megfogalmazásához definiálunk egy  $g \in \mathbb{Z} \rightarrow \mathbb{N}_0$  parciális függvényt, ahol  $g(i)$  azt adja meg, hogy hányszor vette fel az  $f$  függvény a  $k$  értéket az  $[a..i]$  intervallumban.

$$g(a-1) = 0, \forall i \in [a-1..b-1] :$$

$$g(i+1) = \begin{cases} g(i), & \text{ha } f(i+1) \neq k; \\ g(i) + 1, & \text{ha } f(i+1) = k. \end{cases}$$

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{N} \times \mathbb{Z} \times \mathbb{L}$$

$$\quad \quad \quad a \quad b \quad k \quad p \quad i \quad l$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{N}$$

$$\quad \quad \quad a' \quad b' \quad k' \quad p'$$

$$Q : (a = a' \wedge b = b' \wedge a \leq b + 1 \wedge k = k' \wedge p = p')$$

$$R : (Q \wedge l = (\exists j \in [a..b] : g(j) = p) \wedge l \rightarrow (i \in [a..b] \wedge g(i) = p \wedge f(i) = k))$$

feladat		lineáris keresés 2.8
$a$	$\leftrightarrow$	$m$
$b$	$\leftrightarrow$	$n$
$g(i) = p$	$\leftrightarrow$	$\beta(i)$

$i, l := a - 1, \textit{hamis}$
$\neg l \wedge i \neq b$
$l := (g(i + 1) \neq p)$
$i := i + 1$

Felhasználva a rekurzív függvény változóval való helyettesítésének programtranszformációs módszerét  $g$ -re:

$z := 0$
$i, l := a - 1, \textit{hamis}$
$\neg l \wedge i \neq b$
$z := F(i + 1, z)$
$l := (z \neq p)$
$i := i + 1$

Ha ismerjük az elágazással definiált függvény kiszámításának programozási tételét, akkor ebből megkaphatjuk a megoldóprogramot, ha azt  $F$ -re alkalmazzuk:

$z := 0$
$i, l := a - 1, \textit{hamis}$
$\neg l \wedge i \neq b$
$f(i + 1) = k$
$z := z + 1 \quad   \quad z := z (\Leftrightarrow \textit{SKIP})$
$l := (z \neq p)$
$i := i + 1$

**13.11. példa:** Keressük meg az  $[m..n]$  intervallumban  $f : \mathbb{Z} \rightarrow \mathbb{Z}$  függvény egy olyan értékét, amely egyenlő a közvetlen szomszédai átlagával!

**Megoldás:**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{L} \times \mathbb{Z}$$

$m \quad n \quad l \quad i$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$m' \quad n'$

$$Q = (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R = (Q \wedge l = (\exists j \in [m + 1, n - 1] : (f(j) = \frac{f(j-1) + f(j+1)}{2}) \wedge l \rightarrow (i \in [m + 1, n - 1] \wedge f(i) = \frac{f(i-1) + f(i+1)}{2})))$$

A lineáris keresés 2.8 specifikációja hasonló. Lássuk, hogy az alábbi átnevezés után milyen különbségek maradnak:

feladat	lineáris keresés 2.8
$m + 1$	$\leftrightarrow m$
$n - 1$	$\leftrightarrow n$
$f(i) = \frac{f(i-1) + f(i+1)}{2}$	$\leftrightarrow \beta(i)$

Az első két helyettesítés esetében a tétel előfeltétele csak akkor igaz, ha  $m < n$ , ezért ezt fel kell tennünk.

A visszavezetés általánosított, hiszen nem követeljük meg, hogy az első ilyen speciális tulajdonságú elemet találjuk meg.

$i, l := m, \textit{hamis}$
$\neg l \wedge i \neq n - 1$
$l := (f(i + 1) = \frac{f(i) + f(i+2)}{2})$
$i := i + 1$

### 13.7. Feladatok

A feladatokban szereplő függvények (ha más nincs kikötve) egész számok egy intervallumán vannak értelmezve és egész értékűek.

- 13.1. Határozzuk meg az  $f$  függvény azon pozitív értékeinek a számát, amelyek közvetlenül egy negatív érték után állnak!
- 13.2. Állapítsuk meg, hogy az  $n$  természetes számnak van-e páratlan valódi osztója!
- 13.3. Határozzuk meg az  $n$  természetes szám legkisebb páratlan osztóját!
- 13.4. Adottak az  $x$  és  $y$  vektorok, ahol  $y$  elemei az  $x$  indexei közül valók. Keressük meg az  $x$  vektornak az  $y$ -ban megjelölt elemei közül a legnagyobbat!
- 13.5. Adjuk meg, hány olyan elem van az  $x$  vektorban, amely kisebb az indexénél!
- 13.6. Határozzuk meg az  $n$  természetes szám legkisebb egyszeres osztóját!
- 13.7. Adottak az azonos értelmezési tartományú  $f$  és  $g$  függvények. Az  $f$  értékei egészek,  $g$  pedig csak a 0, 1 értékeket veszi fel. Határozzuk meg azoknak a páros  $f$ -értékeknek a számát, amelyek olyan pozícióban vannak, ahol a  $g$  függvény értéke 1!
- 13.8. Keressük meg azt a helyet, ahol az  $f$  függvény értékének decimális alakjában az egyesek helyén a legnagyobb számjegy áll!
- 13.9. Az  $x$  vektor egy szöveget tartalmaz. Állapítsuk meg, hogy visszafelé olvasva a szöveg ugyanaz-e!
- 13.10. Adott egy gráf a csúcsmátrixával. Állapítsuk meg a  $k$ -adik csúcs fokszámát!
- 13.11. Határozzuk meg az  $f$  függvény legnagyobb  $k$ -val osztható értékét!
- 13.12. Határozzuk meg az  $n$  természetes szám valódi páros osztóinak számát!
- 13.13. Határozzuk meg az  $f$  függvény lokális minimumai közül a legnagyobbat! (Egy érték akkor lokális minimum, ha mindkét szomszédjánál kisebb.)
- 13.14. Adjuk meg az  $f$  függvény egy  $k$ -val osztható értékéhez tartozó argumentumát!
- 13.15. Határozzuk meg az  $f$  függvénynek a  $k$ -nál kisebb legnagyobb értékét!
- 13.16. Adott a középpontjával és a sugarával a síkon egy kör, és további  $N$  darab pont. Keressünk egy olyan pontot, amely a körbe esik!
- 13.17. Határozzuk meg az  $f$  függvénynek az  $[a, b]$  intervallumba eső legnagyobb értékét!

- 13.18.** Határozzuk meg az  $f$  függvény azon értékeinek a számát, amelyek vagy az  $[a, b]$  vagy a  $[c, d]$  intervallumba esnek!
- 13.19.** Határozzuk meg az  $f$  függvénynek azt a legnagyobb értékét, amely  $k$ -val osztva 1-et ad maradékkal!
- 13.20.** Adjuk meg az  $f$  függvénynek azt az értékét, amely  $\text{mod } N$  a legnagyobb!
- 13.21.** Adottak az azonos értelmezési tartományú  $f$  és  $g$  függvények. Az  $(f(i), g(i))$  számpárok egy-egy síkbeli pont koordinátái. Számoljuk meg, hogy a pontok közül hány esik az  $(x_0, y_0)$  középpontú,  $r$  sugarú körbe!
- 13.22.** Keressük meg az  $f$  függvénynek az első  $n$ -nél kisebb vagy 0 értékét!
- 13.23.** Adottak az  $x$  és  $y$  vektorok, valamint a  $k$  szám. Az  $y$  vektor az  $x$  indexeinek egy részhalmazát tartalmazza. Számoljuk meg, hány olyan  $k$ -val osztható elem van  $x$ -ben, amelynek indexe megtalálható  $y$ -ban!
- 13.24.** Adott az  $x$  vektorban egy szöveg. Állapítsuk meg, hogy a szöveg tartalmaz-e magánhangzót!
- 13.25.** Keressük meg az  $f$  függvény egy olyan értékét, amely beleesik az  $[a, b]$  és a  $[c, d]$  intervallumba is!
- 13.26.** Az  $x$  vektor egy szöveget tartalmaz. Számoljuk meg, hány magánhangzó van a szövegben!
- 13.27.** Állapítsuk meg, hol van a monoton növekedő  $f$  függvényben a legnagyobb ugrás, azaz az  $f(k) - f(k - 1)$  érték mely  $k$ -ra maximális!
- 13.28.** Határozzuk meg az  $f$  függvény legnagyobb páros értékéhez tartozó argumentumot!
- 13.29.** Keressünk az  $x$  vektorban két olyan szomszédos elemet, amelyek szorzata negatív!
- 13.30.** Adottak az  $[m, n]$  intervallumon értelmezett  $f$  és  $g$  függvények. Állapítsuk meg, hány egész koordinátájú pont esik a függvényértékek közé!
- 13.31.** Adottak az  $x$  és  $b$  vektorok. „Fektessük”  $b$ -t az  $x$  vektorra folyamatosan egymás után, ahányszor csak lehet, és számoljuk meg, hány helyen egyeznek az egymás feletti értékek!
- 13.32.** Adott az  $n$  természetes szám. Határozzuk meg  $n$  egy valódi osztóját!
- 13.33.** Adjuk meg az  $f$  függvénynek azt az értékét, amelynek szomszédai átlagától való eltérése a legnagyobb!

- 13.34.** Keressünk az  $x$  vektorban egy olyan elemet, amely osztható az indexével!
- 13.35.** Adott az  $x$  mátrix, amelynek elemei sorfolytonosan növekvő sorozatot alkotnak. Keressük meg a mátrixban az  $n$  értéket!
- 13.36.** Adott egy  $x$  vektor, amely színeket tartalmaz sötétedő sorrendben (színeken értelmezve van egy úgynevezett sötétségi reláció, amely teljes rendezés). Keressük meg az  $x$  vektorban a világoskéket!
- 13.37.** Adott a síkon  $N$  darab pont. Keressük meg az origótól legtávolabb eső pontot!
- 13.38.** Adjuk meg az  $f$  függvény utolsó pozitív értékének argumentumát!
- 13.39.** Adott az  $x$  vektor, amelynek elemei nullák és egyesek. Számoljuk meg, hány-szor fordul elő a vektorban a '0101' szakasz!
- 13.40.** Állapítsuk meg, hogy van-e az  $f$  függvény értékei között olyan szám, amely  $k$ -hoz relatív prím!
- 13.41.** Határozzuk meg az  $n$  természetes szám legkisebb valódi nem prím osztóját!
- 13.42.** Adottak az  $f$  és  $g$  monoton növekvő függvények, valamint a  $k$  szám. Állapítsuk meg, található-e olyan  $i$  és  $j$  argumentum, amelyre  $f(i) + g(j) = k$ !
- 13.43.** Adjuk meg az  $x$  mátrix egy olyan sorának indexét, amely nem tartalmaz pozitív elemet!
- 13.44.** Állapítsuk meg, hogy a  $b$  vektorban levő szöveg előfordul-e a karakteres  $x$  vektorban!
- 13.45.** Adott az injektív  $f$  függvény. Adjuk meg egy helyet, amelyet legalább egy olyan megelőz, ahol a függvényérték nagyobb!
- 13.46.** Határozzuk meg az első helyet, ahol az  $f$  függvény olyan értéket, vesz fel, amelyet legalább kétszer vesz fel!
- 13.47.** Keressük meg az  $x$  mátrixnak azt a sorát, amelynek minden eleme 1!
- 13.48.** Adjuk meg, hány prímszám van az  $[a, b]$  intervallumban!
- 13.49.** Határozzuk meg az  $n$ -nél kisebb,  $n$ -hez relatív prím természetes számok számát!
- 13.50.** Adott két egybevágó  $2n$ -szög. Mindkettő oldalait véletlenszerűen kékre vagy pirosra festettük. Helyezzük egymásra a két sokszöget úgy, hogy a lehető legtöbb helyen legyenek azonos színű oldalak egymáson!
- 13.51.** Számoljuk meg az  $f : [m, n] \times [m, n] \rightarrow \mathbb{Z}$  függvény nulla értékeit!

- 13.52.** Számoljuk meg, hogy az  $x$  mátrixban hány olyan sor van, amely csak egyetlen nullától különböző elemet tartalmaz!
- 13.53.** Állapítsuk meg, melyik az  $f$  függvény leggyakrabban felvett értéke!
- 13.54.** Határozzuk meg az  $f$  függvénynek azt az értékét, amelyet a legtöbb nála nagyobb elem előz meg!
- 13.55.** Keressük meg a négyzetes  $x$  mátrixnak azt az oszlopát, amelyben a fődiagonális feletti elemek összege a legnagyobb!
- 13.56.** Határozzuk meg a négyzetes  $x$  mátrixnak a fődiagonális alatti legnagyobb elemét!
- 13.57.** Számoljuk meg, hogy az  $x$  mátrixnak hány olyan sora van, amely csak egy nullától különböző elemet tartalmaz!
- 13.58.** Adott a sík  $N$  pontja. Állapítsuk meg, melyik a két legtávolabbi pont!
- 13.59.** Egy sakkbajnokság végeredményét egy  $x$  négyzetes mátrixban tároltuk, ahol az  $i$ -edik sorban a  $j$ -edik elem az  $i$ -edik játékos és a  $j$ -edik játékos közti mérkőzés eredményét jelenti az  $i$ -edik játékos szempontjából ( $x_{i,j}$  értéke 0, ha  $j$  nyert; 2, ha  $i$  nyert; 1, ha a játékosok döntetlenben egyeztek meg). Keressük meg a bajnokság (egyik) győztesét (győztes az, akinek a legtöbb pontja van)!
- 13.60.** Adott egy  $f : [a, b] \rightarrow \mathbb{Z}$  függvény. E függvény értelmezési tartományának egy  $i$  pontját a függvény csúcsának nevezzük, ha  $\forall j \in [i+1, b] : f(j) < f(i)$ . Adjuk meg, hány csúcsa van  $f$ -nek!
- 13.61.** Keressük meg az  $x$  négyzetes mátrixnak azt a fődiagonálissal párhuzamos átlóját, amelyben az elemek összege a legnagyobb!
- 13.62.** Adott a 0, 1 értékeket felvevő  $f$  függvény. Keressük meg a függvény értelmezési tartományának azt az elemét, amely a leghosszabb egyes-értéksorozat kezdete!
- 13.63.** Állapítsuk meg, van-e negatív szám az  $f$  függvényértékeinek (kezdő) részletösszegei között!
- 13.64.** Egy függvény értelmezési tartományának azt a szakaszát, amelyhez tartozó értékek negatívak úgy, hogy a szakaszt jobbról és balról nemnegatív érték vagy az értelmezési tartomány vége határolja, a függvény negatív szigetének nevezzük. Adjuk meg az  $f$  függvény értelmezési tartományában a negatív szigetek számát!
- 13.65.** Adjunk meg egy olyan  $k$  számot, amelyre az  $n$  természetes szám bináris alakjának  $k$ -adik helyi értékén 1-es áll!

- 13.66.** Adjuk meg az  $f$  függvény értelmezési tartományának azt a leghosszabb szakaszát, amelyen belül az értékek növekvőek!
- 13.67.** Állapítsuk meg, hogy az  $x$  szám binárisan felírt alakjában hány darab 1-es szerepel!
- 13.68.** Adott az  $x$  vektor, amelynek elemei karakterek. A vektor szavakat tartalmaz, amiket egy-egy vessző választ el egymástól. Adjuk meg a leghosszabb szónak a kezdőindexét!
- 13.69.** Egy  $f : [a, b] \rightarrow \mathbb{Z}$  függvény értelmezési tartományának azon szakaszát, melynek két végpontja a függvény lokális minimumhelye úgy, hogy a végpontok közötti elemek nem azok, a függvény egy hegyének nevezzük. Adjuk meg a legszélesebb hegy kezdőpontját!
- 13.70.** Egy vektornak azt a szakaszát, amely csupa negatív elemet tartalmaz úgy, hogy a szakaszt jobbról és balról nemnegatív elem vagy a vektor vége határolja, a vektor negatív szigetének nevezzük. Adjuk meg az  $x$  vektor legnagyobb negatív szigetének kezdőindexét!
- 13.71.** Egy múzeumban az  $i$ -dik órában  $x_i$  látogató érkezik és  $y_i$  látogató megy el. Melyik órában volt a legtöbb látogató a múzeumban?
- 13.72.** Adott az egész számok egy vektora és két egész szám. Állapítsuk meg, hogy a két adott szám előfordul-e a vektorban; és ha igen, akkor melyik előbb!
- 13.73.** Helyezzünk el  $n$  darab vezért egy  $n \times n$  méretű sakktáblán úgy, hogy egyik vezér se támadjon másikat!
- 13.74.** Hányféleképpen helyezhetünk el  $n$  darab vezért egy  $n \times n$  méretű sakktáblán úgy, hogy egyik vezér se támadjon másikat?
- 13.75.** Legfeljebb hány vezért lehet egy  $n \times n$  méretű tórikus sakktáblán elhelyezni úgy, hogy egyik vezér se támadjon másikat?
- 13.76.** Adott  $n$  fiú és ugyanennyi lány. Egy  $x$  logikai mátrixban tároljuk a fiúk és lányok közötti szimpátiát (ez egy szimmetrikus reláció) a következőképpen:  $x_{i,j}$  igaz, ha az  $i$ -edik fiú és a  $j$ -edik lány szimpatizál egymással, hamis, ha nem szimpatizálnak egymással. A feladat az, hogy ha lehet, akkor párosítsuk (házassítsuk) össze őket úgy, hogy minden párban a felek szimpatizáljanak!
- 13.77.** Adott egy  $n \times m$  méretű sakktábla  $(i, j)$  mezijén egy huszár. Végig lehet-e vezetni a táblán úgy, hogy minden mezőre lép, de csak egyszer, és minden lépése szabályos (huszárlépés)?
- 13.78.** Hányféleképpen lehet  $n$  forintot kifizetni  $m$  különböző címletű pénzből?



- 13.79.** Hányféleképpen lehet  $n$  forintot kifizetni  $m$  különböző címletű pénzből, ha legfeljebb  $d_1, d_2, \dots, d_m$  használható fel?
- 13.80.** Adott a természetes számok egy  $S$  véges részhalmaza. Kiválasztható-e ebből  $n$  darab elem úgy, hogy az összegük  $m$  legyen?
- 13.81.** Az  $x$  szekvenciális fájl (megengedett művelet az  $sx, dx, x : read$ ) egy vállalat dolgozóiról a következő adatokat tartalmazza:
- a dolgozó azonosító száma;
  - vezető beosztásban van-e;
  - legmagasabb iskolai végzettsége.

Válasszuk ki a  $v$  sorozatba azoknak a dolgozóknak az adatait, akik vezető beosztásban vannak, a  $z$  sorozatba azoknak az azonosítóit, akik vezető beosztásban vannak és nem érettségiztek!

- 13.82.** Az  $x$  szekvenciális fájl (megengedett művelet az  $sx, dx, x : read$ ) földregések adatait tartalmazza. Egy elem a következőkből áll:
- az észlelés helyének koordinátái;
  - a rengés erőssége;
  - a rengés időtartama;
  - a földrész azonosítója;
  - a rengést előre jelezték-e.

Válasszuk ki a  $t$  sorozatba az előre nem jelzett földregések észlelési helyeit, a  $z$  sorozatba pedig a 20 másodpercnél hosszabb földregések adatait!

- 13.83.** Adott a keresztnevek és a virágnevek fájlja, mindkettő ábécésorrendben rendezett (megengedett művelet az  $sx, dx, x : read$ ). Határozzuk meg azokat a keresztneveket, amelyek nem virágnevek!
- 13.84.** Adott egy egész számokat tartalmazó fájl. Ha a fájl tartalmaz pozitív elemet, akkor keressük meg a legnagyobbat, különben a legkisebbet!
- 13.85.** Egy fájlban (megengedett művelet a *lopop* extrémális elemmel) adottak az egyes kaktuszfajtákról a (név, őshaza, virágszín, méret) adatok. Válogassuk ki egy fájlba a mexikói, egy másikba a piros virágú kaktuszokat!
- 13.86.** Adott egy fájlban (megengedett művelet az  $sx, dx, x : read$ ) egy OTP-nyilvántartás (név, összeg) párok alakjában. Adjuk meg annak a nevét, akinek nincs tartozása, de a legkisebb a betétállománya (ha van ilyen)!

- 13.87.** Az  $x$  szekvenciális fájl egész számokat tartalmaz (megengedett művelet az  $sx, dx, x : read$ ). Keressünk a fájlban lokális maximumot, vagyis olyan értéket, amely mindkét közvetlen szomszédjánál nagyobb!
- 13.88.** Adott az  $x$  vektor és az  $y$  szekvenciális fájl, amelynek elemei egyaránt pozitív egész számok.  $x$ -ben és  $y$ -ban egy szám legfeljebb egyszer fordul elő, és mindkettő növekvően rendezett. Az  $y$  egy olyan szekvenciális fájl, amelyre csak a *lopop* művelet megengedett, és a fájl végét egy negatív szám jelzi. Állítsuk elő a rendezett  $z$  sorozatot, ami  $x$  és  $y$  elemeit tartalmazza!
- 13.89.** Adottak az  $x$  szekvenciális fájlban (megengedett művelet az  $sx, dx, x : read$ ) egy évfolyam hallgatóinak adatai. Egy elem a hallgató nevét, csoportszámát és tíz db osztályzatot (a nulla azt jelöli, hogy az osztályzat hiányzik) tartalmaz. A fájl a csoportszámok szerint növekvően rendezett. Állítsuk elő az  $y$  sorozatot, ami a hallgatók nevét, csoportszámát és átlagát tartalmazza!

## 14. fejezet

# Absztrakciós stratégia

A programozási feladatok megoldása során mindig valamilyen absztrakciós eljárást követünk, ami a megoldás során lehet végig azonos, de lépésenként változhat is. A különböző programozási módszereket az jellemzi, hogy döntően milyen absztrakciós stratégiára építenek.

Az előző részben mutattunk egy egyszerű példát az állapotér-transzformációra. Az állapotér-transzformációra épülő stratégiákat *adatabsztrakciós* stratégiának is nevezük, hiszen az állapotér az adatok absztrakciója.

Most ezt a feladatot újra megoldjuk, de egészen más gondolatmenettel. Nem az állapotér alakítjuk át, hanem az eredeti állapotéren definiálunk olyan függvényeket, amelyek segítségével az utófeltétel kényelmesen felírható, és könnyebben kezelhető feladathoz jutunk. Ebben az esetben *függvényabsztrakcióról* beszélünk.

Tehát a feladat a következő: Tegyük fel, hogy van egy karakterekből álló szekvenciális fájlunk, ami egy szöveget tartalmaz. Számoljuk meg, hogy hány olyan csupa betűkből álló rész van a szövegben, amelynek hossza nagyobb, mint egy megadott  $k$  érték!

Legyen  $x$  egy karakterekből álló sorozat. Definiáljuk a következő parciális függvényt:

$$f(0) = 0 \text{ és} \\ \forall i \in [0..dom(x) - 1] :$$

$$f(i + 1) = \begin{cases} f(i) + 1, & \text{ha } x_{i+1} \in BET\ddot{U}; \\ 0, & \text{ha } x_{i+1} \notin BET\ddot{U}. \end{cases}$$

Az  $f$  függvény azt adja meg, hogy az  $i$  helyen hanyadik betűje van egy megszakítás nélküli betűsorozatnak az  $x$ -ben. Ezért a feladat specifikációja:

$$A = X \times \mathbb{N} \text{ ha } X = file(CH),$$

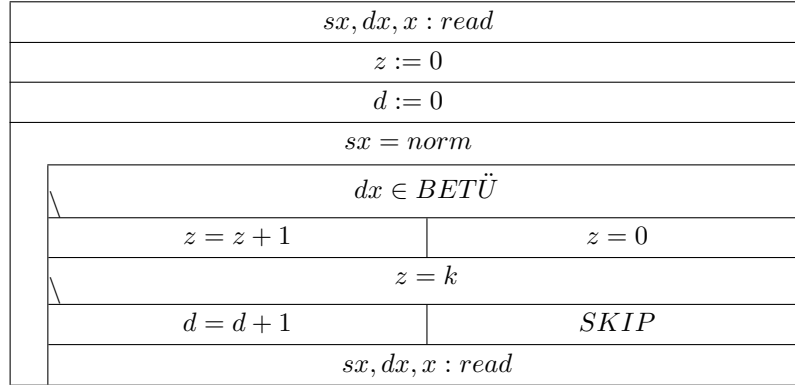
$$x \quad d$$

$$B = x \\ x'$$

$$Q : (x = x')$$

$$R : (k = k' \wedge d = \sum_{i=0}^{dom(x')} \chi(f(i) = k))$$

Erre a feladatra alkalmazva a megszámlálás tételének szekvenciális fájlra vonatkozó változatát és a rekurzív függvény helyettesítése változóval transzformációt kapjuk a következő megoldó programot.



A következőkben egy összetettebb feladaton mutatjuk meg e két alapvető stratégia összehasonlítását.

## 14.1. Az időszerűsítés definíciója

Először néhány, a sorozatok (szekvenciális fájl) estén gyakran használt jelölést, fogalmat vezetünk be. Legyen  $x$  egy sorozat.  $\{x\}$  jelöli a sorozat elemeinek halmazát, azaz

$$\{x\} = \bigcup_{i=1}^{dom(x)} \{x_i\}.$$

Gyakran használunk olyan sorozatot, amelynek *kulcsa* van. Ez azt jelenti, hogy  $S = seq(X)$ , ahol  $X = (k : K, d : D)$  és  $K$  egy rendezett halmaz.  $K$  a kulcsok,  $D$  az adatrészek lehetséges értékeinek halmaza. Legyen  $s \in S$ , ekkor  $\{s.k\}$  az  $s$  kulcsainak halmaza, azaz

$$\{s.k\} = \bigcup_{i=1}^{dom(s)} \{s_i.k\}.$$

Azt mondjuk, hogy az  $s \in S$  sorozat *kulcs szerint rendezett*, ha

$$\forall i \in [1..dom(s) - 1] : s_i.k \leq s_{i+1}.k.$$

Ha  $S$  minden eleme kulcs szerint rendezett, akkor  $S$  kulcs szerint rendezett sorozat típus.

Ha az  $s \in S$  sorozat minden elemének kulcsa különbözik, azaz

$$\forall i, j \in [1..dom(s)] : i \neq j \Rightarrow s_i.k \neq s_j.k,$$

akkor azt mondjuk, hogy  $s$  *egyértelmű kulcsú* sorozat. Ha  $S$  minden eleme egyértelmű kulcsú, akkor  $S$  egyértelmű kulcsú sorozat típus.

Legyen  $S$  egyértelmű kulcsú sorozat típus. Definiáljuk a következő  $\mathcal{K} : S \times K \rightarrow X$  parciális függvényt:

$$(s, q) \in \mathcal{D}_{\mathcal{K}} \Leftrightarrow q \in \{s.k\}$$

és

$$\mathcal{K}(s, q) \in \{s\} \text{ és } \mathcal{K}(s, q).k = q.$$

Most lássuk az időszerűsítés feladatát. Induljunk ki egy olyan adatfájlból, amelyben azonos típusú elemek találhatóak. Ezt a fájlt *törzsfájl*nak fogjuk nevezni. Legyen az elemek típusa  $E$ , ekkor

$$T = seq(E).$$

Legyen adott továbbá egy olyan fájl, amely transzformációk sorozatát tartalmazza. Ezt a fájlt fogjuk *módosítófájl*nak nevezni. Legyen  $F = \{f \mid f : T \rightarrow T\}$ , ekkor

$$M = seq(F).$$

A feladat az, hogy időszerűsítsük a törzsfájlt a módosítófájlban leírt transzformációkkal. Jelölje  $\Upsilon$  az időszerűsítés transzformációt. Ekkor  $\Upsilon : T \times M \rightarrow T$  és

$$\Upsilon(t, m) = m_{dom(m)} \circ \dots \circ m_2 \circ m_1(t).$$

Ahhoz, hogy a feladatra megoldást adjunk, ez a leírás még túl általános, ezért további kikötéseket teszünk a fájlokra.

1. Az időszerűsítés kulcsos.

Legyen  $E = (k : K, d : D)$  és  $F = (k : K, v : V)$ , ahol  $K$  kulcs része;  $D$  a törzsrékörd adat része;  $V$  pedig az elvégzendő transzformációt definiáló típus. Feltesszük még, hogy mind a törzsfájl, mind a módosítófájl a kulcs ( $k$ ) szerint rendezett.

2. A törzsfájl a kulcs szerint egyértelmű.

3. A transzformáció csak a következő háromféle (törlés, beszúrás, javítás) lehet:

$$\begin{aligned} V &= (tr : W_1; be : W_2; jav : W_3); \\ W_1 &= \{\alpha\}, \\ W_2 &= (d : D), \\ W_3 &= (g : G), \quad \text{ahol } G = seq(H), H = \{\gamma \mid \gamma : D \rightarrow D\}. \end{aligned}$$

Hátra van még annak a leírása, hogy hogyan hatnak a fenti transzformációk a törzsfájra. Mivel a törzsfájl kulcs szerint egyértelmű és rendezett, elég csak azt megadni, hogy milyen elemekből áll.

Három esetet különböztetünk meg attól függően, hogy a transzformáció törlés, beszúrás vagy javítás.

(a)  $m_i.tr$ , azaz  $m_i$  törlés:

$$\{m_i(t)\} = \begin{cases} \{e \mid e \in t \text{ és } e.k \neq m_i.k\}, & \text{ha } m_i.k \in \{t.k\}; \\ \{t\} & \text{különben.} \end{cases}$$

(b)  $m_i.be$ , azaz  $m_i$  beszúrás:

$$\{m_i, t\} = \begin{cases} \{t\} \cup \{(m_i.k, m_i.v.d)\}, & \text{ha } m_i.k \notin \{t.k\}; \\ \{t\} & \text{különben.} \end{cases}$$

(c)  $m_i.jav$ , azaz  $m_i$  javítás:

$$\{m_i, t\} = \begin{cases} \{e \mid e \in t \text{ és } e.k \neq m_i.k\} \cup \\ \{(m_i.k, m_i.g_{dom(m_i.g)} \circ \dots \\ \dots \circ m_i.g_1(\mathcal{K}(t, m_i.k).d))\}; & \text{ha } m_i.k \in \{t.k\} \\ \{t\} & \text{különben.} \end{cases}$$

#### 4. A javítás művelet csere.

Ez azt jelenti, hogy  $dom(m_i.g) = 1$  és  $m_i.g_1$  konstans. Ebben az esetben a  $m_i.g_1(\mathcal{K}(t, k).d) = m_i.d$ .

A feladat specifikációja tehát:

$$A = T \times M \times T$$

$$\quad \quad \quad t_0 \quad m \quad t$$

$$B = T \times M$$

$$\quad \quad \quad t'_0 \quad m'$$

$Q : (t_0 = t'_0 \wedge m = m' \text{ és az } 1..x \text{ pontok teljesülnek})$

$R : (t = \Upsilon(t'_0, m'))$

Ha a fenti specifikációban  $x = 3$ , akkor *közönséges*, ha  $x = 4$ , akkor *egyszerű* időszerűsítésről beszélünk.

## 14.2. Időszerűsítés egyértelmű módosítófajllal

A továbbiakban az egyszerű időszerűsítéssel fogunk foglalkozni. Közönséges időszerűsítés esetére csak utalni fogunk.

A feladatot három irányból is megpróbáljuk megoldani: visszavezetjük halmazok uniójára, egyváltozós egyértékű; illetve kétváltozós egyértékű elemenkénti feldolgozásra.

### 14.2.1. Visszavezetés halmazok uniójára

Ez a megoldás Dijkstra könyvében [Dij 76] található. Az alapgondolat az, hogy a  $t$  elemei vagy olyanok, hogy a kulcsuk  $t_0$ -ban is kulcs vagy olyanok, hogy a kulcsuk  $m$ -ben is kulcs. Tehát  $\{t\} = x \cup y$ , ahol  $x = \{e \in \{t\} \mid e.k \in \{t_0.k\}\}$  és  $y = \{e \in \{t\} \mid e.k \in \{m.k\}\}$ .

A transzformált feladat tehát a következő:  $A = T \times T \times T$   
 $x \quad y \quad t$

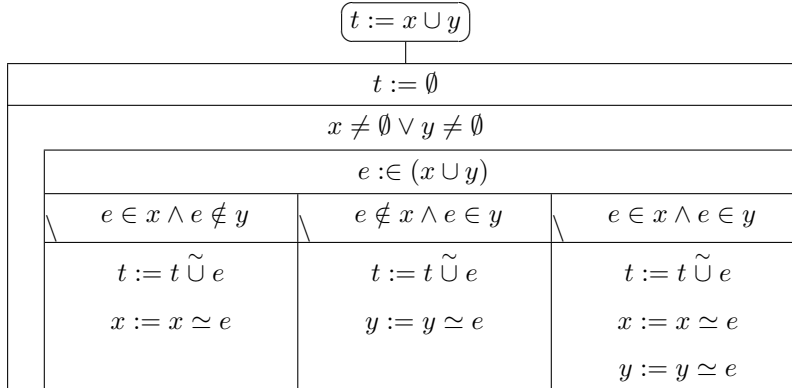
$$B = T \times T$$

$$x' \quad y'$$

$$Q : (x = x' \wedge y = y')$$

$$R : (t = x \cup y)$$

Idézzük fel az *unió* programját:



Most már csak a fenti programot kell transzformálni az eredeti állapottérre.

$$x \neq \emptyset \vee y \neq \emptyset \rightarrow t_0 \neq \emptyset \vee m \neq \emptyset$$

$$e \in (x \cup y) \rightarrow e.k \in (\{t_0.k\} \cup \{m.k\})$$

$$e \in x \rightarrow e.k \in \{t_0.k\}$$

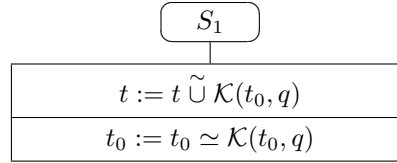
$$e \in y \rightarrow e.k \in \{m.k\}$$

Jelöljük  $q$ -val az  $e.k$  kulcsot, ekkor a megfelelő program:

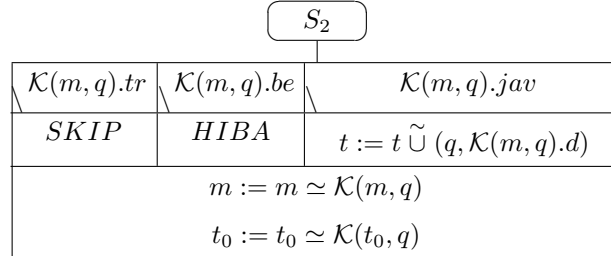
$t := \emptyset$		
$t_0 \neq \emptyset \vee m \neq \emptyset$		
$q \in (\{t_0.k\} \cup \{m.k\})$		
$q \in \{t_0.k\} \wedge q \notin \{m.k\}$	$q \in \{t_0.k\} \wedge q \in \{m.k\}$	$q \notin \{t_0.k\} \wedge q \in \{m.k\}$
$S_1$	$S_2$	$S_3$

Vizsgáljuk meg most, hogy mit kell tenni az elágazás egyes ágaiban:

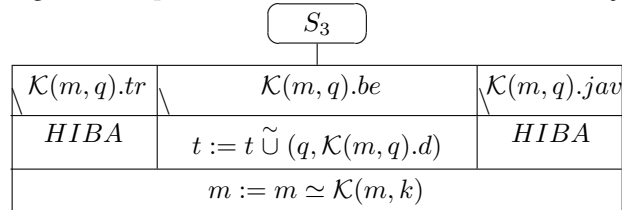
1. Ha a  $q$  kulcs csak a  $t_0$  eredeti törzsfájlból szerepelt, akkor a  $\mathcal{K}(t, q)$  elemre nem vonatkozott módosítás, változtatás nélkül kell kiírni az új törzsfájlból.



2. Ha a  $q$  kulcsérték mind az eredeti törzsfájlból, mind pedig a módosítófájlból szerepelt, akkor a törlés és a javítás műveletek végezhetők el. Ebben az ágban a  $q$  kulcsú elemet mindkét fájlból el kell hagyni.



3. Ha a  $k$  kulcs csak a módosítófájlból szerepelt, akkor csak a beszúrás műveletet lehet elvégezni, és a  $q$  kulcsú elemet ki kell törölni a módosítófájlból.

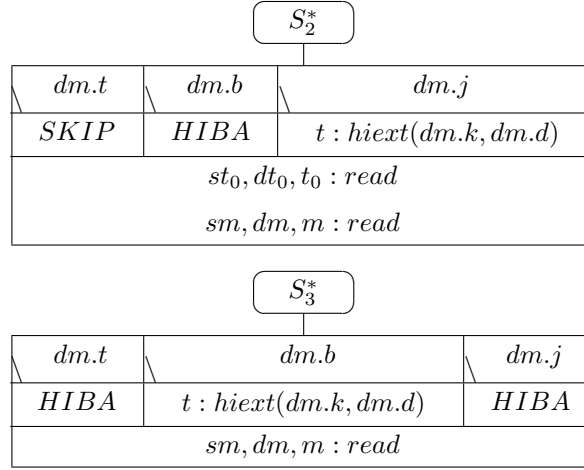




Ha a programnak hibajelzést is kell adnia, akkor azt a fenti struktogramokban *HIBA*-val jelzett helyeken kell megtennie. Térjünk most vissza az eredeti feladatra, ahol halmazok helyett szekvenciális fájlok szerepelnek. Legyen az input fájlok a *read* művelet értelmezve. Ekkor a program az alábbiak szerint alakul:

$st_0, dt_0, t_0 : read; sm, dm, m : read$		
$t := \langle \rangle$		
$st = norm \vee sm = norm$		
$(st_0 = sm \wedge dt_0.k < dm.k)$ $\vee sm = abnorm$	$st_0 = sm \wedge$ $dt_0.k = dm.k$	$(st_0 = sm \wedge dt_0.k > dm.k)$ $\vee st_0 = abnorm$
$t : hiext(dt_0)$	$S_2^*$	$S_3^*$
$st_0, dt_0, t_0 : read$		

ahol



### 14.2.2. Visszavezetés egyváltozós elemenkénti feldolgozásra

Állapottér-transzformációt alkalmazunk. Legyen  $X = seq(Y)$  és  $Y = (k : K, d : D', v : V')$ , ahol  $D' = D \cup \{\text{"üres"}\}$  és  $V' = V \cup \{\text{"üres"}\}$ .

Legyen  $x \in X$  és  $\{x.k\} = \{t_0.k\} \cup \{m.k\}$ .

$$\forall i \in [1..x.dom] : x_i.d = \begin{cases} \text{"üres"}, & \text{ha } x_i.k \notin \{t_0.k\}; \\ \mathcal{K}(t_0, x_i.k).d, & \text{ha } x_i.k \in \{t_0.k\}. \end{cases}$$

$$x_i.v = \begin{cases} \text{"üres"}, & \text{ha } x_i.k \notin \{m.k\}; \\ \mathcal{K}(m, x_i.k).v, & \text{ha } x_i.k \in \{m.k\}. \end{cases}$$

Legyen  $f : X \rightarrow T$ .

$$f(x) = \bigcup_{e \in \{x\}} f(\{e\}) \text{ és}$$

$$f(\{e\}) = \begin{cases} (e.k, e.d), & \text{ha } e.v = \text{"üres"}; \\ \text{"üres"}, & \text{ha } e.v.tr \wedge e.d = \text{"üres"}, \\ (e.k, e.v.d), & \text{ha } e.v.be \wedge e.d = \text{"üres"}, \\ \text{"üres"}, & \text{ha } e.v.jav \wedge e.d = \text{"üres"}; \\ \text{"üres"}, & \text{ha } e.v.tr \wedge e.d \neq \text{"üres"}, \\ (e.k, e.d), & \text{ha } e.v.be \wedge e.d \neq \text{"üres"}, \\ (e.k, e.v.d), & \text{ha } e.v.jav \wedge e.d \neq \text{"üres"}. \end{cases}$$

A specifikáció:

$$A = X \times T$$

$$x \quad t$$

$$B = X$$

$$x'$$

$$Q : (x = x')$$

$$R : (t = f(x'))$$

A halmazokra felírt megoldóprogram:

$t := \emptyset$		
$x \neq \emptyset$		
$e \in x$		
$e.d \neq \text{"üres"} \wedge$ $e.v = \text{"üres"}$	$e.d \neq \text{"üres"} \wedge$ $e.v = \text{"üres"}$	$e.d = \text{"üres"} \wedge$ $e.v = \text{"üres"}$
$d := (e.k, e.d)$	$S_2$	$S_3$
$t := t \tilde{\cup} d$		
$x := x \simeq e$		

ahol

$S_2$		
$e.tr$	$e.be$	$e.jav$
$HIBA$	$d := (e.k, e.v.d)$	$HIBA$

$S_3$		
$e.tr$	$e.be$	$e.jav$
$d := \emptyset$	$HIBA$	$d := (e.k, e.g(e.d))$

Térjünk vissza most az eredeti állapotérre:

$x \neq \emptyset$	$\Rightarrow$	$t_0 \neq \emptyset \vee m \neq \emptyset$						
$e \in x$	$\Rightarrow$	$q \in (\{t_0.k\} \cup \{m.k\})$						
$e.d \neq \text{"üres"} \wedge e.v = \text{"üres"}$	$\Rightarrow$	$q \in \{t_0.k\} \wedge q \notin \{m.k\}$						
$e.d = \text{"üres"} \wedge e.v \neq \text{"üres"}$	$\Rightarrow$	$q \notin \{t_0.k\} \wedge q \in \{m.k\}$						
$e.d \neq \text{"üres"} \wedge e.v \neq \text{"üres"}$	$\Rightarrow$	$q \in \{t_0.k\} \wedge q \in \{m.k\}$						
$x := x \simeq e$	$\Rightarrow$	<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td style="text-align: center;"><math>q \in \{t_0.k\}</math> <math>\wedge q \notin \{m.k\}</math></td> <td style="text-align: center;"><math>q \in \{t_0.k\}</math> <math>\wedge q \in \{m.k\}</math></td> <td style="text-align: center;"><math>q \notin \{t_0.k\}</math> <math>\wedge q \in \{m.k\}</math></td> </tr> <tr> <td style="text-align: center;"><math>t_0 := t_0 \simeq</math> <math>\mathcal{K}(t_0, q)</math></td> <td style="text-align: center;"><math>t_0 := t_0 \simeq</math> <math>\mathcal{K}(t_0, q)</math> <math>m := m \simeq</math> <math>\mathcal{K}(m, q)</math></td> <td style="text-align: center;"><math>m := m \simeq</math> <math>\mathcal{K}(m, q)</math></td> </tr> </table>	$q \in \{t_0.k\}$ $\wedge q \notin \{m.k\}$	$q \in \{t_0.k\}$ $\wedge q \in \{m.k\}$	$q \notin \{t_0.k\}$ $\wedge q \in \{m.k\}$	$t_0 := t_0 \simeq$ $\mathcal{K}(t_0, q)$	$t_0 := t_0 \simeq$ $\mathcal{K}(t_0, q)$ $m := m \simeq$ $\mathcal{K}(m, q)$	$m := m \simeq$ $\mathcal{K}(m, q)$
$q \in \{t_0.k\}$ $\wedge q \notin \{m.k\}$	$q \in \{t_0.k\}$ $\wedge q \in \{m.k\}$	$q \notin \{t_0.k\}$ $\wedge q \in \{m.k\}$						
$t_0 := t_0 \simeq$ $\mathcal{K}(t_0, q)$	$t_0 := t_0 \simeq$ $\mathcal{K}(t_0, q)$ $m := m \simeq$ $\mathcal{K}(m, q)$	$m := m \simeq$ $\mathcal{K}(m, q)$						

Használjuk fel azt a tényt, hogy a  $d := f(\{e\})$  értékadást kiszámító programokban és az  $x := x \simeq e$  értékadás megfelelőjében szereplő elágazások feltételrendszere megegyezik, továbbá a  $t := t \tilde{\cup} d$  értékadást csak azokba az ágakba írjuk bele, amelyekben  $d \neq \emptyset$ . Ekkor ugyanazt a programot kapjuk, mint az első megoldásban.

### 14.2.3. Visszavezetés kétváltozós elemenkénti feldolgozásra

A feladat megoldásának talán legegyszerűbb módja az, ha kétváltozós egyértékű – hibakezelés esetén kétértékű – elemenkénti feldolgozásra vezetjük vissza, úgy, hogy az elemeket a kulcsukkal azonosítjuk. Tekintsük a feladat eredeti specifikációját.

Ha a módosítófájl kulcs szerint egyértelmű, akkor az időszersűsítés függvénye ( $\Upsilon$ ) a kulcsokra nézve elemenként feldolgozható. A kulcsértékekre felírt függvény:

$$\begin{aligned}
\Upsilon(\{q\}, \emptyset) &= \{\mathcal{K}(t_0, q)\}. \\
\Upsilon(\emptyset, \{q\}) &= \begin{cases} \emptyset, & \text{ha } \mathcal{K}(m, q).tr; \\ \{(q, \mathcal{K}(m, q).d)\}, & \text{ha } \mathcal{K}(m, q).be; \\ \emptyset, & \text{ha } \mathcal{K}(m, q).jav. \end{cases} \\
\Upsilon(\{q\}, \{q\}) &= \begin{cases} \emptyset, & \text{ha } \mathcal{K}(m, q).tr; \\ \{\mathcal{K}(t_0, q)\}, & \text{ha } \mathcal{K}(m, q).be; \\ \{(q, \mathcal{K}(m, q).g(\mathcal{K}(t_0, q).d))\}, & \text{ha } \mathcal{K}(m, q).jav. \end{cases}
\end{aligned}$$

Ha ezt a fenti függvényt behelyettesítjük a kétváltozós elemenkénti feldolgozás tételébe, akkor ugyanahhoz a megoldóprogramhoz jutunk – csak sokkal rövidebb úton –, mint az első megoldásban.

A három megoldást összehasonlítva, két észrevételt tehetünk. Meglepő módon három látszólag teljesen különböző megoldási stratégiával ugyanahhoz a programhoz jutottunk. Valójában az eredmény nem annyira meglepő, hiszen tudjuk, hogy az unió kétváltozós elemenkénti feldolgozás.

A másik észrevételünk az, hogy minél erősebb tételt alkalmazunk, annál egyszerűbben jutunk el a megoldóprogramhoz.

### 14.3. Időszerűsítés nem egyértelmű módosítófájllal

Vajon miben változik a feladat, ha a módosítófájl kulcs szerint nem egyértelmű? Ebben az esetben a feladat nem elemenként feldolgozható, hiszen az elemek nem tekintethetők azonosnak a kulcsukkal. Erre a problémára kétféle megoldási módot is megvizsgálunk: az egyik jellemzően *adatabsztrakció*s a másik elsődlegesen *függvényabsztrakció*s megközelítés.

#### 14.3.1. Megoldás adatabsztrakcióval

Mint az előbb már említettük, ha a módosító fájl kulcs szerint nem egyértelmű, akkor a feladat nem elemenként feldolgozható. Próbáljuk meg azzá tenni. Ehhez arra van szükség, hogy a módosítófájlt kulcs szerint egyértelművé tegyük. Ezt egy állapotát-transzformáció segítségével könnyen elérhetjük, ugyanis csak annyit kell tennünk, hogy az azonos kulcsú módosítórekordokat egy új rekordba fogjuk össze. Így az új módosítórekord a következőképpen fog kinézni:

(kulcs, transzformációsorozat)

Az állapotát-transzformációt két lépésben adjuk meg. Legyen  $Z = seq(U)$ , ahol  $U = seq(F)$ .

$$A' = T \times Z \times T$$

$$t_0 \quad z \quad t$$

A  $z$ -re teljesül, hogy  $\forall i \in [1..dom(z)] : \forall j \in [1..dom(z_i)-1] : z_{i_j}.K = z_{i_{j-1}}.K$ ,  $z$  egyértelmű kulcsú, és a kapcsolat  $m$  és  $z$  között:  $seq(m|F) = seq(z|F)$ .

Legyen  $S = seq(V)$ ; és  $F' = (k : K, s : S)$  és  $X = seq(F')$ .

$$A' = T \times X \times T$$

$$t_0 \quad x \quad t$$

$x$  kulcs szerint egyértelmű,  $\{x.K\} = \{z.K\}$  és  $seq(x|V) = seq(z|V)$ .

Ezzel a módosítófájllal tehát eljutottunk a kétváltozós egyértékű (hibafájl használata esetén kétértékű) elemenkénti feldolgozáshoz. Az egyetlen különbség csak az, hogy az adott transzformációsorozat végrehajtását meg kell valósítanunk az eredeti

állapottéren. Ehhez szükségünk lesz az "üres" szimbólumra, amely értéket hozzávesszük az adat rész típusához:  $D' = D \cup \{\text{"üres"}\}$ . Az, hogy egy törzsrekord adat-része "üres", azt jelenti, hogy a rekordot nem kell kiírni az eredményfájlba.

Az elemenként feldolgozható függvényünk:

$$\begin{aligned} \Upsilon(\{q\}, \emptyset) &= \{\mathcal{K}(t_0, q)\} \\ \Upsilon(\emptyset, \{q\}) &= \{(q, \mathcal{K}(x, q).s(\text{"üres"}))\} \\ \Upsilon(\{q\}, \{q\}) &= \{(q, \mathcal{K}(x, q).s(\mathcal{K}(t_0, q).d))\} \end{aligned}$$

A transzformációsorozat elvégzése csak a transzformációk megfelelő sorrendje esetén lehetséges. Ha egy transzformációsorozat egy tagját nem lehet elvégezni, akkor azt a sorozatból ki kell hagyni (esetleg hibát kell jelezni). Ezzel az  $\Upsilon$  függvény egyértelműen definiált.

Írjuk fel tehát a fenti megfontolások alapján a kétváltozós elemenkénti feldolgozás programját a megfelelő behelyettesítésekkel:

$t := 0$		
$st_0, dt_0, t_0 : read$		
$sx, dx, x : read$		
$st_0 = norm \vee sx = norm$		
$sx = abnormal \vee (sx = st_0 \wedge dt_0.k < dx.k)$	$sx = st_0 \wedge dx.k = dt_0.k$	$st_0 = abnormal \vee (sx = st_0 \wedge dx.k < dt_0.k)$
$t : hiext(dt_0)$ $st_0, dt_0, t_0 : read$	$ak := dx.k$ $ad := dx.s(dt_0.d)$ $t : HIEXT(ad, ak)$ $st_0, dt_0, t_0 : read$ $sx, dx, x : read$	$ak := dx.k$ $ad := dx.s(\text{"üres"})$ $t : HIEXT(ad, ak)$ $sx, dx, x : read$

Definiálnunk kell még, hogy mit jelent a fenti struktogramban a transzformációsorozat elvégzése. Ehhez bevezetjük az  $f : \mathbb{N}_0 \rightarrow D'$  rekurzívan definiált függvényt:

$$dx.s(p) = f(dx.v.dom),$$

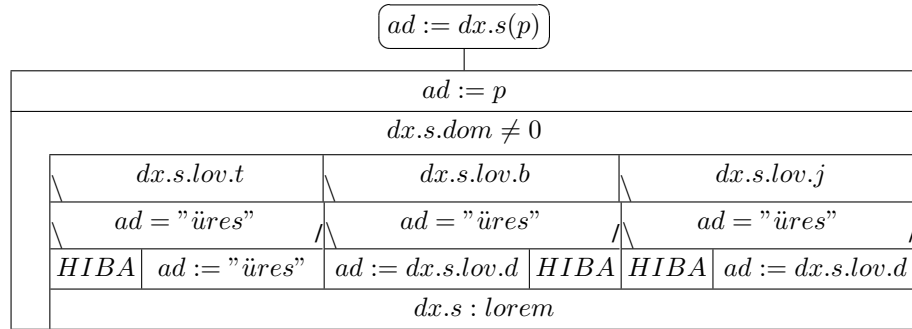
ahol

$$\begin{aligned} f(0) &= p; \\ f(i+1) &= dx.s_{i+1}(f(i)). \end{aligned}$$

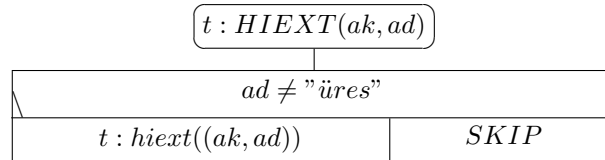
A fenti definícióban szereplő  $dx.v_{i+1}$  művelet elvégzésén az alábbi függvényértékeket értjük. Legyen  $d \in D'$ , ekkor:

$$dx.s_{i+1}(d) = \begin{cases} \text{"üres"}, & \text{ha } dx.s_{i+1}.t \wedge d \neq \text{"üres"}; \\ d, & \text{ha } dx.s_{i+1}.t \wedge d = \text{"üres"}; \\ d, & \text{ha } dx.s_{i+1}.b \wedge d \neq \text{"üres"}; \\ dx.s_{i+1}.d, & \text{ha } dx.s_{i+1}.b \wedge d = \text{"üres"}; \\ dx.s_{i+1}.d, & \text{ha } dx.s_{i+1}.j \wedge d \neq \text{"üres"}; \\ d, & \text{ha } dx.s_{i+1}.j \wedge d = \text{"üres"} \end{cases}$$

Az  $f$  függvényt kiszámító – a módosítássorozatot elvégző – program:



Mivel az aktuális adat értéke lehet "üres" is, a *hiext* művelet helyett az alábbi programot használjuk:



Térjünk most vissza az eredeti állapotérre. Ekkor a főprogram:

$t := 0$		
$st_0, dt_0, t_0 : read$		
$sm, dm, m : read$		
$st_0 = norm \vee sm = norm$		
$sm = abnorm \vee (sm = st_0 \wedge dt_0.k < dm.k)$	$sm = st_0 \wedge dm.k = dt_0.k$	$st_0 = abnorm \vee (sm = st_0 \wedge dm.k < dt_0.k)$
$t : hiext(dt_0)$ $st_0, dt_0, t_0 : read$	$ak := dm.k$ $ad := TR(dt_0.d, ak)$ $t : HIEXT(ad, ak)$ $st_0, dt_0, t_0 : read$	$ak := dm.k$ $ad := TR("üres", ak)$ $t : HIEXT(ad, ak)$

Az  $ad := TR(p, ak)$  a már korábban definiált rekurzív függvényt kiszámító program megvalósítása az eredeti állapottéren:

$ad := TR(p, ak)$					
$ad := p$					
$sm = norm \wedge ak = dm.k$					
$dm.t$		$dm.b$		$dm.j$	
$ad = "üres"$		$ad = "üres"$		$ad = "üres"$	
<i>HIBA</i>	$ad := "üres"$	$ad := dm.d$	<i>HIBA</i>	<i>HIBA</i>	$ad := dm.d$
$sm, dm, m : read$					

### 14.3.2. Megoldás függvényabsztrakcióval

Egy adott feladat megoldását mindig elkerülhetetlenül befolyásolja a specifikáció módja. A függvényabsztrakció lényege abban rejlik, hogy a megoldást egy alkalmasan választott függvény helyettesítési értékének kiszámítására vezetjük vissza.

**Kulcsok egyértelműsítése.** A gyakorlatban sokszor találkozhatunk olyan fájlokkal, amelyek rekordjaiban van valamilyen kulcsmező, ami szerint a fájl rendezett, de a fájl mégsem egyértelmű kulcs szerint és így a kulcsmezőre vonatkoztatva nem elemenként feldolgozható. A következőkben egy olyan technikát fogunk bemutatni, amelyben egy új kulcsot definiálunk a fájlra, és ezen új kulcs szerint a fájl már egyértelmű.

Tegyük fel, hogy a fájl  $U = (k : K, z : Z)$  típusú rekordokból áll. Az új kulcsot úgy kapjuk, hogy az egymás után levő azonos kulcsokat megsorszámozzuk. Legyen tehát  $V = (h : H, z : Z)$ , ahol  $H = (k : K, s : \mathbb{N})$ . Legyen továbbá  $g : seq(U) \rightarrow seq(V)$ :

1.  $g(u).dom = u.dom$
2.  $g(u)_{1.s} = 1$  és  $\forall i \in [1..u.dom - 1]$ :

$$g(u)_{i+1.s} = \begin{cases} 1, & \text{ha } u_i.k \neq u_{i+1}.k; \\ g(u)_{i.s} + 1, & \text{ha } u_i.k = u_{i+1}.k. \end{cases}$$

3.  $\forall i \in [1..u.dom]$ :

$$g(u)_i.k = u_i.k \text{ és } g(u)_i.z = u_i.z.$$

Ekkor tetszőleges  $u \in seq(U)$  esetén – feltéve, hogy  $u$  a  $k$  kulcs szerint rendezett –  $g(u)$  a  $h$  kulcs szerint rendezett és egyértelmű.

Természetesen a fenti megszámozást általában csak az absztrakció leírására használjuk, és csak ritkán fordul elő, hogy a  $g$  függvény által definiált absztrakciót meg is valósítjuk.

**Megoldás extrémális elemmel.** Induljunk ki egy olyan absztrakt fájlból, mint amelyet az egyváltozós elemenkénti feldolgozásra való visszavezetésben használtunk. Természetesen, mivel most a módosítófájl nem egyértelmű kulcs szerint, az  $X$  absztrakt fájl definíciója kissé módosul: ha egy kulcs mindkét fájlban szerepel, akkor a törzsrekordot az első rá vonatkozó módosítórekorddal vonjuk össze, és az esetlegesen előforduló további azonos kulcsú módosítórekordokból pedig egy-egy olyan absztrakt rekordot képezünk, amelynek adat része "üres".

Ez az absztrakció az imént bemutatott egyértelműsítő leképezésen keresztül implicit módon írható le: legyen  $t_0 \in T$ ,  $m \in M$  és  $x \in X$ . Ekkor

$$\{g(x).h\} = \{g(t_0).h\} \cup \{g(m).h\} \cup \{(extr, 1)\},$$

és  $\forall i \in [1..x.dom]$ :

$$g(x)_i.d = \begin{cases} \mathcal{K}(g(t_0), g(x)_i.h).d, & \text{ha } g(x)_i.h \in \{g(t_0).h\}; \\ \text{"üres"} & \text{különbén.} \end{cases}$$

$$g(x)_i.v = \begin{cases} \mathcal{K}(g(m), g(x)_i.h).v, & \text{ha } g(x)_i.h \in \{g(m).h\}; \\ \text{"üres"} & \text{különbén.} \end{cases}$$

Figyeljük meg, hogy ez a megoldás egy a gyakorlatban sokszor hasznos eszközt használ, az utolsó utáni elem bevezetését (*read* extrémális elemmel).



A jobb áttekinthetőség érdekében a függvényt most komponensenként fogjuk felírni.  $f : \mathbb{N}_0 \rightarrow T \times K' \times D'$ ,  $f = (f_1, f_2, f_3)$  :

$$\begin{aligned}
 f(0) &= (\langle \rangle, \text{"üres"}, \text{"üres"}) \\
 f_1(i+1) &= \begin{cases} f_1(i), & \text{ha } f_2(i) = x_{i+1}.k \vee \\ & (f_2(i) \neq x_{i+1}.k \wedge f_3(i) = \text{"üres"}); \\ \text{hiext}(f_1(i), (f_2(i), f_3(i))), & \text{ha } f_2(i) \neq x_{i+1}.k \wedge f_3(i) \neq \text{"üres"}. \end{cases} \\
 f_2(i+1) &= \begin{cases} f_2(i), & \text{ha } f_2(i) = x_{i+1}.k; \\ x_{i+1}.k, & \text{ha } f_2(i) \neq x_{i+1}.k. \end{cases} \\
 f_3(i+1) &= \begin{cases} x_{i+1}.v(f_3(i)), & \text{ha } f_2(i) = x_{i+1}.k; \\ x_{i+1}.v(x_{i+1}.d), & \text{ha } f_2(i) \neq x_{i+1}.k. \end{cases}
 \end{aligned}$$

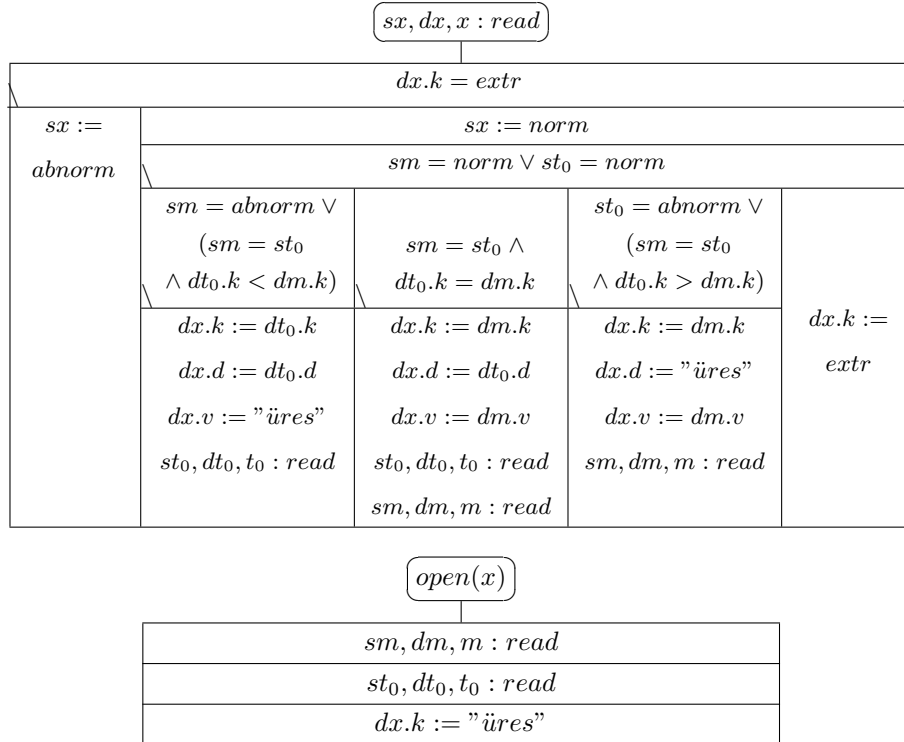
Ezt a függvényt használva a feladat specifikációja:

$$\begin{aligned}
 A &= X \times T \\
 &\quad x \quad t \\
 B &= X \\
 &\quad x' \\
 Q &: (x = x') \\
 R &: (t = f_1(x'.dom))
 \end{aligned}$$

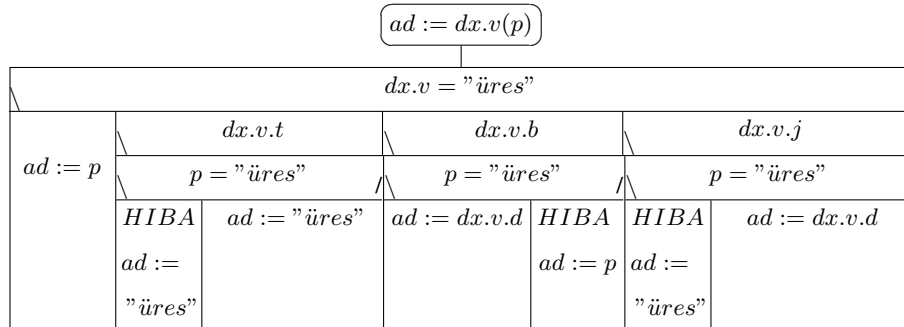
Ez a feladat visszavezethető a fájlra felírt rekurzívan megadott függvény helyettesítési értékének kiszámítására:

$open(x)$
$sx, dx, x : read$
$t, ak, ad := \langle \rangle, \text{"üres"}, \text{"üres"}$
$sx = norm$
$ak = dx.k$
$ad := dx.v(ad)$
$ad = \text{"üres"}$
$SKIP \quad t : \text{hiext}(ak, ad)$
$ak := dx.k$
$ad := dx.v(dx.d)$
$sx, dx, x : read$

Az  $x$  absztrakt fájl műveleteinek megvalósítása:



A transzformáció elvégzésének megvalósítására az  $x$  absztrakt fájl használjuk:



A fenti megoldási módokat összehasonlítva látható, hogy minél magasabb absztrakciós szintű fogalmakat használunk, annál egyszerűbben tudjuk kezelni a feladatot.

Nagyon sok esetben az adat- és függvényabsztrakció is alkalmazható egy feladat megoldásakor, sőt mint azt az iménti példa mutatja, a kettő kombinációja is egy lehetséges út.

**Megoldás függvénykompozícióval.** Olyan megoldást is adhatunk a feladatra, amelynek a specifikációjában az utófeltétel egy kompozícióval adott függvény kiszámítása, ahol a kompozíció egy rekurzív módon megadott függvényből és egy esetválasztással definiált függvényből áll. Ebben az esetben nincs szükség utolsó utáni elemre.

$$A = X \times T$$

$$\begin{array}{cc} x & t \end{array}$$

$$B = X$$

$$x'$$

$$Q : (x = x')$$

$$R : (t = HIEXT(f_1(x'.dom), (f_2(x'.dom), f_3(x'.dom)))),$$

ahol  $f : \mathbb{N}_0 \rightarrow T \times K \times D'$ ,

$$f(0) = (\langle \rangle, \text{"üres"}, \text{"üres"})$$

$$f(i+1) = \begin{cases} (f_1(i), f_2(i), x_{i+1}.v(f_3(i))), & \text{ha } x_{i+1}.k = f_2(i); \\ (hiext(f_1(i), (f_2(i), f_3(i))), \\ x_{i+1}.k, x_{i+1}.v(x_{i+1}.d)), & \text{ha } x_{i+1}.k \neq f_2(i); \end{cases}$$

és  $HIEXT : T \times (K \times D') \rightarrow T$ ,

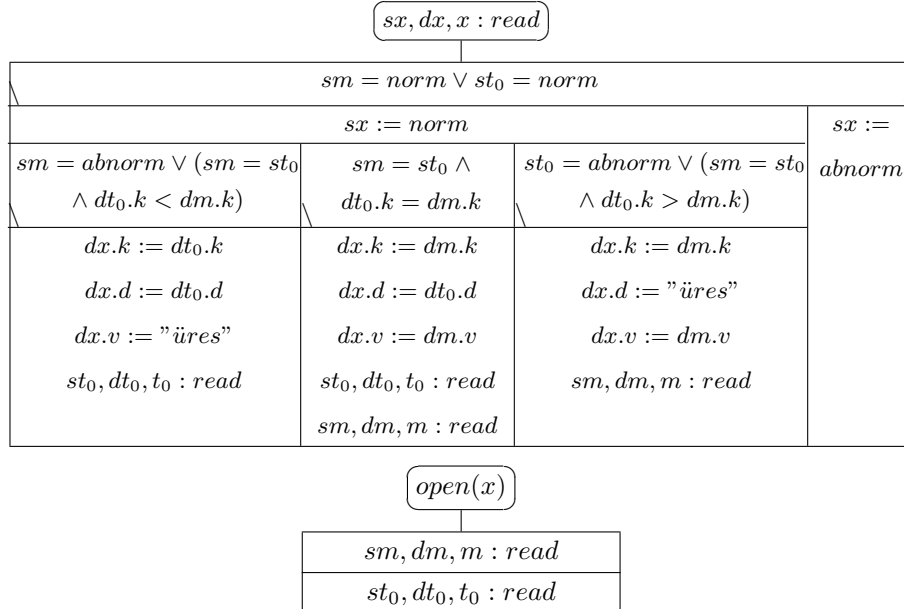
$$HIEXT(t, k, d) = \begin{cases} hiext(t, (k, d)), & \text{ha } d \neq \text{"üres"}; \\ t, & \text{ha } d = \text{"üres"}. \end{cases}$$

Az  $f$  függvény kezdőértékének definíciójában szereplő  $EXTR$  kulcsérték tetszőleges olyan kulcsérték lehet, amely egyik fájlban sem fordul elő.

Mivel az utófeltétel függvénykompozícióval adott, a megoldás egy szekvencia lesz, amelynek első része az  $f$ , második része pedig a  $HIEXT$  függvényt számítja ki. Az  $f$  függvény egyes komponenseinek rendre a  $t, ak, ad$  változók felelnek meg.

$open(x)$
$sx, dx, x : read$
$t, ak, ad := \langle \rangle, "üres", "üres"$
$sx = norm$
$ak = dx.k$
$ad := dx.v(ad)$
$t : HIEXT(ak, ad)$
$ad := dx.v(dx.d)$
$ak := dx.k$
$sx, dx, x : read$
$t : HIEXT(ak, ad)$

Az  $x$  absztrakt fájl műveleteinek megvalósítása:



Az  $ad := dx.v(p)$  transzformáció elvégzésének megvalósítása ugyanaz, mint az előbb.

## 14.4. Feladatok

- 14.1.** Adott az egész számokat tartalmazó  $x$  vektor. Válogassuk ki az  $y$  sorozatba a vektor pozitív elemeit!
- 14.2.** Adott két vektorban egy angol–latin szótár: az egyik vektor  $i$ -edik eleme tartalmazza a másik vektor  $i$ -edik elemének a jelentését. Válasszuk ki egy vektorba azokat az angol szavakat, amelyek szóalakja nem egyezik meg a latin megfelelőjével.
- 14.3.** Adott egy  $x$  sorozat, ami egy vállalat dolgozóinak adataiból áll. Egy dolgozóról a következő adatokat tudjuk:
- azonosító szám;
  - születési adatok (idő, hely, anyja neve);
  - lakcím;
  - iskolai végzettség;
  - a munkaviszony kezdete;
  - beosztás;
  - fizetés.

Adott még az  $y$  sorozat, amely azonosítókat tartalmaz. Mindkét sorozat az azonosító szám szerint rendezett. Adjuk meg a  $z$  sorozatban azoknak a dolgozóknak az adatait, akiknek az azonosítója szerepel  $y$ -ban, és a munkaviszonyuk kezdete egy adott évnél régebbi!

- 14.4.** Az  $x$  sorozat egy szöveget tartalmaz. Tömörítsük a szöveget úgy, hogy minde-  
nütt, ahol több szóköz van egymás mellett, csak egy szóközt hagyjunk meg!
- 14.5.** Adott egy szöveg, ami mondatokból áll, és a mondatok végén pont van. Módosítsuk a szöveget úgy, hogy minden mondat végét jelző pontot pontosvesszőre cserélünk! A mondatokban lehetnek idézetek, és az idézetek is tartalmazhatnak idézeteket tetszőleges mélységben (az idézetet egy kezdő idézőjel vezeti be és egy záró idézőjel jelzi a végét). Azok a pontok, amelyek egy idézet belsejében vannak, nem jelentik a mondat végét! Feltesszük, hogy a szövegben az idézőjelek kiegyensúlyozottak.
- 14.6.** Adott az  $x$  sorozat, amely egy szöveget tartalmaz. Másoljuk át  $x$ -et a  $z$  sorozatba úgy, hogy a kerek zárójelek közé írt szöveget elhagyjuk! (A zárójelekkel együtt.)
- 14.7.** Egy szekvenciális fájl (megengedett művelet az  $sx, dx, x : read$ ) szöveget tartalmaz, melyben a szavakat szóközök (esetleg több szóköz) választják el. Számoljuk meg, hány 5 jelnél rövidebb szó van a szövegben!

**14.8.** Adott egy szekvenciális fájl (megengedett művelet az  $sx, dx, x : read$ ), ami egy bank tranzakcióit tartalmazza: egy ügyfél adatait tartalmazó rekord után olyan rekordok következnek, amelyek az ügyfél tranzakcióit írják le.

- Ügyfél = (Azonosító, Számla összege)
- Tranzakció = (Kivét-betét, Összeg)

Állítsuk elő azt a fájlt, ami az ügyfeleknek a bankban levő pillanatnyi összegeit tartalmazza ügyfél típusú rekordokban!

**14.9.** Adott egy karakterekből álló szekvenciális fájl (megengedett művelet az  $sx, dx, x : read$ ). Számoljuk meg, hogy a szöveg hány szóból áll, ha a 12 jelnél hosszabb szavakat két szónak tekintjük! (A szavakat tetszőleges számú szóköz választhatja el.)

**14.10.** Adott az  $x$  szekvenciális fájl (megengedett művelet az  $sx, dx, x : read$ ), amely egy szöveget tartalmaz. Állapítsuk meg, hány olyan szó van a szövegben, amely tartalmaz „R” betűt!

**14.11.** Adott az  $x$  szekvenciális fájl, melynek elemei egy vezetéknevet és egy keresztnévet tartalmaznak. A fájl a keresztnévek szerint rendezett. Gyűjtsük ki a fájlból a különböző keresztnéveket és azt, hogy hányszor szerepelnek!

**14.12.** Az  $x$  sorozat egy szöveget tartalmaz, ahol a szavakat egy vagy több szóköz választja el. Számoljuk meg, hogy a sorozatban hány  $k$  betűnél hosszabb szó van!

**14.13.** Adott egy évfolyam-nyilvántartás névsor szerint a következő adatokkal: név, csoportszám, átlag. Félévenként módosítják a nyilvántartást, ekkor az alábbi változások történhetnek:

- jöhetnek új hallgatók az évfolyamra;
- elmehetnek hallgatók (törölni kell a nyilvántartás-ból);
- változhat a hallgató átlaga;
- megváltozhat a hallgató csoportszáma.

A változások is névsor szerint rendezett fájlban vannak. Végezzük el az adatok frissítését.

**14.14.** Végezzük el egy áruház osztályain kint lévő árukészlet nyilvántartásának napi frissítését. Az adatbázisban az árukról nyilván van tartva, melyik osztályon árusítják, mi a neve, mennyi az ára, és hány db van belőle a boltban. A nyilvántartás osztályok szerint, azon belül árunév szerint rendezett. Egy módosítófájl tartalmazza ugyanígy rendezve a napi fogyást, egy másik az új szállítmányt, egy harmadik pedig az árváltozásokat.

**14.15.** Adott egy raktárnyilvántartás (áru, mennyiség) adatokkal, névsor szerint rendezve. Minden nap érkezik három fájl:

- a gyártótól: szállítás; melyik áruból mennyit hozott;
- a boltból: igénylés; melyik áruból mennyit rendel a bolt;
- a főnöktől: selejtezés; csak áruneveket tartalmaz, amiket törölni kell a nyilvántartásból.

Végezzük el az adatok módosítását. Ha nincs annyi áru, amennyit a bolt igényel, akkor ezt jelezzük, de amennyit lehet, adjunk ki a boltnak. Ha egy áru mennyisége 0-ra csökken, akkor töröljük ki a nyilvántartásból. Ha a gyártó olyan árut hoz, amilyen még nincs, azt fel kell venni a nyilvántartásba.

**14.16.** Adott a virágokról egy nyilvántartás (virágnév, szín, magasság) adatokkal, virágnév szerint növekvően rendezve egy  $x$  fájlban. Adott továbbá három ugyanilyen szerkezetű módosítófájl ( $a, b, c$ ). Végezzük el az  $x$  fájl frissítését a következők szerint:

- Ha egy virág benne van  $a$ -ban, akkor szűrjük be  $x$ -be.
- Ha egy virág benne van  $b$ -ben, akkor módosítsuk  $x$ -et a  $b$ -beli rekorddal.
- Ha egy virág benne van  $c$ -ben, akkor töröljük  $x$ -ből.

Ha egy rekord az  $a, b, c$  fájlok közül többen is benne van, akkor a módosításokat a fenti sorrendben kell elvégezni (először  $a$ , majd  $b$  és  $c$ ).





## 15. fejezet

# Megoldások

### 15.1. Alapfogalmak

1. Jelölje  $A$  az  $m$  elemű és  $B$  az  $n$  elemű halmazt.

- a)  $A \cap B$  lehet üres is, legalább nulla, legfeljebb  $\min\{n, m\}$  eleme van.
- b)  $A \cup B$ -nek legalább  $\max\{n, m\}$ , legfeljebb  $n + m$  eleme van.
- c)  $A \times B$ -nek minden esetben  $n \cdot m$  eleme van.
- d)  $A \setminus B$ -nek legalább  $\max\{0, m - n\}$ , legfeljebb  $m$  eleme van.

2. a) Két részre bontjuk a bizonyítást, egy balról jobbra és egy jobbról balra irányra.

$\implies$  : Egyrészt minden  $H$  relációra teljesül, hogy (1)  $H \subseteq \mathcal{D}_H \times \mathcal{R}_H$ .  
Másképp  $(x, y) \in \mathcal{D}_H \times \mathcal{R}_H$  esetén  $x \in \mathcal{D}_H \implies \exists b \in B : (x, b) \in H$ , valamint  $y \in \mathcal{R}_H \implies \exists a \in A : (a, y) \in H$ , így a bizonyítandó állítás bal oldalán szereplő feltétel szerint  $(x, y) \in H$ .  
Ezzel beláttuk, hogy (2)  $\mathcal{D}_H \times \mathcal{R}_H \subseteq H$ .

(1)-ből és (2)-ből következik, hogy  $H = \mathcal{D}_H \times \mathcal{R}_H$ . Tehát  $K = \mathcal{D}_H$  és  $L = \mathcal{R}_H$  választással készen is vagyunk.

$\impliedby$  : Legyen  $K \subseteq A$ ,  $L \subseteq B$ ,  $H = K \times L$ . Ha  $(a, b), (c, d) \in H$ , akkor  $a, c \in K$  és  $b, d \in L \implies (a, d) \in K \times L = H$ . Ezzel ezt az irányt is beláttuk.

- b) Ha  $H \neq \emptyset$ , akkor nyilvánvaló, hogy  $K = \mathcal{D}_H$  és  $L = \mathcal{R}_H$ .  
Ha  $H = \emptyset$ , akkor  $K = \emptyset$  vagy  $L = \emptyset$ , a másik pedig tetszőleges.

$$3. R^{-1}(B) = \{a \in \mathcal{D}_R \mid R(a) \subseteq B\} = \mathcal{D}_R$$

$$4. H = \{(1, 1), (1, 2), (1, 3), (2, 1), (2, 2), (3, 1)\}$$

Mivel  $R$  determinisztikus, az inverzkép és az őskép megegyezik,

$$R^{(-1)}(H) = R^{-1}(H) = \{(1, 1), (2, 1)\}.$$

$$5. R \subseteq (\mathbb{N} \times \mathbb{N}) \times (\mathbb{Z} \times \mathbb{Z}) \text{ és } \mathcal{D}_R = \mathbb{N} \times \mathbb{N}.$$

$$\forall (u, v) \in \mathbb{N} \times \mathbb{N} : R((u, v)) = \{(u + v, v), (u - v, v)\}.$$

$$\begin{aligned} R^{(-1)}(H) &= \{(u, v) \in \mathbb{N} \times \mathbb{N} \mid \{(u + v, v), (u - v, v)\} \cap H \neq \emptyset\} = \\ &= \{(u, v) \in \mathbb{N} \times \mathbb{N} \mid (u + v, v) \in H \text{ vagy } (u - v, v) \in H\} = \\ &= \{(u, v) \in \mathbb{N} \times \mathbb{N} \mid (u + v, v \in \mathbb{N} \text{ és } u + 2v < 5) \text{ vagy} \\ &\quad (u - v, v \in \mathbb{N} \text{ és } u < 5)\} = \\ &= \{(u, v) \in \mathbb{N} \times \mathbb{N} \mid u + 2v < 5 \text{ vagy } (u > v \text{ és } u < 5)\} = \\ &= \{(1, 1), (2, 1), (3, 1), (3, 2), (4, 1), (4, 2), (4, 3)\} \end{aligned}$$

$$\begin{aligned} R^{-1}(H) &= \{(u, v) \in \mathbb{N} \times \mathbb{N} \mid \{(u + v, v), (u - v, v)\} \subseteq H\} = \\ &= \{(u, v) \in \mathbb{N} \times \mathbb{N} \mid (u + v, v) \in H \text{ és } (u - v, v) \in H\} = \\ &= \{(u, v) \in \mathbb{N} \times \mathbb{N} \mid (u + v, v \in \mathbb{N} \text{ és } u + 2v < 5) \text{ és} \\ &\quad (u - v, v \in \mathbb{N} \text{ és } u < 5)\} = \\ &= \{(u, v) \in \mathbb{N} \times \mathbb{N} \mid u + 2v < 5 \text{ és } u > v\} = \\ &= \{(2, 1)\} \end{aligned}$$

6. Mivel  $R$  determinisztikus, az inverzkép és az őskép megegyezik,

$$R^{(-1)}(H) = R^{-1}(H) = \{(u, v) \in \mathbb{N} \times \mathbb{N} \mid f(u, v) + v < 5\}.$$

7.  $R^{-1}(B \setminus Q)$  azon pontok halmaza, amelyekből *biztosan*  $B \setminus Q$ -ba jutunk, vagyis  $\mathcal{D}_R$ -ből azokat a pontokat hagyjuk el, amelyekből *eljuthatunk*  $Q$ -ba:

$$\begin{aligned} R^{-1}(B \setminus Q) &= \{a \in \mathcal{D}_R \mid R(a) \subseteq B \setminus Q\} = \\ &= \mathcal{D}_R \setminus \{a \in A \mid R(a) \cap Q \neq \emptyset\} = \mathcal{D}_R \setminus R^{(-1)}(Q). \end{aligned}$$

Mivel  $\mathcal{D}_R \subseteq A$  és  $R^{-1}(Q) \subseteq R^{(-1)}(Q)$ ,

$$\mathcal{D}_R \setminus R^{(-1)}(Q) \subseteq A \setminus R^{(-1)}(Q) \subseteq A \setminus R^{-1}(Q).$$

Tehát  $R^{-1}(B \setminus Q) \subseteq A \setminus R^{-1}(Q)$ .

A fordított irányú tartalmazás azonban nem áll fenn, ugyanis például  $\mathcal{D}_R \subset A$  esetén  $\forall a \in A \setminus \mathcal{D}_R : a \in A \setminus R^{-1}(Q)$ , de  $a \notin R^{-1}(B \setminus Q)$ .

8. Ha létezik olyan  $a \in \mathcal{D}_R$ , amelyre  $R(a)$  valódi részhalmaza az értékkészletnek, akkor  $R(a)$  ősképe  $a \in R^{-1}(R(a))$  miatt biztosan nem üres. Tehát  $\forall a \in \mathcal{D}_R : R(a) = \mathcal{R}_R$  feltétel szükséges, és nyilvánvalóan elégséges is. Pontosabban azok a relációk lesznek megfelelőek, amelyekre ez teljesül.

Például:  $R = \{(1, 1)\}$ ,  $R = \{(1, 1), (1, 2)\}$ ,  $R = \{(1, 1), (1, 2), (2, 1), (2, 2)\}$ .

9.  $F = \{(a, b) \mid b \text{ az } a\text{-nak valódi osztója}\}$ ,  $G = \{(a, b) \mid b \text{ az } a\text{-nak fele}\}$ .

- a)  $G \circ F$  egy természetes számhoz az összes páros valódi osztójának felét rendeli:

$$G \circ F = \{(a, b) \in \mathbb{N} \times \mathbb{N} \mid 2b \mid a \text{ és } 2b \neq a\}.$$

- b)  $G \odot F$  egy olyan természetes számhoz, amelynek minden valódi osztója páros, azok felét rendeli:

$$G \odot F = \{(2^k, 2^l) \mid k, l \in \mathbb{N}_0 \text{ és } l \leq k - 2\}.$$

- c)  $F^{(-1)} \circ G^{(-1)}$  egy természetes számhoz a kétszeresének valódi többszöröseit rendeli. Mivel az 1.5. példa alapján  $F^{(-1)} \circ G^{(-1)} = (G \circ F)^{(-1)}$ , az a) pont felhasználásával:

$$F^{(-1)} \circ G^{(-1)} = (G \circ F)^{(-1)} = \{(b, a) \in \mathbb{N} \times \mathbb{N} \mid 2b \mid a \text{ és } 2b \neq a\}.$$

- d)  $F^{-1}(G^{-1}(Y)) = F^{-1}(\{a \in \mathbb{N} \mid 2 \mid a \text{ és } \frac{a}{2} \in \{1, 2\}\}) =$   
 $= F^{-1}(\{2, 4\}) = \{a \in \mathcal{D}_F \mid F(a) \subseteq \{2, 4\}\} =$   
 $= \underbrace{\{a \in \mathcal{D}_F \mid \{b \in \mathbb{N} \mid b \mid a \text{ és } b \neq 1 \text{ és } b \neq a\} \subseteq \{2, 4\}\}}_{\text{van valódi osztója minden valódi osztója benne van a } \{2, 4\} \text{ halmazban}} = \{4, 8\}.$

- e) Az a) pont alapján  $(G \circ F)^{-1}(Y)$  azon természetes számok halmaza, amelyeknek van páros valódi osztója, és minden páros valódi osztójuk fele benne van az  $Y = \{1, 2\}$  halmazban. Tehát olyan kettőnél nagyobb páros számok halmaza, amelyeknek minden páros valódi osztója benne van a  $\{2, 4\}$  halmazban.

$$(G \circ F)^{-1}(Y) = \{a \in \mathbb{N} \mid 2 \mid a \text{ és } a > 2 \text{ és } \{b \in \mathbb{N} \mid 2 \mid b \text{ és } b \mid a \text{ és } b \neq a\} \subseteq \{2, 4\}\} =$$

$$= \{4, 8\} \cup \{2p \mid p \text{ prím}\}.$$

f) Az a) pont alapján  $(G \circ F)^{(-1)}$  egy természetes számhoz a kétszeresének valódi többszöröseit rendeli:

$$(G \circ F)^{(-1)} = \{(b, a) \in \mathbb{N} \times \mathbb{N} \mid 2b|a \text{ és } 2b \neq a\}.$$

10. Mivel  $f$  függvény,  $\lceil f \rceil = \lfloor f \rfloor = \{1, 2, 3, 5\}$ .  
 $\lceil f \circ R \rceil = \{2, 4\}$ ,  $\lfloor f \circ R \rfloor = \{1, 2, 3, 4\}$ .

11. Nem igaz, ugyanis az 1.5. példa alapján:

$$Q^{(-1)} \circ R^{(-1)} = (R \circ Q)^{(-1)} \neq (R \odot Q)^{(-1)}.$$

Megjegyzés: Mivel  $R \odot Q \subseteq R \circ Q$ , ezért  $(R \odot Q)^{(-1)} \subseteq Q^{(-1)} \circ R^{(-1)}$ .

12. A kompozíciók definíciójából adódik, hogy  $\mathcal{D}_{G \odot F} \subseteq \mathcal{D}_{G \circ F}$ , valamint  $\forall a \in \mathcal{D}_{G \circ F} : G \circ F(a) = G(F(a))$  és  $\forall a \in \mathcal{D}_{G \odot F} : G \odot F(a) = G(F(a))$ . Ezeket felhasználva:

$$\begin{aligned} (G \circ F)^{-1}(Y) &= \{a \in \mathcal{D}_{G \circ F} \mid G(F(a)) \subseteq Y\} \supseteq \\ &\supseteq \{a \in \mathcal{D}_{G \odot F} \mid G(F(a)) \subseteq Y\} = (G \odot F)^{-1}(Y). \end{aligned}$$

Mivel az 1.6. példa alapján  $(G \odot F)^{-1}(Y) = F^{-1}(G^{-1}(Y))$ , ebből azt kapjuk, hogy:

$$(G \circ F)^{-1}(Y) \supseteq F^{-1}(G^{-1}(Y)).$$

A fordított irányú tartalmazás azonban nem áll fenn, ugyanis például a 15.1. ábrán  $(G \circ F)^{-1}(\{1, 2\}) = \{1\}$ , de  $F^{-1}(G^{-1}(\{1, 2\})) = F^{-1}(\{1\}) = \emptyset$ .

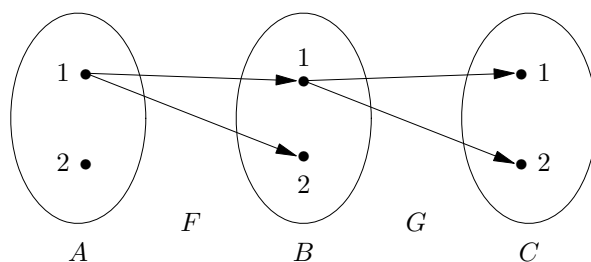
Ha  $G$  vagy  $F$  függvény, akkor  $G \circ F = G \odot F$  miatt:

$$(G \circ F)^{-1}(Y) = (G \odot F)^{-1}(Y) = F^{-1}(G^{-1}(Y)).$$

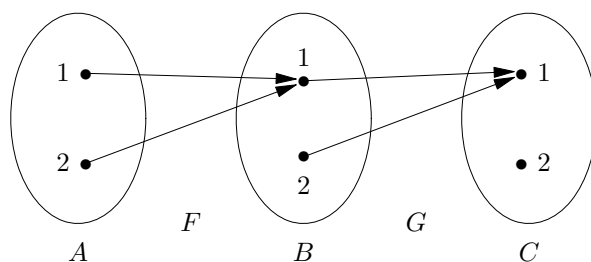
13. Egyik irányú tartalmazás sem áll fenn. Egyrészt a 15.1. ábrán  $G \odot F = \emptyset \implies (G \odot F)^{(-1)} = \emptyset$ , de  $F^{(-1)} \odot G^{(-1)} = \{(1, 1), (2, 1)\}$ , tehát:

$$F^{(-1)} \odot G^{(-1)} \not\subseteq (G \odot F)^{(-1)}.$$

Továbbá a 15.2. ábrán  $(G \odot F)^{(-1)} = \{(1, 1), (1, 2)\}$ , de  $F^{(-1)} \odot G^{(-1)} = \emptyset$ , tehát:



15.1. ábra. Ellenpélda



15.2. ábra. Ellenpélda

$$(G \circ F)^{(-1)} \not\subseteq F^{(-1)} \circ G^{(-1)}.$$

Ha  $G$  vagy  $F$  függvény, akkor  $G \circ F = G \circ F$ , és felhasználva, hogy az 1.5. példa alapján  $(G \circ F)^{(-1)} = F^{(-1)} \circ G^{(-1)}$ :

$$(G \circ F)^{(-1)} = (G \circ F)^{(-1)} = F^{(-1)} \circ G^{(-1)} \supseteq F^{(-1)} \circ G^{(-1)}.$$

A fordított irányú tartalmazás azonban ilyenkor sem áll fenn (ld. 15.2. ábra).

14. Definíció szerint  $\mathcal{D}_{Q \circ F} \subseteq \mathcal{D}_{Q \circ F}$ .

15. Ha  $\forall b \in \mathcal{R}_R \cap \mathcal{D}_f : f(b) = \text{hamis}$ , akkor minden  $a$  pontra  $f \circ R(a) = \{\text{hamis}\}$ , tehát  $\lceil f \circ R \rceil = \lfloor f \circ R \rfloor = \emptyset$ .

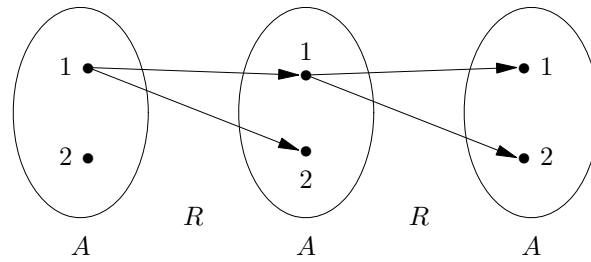
Például:  $f = \text{Hamis}$ ,  $R$  tetszőleges.

16. Igaz, ugyanis felhasználva, hogy az 1.5. példa alapján  $(G \circ F)^{(-1)} = F^{(-1)} \circ G^{(-1)}$ :

$$(R^{(-1)})^2 = R^{(-1)} \circ R^{(-1)} = (R \circ R)^{(-1)} = (R^2)^{(-1)}.$$

17. Nem igaz. Tekintsük például a 15.3. ábrát, és legyen  $H = \{1, 2\}$ . Ekkor ugyanis:

$$R^{-1}(R^{-1}(H)) = R^{-1}(\{1\}) = \emptyset, \text{ de } (R^2)^{-1}(H) = \{1\}.$$



15.3. ábra. Ellenpélda

Megjegyzés:  $R^{-1}(R^{-1}(H)) \subseteq (R^2)^{-1}(H)$  azonban teljesül (ld. 12. feladat).

18.  $Q$  egy páros természetes számhoz a prímosztóit rendeli, így  $Q^{(-1)}$  egy prímszámhoz annak páros többszöröseit rendeli:

$$Q^{(-1)} = \{(b, a) \in \mathbb{N} \times \mathbb{N} \mid 2|a \text{ és } b|a \text{ és } b \text{ prím}\}.$$

a)  $P$  egy természetes számhoz a valódi osztóit rendeli.

–  $Q \circ P$  egy páros összetett számhoz a páros valódi osztóinak prímosztóit rendeli:

$$Q \circ P = \{(a, c) \in \mathbb{N} \times \mathbb{N} \mid \exists b \in \mathbb{N} : b|a \text{ és } b \neq 1 \text{ és } b \neq a \text{ és } 2|b \text{ és } c|b \text{ és } c \text{ prím}\}.$$

–  $a \in \mathcal{D}_{Q \circ P}$ -hez kell, hogy  $a$ -nak legyen valódi osztója, továbbá  $P(a) \subseteq \mathcal{D}_Q$  miatt az összes valódi osztójának párosnak kell lennie, ezért  $a$ -nak nem lehet páratlan prímtenyezője. Így:

$$Q \circ P = \{(2^k, 2) \mid k \in \mathbb{N} \text{ és } k \geq 2\}.$$

b)  $P$  egy természetes számhoz az osztóit rendeli.

–  $Q \circ P = \{(a, c) \in \mathbb{N} \times \mathbb{N} \mid \exists b \in \mathbb{N} : b|a \text{ és } 2|b \text{ és } c|b \text{ és } c \text{ prím}\} = \{(a, c) \in \mathbb{N} \times \mathbb{N} \mid 2|a \text{ és } c|a \text{ és } c \text{ prím}\} = Q.$

- $a \in \mathcal{D}_{Q \odot P}$ -hez  $P(a) \subseteq \mathcal{D}_Q$ -nak teljesülnie kell, vagyis  $a$  összes osztójának párosnak kellene lennie, de mivel az 1 minden természetes számnak osztója, ez nyilván nem teljesülhet. Tehát:

$$Q \odot P = \emptyset.$$

19. a) Definíció szerint igaz.  
 b) Definíció szerint igaz.  
 c) Az a) pont alapján elég belátni, hogy  $\mathcal{D}_{G \circ H} = \mathcal{D}_{G \odot H}$ .  
 $\supseteq$  :  $\mathcal{D}_{G \circ H} \supseteq \mathcal{D}_{G \odot H}$  definíció szerint teljesül.  
 $\subseteq$  :  $a \in \mathcal{D}_{G \circ H} \iff \exists b \in B : (a, b) \in H \text{ és } b \in \mathcal{D}_G$ . Viszont  $|H(a)| = 1$  miatt ilyenkor  $H(a) = \{b\} \subseteq \mathcal{D}_G \implies a \in \mathcal{D}_{G \odot H}$ .  
 d) Definíció szerint igaz.

20. Mindkét kompozíció asszociatív.

- a)  $(F \circ G) \circ H =$   
 $= \{(a, d) \in A \times D \mid \exists b \in B : (a, b) \in H \text{ és } (b, d) \in F \circ G\} =$   
 $= \{(a, d) \in A \times D \mid \exists b \in B : \exists c \in C : (a, b) \in H \text{ és}$   
 $(b, c) \in G \text{ és } (c, d) \in F\} =$   
 $= \{(a, d) \in A \times D \mid \exists c \in C : (a, c) \in G \circ H \text{ és } (c, d) \in F\} =$   
 $= F \circ (G \circ H).$
- b)  $(F \odot G) \odot H =$   
 $= \{(a, d) \in A \times D \mid \exists b \in B : (a, b) \in H \text{ és } (b, d) \in F \odot G \text{ és}$   
 $H(a) \subseteq \mathcal{D}_{F \odot G}\} =$   
 $= \{(a, d) \in A \times D \mid \exists b \in B : \exists c \in C : (a, b) \in H \text{ és } (b, c) \in G \text{ és}$   
 $(c, d) \in F \text{ és } H(a) \subseteq \mathcal{D}_{F \odot G} \text{ és } G(b) \subseteq \mathcal{D}_F\}.$

Továbbá, felhasználva, hogy  $\forall a \in \mathcal{D}_{G \odot H} : G \odot H(a) = G(H(a))$ :

$$\begin{aligned} F \odot (G \odot H) &= \\ &= \{(a, d) \in A \times D \mid \exists c \in C : (a, c) \in G \odot H \text{ és } (c, d) \in F \text{ és} \\ &\quad G \odot H(a) \subseteq \mathcal{D}_F\} = \\ &= \{(a, d) \in A \times D \mid \exists b \in B : \exists c \in C : (a, b) \in H \text{ és } (b, c) \in G \text{ és} \\ &\quad (c, d) \in F \text{ és } H(a) \subseteq \mathcal{D}_G \text{ és } G(H(a)) \subseteq \mathcal{D}_F\}. \end{aligned}$$

Tehát csak azt kell belátnunk, hogy a két megjelölt kifejezés ekvivalens:

$$\begin{aligned} H(a) \subseteq \mathcal{D}_{F \odot G} \text{ és } G(b) \subseteq \mathcal{D}_F &\iff \\ H(a) \subseteq \mathcal{D}_G \text{ és } G(H(a)) \subseteq \mathcal{D}_F \text{ és } G(b) \subseteq \mathcal{D}_F &\iff \\ H(a) \subseteq \mathcal{D}_G \text{ és } G(H(a)) \subseteq \mathcal{D}_F. & \end{aligned}$$

Ugyanis  $(a, b) \in H \implies b \in H(a)$ , tehát a  $G(H(a)) \subseteq \mathcal{D}_F$  feltételből  $G(b) \subseteq \mathcal{D}_F$  következik.

21. a) Egyik irányú következtetés sem teljesül:

$\nRightarrow$ : Legyen  $A = \{1, 2\}$ ,  $Q = \{(1, 1)\}$ ,  $R = \{(1, 1), (1, 2), (2, 1), (2, 2)\}$ ,  
 $S = \{(1, 1), (2, 2)\}$ . Ekkor ugyanis  $Q^{(-1)} = \{(1, 1)\}$ ,  $\widehat{R} = \emptyset$  és  
 $\widehat{S} = \{(1, 2), (2, 1)\}$ , tehát:

$$Q \circ R = \emptyset \subseteq S, \text{ de } Q^{(-1)} \circ \widehat{S} = \{(2, 1)\} \not\subseteq \widehat{R}.$$

$\Leftarrow$ : Legyen  $A = \{1, 2\}$ ,  $Q = \{(1, 1)\}$ ,  $R = \{(1, 1)\}$ ,  $S = \emptyset$ .  
Ekkor ugyanis  $Q^{(-1)} = \{(1, 1)\}$ ,  $\widehat{R} = \{(1, 2), (2, 1), (2, 2)\}$  és  
 $\widehat{S} = \{(1, 1), (1, 2), (2, 1), (2, 2)\}$ , tehát:

$$Q \circ R = \{(1, 1)\} \not\subseteq S, \text{ de } Q^{(-1)} \circ \widehat{S} = \emptyset \subseteq \widehat{R}.$$

b) Nem-szigorú kompozíció esetén az állítás igaz, mindkét irányú következtetés teljesül:

$\implies$ : Tegyük fel, hogy  $Q \circ R \subseteq S$ , és legyen  $(x, z) \in Q^{(-1)} \circ \widehat{S}$ . Ekkor:

$$\exists y \in A : (x, y) \in \widehat{S} \text{ és } (y, z) \in Q^{(-1)} \iff$$

$$\exists y \in A : (1) (x, y) \notin S \text{ és } (2) (z, y) \in Q.$$

Ezek után tegyük fel indirekt módon, hogy  $(x, z) \notin \widehat{R} \implies (x, z) \in R$  és (2)  $(z, y) \in Q$  miatt  $(x, y) \in Q \circ R$ , amiből a feltétel miatt következik, hogy  $(x, y) \in S$ , ami ellentmond az (1) állításnak. Tehát:

$$Q \circ R \subseteq S \implies Q^{(-1)} \circ \widehat{S} \subseteq \widehat{R}.$$

$\Leftarrow$ : Tegyük fel, hogy  $Q^{(-1)} \circ \widehat{S} \subseteq \widehat{R}$ , és legyen  $(x, z) \in Q \circ R$ . Ekkor  $\exists y \in A : (1) (x, y) \in R$  és (2)  $(y, z) \in Q$ . Ezek után tegyük fel indirekt módon, hogy  $(x, z) \notin S \implies (x, z) \in \widehat{S}$  és (2)  $(y, z) \in Q$  miatt  $(z, y) \in Q^{(-1)} \implies (x, y) \in Q^{(-1)} \circ \widehat{S}$ , amiből a feltétel miatt következik, hogy  $(x, y) \in \widehat{R}$ , ami ellentmond az (1) állításnak. Tehát:

$$Q \circ R \subseteq S \Leftarrow Q^{(-1)} \circ \widehat{S} \subseteq \widehat{R}.$$

22. a) Igaz, ugyanis:

$$(a, c) \in R \circ Q \iff$$

$$\exists b \in B : (a, b) \in Q \text{ és } (b, c) \in R \text{ és } Q(a) \subseteq \mathcal{D}_R \implies$$

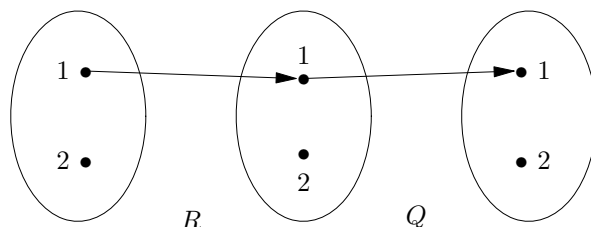
$$\exists b \in B : (a, b) \in Q \text{ és } (b, c) \in S \text{ és } Q(a) \subseteq \mathcal{D}_S \iff$$

$$(a, c) \in S \circ Q.$$

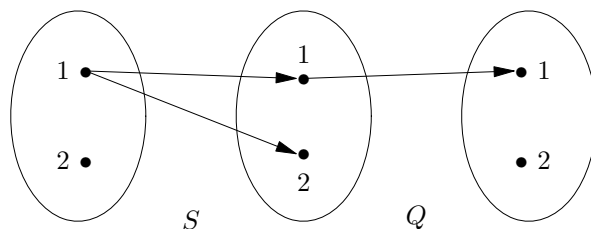


Tehát  $R \circ Q \subseteq S \circ Q$ .

- b) Nem igaz, ugyanis például a 15.4. és a 15.5. ábrán  $Q \circ R = \{(1, 1)\}$ , de  $Q \circ S = \emptyset$ .



15.4. ábra.  $Q \circ R$



15.5. ábra.  $Q \circ S$

23. a)  $R = \{(a, b) \in \mathbb{N} \times \mathbb{N} \mid b = a \text{ vagy } b = 2a\}$ ,  $\mathcal{D}_R = \mathbb{N}$ .  
 $Q = \{(a, b) \in \mathbb{N} \times \mathbb{N} \mid 2|a \text{ és } b = \frac{a}{2}\}$ ,  $\mathcal{D}_Q = \{a \in \mathbb{N} \mid 2|a\}$ .
- b)  $R^k = \{(a, b) \in \mathbb{N} \times \mathbb{N} \mid \exists l \in \mathbb{N}_0 : b = 2^l \cdot a \text{ és } l \leq k\}$ ,  $\mathcal{D}_{R^k} = \mathbb{N}$ .
- c)  $Q \circ R = \{(a, b) \in \mathbb{N} \times \mathbb{N} \mid (2 \nmid a \text{ és } b = a) \text{ vagy } (2|a \text{ és } b \in \{a, \frac{a}{2}\})\} =$   
 $= \{(a, b) \in \mathbb{N} \times \mathbb{N} \mid b = a \text{ vagy } b = \frac{a}{2}\}$ ,  
 $\mathcal{D}_{Q \circ R} = \mathbb{N}$ .
- d)  $Q \circ R = \{(a, b) \in \mathbb{N} \times \mathbb{N} \mid 2|a \text{ és } (b = a \text{ vagy } b = \frac{a}{2})\}$ ,  
 $\mathcal{D}_{Q \circ R} = \{a \in \mathbb{N} \mid 2|a\}$ .
24. a) Igaz, ugyanis:

$$\begin{aligned} \mathcal{D}_{G \circ F} &= \{a \in A \mid \exists b \in B : (a, b) \in F \text{ és } b \in \mathcal{D}_G \text{ és } F(a) \subseteq \mathcal{D}_G\} = \\ &= \{a \in \mathcal{D}_F \mid F(a) \subseteq \mathcal{D}_G\} = F^{-1}(\mathcal{D}_G). \end{aligned}$$

b) Nem igaz, ugyanis:

$$\begin{aligned} \mathcal{D}_{G \circ F} &= \{a \in A \mid \exists b \in B : (a, b) \in F \text{ és } b \in \mathcal{D}_G\} = \\ &= \{a \in \mathcal{D}_F \mid F(a) \cap \mathcal{D}_G \neq \emptyset\} = F^{(-1)}(\mathcal{D}_G). \end{aligned}$$

25.  $\mathcal{D}_P = \{a \in \mathbb{N}_0 \mid a \neq 1 \text{ és } a \text{ nem prím}\}$

Mivel a nullának minden természetes szám osztója,  $P$  a nullához az összes egynél nagyobb természetes számot hozzárendeli.

a)  $\mathcal{D}_{\overline{P}} = \mathbb{N}_0$ ,  $\overline{P} = \{(1, 1)\} \cup \{(a, b) \in \mathbb{N}_0 \times \mathbb{N}_0 \mid b|a \text{ és } b \text{ prím}\}$ .

b)  $\mathcal{D}_{\overline{\overline{P}}} = \mathcal{D}_{\overline{P}} \setminus \{0\} = \mathbb{N}$ , tehát:

$$\overline{\overline{P}} = \overline{P} \setminus \{(0, b) \mid b \in \mathbb{N}_0\} = \{(1, 1)\} \cup \{(a, b) \in \mathbb{N} \times \mathbb{N} \mid b|a \text{ és } b \text{ prím}\}.$$

26.  $R|\pi = \{(1, 1), (2, 2)\} \cup \{(2^k, 2^l) \mid k, l \in \mathbb{N} \text{ és } l < k\}$ ,

$$\overline{R|\pi} = \overline{R|\pi} = id_{\mathbb{N} \setminus \{\pi\}}.$$

27. Legyen  $R \subseteq A \times A$  olyan, hogy  $\mathcal{D}_R = A$ , ekkor ugyanis  $\overline{R} = \overline{\overline{R}} = \emptyset$ , és legyen  $\pi = Hamis$ , így  $R|\pi = \emptyset \implies \overline{R|\pi} = id_A \implies \mathcal{D}_{\overline{R|\pi}} = \mathcal{D}_R = A$ .

*Megjegyzés:* Ha  $R = id_A$ , akkor nem csak az értelmezési tartományok egyeznek meg, hanem  $\overline{R|\pi} = R$  is teljesül (ld. 1.10. példa).

28.  $R \subseteq A \times A$ ,  $\mathcal{D}_R = R^{-1}(\mathcal{D}_R) \iff \forall a \in \mathcal{D}_R : R(a) \subseteq \mathcal{D}_R$ , tehát a  $\mathcal{D}_R$ -beli pontokból indulva biztosan nem jutunk ki  $\mathcal{D}_R$ -ből. Így:

$$\mathcal{D}_{\overline{R}} = \mathcal{D}_{\overline{\overline{R}}} = A \setminus \mathcal{D}_R, \quad \overline{R} = \overline{\overline{R}} = id_{A \setminus \mathcal{D}_R}.$$

29.  $\mathcal{D}_{\overline{R}} = \mathbb{N}_0$ ,

$$\overline{R}(a) = \begin{cases} \{0\}, & \text{ha } a = 3k \text{ vagy } a = 3k + 1 \text{ } (k \in \mathbb{N}_0); \\ \{2\}, & \text{ha } a = 3k + 2 \text{ } (k \in \mathbb{N}_0). \end{cases}$$

$$\mathcal{D}_{\overline{\overline{R}}} = \mathbb{N}_0 \setminus \{3k + 1 \mid k \in \mathbb{N}_0\}, \quad \forall a \in \mathcal{D}_{\overline{\overline{R}}} : \overline{\overline{R}}(a) = \overline{R}(a).$$

30. a)  $[P_q]$  minden eleme összetett szám  $\implies [P_q] \subseteq \mathcal{D}_R \implies R|P_q = R|_{[P_q]}$ . Továbbá mivel  $q$  összetett szám,  $q$  egy pozitív egész kitevős hatványának legnagyobb valódi osztója nem lehet  $q$  hatványa, vagyis  $\forall a \in \mathcal{D}_{R|P_q} : R|P_q(a) = R(a) \notin \mathcal{D}_{R|P_q}$ . Tehát  $\mathcal{D}_{\overline{R|P_q}} = \mathbb{N}$ .

- b) Ha  $q$  prímszám, akkor  $k > 1$  esetén  $R(q^k) = \{q^{k-1}\}$ , de  $R(q) = \emptyset$ , mivel  $q$ -nak nincs valódi osztója. Vagyis:

$$R|P_q(a) = \begin{cases} \{q^{k-1}\}, & \text{ha } a = q^k \text{ és } k > 1; \\ \{q\}, & \text{ha } a = q. \end{cases}$$

$$\text{Tehát } \mathcal{D}_{\overline{R|P_q}} = \mathbb{N} \setminus [P_q].$$

$$31. \overline{R} = \overline{\overline{R}} = \{(3, 3), (5, 5)\}.$$

32.

$$R|\pi(a) = \begin{cases} \{a - 3\}, & \text{ha } a > 2 \text{ és páratlan;} \\ \{3k \mid k \in \mathbb{N}\}, & \text{ha } a = 1. \end{cases}$$

$$\overline{R|\pi}(a) = \overline{\overline{R|\pi}(a)} = \begin{cases} \{6k \mid k \in \mathbb{N}_0\}, & \text{ha } a = 1; \\ \{a - 3\}, & \text{ha } a > 2 \text{ és páratlan;} \\ \{a\}, & \text{ha } a \text{ páros.} \end{cases}$$

33. a) Definíció szerint igaz.  
 b) Definíció szerint igaz.  
 c) Belátható, hogy igaz, de a bizonyítás nem egyszerű.  
 d) Belátható, hogy igaz, de a bizonyítás nem egyszerű.

34.  $\forall x \in [\pi] : R|\pi(x) = \{x - 1\}$ , ahol  $x - 1 \notin [\pi]$ . Így  $\mathcal{D}_{\overline{R|\pi}} = \mathcal{D}_{\overline{\overline{R|\pi}}} = \mathbb{N}_0$ .

$$\overline{R|\pi}(a) = \overline{\overline{R|\pi}(a)} = \begin{cases} \{a\}, & \text{ha } a \text{ páratlan, vagy } a = 0; \\ \{a - 1\}, & \text{ha } a > 0 \text{ és páros.} \end{cases}$$

*Megjegyzés:* Vegyük észre, hogy a feladatban szereplő  $R \subseteq \mathbb{N}_0 \times \mathbb{N}_0$  kikötés miatt  $R(0) \neq \{-1\}$ , hanem  $R(0) = \emptyset$ , vagyis  $0 \notin \mathcal{D}_R$ . Fontos tehát, hogy egy reláció megadásához mindig hozzátartozik az is, hogy milyen direktszorzat részhalmaza.

35. Jelölje  $\alpha$  az  $m$  hosszúságú és  $\beta$  az  $n$  hosszúságú sorozatot.

- a)  $\text{kon}(\text{red}(\alpha), \text{red}(\beta))$  legalább 2, legfeljebb  $n + m$  hosszúságú. Kivéve ha valamelyik vagy mindkét sorozat üres, ugyanis ilyenkor nyilván lehet 1, illetve 0 hosszúságú is.

- b)  $red(kon(\alpha, \beta))$  legalább 1, legfeljebb  $n+m$  hosszúságú. Kivéve ha mindkét sorozat üres, ugyanis ilyenkor nyilván 0 hosszúságú.

36. Definíció szerint igaz.

37. Nem igaz, ugyanis például egy csupa különböző elemből álló sorozat vetülete állhat akár csupa azonos elemből is.

Legyen például  $A = N_1 \times N_2$ , ahol  $N_i = \mathbb{N}$  ( $i = 1, 2$ ) és

$$\alpha = \langle (1, 1), (1, 2), (1, 3) \rangle.$$

Ekkor ugyanis  $red(\alpha) = \alpha$ , viszont:

$$pr_{N_1}(\alpha) = \langle 1, 1, 1 \rangle \implies red(pr_{N_1}(\alpha)) = \langle 1 \rangle.$$

38. a)  $pr_B(\alpha) = \langle (1, 1), (1, 1), (1, 1), (4, 1), (4, 5), (4, 5), (4, 5), (14, 5), (14, 19), (14, 19), (14, 19), \dots \rangle$   
 b)  $red(pr_b(\alpha)) = \langle (1, 1), (4, 1), (4, 5), (14, 5), (14, 19), \dots \rangle$

## 15.2. A programozás alapfogalmai

1. a)  $p(S) = \{(\Phi, \Omega), (\Theta, \Theta), (\Theta, \Omega), (\Theta, \Phi), (\Gamma, \Psi), (\Gamma, \Omega)\}$ .  
 b)  $S$  megoldja  $F$ -et, ugyanis teljesülnek a megoldás feltételei:  
 –  $\mathcal{D}_F = \{\Phi, \Theta\} \subseteq \{\Phi, \Theta, \Gamma\} = \mathcal{D}_{p(S)}$ .  
 –  $p(S)(\Phi) = \{\Omega\} \subseteq \{\Omega, \Psi\} = F(\Phi)$ ,  
 $p(S)(\Theta) = \{\Theta, \Omega, \Phi\} \subseteq \{\Theta, \Omega, \Phi\} = F(\Theta)$ .

2. Egyik állítás sem teljesül.

- a) c) d) Nem igaz, ugyanis a megoldás definíciójában szereplő  $\forall a \in \mathcal{D}_F : p(S)(a) \subseteq F(a)$  feltétel megengedi, hogy  $p(S)$  egy adott  $a \in \mathcal{D}_F$  ponthoz *kevesebb* pontot rendeljen, mint  $F$ . Tehát attól, hogy  $F$  nem determinisztikus,  $S$  és  $p(S)$  még lehet az.

Legyen például  $A = \{1, 2\}$ , és

$$F = \{(1, 1), (1, 2)\},$$

$$S = \{(1, \langle 1 \rangle), (2, \langle 2 \rangle)\}.$$

- b) e) Nem igaz, ugyanis a megoldás feltételei semmit sem követelnek meg a programnak a  $\mathcal{D}_F$ -en kívüli pontokban való viselkedéséről. Azokban a pontokban lehet  $S$  és  $p(S)$  nondeterminisztikus akkor is, ha  $F$  minden pontban determinisztikus.

Legyen például  $A = \{1, 2\}$ , és

$$\begin{aligned} F &= \{(1, 1)\}, \\ S &= \{(1, \langle 1 \rangle), (2, \langle 2 \rangle), (2, \langle 2, 1 \rangle)\}. \end{aligned}$$

- f) Ellenpélda:

$$S = \{(1, \langle 1 \rangle), (1, \langle 1, 2, 1 \rangle), (2, \langle 2 \rangle), (2, \langle 2, 2, \dots \rangle)\}$$

3. Igaz, ugyanis  $\mathcal{D}_S = A$  miatt:

$$\mathcal{D}_{p(S)} = \{a \in \mathcal{D}_S \mid S(a) \subseteq A^*\} = S^{-1}(A^*).$$

4. A megoldás két feltételének teljesülését kell belátnunk:

$$\begin{aligned} q \in \mathcal{D}_F \implies S(q) \subseteq A^* &\iff \mathcal{D}_F \subseteq \mathcal{D}_{p(S)}, \\ q \in \mathcal{D}_F \implies p(S)(q) \subseteq F(q) &\iff \forall q \in \mathcal{D}_F : p(S)(q) \subseteq F(q). \end{aligned}$$

Tehát a feladatban megfogalmazott feltétel nem csupán elégséges, hanem a megoldás definíciójának egy ekvivalens átfogalmazása: egy  $a \in \mathcal{D}_F$  pontban pontosan azt követeljük meg, hogy egyrészt legyen ott értelmezve a program programfüggvénye:  $S(a) \subseteq A^*$ , másrészt pedig, hogy  $p(S)(a) \subseteq F(a)$  teljesüljön.

5. Nem azonosak.  $\mathcal{D}_{F_1} = \mathcal{D}_{F_2}$ , de  $F_2 \subset F_1$ , tehát  $F_2$  szigorúbb, mint  $F_1$ . Ugyanis  $F_2$ -ben azt is megköttjük, hogy az első komponens ne változzon.

Például:  $((4, 1), (2, 2)) \in F_1$ , de  $((4, 1), (2, 2)) \notin F_2$ .

6. Igaz, ugyanis a 2.3. példa alapján beláthatjuk a megoldás két feltételének teljesülését:

- i. Mivel  $S_1$  és  $S_2$  is megoldja  $F$ -et,  $\mathcal{D}_F \subseteq \mathcal{D}_{p(S_1)}$  és  $\mathcal{D}_F \subseteq \mathcal{D}_{p(S_2)}$ , tehát:

$$\mathcal{D}_F \subseteq \mathcal{D}_{p(S_1)} \cap \mathcal{D}_{p(S_2)} = \mathcal{D}_{p(S_1 \cup S_2)}.$$

- ii.  $\forall a \in \mathcal{D}_F : p(S_1)(a) \subseteq F(a)$  és  $p(S_2)(a) \subseteq F(a)$ , tehát:

$$p(S_1 \cup S_2)(a) = p(S_1)(a) \cup p(S_2)(a) \subseteq F(a).$$

7. Legyen  $X = \{1, 2, 3, 4, 5, 6, 7, 8\}$ ,  $H = X \times X$ , az állapottér pedig  $A = H \times H \times H$ , ahol mindegyik  $H$  komponens egy-egy bábú helyzetét határozza meg a sakktáblán.

Ezek után az  $F$  reláció megadható az alábbi formában:

$$F = \{((a_1, a_2, a), (b_1, b_2, b)) \in A \times A \mid b_1 = a_1 \text{ és } b_2 = a_2 \text{ és } \text{ütésben}(a_1, a_2, b)\},$$

ahol az  $\text{ütésben}(a_1, a_2, b)$  feltétellel azt adjuk meg, hogy a  $b$  pozíció az  $a_1$  és  $a_2$  pozíción lévő bástya ütésében álljon:

$$\begin{aligned} \text{ütésben}(a_1, a_2, b) &= \text{ütésben}((x_1, y_1), (x_2, y_2), (x, y)) = \\ &= \begin{cases} x = x_1 \text{ és } \min\{y_1, y_2\} < y < \max\{y_1, y_2\}, & \text{ha } x_1 = x_2; \\ y = y_1 \text{ és } \min\{x_1, x_2\} < x < \max\{x_1, x_2\}, & \text{ha } y_1 = y_2; \\ (x = x_1 \text{ és } y = y_2) \text{ vagy } (x = x_2 \text{ és } y = y_1) & \text{egyébként.} \end{cases} \end{aligned}$$

*Megjegyzés:* Vegyük észre, hogy ha a megadott  $b$  pozíció megegyezik valamelyik bástya pozíciójával ( $b = a_1$  vagy  $b = a_2$ ), illetve ha a két bástya pozíciója megegyezik ( $a_1 = a_2$ ), akkor az  $\text{ütésben}(a_1, a_2, b)$  feltétel nem teljesül.

8. Igaz, ugyanis egy  $a \in A$  pont esetén:

$$\begin{aligned} S(a) \not\subseteq A^* &\iff a \notin \mathcal{D}_{p(S)} \implies a \notin \mathcal{D}_F, \\ p(S)(a) \not\subseteq F(a) &\implies a \notin \mathcal{D}_F. \end{aligned}$$

Mivel a megoldás definíciója szerint  $a \in \mathcal{D}_F$  esetén  $a \in \mathcal{D}_{p(S)}$  és  $p(S)(a) \subseteq F(a)$  is teljesül.

9. a) Nem igaz. Legyen ugyanis  $A = \{1, 2\}$ ,  $F = \{(1, 1)\}$ ,  $S$  pedig olyan program, amelyre  $p(S) = \{(1, 1), (1, 2)\}$ . Ekkor  $\mathcal{D}_{FP} = \mathcal{D}_F = \{1\}$ , viszont  $S$  nyilván nem oldja meg  $F$ -et.

- b) Igaz. Ha  $S$  megoldja  $F$ -et, akkor  $\mathcal{D}_{FP} = \mathcal{D}_F$ , ugyanis:

$$\subseteq : FP = F \cap p(S) \implies \mathcal{D}_{FP} \subseteq \mathcal{D}_F.$$

$$\supseteq : \text{Legyen } a \in \mathcal{D}_F. \text{ Ekkor mivel } S \text{ megoldja } F\text{-et, } a \in \mathcal{D}_{p(S)} \text{ és } p(S)(a) \subseteq F(a), \text{ tehát (1) } \exists(a, b) \in p(S), \text{ amelyre } p(S)(a) \subseteq F(a) \text{ miatt nyilván (2) } (a, b) \in F \text{ is teljesül. Így (1) és (2) miatt } (a, b) \in F \cap p(S) = FP, \text{ tehát } a \in \mathcal{D}_{FP}.$$

### 15.3. Specifikáció

$$1. p(S) = \{(2, 1), (2, 4), (4, 1), (4, 2), (4, 4), (5, 4)\} \implies \lceil lf(S, R) \rceil = \emptyset.$$

$$2. \lceil lf(S, Igaz) \rceil = \{a \in \mathcal{D}_{p(S)} \mid p(S)(a) \subseteq \lceil Igaz \rceil = A\} = \mathcal{D}_{p(S)}, \text{ tehát:}$$

$$lf(S, Igaz) = \mathcal{P}(\mathcal{D}_{p(S)}).$$

3. Állítás:

$$(1) (\exists n \in \mathbb{N} : lf(S, Q_n)) \Rightarrow lf(S, (\exists n \in \mathbb{N} : Q_n)),$$

viszont:

$$(2) (\exists n \in \mathbb{N} : lf(S, Q_n)) \not\Leftarrow lf(S, (\exists n \in \mathbb{N} : Q_n)).$$

*Bizonyítás:*

Először is értelmeznünk kell a feladatban szereplő két logikai függvényt, amit könnyebbé tehet, ha a  $(\exists n \in \mathbb{N} : \dots)$  kifejezéseket  $(\bigvee_{n \in \mathbb{N}} \dots)$  alakban írjuk fel:

$$- \lceil \exists n \in \mathbb{N} : lf(S, Q_n) \rceil = \lceil \bigvee_{n \in \mathbb{N}} lf(S, Q_n) \rceil = \bigcup_{n \in \mathbb{N}} \lceil lf(S, Q_n) \rceil.$$

Tehát  $(\exists n \in \mathbb{N} : lf(S, Q_n))$  igazsághalmazában olyan  $a$  pontok vannak, amelyekre létezik olyan  $Q_n$ , hogy  $a \in \lceil lf(S, Q_n) \rceil$ , azaz  $a$ -ból  $p(S)$  szerint biztosan  $\lceil Q_n \rceil$ -ba jutunk.

$$- \lceil lf(S, (\exists n \in \mathbb{N} : Q_n)) \rceil = \lceil lf(S, \bigvee_{n \in \mathbb{N}} Q_n) \rceil.$$

Tehát  $lf(S, (\exists n \in \mathbb{N} : Q_n))$  igazsághalmazában olyan  $a$  pontok vannak, amelyekből  $p(S)$  szerint biztosan  $\lceil \bigvee_{n \in \mathbb{N}} Q_n \rceil = \bigcup_{n \in \mathbb{N}} \lceil Q_n \rceil$ -ba jutunk.

Ezek után a fenti két állítást bizonyíthatjuk a következőképpen:

(1): Legyen  $a \in \lceil \exists n \in \mathbb{N} : lf(S, Q_n) \rceil$ , ekkor a fentiek alapján  $\exists m \in \mathbb{N} : a \in \lceil lf(S, Q_m) \rceil$ . Mivel  $Q_m \Rightarrow \bigvee_{n \in \mathbb{N}} Q_n$  triviálisan teljesül, az  $lf$  monotonitása miatt  $lf(S, Q_m) \Rightarrow lf(S, \bigvee_{n \in \mathbb{N}} Q_n)$ , tehát  $a \in \lceil lf(S, Q_m) \rceil \implies a \in \lceil lf(S, \bigvee_{n \in \mathbb{N}} Q_n) \rceil = \lceil lf(S, (\exists n \in \mathbb{N} : Q_n)) \rceil$ . Így:

$$(\exists n \in \mathbb{N} : lf(S, Q_n)) \Rightarrow lf(S, (\exists n \in \mathbb{N} : Q_n)).$$

*Megjegyzés:* A bizonyítás során nem használtuk ki a  $\forall i \in \mathbb{N} : Q_i \Rightarrow Q_{i+1}$  feltételt, így az (1) állítás tetszőleges  $Q_i : A \rightarrow \mathbb{L}$  ( $i \in \mathbb{N}$ ) függvényekre teljesül. Ezért tekinthető az  $lf$  tulajdonságai között kimondott (4)  $lf(S, Q) \vee lf(S, R) \Rightarrow lf(S, Q \vee R)$  állítás megszámlálhatóan sok

függvényre vonatkozó általánosításának. (Véges sok függvényre az eredeti állítás alapján nyilvánvalóan teljesül.) Ezek alapján kimondhatjuk, hogy tetszőleges  $I$  véges vagy megszámlálható halmaz és  $Q_i : A \rightarrow \mathbb{L}$  ( $i \in I$ ) függvények esetén:

$$\bigvee_{i \in I} lf(S, Q_i) \Rightarrow lf(S, \bigvee_{i \in I} Q_i).$$

(2) : Legyen  $A = \mathbb{N}$ ,  $[Q_i] = \{x \in \mathbb{N} \mid x \leq i\}$  ( $i \in \mathbb{N}$ ), tehát  $\forall i \in \mathbb{N} : Q_i \Rightarrow Q_{i+1}$  nyilván teljesül. Legyen  $S$  olyan program amelyre  $p(S)(1) = \mathbb{N}$ . Ekkor  $1 \in [lf(S, (\exists n \in \mathbb{N} : Q_n))]$ , ugyanis  $1 \in \mathcal{D}_{p(S)}$  és  $p(S)(1) \subseteq [\exists n \in \mathbb{N} : Q_n] = \bigcup_{n \in \mathbb{N}} [Q_n] = \mathbb{N}$ . Viszont  $\nexists n \in \mathbb{N} : 1 \in [lf(S, Q_n)]$ , hiszen minden  $n \in \mathbb{N}$  esetén  $[Q_n]$  véges ( $n$  elemű) halmaz, de  $p(S)(1) = \mathbb{N}$ , így  $p(S)(1) \not\subseteq [Q_n]$ . Tehát:

$$(\exists n \in \mathbb{N} : lf(S, Q_n)) \not\Leftarrow lf(S, (\exists n \in \mathbb{N} : Q_n)).$$

*Megjegyzés:* A feladat egyfajta folytonossági tulajdonságot fogalmaz meg az  $lf$ -re, amely az analízisből ismert átviteli elvvel analóg. Arról van szó ugyanis, hogy ha  $\forall n \in \mathbb{N} : Q_n \Rightarrow Q_{n+1}$  teljesül, akkor  $\bigvee_{n \in \mathbb{N}} Q_n$  tekinthető ezen  $Q_n$  feltételek határértékének, és így a feladatban azt vizsgáljuk, hogy az egyes  $Q_n$ -ekre kapott  $lf(S, Q_n)$  függvényértékek határértéke ( $\bigvee_{n \in \mathbb{N}} lf(S, Q_n)$ ) megegyezik-e a  $Q_n$ -ek határértékében felvett függvényértékkel ( $lf(S, \bigvee_{n \in \mathbb{N}} Q_n)$ ). Rögzített  $S$  esetén ugyanis az  $lf$  egy feltételekhez feltételeket rendelő függvénynek tekinthető.

Azonban a fentiekben bizonyított (2) állítás alapján ez a folytonossági tulajdonság nem teljesül. Fontos megjegyeznünk viszont, hogy ez azon múltott, hogy a modellünkben megengedünk olyan programokat is, amelyek programfüggvénye korlátlanul nemdeterminisztikus, azaz létezik legalább egy olyan pont, amelyhez végtelen sok különböző pontot rendel. Ha ezt nem engednénk meg, illetve ha csak véges sok  $Q_n$  esetén vizsgálnánk, akkor a feladatban szereplő tulajdonság teljesülne.

4. Igaz, ugyanis a 2.3. példa alapján:

$$\begin{aligned} [lf(S_1 \cup S_2, R)] &= \{a \in \mathcal{D}_{p(S_1 \cup S_2)} \mid p(S_1 \cup S_2)(a) \subseteq [R]\} = \\ &= \{a \in \mathcal{D}_{p(S_1)} \cap \mathcal{D}_{p(S_2)} \mid p(S_1)(a) \cup p(S_2)(a) \subseteq [R]\} = \\ &= \{a \in A \mid a \in \mathcal{D}_{p(S_1)} \text{ és } p(S_1)(a) \subseteq [R] \text{ és} \\ &\quad a \in \mathcal{D}_{p(S_2)} \text{ és } p(S_2)(a) \subseteq [R]\} = \\ &= [lf(S_1, R)] \cap [lf(S_2, R)] = [lf(S_1, R) \wedge lf(S_2, R)]. \end{aligned}$$

És mivel feltettük, hogy  $lf(S_1, R) = lf(S_2, R)$ ,

$$lf(S_1 \cup S_2, R) = lf(S_1, R) \wedge lf(S_2, R) = lf(S_1, R) \vee lf(S_2, R).$$

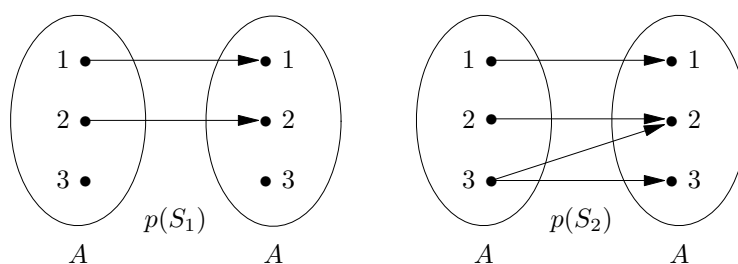


5. Nem igaz, ugyanis a feladatban szereplő feltétel azzal ekvivalens, hogy:

$$\forall y \in A : p(S_1)^{-1}(\{y\}) = p(S_2)^{-1}(\{y\}).$$

Tehát  $p(S_1)$  és  $p(S_2)$  relációk szerint csak a pontonként vett ősképekről tudjuk, hogy megegyeznek, amiből nem következik, hogy  $\mathcal{D}_{p(S_1)} = \mathcal{D}_{p(S_2)}$ . Tekintsük például a 15.6. ábrát.

*Megjegyzés:* Ha bármely halmaz ősképe megegyezik a két programfüggvény szerint, akkor a programok ekvivalensek (ld. következő feladat).



15.6. ábra. Ellenpélda

6. Igaz.

–  $\mathcal{D}_{p(S_1)} = \mathcal{D}_{p(S_2)}$ , ugyanis  $H = \text{Igaz}$  választással:

$$\mathcal{D}_{p(S_1)} = \lceil lf(S_1, \text{Igaz}) \rceil = \lceil lf(S_2, \text{Igaz}) \rceil = \mathcal{D}_{p(S_2)}.$$

– Minden  $a \in \mathcal{D}_{p(S_1)} = \mathcal{D}_{p(S_2)}$  esetén  $p(S_1)(a) = p(S_2)(a)$ , ugyanis  $\lceil H_1 \rceil = p(S_1)(a)$ , illetve  $\lceil H_2 \rceil = p(S_2)(a)$  választással adódik a kölcsönös tartalmazás:

$$a \in \lceil lf(S_1, H_1) \rceil = \lceil lf(S_2, H_1) \rceil = p(S_2)^{-1}(p(S_1)(a)) \implies p(S_2)(a) \subseteq p(S_1)(a),$$

és hasonlóan:

$$a \in \lceil lf(S_2, H_2) \rceil = \lceil lf(S_1, H_2) \rceil = p(S_1)^{-1}(p(S_2)(a)) \implies p(S_1)(a) \subseteq p(S_2)(a).$$

7.  $\lceil lf(S, H) \rceil = \{x \in \mathbb{N} \mid x \equiv 3 \pmod{4} \text{ vagy } x \equiv 0 \pmod{4}\}$

8. Nem azonosak.  $\mathcal{D}_{F_1} = \mathcal{D}_{F_2}$ , de  $F_2 \subset F_1$ , tehát  $F_2$  szigorúbb, mint  $F_1$ . Ugyanis  $F_2$ -ben azt is megköjtjük, hogy az első két komponens ne változzon.

Például:  $(a, b) = ((2, 1, hamis), (1, 3, igaz))$  esetén  $(a, b) \in F_1$ , de  $(a, b) \notin F_2$ .

9. Legyen  $a \in \mathbb{N}$ . Ekkor egyrészt:

$$F_1((a^2, b)) = \{(a^2, a), (a^2, -a)\} \not\subseteq \{(a^2, a)\} = F_2((a^2, b)),$$

másrészt pedig:

$$F_1((-a^2, b)) = \emptyset \not\supseteq \{(-a^2, -a)\} = F_2((-a^2, b)).$$

Tehát  $F_1 \not\subseteq F_2$  és  $F_2 \not\subseteq F_1$ .

Viszont  $\mathcal{D}_{F_1} \subseteq \mathcal{D}_{F_2}$  és  $\forall a \in \mathcal{D}_{F_1} : F_2(a) \subseteq F_1(a)$ , tehát  $F_2$  szigorúbb, mint  $F_1$ .

*Megjegyzés:* Vegyük észre, hogy a feladatban adott specifikáció a 3.2. állítás értelmében *nem jó specifikáció*, ugyanis a  $Q$  előfeltételből nem tudjuk leolvasni a feladat értelmezési tartományát, ezért nem létezik olyan program, amely teljesítené a specifikáció tételének feltételeit. Ezt úgy kerülhetnénk el, ha a  $Q$  előfeltételben kikötnénk, hogy  $x \geq 0$ . (Ilyen kérdésekkel részletesebben a 11. és 18. feladatok kapcsán foglalkozunk.)

10. Az  $m$  és  $n$  változók egy intervallumot határoznak meg, és  $i \in [m..n]$  esetén  $g(i) = 1$  pontosan akkor teljesül, ha  $i$  az  $f$  függvény egy maximumhelye az  $[m..n]$  intervallumon, egyébként pedig  $g(i) = 0$ . Tehát a feladat az, hogy adjuk meg az  $f$  függvény  $[m..n]$  intervallumon vett maximumhelyeinek számát.

11. A specifikáció tételének kimondásakor is hangsúlyoztuk, hogy a tétel általában nem megfordítható, és a 3.2. állításban megfogalmaztuk a *jó specifikációra* vonatkozó kritériumokat. Az alábbiakban arra mutatunk példát, hogy ilyen feltételekre valóban szükség van, vagyis ha ezek nem teljesülnek, a tétel nem feltétlenül megfordítható.

Legegyszerűbben olyan ellenpéldát adhatunk, ahol létezik olyan  $b \in B$  paraméterterületi pont, amelyre  $\lceil Q_b \rceil \neq \emptyset$ , de  $\lceil R_b \rceil = \emptyset$ . Ekkor ugyanis nem létezik olyan program, amely teljesítené a specifikáció tételének feltételeit, hiszen egy ilyen  $b \in B$  pontra tetszőleges  $S$  program esetén (az  $lf$  (1) tulajdonságát felhasználva):

$$[lf(S, R_b)] = [lf(S, Hamis)] = \emptyset.$$

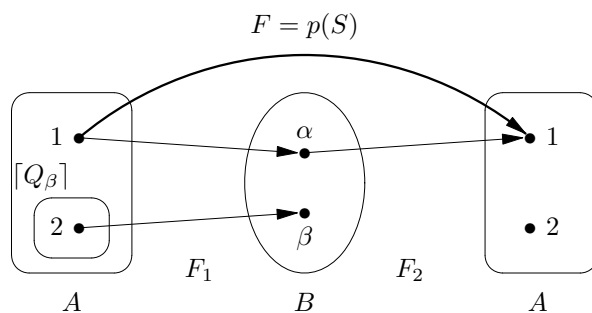
Tehát  $[Q_b] \neq \emptyset$  miatt  $[Q_b] \not\subseteq [lf(S, R_b)]$ , vagyis  $Q_b \Rightarrow lf(S, R_b)$  nem teljesül.

Tekintsük például a 15.7. ábrát. Legyen  $A = \{1, 2\}$ ,  $B = \{\alpha, \beta\}$ ,  $F_1 = \{(1, \alpha), (2, \beta)\}$ ,  $F_2 = \{(\alpha, 1)\}$ ,  $F = \{(1, 1)\}$ . Ekkor nyilván teljesül, hogy:

$$F_1 \subseteq A \times B, F_2 \subseteq B \times A \text{ és } F = F_2 \circ F_1.$$

Legyen  $S$  olyan program, amelyre  $p(S) = F$ . Ekkor  $S$  megoldja  $F$ -et, viszont:

$$[Q_\beta] = \{2\} \not\subseteq \emptyset = [lf(S, R_\beta)], \text{ tehát } Q_\beta \not\Rightarrow lf(S, R_\beta).$$



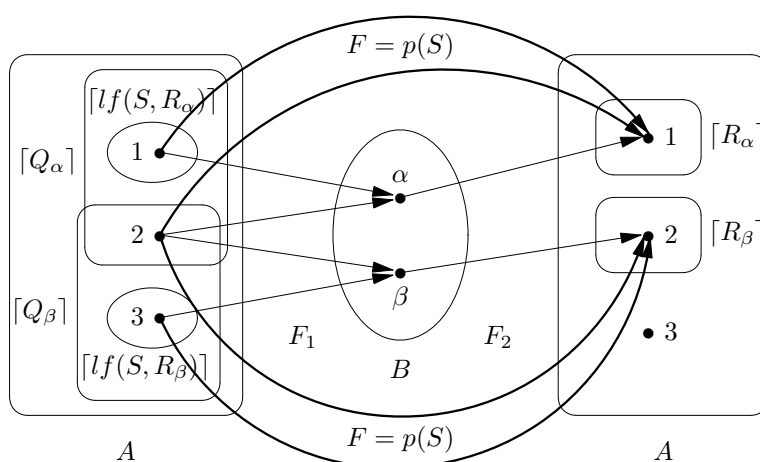
15.7. ábra. Ellenpélda

Általában azonban az ilyen specifikációkat – éppen ezért – kerülni szoktuk, olyan  $F_1$  és  $F_2$  relációkat adunk meg, amelyekre  $\mathcal{R}_{F_1} \subseteq \mathcal{D}_{F_2}$ . Viszont ekkor sem teljesül minden esetben a specifikáció tételének megfordítása, ugyanis a különböző  $b \in B$  pontokhoz tartozó  $[Q_b]$  halmazok nem feltétlenül diszjunktak.

Tekintsük például a 15.8. ábrát. Legyen  $A = \{1, 2, 3\}$ ,  $B = \{\alpha, \beta\}$ ,  $F_1 = \{(1, \alpha), (2, \alpha), (2, \beta), (3, \beta)\}$ ,  $F_2 = \{(\alpha, 1), (\beta, 2)\}$ . Ekkor  $F = F_2 \circ F_1 = \{(1, 1), (2, 1), (2, 2), (3, 2)\}$ , és  $S$  legyen olyan program, amelyre  $p(S) = F$ , tehát  $S$  nyilván megoldja  $F$ -et, viszont:

$$Q_\alpha \not\Rightarrow lf(S, R_\alpha) \text{ és } Q_\beta \not\Rightarrow lf(S, R_\beta).$$

Tehát a paraméterter egyetlen pontjában sem teljesül a specifikáció tételének feltétele, de az  $S$  program megoldja az  $F$  feladatot.



15.8. ábra. Ellenpélda

12. A feladat az, hogy adjuk meg a  $k$  pozitív egészhez legközelebb eső (egyik) prímszámot.  $F$  egy  $(k, q)$  párhoz olyan  $(k, p)$  párt rendel, amelyre  $p$  prímszám, és nem létezik  $k$ -hoz  $p$ -nél közelebb eső prím ( $p$  nem minden esetben egyértelmű). Ezek alapján:

$$F((10, 1)) = \{(10, 11)\} \text{ és } F((9, 5)) = \{(9, 7), (9, 11)\}.$$

13.  $\mathcal{D}_{F_1} = \mathcal{D}_{F_2}$  és  $F_1 \subset F_2$ , tehát  $F_1$  szigorúbb, mint  $F_2$ . Ugyanis  $F_1$  egy  $(a, b, c)$  ponthoz olyan  $(a, b, d)$  pontot rendel, amelyre  $d$  az  $a$  és  $b$  legkisebb közös többszöröse,  $F_2$  pedig olyan  $(a, b, e)$  pontokat, amelyekre  $e$  az  $a$  és  $b$  számoknak egy a szorzatuknál nem nagyobb közös többszöröse.

$$\begin{aligned} \text{Például: } F_1(4, 6, 1) &= \{(4, 6, 12)\}, \\ F_2(4, 6, 1) &= \{(4, 6, 12), (4, 6, 24)\}. \end{aligned}$$

14. Nem azonosak.  $\mathcal{D}_{F_1} = \mathcal{D}_{F_2}$  és  $F_1 \subset F_2$ , tehát  $F_1$  szigorúbb, mint  $F_2$ . Két fontos különbség is van a feladatok között, ugyanis  $F_1$ -ben az  $f$  függvénynek az  $[m..n]$  intervallumon vett első maximumhelyét keressük úgy, hogy az első két változó (az intervallum határai) ne változzon,  $F_2$ -ben pedig az  $[m..n]$  intervallumon vett valamelyik maximumhelyet keressük és az első két komponens is megváltozhat.

*Megjegyzés:* Vegyük észre, hogy a feladatban adott két specifikáció a 3.2. állítás értelmében *nem jó specifikáció*, ugyanis a  $Q$  előfeltételből nem tudjuk leolvasni

a feladat értelmezési tartományát, ezért nem létezik olyan program, amely teljesítené a specifikáció tételének feltételeit. Ezt mindkét esetben úgy kerülhetnénk el, ha a  $Q$  előfeltételben kikötnénk, hogy  $m \leq n$ . (Ilyen kérdésekkel részletesebben a 11. és 18. feladatok kapcsán foglalkozunk.)

- 15.** A feladat szövegesen megfogalmazva az, hogy összegezzük a  $v : \mathbb{N} \rightarrow \{0, 1\}$  függvényt az  $[1..n]$  intervallumon. Vegyük észre, hogy az állapotternek egyetlen komponense van, amelynek értékétől nem függ a feladat, vagyis minden ponthoz ugyanazt rendeli. (Az  $n$  szám ugyanis nem az állapotter egyik komponensében adott, hanem a  $v$  függvényhez hasonlóan rögzített.) Tehát nincsenek olyan komponensek, amelyek paraméterezik a feladatot, ezért most nem az állapotter egy alterét választjuk paraméterternek, hanem egy tetszőleges egyelemű halmazt, és az  $F_1$  reláció minden állapotterbeli ponthoz hozzárendeli ezt az elemet.

$$A = \mathbb{N}$$

$$s$$

$$B = \{\chi\}$$

$$Q : \text{Igaz}$$

$$R : (s = \sum_{k=1}^n v(k))$$

**16.**  $A = \mathbb{N} \times \mathbb{N}$   
 $n \quad d$

$$B = \mathbb{N}$$

$$n'$$

$$Q : (n = n')$$

$$R : (Q \wedge d|n)$$

**17.**  $A = \mathbb{N} \times \mathbb{N}$   
 $n \quad d$

$$B = \mathbb{N}$$

$$n'$$

$$Q : (n = n' \wedge \exists a \in \mathbb{N} : (a|n \wedge a \neq 1 \wedge a \neq n))$$

$$R : (Q \wedge d|n \wedge d \neq 1 \wedge d \neq n)$$

18. A feladat szövege alapján közvetlenül az alábbi specifikáció adódik:

$$\begin{aligned}
 A &= \mathbb{N} \times \mathbb{N} \\
 &\quad n \quad d \\
 B &= \mathbb{N} \\
 &\quad n' \\
 Q &: (n = n') \\
 R &: (Q \wedge d|n \wedge d \neq 1 \wedge d \neq n)
 \end{aligned}$$

Azonban vegyük észre, hogy a fenti specifikációban nem foglalkoztunk azzal, hogy nem minden természetes számnak létezik valódi osztója. Így olyan  $F_1$  és  $F_2$  relációkat kaptunk, amelyekre az  $F = F_2 \circ F_1$  feladat értelmezési tartománya nem egyezik meg  $F_1$  értelmezési tartományával ( $\mathcal{D}_F \subset \mathcal{D}_{F_1}$ ), tehát a  $Q$  előfeltételből nem tudjuk leolvasni a feladat értelmezési tartományát. Vagyis ez a specifikáció nem teljesíti a 3.2. állításban megfogalmazott kritériumokat, *nem jó specifikáció*. Léteznek ugyanis olyan  $b \in B$  paraméterterbeli pontok (a nem összetett számok), amelyekre  $[Q_b] \neq \emptyset$ , de  $[R_b] = \emptyset$ , ezért nem létezik olyan program, amely teljesítené a specifikáció tételének feltételeit. Hiszen minden ilyen  $b \in B$  pontra tetszőleges  $S$  program esetén (az  $lf$  (1) tulajdonságát felhasználva):

$$[lf(S, R_b)] = [lf(S, Hamis)] = \emptyset.$$

Tehát  $[Q_b] \neq \emptyset$  miatt  $Q_b \not\Rightarrow lf(S, R_b)$ . Ezért az ilyen specifikációkat nyilván kerülni fogjuk.

Erre jelen esetben is, mint ahogy általában, két megoldás kínálkozik. Az egyik az, hogy már eleve kikötjük az előfeltételben, hogy csak összetett természetes számokra van értelmezve a feladat. Ekkor pontosan ugyanazt a specifikációt és szövegesen megfogalmazva is ugyanazt a feladatot kapjuk, mint az előző feladatban. Vegyük észre, hogy ez a specifikáció ugyanazt az  $F = F_2 \circ F_1$  relációt határozza meg, mint az elsőként felírt változat (habár  $F_1$  nem ugyanaz), de a specifikáció tételének alkalmazása szempontjából nagyon lényeges különbség van a kettő között.

A másik megoldás, hogy átfogalmazzuk a feladatot úgy, hogy minden természetes számra értelmezhető legyen: döntsük el egy természetes számról, hogy van-e valódi osztója, és ha van, akkor adjunk meg egyet. Ekkor az állapototteret is bővítenünk kell egy logikai komponenssel, és nyilván egy teljesen eltérő specifikációt kapunk, teljesen más programok oldják meg ezt a feladatot, mint az előzőeket.

$$A = \mathbb{N} \times \mathbb{L} \times \mathbb{N}$$

$$\quad \quad \quad n \quad \quad l \quad \quad d$$

$$B = \mathbb{N}$$

$$\quad \quad \quad n'$$

$$Q : (n = n')$$

$$R : (Q \wedge l = (\exists a \in \mathbb{N} : (a|n \wedge a \neq 1 \wedge a \neq n))) \wedge l \rightarrow (d|n \wedge d \neq 1 \wedge d \neq n)$$

$$19. \quad A = \mathbb{N} \times H$$

$$\quad \quad \quad n \quad \quad h$$

$$B = \mathbb{N}$$

$$\quad \quad \quad n'$$

Ahol  $H = \{X \in \wp(\mathbb{N}) \mid |X| < \infty\} = \mathfrak{F}(\mathbb{N})$ , azaz  $\mathbb{N}$  véges részhalmazainak halmaza.  $H$  bevezetése azért szükséges, mert  $\wp(\mathbb{N})$  nem megszámlálható halmaz, ezért definíció szerint nem szerepelhet állapottér komponenseként.

$$Q : (n = n')$$

$$R : (Q \wedge h = \{d \in \mathbb{N} \mid d|n \wedge d \neq 1 \wedge d \neq n\})$$

*Megjegyzés:* Ebben a feladatban nem okoznak gondot a nem összetett számok, ugyanis azoknak is ugyanúgy létezik a valódi osztók halmaza, nevezetesen az üres halmaz.

$$20. \quad A = \mathbb{N} \times \mathbb{N}$$

$$\quad \quad \quad n \quad \quad p$$

$$B = \mathbb{N}$$

$$\quad \quad \quad n'$$

$$Q : (n = n' \wedge n \neq 1)$$

$$R : (Q \wedge p|n \wedge \text{prim}(p) \wedge \forall q \in \mathbb{N} : ((q|n \wedge \text{prim}(q)) \rightarrow q \leq p))$$

A természetes számok közül egyedül az 1-nek nincs prímosztója, ezért azt az előfeltételben kizártuk.

$$21. \quad A = \mathbb{N} \times \mathbb{N}_0$$

$$\quad \quad \quad n \quad \quad s$$

$$B = \mathbb{N}$$

$$\quad \quad \quad n'$$

$$Q : (n = n')$$

$$R : (Q \wedge s = \sum_{k=2}^{n-1} oszto(k, n))$$

Ahol  $oszto : \mathbb{N} \times \mathbb{N} \rightarrow \{0, 1\}$ ,

$$oszto(k, n) = \begin{cases} 1, & \text{ha } k|n; \\ 0 & \text{különben.} \end{cases}$$

*Megjegyzés:* Ebben a feladatban sem okoznak gondot a nem összetett számok, azok valódi osztóinak száma nulla.

- 22.** Az  $[m..n]$  egész intervallumot minden esetben úgy értjük, hogy  $n - m + 1$  eleme van, tehát üres intervallumot pontosan akkor kapunk, ha a végpontja eggyel kisebb a kezdőpontjánál, azaz  $n = m - 1$ . Ezért egy  $[m..n]$  intervallum megadásánál megköjtjük, hogy  $m \leq n + 1$ , illetve ha az intervallum nem lehet üres, akkor  $m \leq n$ .

Mivel nem minden intervallumba esik összetett szám, ezért be kell vezetnünk egy logikai változót, és a feladatot az alábbi pontosabb megfogalmazás szerint kell értelmeznünk: adjuk meg, hogy az  $[m..n]$  intervallumban található-e olyan szám, amelynek van valódi osztója, és ha igen, akkor adjuk meg az első ilyen.

$$A = \mathbb{N} \times \mathbb{N} \times \mathbb{L} \times \mathbb{N}$$

$m \quad n \quad l \quad i$

$$B = \mathbb{N} \times \mathbb{N}$$

$m' \quad n'$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R : (Q \wedge l = (\exists j \in [m..n] : \beta(j)) \wedge l \rightarrow (i \in [m..n] \wedge \beta(i) \wedge \forall j \in [m..i - 1] : \neg \beta(j)))$$

Ahol  $\beta : \mathbb{N} \rightarrow \mathbb{L}$ ,  $[\beta] = \{a \in \mathbb{N} \mid \exists b \in \mathbb{N} : b|a \text{ és } b \neq 1 \text{ és } b \neq a\}$ , vagyis  $\beta(a)$  pontosan azon  $a \in \mathbb{N}$  számokra lesz igaz, amelyeknek van valódi osztója.

*Megjegyzés:* Ez a feladat a 12. fejezetben megfogalmazott *lineáris keresés* egy konkrét esete.

- 23.** Az előző feladathoz hasonlóan itt is szükségünk lesz egy logikai változóra is, és pontosítanunk kell a feladatot, ugyanis nem minden intervallumban van 6-tal nem osztható szám.



$$A = \mathbb{N} \times \mathbb{N} \times \mathbb{L} \times \mathbb{N}$$

$$m \quad n \quad l \quad i$$

$$B = \mathbb{N} \times \mathbb{N}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R : (Q \wedge l = (\exists j \in [m..n] : \neg j) \wedge l \rightarrow (i \in [m..n] \wedge \neg i \wedge \forall j \in [m..n] : (\neg j \rightarrow f(j) \leq f(i))))$$

Ahol az  $f : \mathbb{N} \rightarrow \mathbb{N}_0$  függvény egy természetes szám valódi osztóinak számát adja meg:

$$f(n) = \sum_{k=2}^{n-1} oszto(k, n),$$

ahol  $oszto : \mathbb{N} \times \mathbb{N} \rightarrow \{0, 1\}$ ,

$$oszto(k, n) = \begin{cases} 1, & \text{ha } k|n; \\ 0 & \text{különben.} \end{cases}$$

24. Az előző két feladattal ellentétben most nem engedjük meg az üres intervallumot, kikötjük, hogy  $m \leq n$ .

$$A = \mathbb{N} \times \mathbb{N} \times \mathbb{N}$$

$$m \quad n \quad i$$

$$B = \mathbb{N} \times \mathbb{N}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n)$$

$$R : (Q \wedge i \in [m..n] \wedge \forall j \in [m..n] : f(j) \leq f(i))$$

Ahol  $f$  az előző feladatban definiált függvény.

## 15.4. Kiterjesztés

1.  $F' = \{(q, a), (r, b) \in A \times A \mid r = q + 1\}$

2. Nem igaz. Legyen például  $A = \{1, 2\} \times \{a, b\}$ ,  $B = \{1, 2\}$ ,  $S \subseteq A \times A^{**}$ ,

$$S = \{ ((1, a), \langle (1, a), (2, b) \rangle), ((1, b), \langle (1, b), (2, a) \rangle), \\ ((2, a), \langle (2, a), (1, b) \rangle), ((2, b), \langle (2, b), (1, a) \rangle) \}.$$

Ekkor  $pr_B(S) = \{ (1, \langle 1, 2 \rangle), (2, \langle 2, 1 \rangle) \}$ , tehát:

$$pr_B(S)' = \{ ((1, a), \langle (1, a), (2, a) \rangle), ((1, b), \langle (1, b), (2, b) \rangle), \\ ((2, a), \langle (2, a), (1, a) \rangle), ((2, b), \langle (2, b), (1, b) \rangle) \}.$$

*Megjegyzés:* Vegyük észre, hogy egy program projekciója nem is feltétlenül program (ld. 4. feladat).

3. Legyen  $B$  altere az  $A$  állapotérnek,  $S \subseteq B \times B^{**}$  program,  $S'$  az  $S$  kiterjesztje  $A$ -ra. Ekkor nyilván  $S' \subseteq A \times A^{**}$ , tehát azt kell belátnunk, hogy teljesül a program definíciójában szereplő három feltétel.

- i.  $S'$  definíciójából adódik, hogy  $\forall a \in A : S'(a) \neq \emptyset$ , tehát  $\mathcal{D}_{S'} = A$ .
- ii. Legyen  $\alpha \in \mathcal{R}_{S'} \implies \exists a \in A : (a, \alpha) \in S' \implies (pr_B(a), pr_B(\alpha)) \in S$ . Mivel  $S$  program,  $pr_B(\alpha) = red(pr_B(\alpha))$ , tehát  $\alpha = red(\alpha)$ , ugyanis  $\forall i \in \mathcal{D}_\alpha : pr_{B'}(\alpha_i) = pr_{B'}(a)$  miatt az  $\alpha$  sorozat két tagja pontosan akkor lesz azonos, ha a  $pr_B(\alpha)$  sorozat megfelelő tagjai azonosak.
- iii. Legyen  $a \in A$  és  $\alpha \in S'(a)$ . Ekkor  $(pr_B(a), pr_B(\alpha)) \in S \implies |pr_B(\alpha)| \neq 0$  és  $pr_B(\alpha)_1 = pr_B(a)$ . Tehát  $|\alpha| \neq 0$ ,  $pr_B(\alpha_1) = pr_B(a)$  és  $pr_{B'}(\alpha_1) = pr_{B'}(a)$ , vagyis  $\alpha_1 = a$ .

4. Legyen  $S$  olyan program, amelyre:

$$S((1, 1, \dots, 1)) = \{ \langle (1, 1, 1, \dots, 1), (2, 1, 1, \dots, 1), (2, 2, 1, \dots, 1), \dots, \\ (2, 2, \dots, 2, 1), (2, 2, 2, \dots, 2) \rangle \}.$$

Legyen  $B$  az  $A$  egy valódi altere. Ekkor létezik olyan  $k \in [1..n]$ , amelyre  $A_k$  nem szerepel a  $B$  komponensei között, így  $pr_B(S)$  az  $(1, 1, \dots, 1) \in B$  ponthoz egy olyan véges sorozatot rendel, amelynek a  $k$ . és  $(k+1)$ . tagja azonos, tehát nem lehet redukált.

5. a) Nem igaz. Ha  $F = pr_A(F'')$ , akkor  $F'' \subseteq F'$ .  $F'$  a legbővebb ilyen tulajdonságú reláció, hiszen minden olyan rendezett párt tartalmaz, amelynek vetülete  $F$ -beli.
- b) Mivel a projekció függvény,  $pr_A^{(-1)}(F) = pr_A^{-1}(F)$ .

$$\begin{aligned} pr_A^{(-1)}(F) &= pr_A^{-1}(F) = \{x \in \mathcal{D}_{pr_A} \mid pr_A(x) \in F\} = \\ &= \{(a, b) \in B \times B \mid pr_A((a, b)) \in F\} = F' \end{aligned}$$

*Megjegyzés:* Jelölésben általában nem különböztetjük meg a  $pr_A : B \rightarrow A$  és a  $pr_A : B \times B \rightarrow A \times A$  függvényt, de itt nyilván az utóbbiról van szó, ezért  $\mathcal{D}_{pr_A} = B \times B$ . Továbbá mivel  $pr_A(x)$  egy  $A$ -beli elemet jelöl, nem pedig egy egyelemű halmazt, az őskép definíciójában  $\in$  jelet kell írunk a  $\subseteq$  helyett.

6. A kiterjesztés definíciójából adódik, hogy:

$$\begin{aligned} F' &= \{((a, a_1), (b, b_1)) \in B \times B \mid (a, b) \in F\}, \\ F'' &= \{((a, a_2), (b, b_2)) \in C \times C \mid (a, b) \in F\}, \\ F''' &= \{((a, a_1, a_2), (b, b_1, b_2)) \in D \times D \mid (a, b) \in F\}. \end{aligned}$$

Jelölje  $F^\circ$  az  $F''$  kiterjesztését  $D$ -re. Ekkor  $F^\circ \subseteq D \times D$ ,

$$\begin{aligned} F^\circ &= \{((a, a_1, a_2), (b, b_1, b_2)) \mid pr_C(((a, a_1, a_2), (b, b_1, b_2))) \in F''\} = \\ &= \{((a, a_1, a_2), (b, b_1, b_2)) \mid ((a, a_2), (b, b_2)) \in F''\} = \\ &= \{((a, a_1, a_2), (b, b_1, b_2)) \mid (a, b) \in F\} = F'''. \end{aligned}$$

Tehát beláttuk, hogy  $F'''$  az  $F''$  kiterjesztése  $D$ -re.

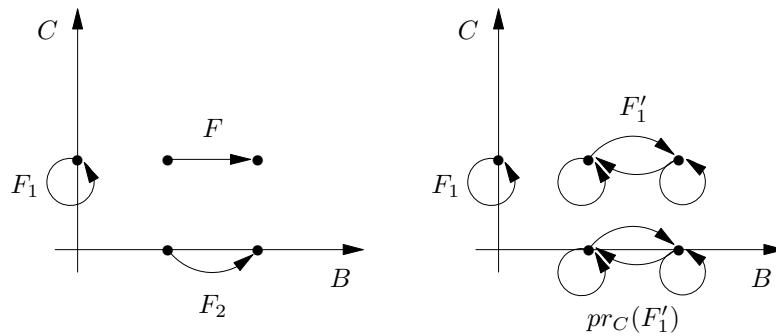
Mivel  $F'$  és  $F''$   $A$ -ra vett projekciója  $F$ , ezért  $F'''$  az  $F'$   $A$ -ra vett projekciójának kiterjesztése  $C$ -re, és hasonlóan  $F'$  az  $F''$   $A$ -ra vett projekciójának kiterjesztése  $B$ -re.

7. Nem igaz. Tekintsük például a 15.9. ábrát. Legyen  $A = B \times C$ ,  $B = \{1, 2\}$ ,  $C = \{\alpha\}$ ,  $F_2 = \{(1, 2)\} \implies F = \{(1, \alpha), (2, \alpha)\} \implies F_1 = \{(\alpha, \alpha)\}$ . Ekkor:

$$\begin{aligned} F'_1 &= \{((1, \alpha), (1, \alpha)), ((1, \alpha), (2, \alpha)), ((2, \alpha), (1, \alpha)), ((2, \alpha), (2, \alpha))\}, \\ pr_C(F'_1) &= \{(1, 1), (1, 2), (2, 1), (2, 2)\} \neq \{(1, 2)\} = F_2. \end{aligned}$$

## 15.5. Programkonstrukciók

1. a)  $(S_1; S_2) = \left\{ \begin{array}{lll} 1 \rightarrow 1463, & 1 \rightarrow 1451 \dots, & 1 \rightarrow 12 \dots, \\ 2 \rightarrow 2132132 \dots, & 3 \rightarrow 361 \dots, & 4 \rightarrow 4636, \\ 4 \rightarrow 45134, & 4 \rightarrow 45121, & 5 \rightarrow 5636, \\ 6 \rightarrow 612132 \dots & & \end{array} \right\}$
- $$p((S_1; S_2)) = \{(4, 1), (4, 4), (4, 6), (5, 6)\}$$



15.9. ábra. Ellenpélda

$$\begin{aligned}
 \text{b) } IF(\pi_1 : S_1, \pi_2 : S_2) &= \\
 &= \left\{ \begin{array}{lll} 1 \rightarrow 14, & 1 \rightarrow 12 \dots, & 2 \rightarrow 2132, \\ 3 \rightarrow 36, & 4 \rightarrow 463, & 4 \rightarrow 451 \end{array} \right\} \cup \\
 &\cup \left\{ \begin{array}{lll} 1 \rightarrow 134, & 1 \rightarrow 121, & 3 \rightarrow 36, \\ 4 \rightarrow 463, & 4 \rightarrow 451 \dots, & 5 \rightarrow 5632 \end{array} \right\} \cup \\
 &\cup \left\{ \begin{array}{l} 6 \rightarrow 6 \dots \end{array} \right\} = \\
 &= \left\{ \begin{array}{lll} 1 \rightarrow 14, & 1 \rightarrow 12 \dots, & 1 \rightarrow 134, \\ 1 \rightarrow 121, & 2 \rightarrow 2132, & 3 \rightarrow 36, \\ 4 \rightarrow 463, & 4 \rightarrow 451, & 4 \rightarrow 451 \dots, \\ 5 \rightarrow 5632, & 6 \rightarrow 6 \dots & \end{array} \right\} \\
 p(IF) &= \{(2, 2), (3, 6), (5, 2)\}
 \end{aligned}$$

- c) Vezessünk be néhány jelölést, hogy könnyebben le tudjuk írni a ciklust. Ha egy sorozat valamely részsorozata fölött egy vonás szerepel, akkor az jelentse azt, hogy ott a megfelelő részsorozat véges sokszori ismétlésével kapott összes sorozat állhat. Tehát  $a \rightarrow \alpha \bar{\beta} \gamma$  az alábbi halmazt jelöli:

$$\{(a, \text{kon}(\alpha, \omega, \gamma)) \mid \exists n \in \mathbb{N} : \omega = \text{kon}(\beta^1, \beta^2, \dots, \beta^n) \text{ és } \forall i \in [1..n] : \beta^i = \beta\}.$$

Ha pedig egy sorozat valamely részsorozata fölött  $\rightarrow$  jel szerepel, akkor az jelentse a megfelelő rész végtelen sokszor történő ismétlésével kapott sorozatot. Tehát:

$$\alpha \overrightarrow{\beta} \dots = \text{kon}(\alpha, \beta, \beta, \beta, \dots).$$

Ezen jelöléseket használva:

$$DO = \left\{ \begin{array}{lll} 1 \rightarrow 14636, & 1 \rightarrow 12\dots, & 1 \rightarrow \overline{1451}4636, \\ 1 \rightarrow \overline{1451}2\dots, & 1 \rightarrow \overline{1451}\dots, & 2 \rightarrow \overline{213}\dots, \\ 3 \rightarrow 36, & 4 \rightarrow \overline{4636}, & 4 \rightarrow \overline{451}4636, \\ 4 \rightarrow \overline{451}2\dots, & 4 \rightarrow \overline{451}\dots & \end{array} \right\} \cup$$

$$\cup \left\{ \begin{array}{ll} 5 \rightarrow 5, & 6 \rightarrow 6 \end{array} \right\},$$

$$p(DO) = \{(3, 6), (5, 5), (6, 6)\}.$$

*Megjegyzés:* Ebben a feladatban a programokban szereplő  $(a, \langle a, b, c, \dots \rangle)$  párokat az egyszerűség kedvéért  $a \rightarrow abc\dots$  alakban adtuk meg.

$$2. IF(\pi_1 : S_1, \pi_2 : S_2, \pi_3 : S_3) =$$

$$= \left\{ \begin{array}{llll} 1 \rightarrow 12\dots, & 1 \rightarrow 12, & 2 \rightarrow 23, & 2 \rightarrow 24, \\ 3 \rightarrow 3456, & 3 \rightarrow 3\dots, & 4 \rightarrow 463, & 4 \rightarrow 43, \\ 4 \rightarrow 432, & 5 \rightarrow 5\dots, & 6 \rightarrow 63\dots & \end{array} \right\},$$

$$\mathcal{D}_{p(IF)} = \{2, 4\}, \quad p(IF) = \{(2, 3), (2, 4), (4, 2), (4, 3)\}.$$

*Megjegyzés:* Az előző feladathoz hasonlóan a programban szereplő  $(a, \langle a, b, c, \dots \rangle)$  párokat itt is  $a \rightarrow abc\dots$  alakban adtuk meg.

3. Nem igaz. Ugyanis egy  $a \in A$  pont esetén  $S_2 \circ \tau \circ S_1(a) = S_2(\tau(S_1(a)))$ , tehát  $\mathcal{R}_{S_2 \circ \tau \circ S_1} \subseteq \mathcal{R}_{S_2}$ , viszont  $\mathcal{R}_{(S_1; S_2)} \not\subseteq \mathcal{R}_{S_2}$ . (Például minden  $\mathcal{R}_{S_1}$ -beli végtelen sorozat definíció szerint benne van  $\mathcal{R}_{(S_1; S_2)}$ -ben is.)

Úgy is gondolkozhatunk, hogy amikor képezzük a  $\tau \circ S_1$  kompozíciót, akkor az  $S_1$  által az állapottér egyes pontjaihoz rendelt sorozatokból csak a véges sorozatok végpontjai maradnak meg, a végtelen sorozatokat és a véges sorozatok többi tagját elveszítjük. Viszont a szekvencia definíciója szerint az  $\mathcal{R}_{S_1}$ -beli sorozatok minden tagjára szükségünk van, tehát  $S_2 \circ \tau \circ S_1$  nem egyezhet meg  $(S_1; S_2)$ -vel.

Legyen például  $A = \{a, b\}$ ,

$$S_1 = \{(a, \langle a, b \rangle), (b, \langle b, b, \dots \rangle)\},$$

$$S_2 = \{(a, \langle a \rangle), (b, \langle b \rangle)\}.$$

Ekkor ugyanis:

$$(S_1; S_2) = \{(a, \langle a, b \rangle), (b, \langle b, b, \dots \rangle)\} = S_1,$$

$$S_2 \circ \tau \circ S_1 = \{(a, \langle b \rangle)\}.$$

Tehát  $S_2 \circ \tau \circ S_1 \neq (S_1; S_2)$ , sőt egyik irányú tartalmazás sem áll fenn.

*Megjegyzés:* Vegyük észre, hogy  $S_2 \circ \tau \circ S_1$  nem is program.

4. a) Igaz, ugyanis felhasználva a definíciókból egyszerűen adódó  $\mathcal{D}_{G \circ F} = F^{-1}(\mathcal{D}_G)$  összefüggést:

$$\mathcal{D}_{p(S)} = \mathcal{D}_{p(S_2) \circ p(S_1)} = p(S_1)^{-1}(\mathcal{D}_{p(S_2)}) = [lf(S_1, \mathcal{P}(\mathcal{D}_{p(S_2)}))].$$

- b) Igaz, ugyanis egyrészt:

$$\begin{aligned} [lf(S_1, lf(S_2, R))] &= \\ &= \{a \in \mathcal{D}_{p(S_1)} \mid p(S_1)(a) \subseteq [lf(S_2, R)]\} = \\ &= \{a \in \mathcal{D}_{p(S_1)} \mid p(S_1)(a) \subseteq \{b \in \mathcal{D}_{p(S_2)} \mid p(S_2)(b) \subseteq [R]\}\} = \\ &= \{a \in \mathcal{D}_{p(S_1)} \mid p(S_1)(a) \subseteq \mathcal{D}_{p(S_2)} \text{ és } p(S_2)(p(S_1)(a)) \subseteq [R]\}. \end{aligned}$$

Másrészt pedig felhasználva, hogy  $\mathcal{D}_{G \circ F} = F^{-1}(\mathcal{D}_G)$  és  $\forall a \in \mathcal{D}_{G \circ F} : G \circ F(a) = G(F(a))$ :

$$\begin{aligned} [lf((S_1; S_2), R)] &= \\ &= \{a \in \mathcal{D}_{p(S_2) \circ p(S_1)} \mid p(S_2) \circ p(S_1)(a) \subseteq [R]\} = \\ &= \{a \in p(S_1)^{-1}(\mathcal{D}_{p(S_2)}) \mid p(S_2)(p(S_1)(a)) \subseteq [R]\} = \\ &= \{a \in \mathcal{D}_{p(S_1)} \mid p(S_1)(a) \subseteq \mathcal{D}_{p(S_2)} \text{ és } p(S_2)(p(S_1)(a)) \subseteq [R]\}. \end{aligned}$$

5. Legyen  $A$  egy tetszőleges állapotter, és a 8. fejezet definícióival összhangban vezessük be a *SKIP* és az *ABORT* jelöléseket két speciális programra:

$$\begin{aligned} SKIP &= \{(a, \langle a \rangle) \in A \times A^{**} \mid a \in A\}, \\ ABORT &= \{(a, \langle a, a, \dots \rangle) \in A \times A^{**} \mid a \in A\}. \end{aligned}$$

A *SKIP* és az *ABORT* program is felírható mindhárom programkonstrukció segítségével. Legyen  $S$  tetszőleges program,  $\pi$  tetszőleges feltétel. Ekkor:

$$\begin{aligned} SKIP &= (SKIP; SKIP) \\ SKIP &= IF(Igaz : SKIP) = IF(\pi : SKIP, \neg\pi : SKIP) \\ SKIP &= DO(Hamis, S) \end{aligned}$$

$$\begin{aligned} ABORT &= (ABORT; S) = (SKIP; ABORT) \\ ABORT &= IF(Igaz : ABORT) = IF(\pi : ABORT, \neg\pi : ABORT) \\ ABORT &= IF(Hamis : S) \\ ABORT &= DO(Igaz, SKIP) \end{aligned}$$

6. Ciklusként nem írható fel minden program, csak szekvenciaként és elágazás-ként. Legyen  $S$  egy tetszőleges állapotterén adott program. Ekkor az előző feladatban bevezetett jelölésekkel:

a)  $S = (SKIP; S) = (S; SKIP)$

b)  $S = IF(Igaz : S) = IF(\pi : S, \neg\pi : S)$  ( $\pi$  tetszőleges feltétel)

c) Legyen például  $A = \{a, b\}$ ,

$$S = \{(a, \langle a \rangle), (a, \langle a, b \rangle), (b, \langle b \rangle)\}.$$

$S$  nem írható fel ciklusként. Tegyük fel ugyanis indirekt módon, hogy létezik olyan  $\pi$  feltétel és  $S_0$  program, amelyekre  $S = DO(\pi, S_0)$ . Ekkor:

$$- a \notin [\pi] \implies S_0(a) = \{\langle a \rangle\},$$

$$- a \in [\pi] \implies \forall \alpha \in S_0(a) \cap A^* : \tau(\alpha) \notin [\pi] \implies \langle a \rangle \notin S_0(a).$$

Mivel  $S$  programra egyik sem teljesül,  $S \neq DO(\pi, S_0)$ .

7. Nem igaz.  $\bigcup_{k=1}^n ([\pi_k] \cap \mathcal{D}_{p(S_k)})$  azon  $a$  pontokat tartalmazza, amelyekre van olyan ága az elágazásnak, amelynek a feltétele igaz  $a$ -ban, és a megfelelő  $S_k$  program biztosan terminál  $a$ -ból indulva. Ezzel szemben  $\mathcal{D}_{p(IF)}$ -ben csak olyan  $a$  pontok szerepelnek, amelyekre legalább egy feltétel teljesül, és minden olyan  $S_k$  program, amelyhez tartozó feltétel igaz, biztosan terminál  $a$ -ból.

Ha  $a \in \mathcal{D}_{p(IF)}$ , akkor nyilván  $\exists k \in [1..n] : a \in [\pi_k]$  és  $a \in \mathcal{D}_{p(S_k)}$ , tehát  $\mathcal{D}_{p(IF)} \subseteq \bigcup_{k=1}^n ([\pi_k] \cap \mathcal{D}_{p(S_k)})$ . A fordított irányú tartalmazás azonban nem áll fenn. Legyen például  $A = \{1, 2\}$ ,  $[\pi_1] = \{1, 2\}$ ,  $[\pi_2] = \{1\}$ ,

$$S_1 = \{(1, \langle 1, 1, \dots \rangle), (2, \langle 2 \rangle)\},$$

$$S_2 = \{(1, \langle 1 \rangle), (2, \langle 2, 2, \dots \rangle)\}.$$

Ekkor  $\mathcal{D}_{p(IF)} = \{2\}$ , viszont  $\bigcup_{k=1}^2 ([\pi_k] \cap \mathcal{D}_{p(S_k)}) = \{1, 2\}$ .

8.  $\mathcal{D}_{p(S)} = \emptyset \iff S_1 \cup S_2 \cup \dots \cup S_n$  minden ponthoz rendel végtelen sorozatot.  
 $\mathcal{D}_{p(IF)} = A \iff$  A  $\pi_k$  feltételek lefedik a teljes állapotteret, és minden  $S_k$  program az  $a \in [\pi_k]$  pontokhoz csak véges sorozatot rendel.

Például:  $\pi_1 = Hamis, S_1 = ABORT,$   
 $\pi_k = Igaz, S_k = SKIP \quad (k \in [2..n]),$

ahol *SKIP* és *ABORT* a 8. fejezetben bevezetett programok.

9. Nem igaz, egyik irányú tartalmazás sem áll fenn. Egyrészt az előző feladatban láttuk, hogy  $\mathcal{D}_{p(IF)} \not\subseteq \mathcal{D}_{p(S)}$ , tehát  $p(IF) \not\subseteq p(S)$ . Másrészt legyen például  $S_k = \text{SKIP}$ ,  $\pi_k = \text{Hamis}$  ( $k \in [1..n]$ ), ahol *SKIP* a 8. fejezetben bevezetett program. Ekkor nyilván  $p(IF) = \emptyset$ , de  $p(S) = p(\text{SKIP}) = id_A$ , tehát  $p(S) \not\subseteq p(IF)$ .

10. Igaz, ugyanis:

$$\begin{aligned} \mathcal{D}_{p(IF)} &= \{a \in A \mid a \in [\pi_1] \cup [\pi_2] \text{ és } a \in [\pi_1] \implies a \in \mathcal{D}_{p(S_1)} \text{ és} \\ &\quad a \in [\pi_2] \implies a \in \mathcal{D}_{p(S_2)}\} = \\ &= \{a \in A \mid (a \in [\pi_1] \text{ és } a \in [\pi_2] \text{ és } a \in \mathcal{D}_{p(S_1)} \text{ és } a \in \mathcal{D}_{p(S_2)}) \text{ vagy} \\ &\quad (a \in [\pi_1] \text{ és } a \notin [\pi_2] \text{ és } a \in \mathcal{D}_{p(S_1)}) \text{ vagy} \\ &\quad (a \notin [\pi_1] \text{ és } a \in [\pi_2] \text{ és } a \in \mathcal{D}_{p(S_2)})\} = \\ &= ([\pi_1] \cap [\pi_2] \cap \mathcal{D}_{p(S_1)} \cap \mathcal{D}_{p(S_2)}) \cup (([\pi_1] \setminus [\pi_2]) \cap \mathcal{D}_{p(S_1)}) \cup \\ &\quad \cup (([\pi_2] \setminus [\pi_1]) \cap \mathcal{D}_{p(S_2)}). \end{aligned}$$

11. Ebben a feladatban hivatkozunk a 8. fejezet definícióira.

$$\begin{aligned} S_1 &= \{(1, \langle 1, 2 \rangle), (2, \langle 2, 4 \rangle), (3, \langle 3, 3, \dots \rangle), (4, \langle 4, 4, \dots \rangle)\} \\ S_2 &= \{(1, \langle 1 \rangle), (2, \langle 2 \rangle), (3, \langle 3 \rangle), (4, \langle 4 \rangle)\} \\ IF &= \{(1, \langle 1, 2 \rangle), (1, \langle 1 \rangle), (2, \langle 2 \rangle), (3, \langle 3, 3, \dots \rangle), (4, \langle 4, 4, \dots \rangle)\} \end{aligned}$$

12. a) Mivel a szekvencia programfüggvénye a két komponensprogram programfüggvényének *szigorú* kompozíciója ( $p(S) = p(S_2) \odot p(S_1)$ ),  $F$  pedig a két feladat kompozíciója, ezért  $\mathcal{D}_F \subseteq \mathcal{D}_{p(S)}$  nem mindig teljesül,  $S$  nem feltétlenül oldja meg  $F$ -et.

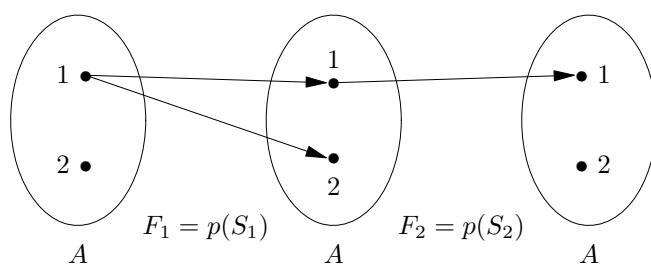
Tekintsük például a 15.10. ábrát. Legyen  $A = \{1, 2\}$ ,

$$\begin{aligned} F_1 &= \{(1, 1), (1, 2)\}, \quad F_2 = \{(1, 1)\}, \\ S_1 &= \{(1, \langle 1 \rangle), (1, \langle 1, 2 \rangle), (2, \langle 2, 2, \dots \rangle)\}, \\ S_2 &= \{(1, \langle 1 \rangle), (2, \langle 2, 2, \dots \rangle)\}. \end{aligned}$$

Ekkor  $p(S_1) = F_1$  és  $p(S_2) = F_2$ , tehát  $S_1$  megoldja  $F_1$ -et,  $S_2$  megoldja  $F_2$ -t, viszont:

$$\begin{aligned} S &= (S_1; S_2) = \{(1, \langle 1 \rangle), (1, \langle 1, 2, \dots \rangle), (2, \langle 2, \dots \rangle)\}, \\ F &= F_2 \circ F_1 = \{(1, 1)\}. \end{aligned}$$





15.10. ábra. Ellenpélda

Tehát  $p(S) = \emptyset$ , így  $S$  nem oldja meg  $F$ -et.

b) Mivel itt  $F$  a két feladat *szigorú* kompozíciója,  $S$  megoldja  $F$ -et.

Tudjuk, hogy  $S_1$  megoldja  $F_1$ -et,  $S_2$  megoldja  $F_2$ -t, tehát:

$$\begin{aligned} (1) \mathcal{D}_{F_1} &\subseteq \mathcal{D}_{p(S_1)} & (3) \mathcal{D}_{F_2} &\subseteq \mathcal{D}_{p(S_2)} \\ (2) \forall a \in \mathcal{D}_{F_1} : p(S_1)(a) &\subseteq F_1(a) & (4) \forall a \in \mathcal{D}_{F_2} : p(S_2)(a) &\subseteq F_2(a) \end{aligned}$$

Ezeket felhasználva lássuk be a megoldás két feltételének teljesülését:

$$\begin{aligned} i. \mathcal{D}_{F_2 \circ F_1} &\subseteq \mathcal{D}_{p(S_2) \circ p(S_1)}, \\ ii. \forall a \in \mathcal{D}_{F_2 \circ F_1} : p(S_2) \circ p(S_1)(a) &\subseteq F_2 \circ F_1(a). \end{aligned}$$

i. Legyen  $a \in \mathcal{D}_{F_2 \circ F_1}$ . Ekkor felhasználva a definíciókból egyszerűen adódó  $\mathcal{D}_{G \circ H} = H^{-1}(\mathcal{D}_G)$  összefüggést:

$$\begin{aligned} a \in \mathcal{D}_{F_2 \circ F_1} &= F_1^{-1}(\mathcal{D}_{F_2}) \iff \\ a \in \mathcal{D}_{F_1} \text{ és } F_1(a) &\subseteq \mathcal{D}_{F_2} \implies^{(1),(3)} \\ a \in \mathcal{D}_{p(S_1)} \text{ és } F_1(a) &\subseteq \mathcal{D}_{p(S_2)} \implies^{(2)} \\ a \in \mathcal{D}_{p(S_1)} \text{ és } p(S_1)(a) &\subseteq \mathcal{D}_{p(S_2)} \iff \\ a \in p(S_1)^{-1}(\mathcal{D}_{p(S_2)}) &= \mathcal{D}_{p(S_2) \circ p(S_1)}. \end{aligned}$$

Tehát beláttuk, hogy  $\mathcal{D}_F \subseteq \mathcal{D}_{p(S)}$ .

ii. Legyen  $a \in \mathcal{D}_{F_2 \circ F_1}$  rögzített. Ekkor a fentiek alapján (5)  $a \in \mathcal{D}_{F_1}$  és (6)  $F_1(a) \subseteq \mathcal{D}_{F_2}$ . Felhasználva, hogy  $\forall a \in \mathcal{D}_{G \circ H} : G \circ H(a) = G(H(a))$ :

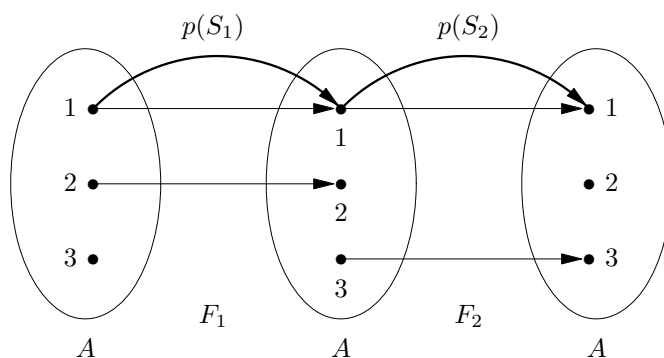
$$\begin{aligned} p(S_2) \circ p(S_1)(a) &= p(S_2)(p(S_1)(a)) \subseteq^{(5),(2)} \\ &\subseteq p(S_2)(F_1(a)) \subseteq^{(6),(4)} F_2(F_1(a)) = F_2 \circ F_1(a). \end{aligned}$$

Tehát beláttuk, hogy  $\forall a \in \mathcal{D}_F : p(S)(a) \subseteq F(a)$ .

13. Egyik sem teljesül. Tekintsük például a 15.11. ábrát. Legyen  $A = \{1, 2, 3\}$ ,

$$F_1 = \{(1, 1), (2, 2)\}, \quad F_2 = \{(1, 1), (3, 3)\},$$

$$S_1 = S_2 = \{(1, \langle 1 \rangle), (2, \langle 2, 2, \dots \rangle), (3, \langle 3, 3, \dots \rangle)\}.$$



15.11. ábra. Ellenpélda

Ekkor  $p(S_2) \circ p(S_1) = F_2 \circ F_1 = \{(1, 1)\}$ , tehát  $S$  megoldja  $F_2 \circ F_1$ -et, viszont  $p(S_1) = p(S_2) = \{(1, 1)\}$ , tehát  $S_1$  nem oldja meg  $F_1$ -et, és  $S_2$  nem oldja meg  $F_2$ -t.

14. a) Nem, ugyanis ha  $\mathcal{D}_F \not\subseteq \left[ \bigvee_{i=1}^n \pi_i \right] = \bigcup_{i=1}^n \lceil \pi_i \rceil$ , akkor létezik olyan  $a \in \mathcal{D}_F$  pont, amelyre  $a \notin \bigcup_{i=1}^n \lceil \pi_i \rceil$ , tehát  $a \notin \mathcal{D}_{p(IF)} \implies \mathcal{D}_F \not\subseteq \mathcal{D}_{p(IF)}$ .

Legyen például  $F = id_A$ ,  $\pi_k = Hamis$ ,  $S_k$  tetszőleges ( $k \in [1..n]$ ). Ekkor  $\forall k \in [1..n]$  esetén  $F|_{\lceil \pi_k \rceil} = \emptyset$ , tehát tetszőleges  $S_k$  program megoldja, viszont  $\forall k \in [1..n] : \pi_k = Hamis$  miatt nyilván  $p(IF) = \emptyset$ , tehát  $IF$  nem oldja meg  $F$ -et.

b) Ez következik a c) pontból, ugyanis itt  $\lceil \pi_1 \vee \pi_2 \vee \dots \vee \pi_n \rceil = A$ .

c) Igen. Tudjuk, hogy  $\forall k \in [1..n]$  esetén  $S_k$  megoldja az  $F|_{\lceil \pi_k \rceil}$ -t, tehát:

- (1)  $\forall k \in [1..n] : \mathcal{D}_F \cap \lceil \pi_k \rceil \subseteq \mathcal{D}_{p(S_k)}$ ,
- (2)  $\forall k \in [1..n] : \forall a \in \mathcal{D}_F \cap \lceil \pi_k \rceil : p(S_k)(a) \subseteq F|_{\lceil \pi_k \rceil}(a) = F(a)$ .

Ezeket felhasználva lássuk be a megoldás két feltételének teljesülését.

*i.* Legyen  $a \in \mathcal{D}_F$ . Ekkor egyrészt a feladatban szereplő feltétel miatt  $a \in \left[ \bigvee_{i=1}^n \pi_i \right] = \bigcup_{i=1}^n \lceil \pi_i \rceil$ , másrészt  $\forall k \in [1..n]$  esetén ha  $a \in \lceil \pi_k \rceil$ ,

akkor (1) miatt  $a \in \mathcal{D}_{p(S_k)}$ , tehát  $a \in \mathcal{D}_{p(IF)}$ . Ezzel beláttuk, hogy  $\mathcal{D}_F \subseteq \mathcal{D}_{p(IF)}$ .

ii. Legyen  $a \in \mathcal{D}_F$  rögzített. Ekkor  $a \in \mathcal{D}_{p(IF)}$ , tehát:

$$p(IF)(a) = \bigcup_{k=1}^n p(S_k)|_{\lceil \pi_k \rceil}(a) = \bigcup_{k \in [1..n], a \in \lceil \pi_k \rceil} p(S_k)(a).$$

Mivel minden  $k \in [1..n]$  és  $a \in \lceil \pi_k \rceil$  esetén (2) miatt  $p(S_k)(a) \subseteq F(a)$ , ezen halmazok úniója is része  $F(a)$ -nak. Ezzel beláttuk, hogy  $\forall a \in \mathcal{D}_F : p(IF)(a) \subseteq F(a)$ .

**15. Igaz. Tudjuk, hogy  $IF$  megoldja  $F$ -et, tehát:**

$$(1) \mathcal{D}_F \subseteq \mathcal{D}_{p(IF)} \iff \forall a \in \mathcal{D}_F : (a \in \bigcup_{i=1}^n \lceil \pi_i \rceil \text{ és } \forall k \in [1..n] : \\ (a \in \lceil \pi_k \rceil \implies a \in \mathcal{D}_{p(S_k)})),$$

$$(2) \forall a \in \mathcal{D}_F : p(IF)(a) \subseteq F(a).$$

Legyen  $k \in [1..n]$  rögzített, és lássuk be, hogy  $S_k$  megoldja  $F|_{\lceil \pi_k \rceil}$ -t.

i.  $\mathcal{D}_{F|_{\lceil \pi_k \rceil}} = \mathcal{D}_F \cap \lceil \pi_k \rceil \subseteq \mathcal{D}_{p(S_k)}$ , ugyanis minden  $a \in \mathcal{D}_F \cap \lceil \pi_k \rceil$  esetén (1) miatt  $a \in \mathcal{D}_{p(S_k)}$ .

ii. Legyen  $a \in \mathcal{D}_F \cap \lceil \pi_k \rceil$  rögzített. Ekkor (1) miatt  $a \in \mathcal{D}_{p(IF)}$  és (2) miatt:

$$p(IF)(a) = \bigcup_{i=1}^n p(S_i)|_{\lceil \pi_i \rceil}(a) \subseteq F(a).$$

Mivel  $a \in \lceil \pi_k \rceil$ ,  $F|_{\lceil \pi_k \rceil}(a) = F(a)$  és  $p(S_k)|_{\lceil \pi_k \rceil}(a) = p(S_k)(a)$ . Tehát:

$$p(S_k)(a) = p(S_k)|_{\lceil \pi_k \rceil}(a) \subseteq p(IF)(a) \subseteq F(a) = F|_{\lceil \pi_k \rceil}(a).$$

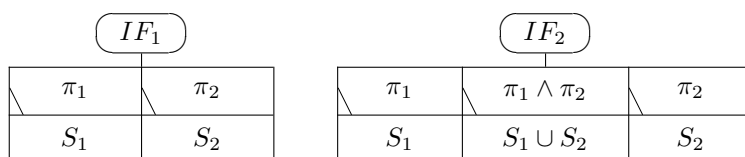
Ezzel beláttuk, hogy  $\forall a \in \mathcal{D}_{F|_{\lceil \pi_k \rceil}} : p(S_k)(a) \subseteq F|_{\lceil \pi_k \rceil}(a)$ .

**16. Nem.** Legyen például  $A = \{1, 2\}$ ,  $\lceil \pi_k \rceil = A$  ( $k \in [1..n]$ ) és

$$F_1 = \{(1, 1)\}, \quad S_1 = \{(1, \langle 1 \rangle), (2, \langle 2, 2, \dots \rangle)\}; \\ F_k = \{(2, 2)\}, \quad S_k = \{(1, \langle 1, 1, \dots \rangle), (2, \langle 2 \rangle)\} \quad (k \in [2..n]).$$

Ekkor  $S_k$  megoldja  $F_k$ -t ( $k \in [1..n]$ ), viszont  $p(IF) = \emptyset$ , tehát  $IF$  nyilván nem oldja meg az  $F = F_1 \cup \dots \cup F_n \neq \emptyset$  feladatot.

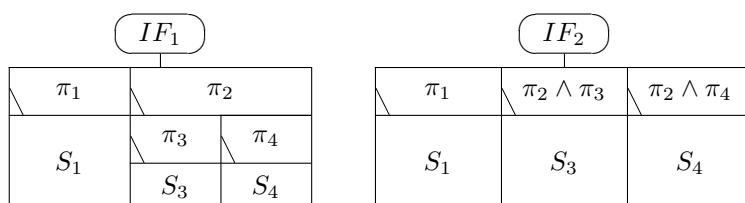
17. Igaz, ugyanis ez a feladat speciális esete a 15. feladatnak, amelyről beláttuk, hogy igaz. Itt meg van adva egy plusz feltétel, de azt nem kell kihasználnunk a bizonyításhoz.
18. Nem, tekintsük például a 16. feladatban adott ellenpéldát. Megjegyezzük azonban, hogy a  $\forall k \in [1..n] : \mathcal{D}_{F_k} = \lceil \pi_k \rceil$  szigorúbb feltétel mellett belátható, hogy  $IF$  megoldja  $F$ -et.
19. Egyenlőek, tehát ekvivalensek is. Tekintsük a struktogramjukat:



Az elágazás definíciója szerint  $\forall a \in A$  esetén:

$$IF_2(a) = \begin{cases} S_1(a), & \text{ha } a \in [\pi_1] \setminus [\pi_2]; \\ S_2(a), & \text{ha } a \in [\pi_2] \setminus [\pi_1]; \\ S_1(a) \cup S_2(a), & \text{ha } a \in [\pi_1 \wedge \pi_2]; \\ \{ \langle a, a, \dots \rangle \}, & \text{ha } a \notin [\pi_1 \vee \pi_2]. \end{cases} = IF_1(a)$$

20. Nem egyenlőek és nem is ekvivalensek. Tekintsük a struktogramjukat:



Első ránézésre azonosnak tűnnek, de nem azok. Olyan  $a \in A$  pontok estén különbözhetnek, amelyekre  $a \in [\pi_1]$ ,  $a \in [\pi_2]$ , de  $a \notin [\pi_3]$  és  $a \notin [\pi_4]$ . Ekkor ugyanis  $IF_{34}(a) = \{ \langle a, a, \dots \rangle \}$ ,  $IF_1(a) = S_1(a) \cup IF_{34}(a) = S_1(a) \cup \{ \langle a, a, \dots \rangle \}$ , viszont  $IF_2(a) = S_1(a)$ .

Legyen például  $S_1 = S_3 = S_4 = SKIP$ , ahol  $SKIP$  a 8. fejezetben bevezetett program,  $\pi_i$ -k pedig olyan feltételek, amelyekre  $\exists a \in [\pi_1 \wedge \pi_2 \wedge \neg \pi_3 \wedge \neg \pi_4]$ . Ekkor ugyanis  $a \notin \mathcal{D}_{p(IF_1)}$ , de  $a \in \mathcal{D}_{p(IF_2)}$ . Tehát  $IF_1$  és  $IF_2$  nem egyenlő és nem is ekvivalens.

**21.** Igen. Tudjuk, hogy  $S_0$  megoldja  $F$ -et, tehát:

- (1)  $\mathcal{D}_F \subseteq \mathcal{D}_{p(S_0)}$ ,
- (2)  $\forall a \in \mathcal{D}_F : p(S_0)(a) \subseteq F(a)$ .

Ezeket felhasználva lássuk be a megoldás két feltételének teljesülését:

- i.  $\mathcal{D}_{\overline{F|\pi}} \subseteq \mathcal{D}_{p(DO)} = \mathcal{D}_{\overline{p(S_0)|\pi}}$ ,
- ii.  $\forall a \in \mathcal{D}_{\overline{F|\pi}} : p(DO)(a) = \overline{p(S_0)|\pi}(a) \subseteq \overline{F|\pi}(a)$ .

Máshogy fogalmazva azt kell belátnunk, hogy egy  $a \in A$  pont esetén ha  $a \in \mathcal{D}_{\overline{F|\pi}}$ , akkor egyrészt  $a \in \mathcal{D}_{\overline{p(S_0)|\pi}}$ , másrészt  $\overline{p(S_0)|\pi}(a) \subseteq \overline{F|\pi}(a)$ . Bontsuk a bizonyítást három részre: ( $a \in \lceil \pi \rceil$  és  $a \notin \mathcal{D}_F$ ) vagy ( $a \in \lceil \pi \rceil$  és  $a \in \mathcal{D}_F$ ) vagy pedig  $a \notin \lceil \pi \rceil$ .

- Ha  $a \in \lceil \pi \rceil$  és  $a \notin \mathcal{D}_F$ , akkor  $a \in \mathcal{D}_{F|\pi}$ ,  $F|\pi(a) = \{a\}$ , tehát  $a \notin \mathcal{D}_{\overline{F|\pi}}$ .
- Legyen  $a \in \lceil \pi \rceil$  és  $a \in \mathcal{D}_F$ , és tegyük fel, hogy  $a \in \mathcal{D}_{\overline{F|\pi}}$ . Ekkor nem létezik olyan végtelen sorozat, amelynek tagjait  $a$ -ból indulva  $F|\pi$  reláció egymás utáni alkalmazásaival kapjuk. Ebből pedig következik, hogy  $F|\pi$  egymás utáni alkalmazásaival nem juthatunk  $\lceil \pi \rceil \setminus \mathcal{D}_F$ -be, tehát amíg nem jutunk ki  $\mathcal{D}_{F|\pi} = \lceil \pi \rceil$ -ből, addig biztosan  $\mathcal{D}_F$ -ben maradunk, ezért (1) és (2) miatt minden így kapott  $x$  pontra  $x \in \mathcal{D}_{p(S_0)}$  és  $p(S_0)(x) \subseteq F(x)$ . Vagyis  $a$ -ból indulva  $p(S_0)|\pi$  egymás utáni alkalmazásaival csak olyan sorozatokat kaphatunk, amelyeket  $F|\pi$  szerint is megkapunk. Ebből pedig  $a \in \mathcal{D}_{\overline{F|\pi}}$  miatt következik, hogy egyrészt  $a \in \mathcal{D}_{\overline{p(S_0)|\pi}}$ , másrészt  $\overline{p(S_0)|\pi}(a) \subseteq \overline{F|\pi}(a)$ .
- Ha  $a \notin \lceil \pi \rceil$ , akkor nyilván  $a \in \mathcal{D}_{\overline{F|\pi}}$  és  $a \in \mathcal{D}_{\overline{p(S_0)|\pi}}$ , és  $\overline{F|\pi}(a) = \overline{p(S_0)|\pi}(a) = \{a\}$ .

Ezzel minden  $a \in A$  pontra beláttuk, hogy ha  $a \in \mathcal{D}_{\overline{F|\pi}}$ , akkor  $a \in \mathcal{D}_{\overline{p(S_0)|\pi}}$  és  $\overline{p(S_0)|\pi}(a) \subseteq \overline{F|\pi}(a)$ , tehát a  $DO(\pi, S_0)$  program megoldja az  $F|\pi$  feladatot.

**22.** Használjuk a 8. fejezetben bevezetett *SKIP* és *ABORT* programokat.

- a) Nem igaz. Legyen például  $S = ABORT$  és  $\pi = Hamis$ . Ekkor ugyanis  $DO(\pi, S) = SKIP$ , tehát:

$$p(DO) = id_A \not\subseteq \emptyset = p(S).$$

- b) Nem igaz. Legyen például  $S = SKIP$  és  $\pi = Igaz$ . Ekkor ugyanis  $DO(\pi, S) = ABORT$ , tehát:

$$p(DO) = \emptyset \not\subseteq id_A = p(S).$$

23. Ebben a feladatban hivatkozunk a 8. fejezet definícióira.

$$S((2, 4)) = \{ \langle (2, 4), (0, 4), (1, 4), (3, 4), (5, 4), (7, 4), (9, 4), \dots \rangle \}$$

$$S((3, 7)) = \{ \langle (3, 7), (0, 7), (1, 7), (3, 7), (5, 7), (7, 7) \rangle \}$$

## 15.6. Levezetési szabályok

1. Legyen  $DO(\pi, S_0)$  ciklus, amelyre teljesülnek a ciklus levezetési szabályának feltételei, tehát:

- (1)  $Q \Rightarrow P$ ,
- (2)  $P \wedge \neg\pi \Rightarrow R$ ,
- (3)  $P \wedge \pi \Rightarrow t > 0$ ,
- (4)  $P \wedge \pi \Rightarrow lf(S_0, P)$ ,
- (5)  $P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_0, t < t_0)$ .

- a) Nem. Tegyük fel ugyanis indirekt módon, hogy  $\lceil P \wedge R \rceil = \emptyset$ . Ekkor (2) miatt nyilván  $\lceil P \wedge \neg\pi \rceil = \emptyset$ , vagyis (6)  $\lceil P \rceil \subseteq \lceil \pi \rceil$ , és mivel  $\lceil Q \rceil \neq \emptyset$ , (1) miatt  $\lceil P \rceil \neq \emptyset$ . Legyen  $a \in \lceil P \rceil$ , ekkor (6) miatt  $a \in \lceil \pi \rceil$  és (3) miatt  $t(a) > 0$ . Továbbá (4) és (5) feltételekből adódik, hogy  $a \in \lceil lf(S_0, P) \rceil$  és  $a \in \lceil lf(S_0, t < t(a)) \rceil$ , tehát biztosan létezik olyan  $b \in p(S_0)(a)$  pont, amelyre  $b \in \lceil P \rceil$  és  $t(b) < t(a)$ . Viszont (6) miatt ekkor  $b \in \lceil \pi \rceil$ , így (3) miatt  $t(b) > 0$ . Vagyis beláttuk, hogy  $\forall a \in \lceil P \rceil : \exists b \in \lceil P \rceil : 0 < t(b) < t(a)$ . Mivel azonban  $\exists q \in \lceil P \rceil$ , a fentiek alapján  $\exists \alpha \in \lceil P \rceil^\infty : \alpha_1 = q$  és  $\forall i \in \mathbb{N} : 0 < t(\alpha_{i+1}) < t(\alpha_i)$ . Tehát a  $t(\alpha_i)$  értékek egy természetes számokból álló szigorúan monoton csökkenő végtelen sorozatot alkotnak, ami nyilván ellentmondás. Ezzel beláttuk, hogy  $\lceil P \wedge R \rceil$  nem lehet üres.
- b) Nem, ugyanis (2) miatt  $\lceil P \wedge \neg\pi \wedge R \rceil = \lceil P \wedge \neg\pi \rceil$ , amelyről pedig az a) pontban beláttuk, hogy nem lehet üres.

2. Nem. Legyen ugyanis  $DO(\pi, S_0)$  ciklus, amelyre teljesülnek a ciklus levezetési szabályának (1) – (5) feltételei. Ekkor a tétel alapján  $Q \Rightarrow lf(DO, R)$ , tehát:

$$\lceil Q \wedge \pi \rceil \subseteq \lceil Q \rceil \subseteq \lceil lf(DO, R) \rceil.$$

Továbbá:

$$\lceil Q \wedge \pi \rceil \subseteq^{(1)} \lceil P \wedge \pi \rceil \subseteq^{(4)} \lceil lf(S_0, P) \rceil.$$

Így  $\lceil Q \wedge \pi \rceil \subseteq \lceil lf(S_0, P) \rceil \cap \lceil lf(DO, R) \rceil$ , és mivel  $\lceil Q \wedge \pi \rceil$  nem üres,  $\lceil lf(S_0, P) \rceil \cap \lceil lf(DO, R) \rceil$  sem lehet üres.

**3.** Legyen  $DO(\pi, S_0)$  ciklus, amelyre teljesülnek a ciklus levezetési szabályának (1) – (5) feltételei.

a) Igaz, ugyanis a ciklus levezetési szabályának bizonyítása előtt kimondott 1. állításból következik.

b) Igaz. Lássuk be  $k$  szerinti teljes indukcióval.

– Ha  $k = 0$ , akkor  $b \in g^0(q) \cap \lceil \pi \rceil \cap \lceil P \rceil = \{q\} \implies b = q \implies t(b) = t(q) - 0$ .

– Tegyük fel, hogy  $\forall b \in g^k(q) \cap \lceil \pi \rceil \cap \lceil P \rceil : t(b) \leq t(q) - k$ , és legyen  $c \in g^{k+1}(q) \cap \lceil \pi \rceil \cap \lceil P \rceil \implies \exists b \in g^k(q) : c \in g(b)$ . Ekkor az a) pont alapján  $b \in \lceil P \rceil$ , továbbá  $b \in \lceil \pi \rceil$ , ugyanis különben  $g(b) = p(S_0)|_{\lceil \pi \rceil}(b)$  üres lenne. Tehát az indukciós feltevés alapján  $t(b) \leq t(q) - k$ , továbbá az (5) feltételből adódik, hogy  $b \in \lceil lf(S_0, t < t(b)) \rceil$ , tehát  $\forall x \in p(S_0)(b) : t(x) < t(b)$ . Mivel azonban  $c \in g(b) = p(S_0)(b)$ , ezért  $t(c) < t(b) \leq t(q) - k$ , így  $t(c) \leq t(q) - (k + 1)$ .

Ezzel az állítást beláttuk.

c) Igaz, ugyanis:

$$g|\pi = (p(S_0)|_{\lceil \pi \rceil})|\pi = p(S_0)|_{\lceil \pi \rceil} \cup id_{\lceil \pi \rceil \setminus \mathcal{D}_{p(S_0)}} = p(S_0)|\pi.$$

d) Igaz. Először is vegyük észre, hogy ha valamilyen  $k \in \mathbb{N}_0$ -ra  $g^k(q) \subseteq \lceil \neg\pi \rceil$  teljesül, akkor  $(k + 1)$ -re is teljesül, ugyanis  $\mathcal{D}_g \subseteq \lceil \pi \rceil$  miatt  $g^{k+1}(q) = g(g^k(q)) = \emptyset$ . Tehát ha létezik ilyen  $k \leq t(q)$ , akkor  $k = t(q)$  is ilyen.

Legyen tehát  $k = t(q)$ , és tegyük fel indirekt módon, hogy  $g^k(q) \not\subseteq \lceil \neg\pi \rceil \implies \exists x \in g^k(q) \cap \lceil \pi \rceil$ . Ekkor az a) pont alapján  $x \in \lceil P \rceil$  is teljesül, így a b) pont miatt  $t(x) \leq t(q) - k = 0$ . Viszont  $x \in \lceil P \rceil \cap \lceil \pi \rceil$ , ezért (3) miatt  $t(x) > 0$ , így ellentmondásra jutottunk. Ezzel beláttuk, hogy  $g^{t(q)}(q) \subseteq \lceil \neg\pi \rceil$ .

**4.** Igen. Legyen például  $A = \{1, 2\}$ ,  $\lceil Q \rceil = \{1\}$ ,  $\lceil Q' \rceil = \{1, 2\}$ ,  $\lceil R \rceil = \{2\}$ ,  $S_1 = S_2 = \{(1, \langle 1, 2 \rangle), (2, \langle 2 \rangle)\}$ . Ekkor ugyanis  $S = S_1 = S_2$ , tehát:

$$\begin{aligned} \lceil Q \rceil = \{1\} &\subseteq \{1, 2\} = \lceil lf(S_1, Q') \rceil, \\ \lceil Q' \rceil = \{1, 2\} &\subseteq \{1, 2\} = \lceil lf(S_2, R) \rceil, \\ \lceil Q \rceil = \{1\} &\subseteq \{1, 2\} = \lceil lf(S, R) \rceil. \end{aligned}$$

Viszont  $[Q] \cap [R] = \emptyset$ ,  $[Q] \cap [Q'] = \{1\} \neq \emptyset$  és  $[Q'] \cap [R] = \{2\} \neq \emptyset$ .

5. Ebben a feladatban hivatkozunk a 8. fejezetben szerepelő definíciókra és állításokra. Az  $S_0$  a definíciója alapján megegyezik az alábbi struktogrammal megadott programmal:

$y \geq 1$	$y = 0$
$x := x - 1$	<i>SKIP</i>
$y := y - 1$	

Tehát a *DO* program az alábbi struktogrammal adható meg:

$\pi$	
$y \geq 1$	$y = 0$
$x := x - 1$	<i>SKIP</i>
$y := y - 1$	

A feladat szövegesen megfogalmazva az, hogy az  $x$  és az  $y$  változó értékét csökkentjük az  $y$  értékével. Az  $S_0$  program pedig minden ciklusiterációban az  $x$  és az  $y$  értékét eggyel csökkenti, tehát a ciklusinvariánsban azt kell megfogalmaznunk, hogy az  $x$  és az  $y$  az eredeti értékéhez képest ugyanannyival csökkent, vagyis:

$$P = (\exists z \in \mathbb{N} : (x = x' - z \wedge y = y' - z)).$$

Így ciklusfeltételnek  $\pi = (y \neq 0)$  megfelelő lesz, ugyanis ekkor nyilván teljesül a ciklus levezetési szabályának (2) pontja. Továbbá mivel  $y$  értéke  $\pi$  teljesülése esetén pozitív, és minden iterációban csökken,  $t = y$  megfelelő termináló függvény.

Ezek után a szekvencia és az elágazás levezetési szabályának felhasználásával a 12. fejezetben szereplő levezetések során alkalmazott módszerekkel formálisan is belátható, hogy erre a *DO* programra a fent megadott  $P$ ,  $\pi$  és  $t$  esetén teljesülnek a ciklus levezetési szabályának feltételei.

6. Ebben a feladatban hivatkozunk a 8. fejezetben szerepelő definíciókra és állításokra. Az  $S = (S_1; (S_2; S_3))$  program struktogramja:



$S_1 :$	$k := 5$	
$S_2 :$	$a > b$	$a \leq b$
	$x := a - b$	$x := b - a$
$S_3 :$	$i := i + 1$	

Vezessünk be egy  $Q'$  és egy  $Q''$  köztes feltételt:

$$Q' = (Q \wedge k = 5),$$

$$Q'' = lf(S_3, R) = R^{i \leftarrow i+1} = (a = a' \wedge b = b' \wedge k \cdot (i + 1) \leq x).$$

Ekkor a 12. fejezetben szereplő levezetések során alkalmazott módszerekkel belátható, hogy (1)  $Q \Rightarrow lf(S_1, Q')$ , (2)  $Q' \Rightarrow lf(S_2, Q'')$ , továbbá  $Q''$  definíciójából adódik, hogy (3)  $Q'' \Rightarrow lf(S_3, R)$ . Ekkor a szekvencia levezetési szabálya alapján a (2) és (3) állításokból következik, hogy (4)  $Q' \Rightarrow lf((S_2; S_3), R)$ , valamint az (1) és (4) állításokból következik, hogy:

$$Q \Rightarrow lf((S_1; (S_2; S_3)), R) = lf(S, R).$$

*Megjegyzés:* Vegyük észre, hogy ebben a feladatban a levezetési szabályokat *helyességbizonyításra* használtuk, azaz egy már meglévő programról bizonyítottuk be, hogy megold egy adott feladatot. Azonban a 12. fejezetben bemutatott levezetési módszer során nem ezt az utat követjük, hanem a feladatból (annak specifikációjából) kiindulva, azt a levezetési szabályok mentén részfeladatokra bontva állítunk elő egy eleve helyes megoldó programot.

## 15.7. Elemi programok

1. a) Definíció szerint igen.
- b) A kiterjesztés definíciójából adódik, hogy  $S_1$  elemi program, de  $M$  nem feltétlenül. Legyen például  $M \subseteq C \times C^{**}$  olyan program, amelyre:

$$M((2, a)) = \{ \langle (2, a), (2, b), (2, a), (2, b), (2, a), \dots \rangle \} \text{ és}$$

$$M((x, y)) = S_1((x, y)), \text{ ha } (x, y) \neq (2, a).$$

Ekkor  $pr_A(M) = S$  is teljesül, nem csak az, hogy  $M$  ekvivalens  $S$ -sel  $A$ -n, viszont  $M$  nyilván nem elemi program.

2.  $lf(x, y := F(x, y), R) = lf(x, y := y, x, x < y) = (x < y)^{x \leftarrow y. y \leftarrow x} = y < x$ .

3. Legyen  $F$  olyan feladat  $A$ -n, amelyet a  $SKIP$  program megold. Ekkor  $p(SKIP) = id_A$  miatt  $\mathcal{D}_F \subseteq A$ , és  $\forall a \in \mathcal{D}_F : id_A(a) = \{a\} \subseteq F(a)$ . Tehát a  $SKIP$  program pontosan azokat a feladatokat oldja meg, amelyekre  $id_{\mathcal{D}_F} \subseteq F$ .
4. Mivel  $p(ABORT) = \emptyset$ , ezért  $\mathcal{D}_F \subseteq \mathcal{D}_{p(ABORT)}$  csak  $F = \emptyset$  esetén teljesül, tehát az  $ABORT$  program csak az  $F = \emptyset$  feladatot oldja meg.

## 15.8. Típus

1. Legyen  $\mathcal{T}_s = (H, I_s, \mathbb{F})$ , ahol

$$\begin{aligned} H &= [0..99999], \\ I_s &= Iga_z, \\ \mathbb{F} &= \{F_k, F_e\}. \end{aligned}$$

Felhívjuk a figyelmet arra, hogy mindegyik feladatban a 10.1. és 10.2. példákhoz hasonlóan  $T_{\mathcal{T}}$  elemeit az egyszerűség kedvéért  $(\mathcal{T}, x)$  helyett  $x$ -szel jelöljük, és  $T_{\mathcal{T}}$  helyett  $T$ -t írunk, így  $T = [I_s] \subseteq H$ . Jelen esetben  $T = H = [0..99999]$ .

$$A_k = T, F_k \subseteq A_k \times A_k,$$

$$F_k = \{(a, a') \mid a' = f_k(a)\},$$

ahol  $f_k : T \rightarrow T$ ,

$$f_k(x) = \begin{cases} x + 1, & \text{ha } x \neq 99999; \\ 0, & \text{ha } x = 99999. \end{cases}$$

$$A_e = T, F_e \subseteq A_e \times A_e,$$

$$F_e = \{(a, a') \mid a' = f_e(a)\},$$

ahol  $f_e : T \rightarrow T$ ,

$$f_e(x) = \begin{cases} x - 1, & \text{ha } x \neq 0; \\ 99999, & \text{ha } x = 0. \end{cases}$$

Adjunk a típusspecifikációnak megfelelő típust. A feladat szövege alapján adódik, hogy reprezentáljuk a típusértékeket a 10-es számrendszerbeli alakjukkal, mint decimális számjegyek sorozatával, amire két kézenfekvő megoldás kínálkozik. Az egyik az, hogy *legfeljebb* öt hosszúságú sorozatokat engedünk meg, és

megkötjük, hogy nullával csak akkor kezdődhetnek, ha egy hosszúak, a másik pedig az, hogy a sorozatok mindig *pontosan* öt hosszúak. Ekkor egyes esetekben hosszabb reprezentáló sorozatokat kapunk, viszont az  $I$  invariáns felírása és a műveletek megvalósítása egyszerűbb, ezért ezt az utat választjuk.

Legyen tehát  $\mathcal{T} = (\varrho, I, \mathbb{S})$ , ahol  $E = [0..9]$ ,  $I : E^* \rightarrow \mathbb{L}$ ,

$$[I] = \{\alpha \in E^* \mid |\alpha| = 5\};$$

$\varrho \subseteq E^* \times T, \forall \alpha \in E^* :$

$$\varrho(\alpha) = \{g(\alpha)\},$$

ahol  $g : E^* \rightarrow T, \forall \alpha \in E^* :$

$$g(\alpha) = \sum_{i=1}^{|\alpha|} \alpha_i \cdot 10^{|\alpha|-i}.$$

$\mathbb{S} = \{S_k, S_e\}$ , ahol  $B_k = E^*$  és  $S_k \subseteq B_k \times B_k^{**}$  olyan program, amelyre:

$$p(S_k) = \{(\alpha, \alpha') \mid \alpha \in [I] \text{ és } \alpha' \in [I] \text{ és } g(\alpha') = f_k(g(\alpha))\};$$

$B_e = E^*$  és  $S_e \subseteq B_e \times B_e^{**}$  olyan program, amelyre:

$$p(S_e) = \{(\alpha, \alpha') \mid \alpha \in [I] \text{ és } \alpha' \in [I] \text{ és } g(\alpha') = f_e(g(\alpha))\}.$$

Ekkor ez a típus megfelel a fenti típuspecifikációnak, ugyanis egyrészt  $\varrho$  definíciója alapján nyilvánvaló, hogy a reprezentáció helyes ( $\varrho([I]) = T$ ), másrészt a programfüggvények és a reprezentációs függvény megadásából egyszerűen adódik, hogy a programok  $\varrho$ -n keresztül megoldják a megfelelő feladatokat.

2. Vegyük észre, hogy az  $F$  feladat nem más, mint az összeadás 10 szerinti maradékkal. Legyen  $\mathcal{T} = (\varrho, I, \mathbb{S})$ , ahol  $\mathbb{S} = \{S\}$ . Az előző feladathoz hasonlóan az első gondolatunk itt is az lehetne, hogy mivel  $E = \{0, 1, 2\}$ , reprezentáljuk a típusértékeket a 3-as számrendszerbeli alakjukkal.

Ekkor  $\varrho \subseteq E^* \times T, \forall \alpha \in E^* :$

$$\varrho(\alpha) = \{g(\alpha)\},$$

ahol  $g : E^* \rightarrow T, \forall \alpha \in E^* :$

$$g(\alpha) = \sum_{i=1}^{|\alpha|} \alpha_i \cdot 3^{|\alpha|-i}.$$

Azonban így azt az  $I : E^* \rightarrow \mathbb{L}$  típusinvariánst kapnánk, amelyre:

$$[I] = \{\alpha \in E^* \mid |\alpha| = 3 \text{ és } \alpha_1 \leq 1 \text{ és } (\alpha_1 = 1 \implies \alpha_2 = \alpha_3 = 0)\}.$$

És nyilván a művelet megvalósítása is viszonylag bonyolult lenne.

Próbáljunk meg inkább olyan reprezentációt adni, amely mellett könnyebben tudjuk megfogalmazni a műveletet a reprezentáló sorozatokon, mint az első esetben. Vegyük észre, hogy semmi sem követeli meg, hogy 3-as számrendszerben való reprezentációban gondolkozzunk. Sokkal egyszerűbb reprezentációt kapunk akkor, ha az  $E$  elemei közül csak az egyiket (például az 1-et) használva egy  $T$ -beli elemet egyszerűen az értékével megegyező hosszúságú sorozattal reprezentálunk, tehát  $\varrho \subseteq E^* \times T, \forall \alpha \in E^*$ :

$$\varrho(\alpha) = \{|\alpha|\};$$

$$I : E^* \rightarrow \mathbb{L},$$

$$[I] = \{\alpha \in E^* \mid |\alpha| < 10 \text{ és } \forall i \in [1..|\alpha|] : \alpha_i = 1\}.$$

Ekkor ugyanis a művelet megvalósítása sokkal egyszerűbbé válik. Legyen  $B = E^* \times E^* \times E^*$  és  $S \subseteq B \times B^{**}$  olyan program, amelyre:

$$p(S) = \{((\alpha, \beta, \gamma), (\alpha', \beta', \gamma')) \mid \alpha \in [I] \text{ és } \beta \in [I] \text{ és } \gamma' \in [I] \text{ és} \\ \alpha = \alpha' \text{ és } \beta = \beta' \text{ és } |\gamma'| = h(|\alpha|, |\beta|)\},$$

ahol  $h : \mathbb{N}_0 \times \mathbb{N}_0 \rightarrow \mathbb{N}_0$ ,

$$h(a, b) = \begin{cases} a + b, & \text{ha } a + b < 10; \\ a + b - 10, & \text{ha } a + b \geq 10. \end{cases}$$

Vegyük észre, hogy sehol nem használtuk ki, hogy a sorozatok minden tagja azonos, tehát a típusinvariánsból a  $\forall i \in [1..|\alpha|] : \alpha_i = 1$  kikötés el is hagyható. Ekkor olyan reprezentációt kapnánk, amely nem kölcsönösen egyértelmű, ugyanis  $\varrho|_{[I]}^{(-1)}$  nem determinisztikus, ez azonban nem okoz gondot.

A művelet megvalósítása még ennél is egyszerűbbé válna akkor, ha úgy módosítanánk a reprezentációt, hogy egy  $a \in T$  elemet nem feltétlenül az értékével megegyező hosszúságú sorozattal reprezentálunk, hanem egy tetszőleges olyan sorozattal, amely hosszának tízzel való osztási maradéka megegyezik  $a$ -val, tehát  $\varrho \subseteq E^* \times T, \forall \alpha \in E^*$ :

$$\varrho(\alpha) = \{\text{az } |\alpha| \text{ 10-zel való osztási maradéka}\}.$$

Ekkor ugyanis  $I$  a fentiek alapján választható akár  $I_{gaz}$ -nak is, és a művelet egyszerűen két sorozat konkatenációjával megvalósítható. Azonban ilyenkor  $\varrho|_{\Gamma I}^{(-1)}$  nem determinisztikus, sőt korlátlanul nem determinisztikus! Ráadásul a műveletünk is olyan, hogy többször alkalmazva mindig egyre hosszabb sorozatokat eredményez, ezért egy ilyen reprezentáció a gyakorlatban nem használható.

6. A típusértékhalmoz a természetes számok halmaza, amelyre az alábbi két feladatot fogalmaztuk meg:

- $F_1$  : adjuk meg egy természetes szám 8-cal való osztási maradékát;  
 $F_2$  : döntsük el két természetes számról, hogy azonosak-e.

Egy természetes számot a 8-as számrendszerbeli alakjával reprezentálunk. Mindkét esetben megkötjük, hogy egy reprezentáló sorozat legalább egy hosszúságú és az a) esetben még azt is, hogy nullával csak akkor kezdődhet, ha egy hosszúságú (vagyis ha a nullát reprezentálja). Tehát a b) esetben a reprezentációnk nem kölcsönösen egyértelmű, ugyanis  $\varrho|_{\Gamma I}^{(-1)}$  nem determinisztikus.

Az  $S_1$  program egy  $e \in E^*$  elemhez olyan sorozatot rendel, amelynek tagjai  $e$  utolsó valahány tagját tartalmazzák, mégpedig minden tag eggyel kevesebbet, mint az előző, így a sorozat utolsó tagja az  $e$  utolsó tagjából alkotott sorozat lesz. Például:

$$S_1(\langle 4, 1, 7, 3 \rangle) = \{ \langle 4, 1, 7, 3 \rangle, \langle 1, 7, 3 \rangle, \langle 7, 3 \rangle, \langle 3 \rangle \}.$$

Tehát  $p(S_1)$  egy  $e \in E^*$  elemhez az utolsó tagjából alkotott sorozatot rendeli, amely nyilván az  $e$  által reprezentált számnak a 8-cal való osztási maradékát reprezentálja.

Az  $S_2$  program egy  $(e, d, l) \in E^* \times E^* \times \mathbb{L}$  elemhez olyan sorozatot rendel, amelynek  $(ee, dd, ll)$  tagjaiban  $ee$  és  $dd$  mindig eggyel hosszabb kezdőszelete  $e$ -nek, illetve  $d$ -nek, és  $ll$  pedig azt adja meg, hogy  $ee$  és  $dd$  megegyezik-e, egészen addig, amíg valamelyik sorozatnak a végére nem érünk. Például:

$$S_2(\langle \langle 4, 1, 7, 3 \rangle, \langle 4, 1, 7, 3, 2 \rangle, h \rangle) = \{ \langle \langle 4, 1, 7, 3 \rangle, \langle 4, 1, 7, 3, 2 \rangle, h \rangle, \langle \langle 4 \rangle, \langle 4 \rangle, i \rangle, \langle \langle 4, 1 \rangle, \langle 4, 1 \rangle, i \rangle, \langle \langle 4, 1, 7 \rangle, \langle 4, 1, 7 \rangle, i \rangle, \langle \langle 4, 1, 7, 3 \rangle, \langle 4, 1, 7, 3 \rangle, i \rangle \}.$$

Tehát  $p(S_2)$  egy  $(e, d, l) \in E^* \times E^* \times \mathbb{L}$  elemhez olyan  $(e', d', l') \in E^* \times E^* \times \mathbb{L}$  elemet rendel, amelyre  $e'$  és  $d'$  azonos hosszúságú kezdőszelete

$e$ -nek, illetve  $d$ -nek, úgy hogy  $e = e'$  és  $d = d'$  közül legalább az egyik teljesül,  $l'$  pedig azt adja meg, hogy  $e'$  és  $d'$  megegyezik-e.

Ezek alapján látható, hogy a reprezentáció az a) és a b) esetben egyaránt helyes, továbbá hogy  $S_1$  mindkét esetben megoldja  $F_1$ -et  $\varrho$ -n keresztül. Azonban a fenti példából is látszik, hogy  $S_2$  nem oldja meg  $F_2$ -t  $\varrho$ -n keresztül, ugyanis egyrészt  $p(S_2)$  nem minden esetben tartja meg mindkét sorozatot (csak ha azonos hosszúságúak), és olyankor is *igaz* választ ad, amikor a két sorozat nem egyezik meg, csak az egyik kezdőszellete a másikkal. Tehát a típus egyik esetben sem felel meg a specifikációnak.

7. A továbbiakban az egyes feladatok kapcsán csak azt vizsgáljuk meg, hogy milyen reprezentációt érdemes választani, ez alapján már a 10. fejezetben szereplő példákhoz és az előző feladatokhoz hasonlóan megadható a típus-specifikáció és a típus.

Ebben a feladatban a legtermészetesebb reprezentáció az, hogy a síkvektorokat valamilyen bázisban vett koordinátáikkal írjuk le. Ekkor a szükséges műveletek könnyen megvalósíthatók. Tehát ha például a típusértékeink egy rögzített bázisban vett egész koordinátájú vektorok, akkor  $E$  lehet az egész számok halmaza, és így minden vektort le tudunk írni egy kettő hosszúságú sorozattal.

8. Az előző feladathoz hasonlóan a térvektorokat is reprezentálhatjuk egy rögzített bázisban vett koordinátáikkal (három hosszúságú sorozatokkal), és így nyilván a műveletek is egyszerűen megvalósíthatók.

9. Ebben a feladatban egy komplex számot az algebrai alakjának ( $z = a + ib$ ) megfelelően érdemes reprezentálni, ekkor ugyanis az összeadás és a képzetes rész meghatározás egyszerűen megvalósítható. Ha például csak a Gauss-egészekkel foglalkozunk (vagyis azon komplex számokkal, amelyeknek a valós és a képzetes része is egész szám), akkor  $E = \mathbb{Z}$  választással minden típusértéket reprezentálhatunk egy kettő hosszúságú sorozattal.

10. Ebben a feladatban a típusértékeink ugyanúgy komplex számok, mint az előzőben, viszont más műveleteket kell megvalósítanunk. A szorzás az előző feladatban szereplő reprezentáció esetén is könnyen megvalósítható, viszont a természetes kitevőjű hatványozás sokkal egyszerűbb abban az esetben, ha egy komplex számot nem az algebrai, hanem a trigonometrikus alakjának megfelelően reprezentálunk. Ekkor ugyanis egy  $z = r(\cos\varphi + i\sin\varphi)$  komplex szám  $n$ . hatványa ( $n \in \mathbb{N}$ ):  $z^n = r^n(\cos(n\varphi) + i\sin(n\varphi))$ . Tehát feltéve, hogy van egy olyan típusunk, amelyben egy a valós számok halmazát „elég sűrűn” lefedő

számhalmazt a szokásos aritmetikai műveletekkel megvalósítottunk, akkor ezt a típusérték-halmazt választva  $E$ -nek, egy komplex számot reprezentálhatunk a trigonometrikus alakjának megfelelően egy kettő hosszúságú sorozattal, amelynek az egyik tagja a komplex szám abszolútértékét határozza meg, tehát arról meg kell kötnünk, hogy nemnegatív.

*Megjegyzés:* Emlékeztetünk arra, hogy egy állapotér komponensei definíció szerint legfeljebb megszámlálhatóak lehetnek, így a komplex, illetve a valós számok halmaza nem szerepelhet típusérték-halmazként.

## 15.9. Levezetés

1.

*Specifikáció:*

$$A = \begin{array}{cccc} \mathbb{Z} & \times & \mathbb{Z} & \times & \mathbb{Z} & \times & \mathbb{Z} \\ & & m & & n & & l & & b \end{array}$$

$$B = \begin{array}{cc} \mathbb{Z} & \times & \mathbb{Z} \\ & & m' & & n' \end{array}$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n)$$

$$R : (Q \wedge b \in [m..n] \wedge l \in [m..n] \wedge \forall i \in [m..n] : f(b) \geq f(i) \geq f(l))$$

A specifikációból kiolvasható, hogy nem engedjük meg az üres intervallumon való futtatást, illetve az eredményt az állapottér két változója ( $b$  és  $l$ , jelentése rendre legnagyobb, legkisebb index) szolgáltatja.

*Megoldás:*

A feladatot az intervallumokon megadott programozási tételeknél használt módon oldjuk meg, először megadjuk a  $\varphi$  függvényt.

$$\varphi(m, n, b, l, be, le) = (b \in [m..n] \wedge l \in [m..n] \wedge f(l) = le \wedge f(b) = be \wedge \forall i \in [m..n] : be \geq f(i) \geq le)$$

Ezzel a  $\varphi$ -vel igazából a feladat egy szigorítását oldjuk meg, ami azt is megadja, hogy mik a függvény értékek a szóban lévő pontokban.

Ennek megfelelően a szigorított utófeltétel, az invariáns tulajdonság és a  $Q'$ , illetve  $Q''$  állítás:

$$R = (Q \wedge b \in [m..n] \wedge l \in [m..n] \wedge f(l) = le \wedge f(b) = be \wedge \forall i \in [m..n] : be \geq f(i) \geq le)$$

$$P = (Q \wedge k \in [m..n] \wedge b \in [m..k] \wedge l \in [m..k] \wedge f(l) = le \wedge f(b) = be \wedge \forall i \in [m..k] : be \geq f(i) \geq le)$$

$$Q' = (Q \wedge k = m \wedge b = m \wedge l = m \wedge le = be = f(m))$$

$$Q'' = (Q \wedge k + 1 \in [m..n] \wedge b \in [m..k + 1] \wedge l \in [m..k + 1] \wedge f(l) = le \wedge f(b) = be \wedge \forall i \in [m..k + 1] : be \geq f(i) \geq le \wedge t = t_0)$$

$S_1$  a  $k, b, l, be, le := m, m, m, f(m), f(m)$  értékadás. De vajon mi lehet  $S_2$ ? A megoldás természetesen egy elágazás lesz, aminek a  $P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_2, Q'')$  követelményt kell teljesítenie.

(S<sub>2</sub>)

$f(k + 1) \leq le$	$f(k + 1) \geq be$	$le \leq f(k + 1) \leq be$
$l, le := k + 1, f(k + 1)$	$b, be := k + 1, f(k + 1)$	<i>SKIP</i>

Vizsgáljuk meg ezt az elágazást!



1.  $P \wedge \pi \wedge t = t_0 \Rightarrow \left( \bigvee_{i=1}^n \pi_i \right)$   
 $f(k+1) \leq le \vee f(k+1) \geq be \vee le \leq f(k+1) \leq be \Leftrightarrow (f(k+1) \leq le \vee f(k+1) \geq be \vee le \leq f(k+1)) \wedge (f(k+1) \leq le \vee f(k+1) \geq be \vee be \geq f(k+1)) \Leftrightarrow igaz.$  És igazra minden következik.

2.  $\forall i \in [1..n] : P \wedge \pi \wedge t = t_0 \wedge \pi_i \Rightarrow lf(S_i, Q'')$

(a)  $f(k+1) \leq le$ :  
 $(Q \wedge k \in [m..n-1] \wedge b \in [m..k] \wedge l \in [m..k] \wedge f(l) = le \wedge f(b) = be \wedge \forall i \in [m..k] : be \geq f(i) \geq le \wedge t = t_0 \wedge f(k+1) \leq le)$   
 $\xrightarrow{?}$   
 $lf(l, le := k+1, f(k+1), Q'') = (Q \wedge k+1 \in [m..n] \wedge b \in [m..k+1] \wedge k+1 \in [m..k+1] \wedge f(k+1) = k+1 \wedge f(b) = be \wedge \underbrace{\forall i \in [m..k+1] : be \geq f(i) \geq f(k+1)}_{\forall i \in [m..k] : be \geq f(i) \geq f(k+1)} \wedge t = t_0))$

✓

(b)  $f(k+1) \geq be$ : Hasonlóan

(c)  $le \leq f(k+1) \leq be$ :  
 $(Q \wedge k \in [m..n-1] \wedge b \in [m..k] \wedge l \in [m..k] \wedge f(l) = le \wedge f(b) = be \wedge \forall i \in [m..k] : be \geq f(i) \geq le \wedge t = t_0 \wedge le \leq f(k+1) \leq be)$   
 $\xrightarrow{?}$   
 $lf(SKIP, Q'') = (Q \wedge k+1 \in [m..n] \wedge b \in [m..k+1] \wedge l \in [m..k+1] \wedge f(l) = le \wedge f(b) = be \wedge \underbrace{\forall i \in [m..k+1] : be \geq f(i) \geq le}_{\forall i \in [m..k] : be \geq f(i) \geq le \wedge be \geq f(k+1) \geq le} \wedge t = t_0)$

$t_0$

✓

$k, b, l, be, le := m, m, m, f(m), f(m)$		
$k \neq n$		
$f(k+1) \leq le$	$f(k+1) \geq be$	$le \leq f(k+1) \leq be$
$l, le := k+1, f(k+1)$	$b, be := k+1, f(k+1)$	<i>SKIP</i>
$k := k+1$		

2.

*Specifikáció:*

$$A = \mathbb{N} \times \mathbb{N} \times \mathbb{N}$$

$$x \quad y \quad a$$

$$B = \mathbb{N} \times \mathbb{N}$$

$$Q : (x = x' \wedge y = y')$$

$$R : (Q \wedge a = \text{lnko}(x, y))$$

Megoldás:

A megoldás egy ciklus, melynek az invariánsa:

$$P = (Q \wedge a \in [1..x] \wedge b \in [1..y] \wedge \text{lnko}(a, b) = \text{lnko}(x, y))$$

Ellenőrizzük le a ciklus feltételeit:

$$1. Q \Rightarrow P$$

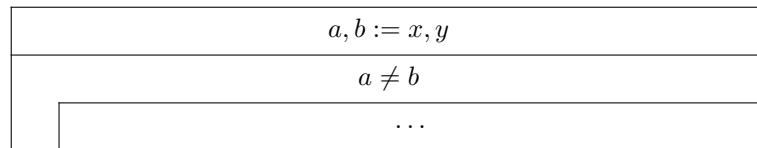
A szokásos módon bevezetjük az új  $Q'$  állapotot, amire már teljesül ez az állítás, továbbá megadjuk azt az  $S_1$  programot is, amivel  $Q \Rightarrow \text{lf}(S_1, Q')$ .

$$Q' = (Q \wedge a = x \wedge b = y)$$

$$S_1 = (a, b := x, y)$$

$$2. P \wedge \neg\pi \Rightarrow R$$

Ez a feltétel kezünkbe adja a ciklusfeltételt, hiszen  $P$  és  $R$  összehasonlításából  $\neg\pi$ -re  $a = b$  adódik (hiszen az  $a = b = \text{lnko}(a, b)$  csak így teljesülhet). Tehát  $\pi = (a \neq b)$ .

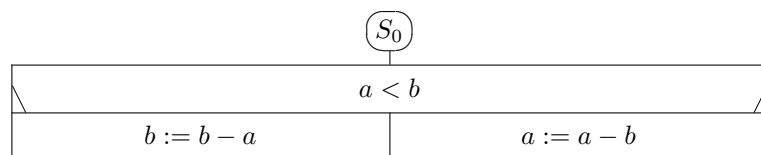


$$3. P \wedge \pi \Rightarrow t > 0$$

Jelen esetben ez:  $Q \wedge a \in [1..x] \wedge b \in [1..y] \wedge \text{lnko}(a, b) = \text{lnko}(x, y) \Rightarrow t > 0$ . Tehát  $t := a + b$  egy megfelelő terminálófüggvény.

$$5. P \wedge \pi \wedge t = t_0 \Rightarrow \text{lf}(S_0, t < t_0)$$

Ellenőrizzük, hogy a következő program biztosítja ezt a tulajdonságot:



$$5/1. P \wedge \pi \wedge t = t_0 \Rightarrow \left( \bigvee_{i=1}^n \pi_i \right)$$

Ez nyilván teljesül (sőt az is igaz, hogy a teljes állapotteret leírjuk).

$$5/2. \forall i \in [1..n] : P \wedge \pi \wedge t = t_0 \wedge \pi_i \Rightarrow lf(S_i, t < t_0)$$

$$(a) a < b: P \wedge \pi \wedge a + b = t_0 \stackrel{?}{\Rightarrow} lf(b := b - a, t < t_0) = a + b - a = b < t_0 \checkmark (\text{hiszen } a > 0)$$

$$(b) a \geq b: P \wedge \pi \wedge a + b = t_0 \stackrel{?}{\Rightarrow} lf(a := a - b, t < t_0) = a - b + b = a < t_0 \checkmark (\text{hiszen } b > 0)$$

$$4. P \wedge \pi \Rightarrow lf(S_0, P)$$

A mi esetünkben:

$$(Q \wedge a \in [1..x] \wedge b \in [1..y] \wedge lnko(a, b) = lnko(x, y) \wedge a \neq b) \stackrel{?}{\Rightarrow} lf(S_0, P)$$

$$4/1. P \wedge \pi \Rightarrow \left( \bigvee_{i=1}^n \pi_i \right) \checkmark$$

$$4/2. \forall i \in [1..n] : P \wedge \pi \wedge \pi_i \Rightarrow lf(S_i, P)$$

$$(a) a < b: P \wedge \pi \wedge a < b \stackrel{?}{\Rightarrow} lf(b := b - a, P) \\ (Q \wedge a \in [1..x] \wedge b \in [1..y] \wedge lnko(a, b) = lnko(x, y) \wedge a < b) \\ \stackrel{?}{\Rightarrow} \\ lf(b := b - a, P) = (Q \wedge a \in [1..x] \wedge b - a \in [1..y] \wedge lnko(a, b - a) = lnko(x, y)) \checkmark$$

$$(b) a \geq b: P \wedge \pi \wedge a > b \stackrel{?}{\Rightarrow} lf(a := a - b, P) \\ (Q \wedge a \in [1..x] \wedge b \in [1..y] \wedge lnko(a, b) = lnko(x, y) \wedge a > b) \\ \stackrel{?}{\Rightarrow} \\ lf(a := a - b, P) = (Q \wedge a - b \in [1..x] \wedge b \in [1..y] \wedge lnko(a - b, b) = lnko(x, y)) \checkmark$$

Ezen következtetések felismeréséhez szükséges az alábbi tétel:

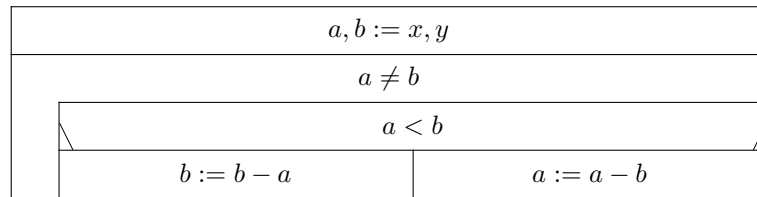
$$lnko(x, y) = q \Rightarrow lnko(x - y, x) = q, \text{ ha } x > y.$$

Bizonyítás:

$$\mathcal{C} := \{c \in \mathbb{N} : c|x \wedge c|y\} \Rightarrow \forall c \in \mathcal{C} : (\exists k, n \in \mathbb{N} : kc = x \wedge nc = y \wedge k > n \wedge (k - n)c = x - y) \Rightarrow \forall c \in \mathcal{C} : c|x - y$$

$$\text{Ugyanakkor } \mathcal{D} = \{d \in \mathbb{N} : d|x \wedge d|x - y\} \Rightarrow \forall c \in \mathcal{C} : d|y$$

Tehát  $\mathcal{C} = \mathcal{D} = \{e \in \mathbb{N} : e|x \wedge e|y \wedge e|x - y\}$ . De ha a közös osztók halmaza pontosan egyenlő (és véges), akkor a maximális elem (az lnko) is egyezik. Ezzel a tételt megmutattuk.



3.

*Specifikáció:*

$$A = \mathbb{N} \times \mathbb{N} \times \mathbb{N}$$

$$x \quad y \quad a$$

$$B = \mathbb{N} \times \mathbb{N}$$

$$x' \quad y'$$

$$Q : (x = x' \wedge y = y')$$

$$R : (Q \wedge a = lkkt(x, y))$$

A specifikációból kiolvasható, hogy az eredményt az állapotór  $a$  változója szolgáltatja.

*Megoldás:*

A specifikációnak megfelelő megoldó program egy szekvenciából fog állni, aminek első fele a már ismertett legkisebb közös osztó számító algoritmus (2. feladat). Ennek utófeltételét nevezzük  $R'$ -nek:

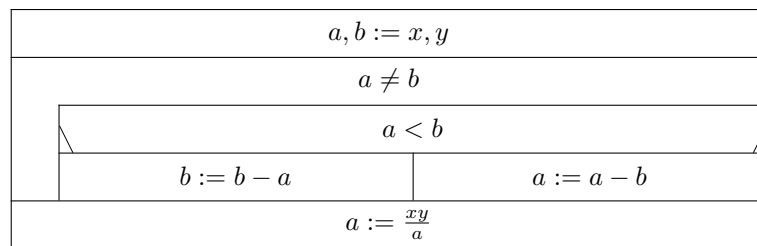
$$R' = (Q \wedge a = lnko(x, y))$$

Nézzük most a szekvencia levezetési szabályát:

$$(Q \Rightarrow lf(S_0, R') \wedge R' \Rightarrow lf(S_1, R)) \Rightarrow (Q \Rightarrow lf((S_0; S_1), R))$$

A feltétel rész első felét a 2. feladat bizonyítása igazolja, tehát mi csak  $R' \Rightarrow lf(S_1, R)$  állítást bizonyítjuk  $S_1 := (a := \frac{xy}{a})$  programra.

$lf(S_1, R) = (Q \wedge \frac{xy}{a} = lkkt(x, y))$ , ahol  $a = lnko(x, y)$  (figyelembe véve  $R'$ -t). Tehát a matematikából jól ismert tételre ( $lkkt(x, y) = \frac{xy}{lnko(x, y)}$ ) jutottunk.



4.

- 1. megoldás

*Specifikáció:*

$$A = \mathbb{Z} \times \mathbb{N} \times \mathbb{Z}$$

$$\quad \quad \quad x \quad \quad n \quad \quad r$$

$$B = \mathbb{Z} \times \mathbb{N}$$

$$\quad \quad \quad x' \quad \quad n'$$

$$Q : (x = x' \wedge n = n' \wedge n \neq 0)$$

$$R : (Q \wedge r = x^n)$$

*Megoldás:*

Most keressük meg a specifikációnak megfelelő megoldó programot! A megoldást úgy sejtjük, hogy egy ciklussal találhatjuk meg. Nos, mi legyen ennek a ciklusnak az invariánsa?

$$P = (Q \wedge k \in [1..n] \wedge r = x^k)$$

Ellenőrizzük le a ciklus feltételeit:

$$1. Q \Rightarrow P$$

$$Q' = (Q \wedge k = 1 \wedge r = x)$$

$$Q' \Rightarrow P \text{ és } Q \Rightarrow lf(k, r := 1, x, Q') = Q \wedge 1 = 1 \wedge x = x \Leftrightarrow Q.$$

$$2. P \wedge \neg\pi \Rightarrow R$$

$P$  és  $R$  összehasonlításából  $\neg\pi$ -re  $k = n$  adódik. Tehát  $\pi = (k \neq n)$ .

$k, r := 1, x$
$k \neq n$
$\dots$

$$3. P \wedge \pi \Rightarrow t > 0$$

Jelen esetben ez:  $Q \wedge k \in [1..n-1] \wedge r = x^k \Rightarrow t > 0$ . Tehát  $t := n - k$  egy megfelelő terminálófüggvény.

$$4./5. P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_0, P \wedge t < t_0)$$

A megoldást egy szekvencia adja, amely közbülső állapotának az  $lf(k := k + 1, P) \wedge t = t_0$  állítást választjuk.

$$Q'' = (Q \wedge k + 1 \in [1..n] \wedge r = x^{k+1} = x \cdot x^k \wedge t = t_0)$$

Ebbe az állapotba  $P \wedge t = t_0$ -ből az  $r := x \cdot r$  értékadással juthatunk, ugyanis  $P$  szerint  $r = x^k$ .

$$Q \wedge k \in [1..n-1] \wedge r = x^k \wedge t = t_0 \stackrel{?}{\Rightarrow} lf(r := x \cdot r, Q'') = Q \wedge k + 1 \in [1..n] \wedge x \cdot r = x \cdot x^k \wedge t = t_0 \checkmark$$

$$Q \wedge k + 1 \in [1..n] \wedge r = x^{k+1} = x \cdot x^k \wedge n - k = t_0 \stackrel{?}{\Rightarrow} lf(k :=$$

$$k + 1, P) \wedge t < t_0 = Q \wedge k + 1 \in [1..n] \wedge r = x^{k+1} = x \cdot x^k \wedge n - k - 1 < t_0 \checkmark$$

$k, r := 1, x$	
$k \neq n$	
$r := x \cdot r$	$k := k + 1$

• 2. megoldás

*Specifikáció:*

$$A = \mathbb{Z} \times \mathbb{N} \times \mathbb{Z}$$

$$x \quad n \quad r$$

$$B = \mathbb{Z} \times \mathbb{N}$$

$$x' \quad n'$$

$$Q : (x = x' \wedge n = n' \wedge n \neq 0)$$

$$R : (Q \wedge r = x^n)$$

*Megoldás:*

Az első megoldástól csupán a 4./5. pontban térünk el. Az új ciklusmag:

$2 \cdot k \leq n$	
$k, r := 2 \cdot k, r \cdot r$	$k, r := k + 1, r \cdot x$

Az, hogy ez az elágazás csökkenti a termináló függvény értékét trivialis.

$$4. P \wedge \pi \Rightarrow lf(S_0, P)$$

$$4/1. P \wedge \pi \Rightarrow \left( \bigvee_{i=1}^n \pi_i \right)$$

A fenti formájú egyszerűsített elágazásoknál ez mindig igaz, hiszen a második feltétel igazából az első pontos negáltja, így az állapotter összes pontjában igaz kettőjük valamelyike.

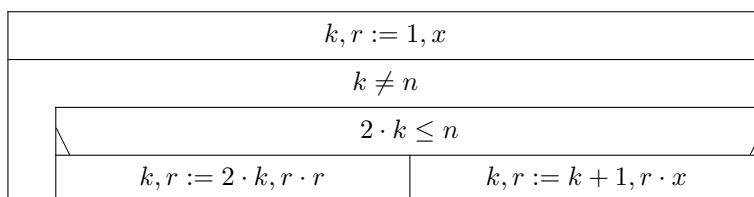
$$4/2. \forall i \in [1..n] : P \wedge \pi \wedge \pi_i \Rightarrow lf(S_i, P)$$

(a)  $k \cdot 2 \leq n$ :

$$Q \wedge k \in [1..n-1] \wedge r = x^k \wedge k \cdot 2 \leq n \stackrel{?}{\Rightarrow} Q \wedge 2 \cdot k \in [1..n] \wedge r \cdot r = x^{2 \cdot k} = x^k \cdot x^k \checkmark$$

(b)  $k \cdot 2 > n$ :

$$Q \wedge k \in [1..n-1] \wedge r = x^k \wedge k \cdot 2 > n \stackrel{?}{\Rightarrow} Q \wedge k + 1 \in [1..n] \wedge r \cdot x = x^{k+1} = x^k \cdot x \checkmark$$



Figyeljük meg, hogy ebben a második megoldásban ugyanahhoz a ciklusinvariánshoz írtunk egy máshogy működő ciklusmagot.

• 3. megoldás

*Specifikáció:*

$$A = \mathbb{Z} \times \mathbb{N} \times \mathbb{Z} \times \mathbb{Z}$$

$x \quad n \quad r \quad b$

$$B = \mathbb{Z} \times \mathbb{N}$$

$x' \quad n'$

$$Q : (x = x' \wedge n = n' \wedge n \neq 0)$$

$$R : (Q \wedge r = x^n)$$

*Megoldás:*

$$P = (Q \wedge k \in [0..n] \wedge x^n = r \cdot b^k)$$

Ellenőrizzük le a ciklus feltételeit:

1.  $Q \Rightarrow P$

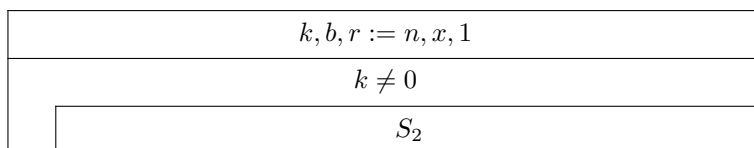
$$Q' = (Q \wedge k = n \wedge b = x \wedge r = 1)$$

$$Q' \Rightarrow P \text{ és } Q \Rightarrow lf(k, b, r := n, x, 1, Q') = Q.$$

2.  $P \wedge \neg\pi \Rightarrow R$

$P$  és  $R$  összehasonlításából  $\neg\pi$ -re  $k = 0$  adódik. Tehát  $\pi = (k \neq 0)$ .

(Megjegyzés: jelenleg  $b = 1$ -re is gondolhatnánk, de az a feladat későbbi részét tekintve nem lenne célszerű.)



3.  $P \wedge \pi \Rightarrow t > 0$

Jelen esetben ez:  $Q \wedge k \in [1..n] \wedge x^n = r \cdot b^k \Rightarrow t > 0$ . Tehát  $t := k$  egy megfelelő terminálófüggvény.

5.  $P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_0, t < t_0)$

Mivel  $t = k$ , a megoldás csak  $k$  csökkentése lehet. Nézzük az alábbi struktogramot  $S_2$ -ként:

$2 k$	
$k, b := k/2, b \cdot b$	$k, r := k - 1, r \cdot b$

Annak bizonyítását, hogy ez az elágazás mindig csökkenti  $k$  értékét, az olvasóra bízom.

4.  $P \wedge \pi \Rightarrow lf(S_0, P)$

Ezen állítás igazolásához az elágazás levezetési szabályának feltételeit kell ellenőriznünk:

4/1.  $P \wedge \pi \Rightarrow \left( \bigvee_{i=1}^n \pi_i \right)$

A fenti formájú egyszerűsített elágazásoknál ez mindig igaz, hiszen a második feltétel igazából az első pontos negáltja, így az állapotér összes pontjában igaz kettőjük valamelyike.

4/2.  $\forall i \in [1..n] : P \wedge \pi \wedge \pi_i \Rightarrow lf(S_i, P)$

(a)  $2|k$ :

$$Q \wedge k \in [1..n] \wedge x^n = r \cdot b^k \wedge 2|k \stackrel{?}{\Rightarrow} Q \wedge k/2 \in [0..n] \wedge x^n = r \cdot (b \cdot b)^{k/2} \sqrt{\quad}$$

(b)  $\neg(2|k)$ :

$$Q \wedge k \in [1..n] \wedge x^n = r \cdot b^k \wedge \neg(2|k) \stackrel{?}{\Rightarrow} Q \wedge k - 1 \in [0..n] \wedge x^n = r \cdot b \cdot b^{k-1} \sqrt{\quad}$$

$k, b, r := n, x, 1$	
$k \neq 0$	
$2 k$	
$k, b := k/2, b \cdot b$	$k, r := k - 1, r \cdot b$

7.

*Specifikáció:*

$$V = \text{vekt}(\mathcal{H}, \mathbb{Z})$$

$$A = V \times V \times \mathbb{Z}$$

$$x \quad y \quad r$$

$$B = V \times V$$

$$x' \quad y'$$

$$Q : (x = x' \wedge y = y' \wedge x.\text{dom} = y.\text{dom})$$

$$R : (Q \wedge$$

$$r = \sum_{i=0}^{x.\text{dom}-1} (x_{\text{succ}^i(x.\text{lob})} + y_{\text{succ}^i(y.\text{lob})}) \cdot (x_{\text{succ}^i(x.\text{lob})} - y_{\text{succ}^i(y.\text{lob})}))$$



*Megoldás:*

A megoldó ciklus invariánsa:

$$P = (Q \wedge k \in [0..x.dom])$$

$$\wedge r = \sum_{i=0}^{k-1} (x_{succ^i(x.lob)} + y_{succ^j(y.lob)}) \cdot (x_{succ^i(x.lob)} - y_{succ^j(y.lob)})$$

Ellenőrizzük le a ciklus feltételeit:

$$1. Q \Rightarrow P$$

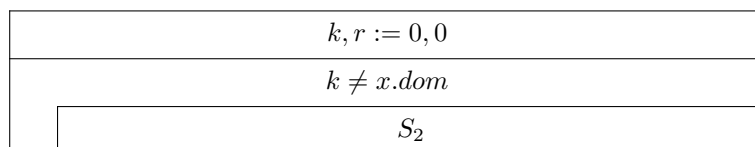
$$Q' = (Q \wedge k = 0 \wedge r = 0)$$

$$\text{Látható, hogy } Q' \Rightarrow P \text{ és } Q \Rightarrow lf(k, r := 0, 0, Q') = Q.$$

$$2. P \wedge \neg \pi \Rightarrow R$$

$$\pi = (k \neq x.dom).$$

A program eddig így néz ki:



$$3. P \wedge \pi \Rightarrow t > 0$$

Jelen esetben ez:  $(Q \wedge k \in [0..x.dom - 1] \wedge r = \dots) \Rightarrow t > 0$ , tehát a  $t = x.dom - k$  függvény egy megfelelő terminálófüggvény.

$$4./5. P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_0, P \wedge t < t_0)$$

Szekvencia legyen az  $S_2$ , közbülső állapot:

$$Q'' = (lf(k := k + 1, P) \wedge t = t_0) = (Q \wedge k + 1 \in [0..x.dom]) \wedge$$

$$\wedge r = \sum_{i=0}^k (x_{succ^i(x.lob)} + y_{succ^j(y.lob)}) \cdot (x_{succ^i(x.lob)} - y_{succ^j(y.lob)}) \wedge t = t_0$$

$$lf(P \wedge \pi \wedge t = t_0, r := r + (x_{succ^k(x.lob)} + y_{succ^k(y.lob)}) \cdot (x_{succ^k(x.lob)} - y_{succ^k(y.lob)})) \stackrel{?}{\Rightarrow} Q'' \checkmark$$

$$Q'' = (lf(k := k + 1, P) \wedge t = t_0 \stackrel{?}{\Rightarrow} lf(k := k + 1, P \wedge t < t_0) = lf(k := k + 1, P) \wedge \underbrace{lf(k := k + 1, t < t_0)}_{x.dom - k - 1 < t_0}) \checkmark$$

$k, r := 0, 0$
$k \neq x.dom$
$r := r + (x_{succ^k(x.lob)} + y_{succ^k(y.lob)}) \cdot (x_{succ^k(x.lob)} - y_{succ^k(y.lob)})$
$k := k + 1$

Megjegyzés: ha nem tekintjük megengedett műveletnek a programban a *succ* valamelyik hatványára való hivatkozást, akkor a kérdéses függvényeket helyettesíthetjük változóval, ekkor az alábbi programot kapjuk:

$xi, yi, r, k := pred(x.lob), pred(y.lob), 0, 0$
$k \neq x.dom$
$r := r + (x_{succ(xi)} + y_{succ(yi)}) \cdot (x_{succ(xi)} - y_{succ(yi)})$
$k, xi, yi := k + 1, succ(xi), succ(yi)$

Továbbá, ha a vektort (kevésbé általános módon)  $vekt(\mathbb{Z}, \mathbb{Z})$  alakúnak képzeljük, akkor a  $succ^i(x.lob)$  kifejezéshez hasonló formulák összeadásra, azaz  $x.lob + i$  alakúra egyszerűsödnek:

$k, r := 0, 0$
$k \neq x.dom$
$r := r + (x_{x.lob+k} + y_{y.lob+k}) \cdot (x_{x.lob+k} - y_{y.lob+k})$
$k := k + 1$

## 8.

*Specifikáció:*

$$V = vekt(\mathbb{Z}, \mathbb{Z})$$

$$A = V \times \mathbb{Z}$$

$$x \quad r$$

$$B = V$$

$$x'$$

$$Q : (x = x')$$

$$R : \left( Q \wedge r = \sum_{i=x.lob}^{x.hib} (-1)^i \cdot x_i \right)$$

*Megoldás:*

A megoldáshoz tartozó ciklusinvariáns:

$$P = \left( Q \wedge k \in [x.lob - 1..x.hib] \wedge r = \sum_{i=x.lob}^k (-1)^i \cdot x_i \right)$$

Ellenőrizzük le a ciklus feltételeit:

1.  $Q \Rightarrow P$

$$Q' = (Q \wedge k = x.lob - 1 \wedge r = 0)$$

$$Q' \Rightarrow P \text{ és } Q \Rightarrow lf(k, r := x.lob - 1, 0, Q') = Q.$$

2.  $P \wedge \neg\pi \Rightarrow R$

$$\pi = (k \neq x.hib).$$

$k, r := x.lob - 1, 0$
$k \neq x.hib$
$S_2$

3.  $P \wedge \pi \Rightarrow t > 0$

Jelen esetben ez:  $(Q \wedge k \in [x.lob - 1..x.hib - 1] \wedge r = \dots) \stackrel{?}{\Rightarrow} t > 0$ , tehát a termináló függvény:  $t := x.hib - k$ . Ez a függvény  $P \wedge \pi$  esetén nyilván pozitív.

4/5.  $P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_0, P \wedge t < t_0)$

A megoldást szekvenciaként keressük, aminek második utasításaként a  $k := k + 1$  tűnik praktikusnak, így a  $Q''$  feltételt az  $lf(k := k + 1, P) \wedge t = t_0$  alakban írjuk fel:

$$Q'' = \left( Q \wedge k + 1 \in [x.lob - 1..x.hib] \wedge r = \sum_{i=x.lob}^{k+1} (-1)^i \cdot x_i \wedge t = t_0 \right)$$

$Q''$ -t pedig az  $S_{21} := (r := r + (-1)^{k+1} \cdot x_{k+1})$  értékadással érjük el a szekvencia első részében.

$$P \wedge \pi \wedge t = t_0 = (Q \wedge k \in [x.lob - 1..x.hib - 1] \wedge r = \sum_{i=x.lob}^k (-1)^i \cdot x_i \wedge t = t_0) \stackrel{?}{\Rightarrow} lf(S_{21}, Q'') = (Q \wedge k + 1 \in [x.lob - 1..x.hib] \wedge r + (-1)^{k+1} \cdot x_{k+1} = \sum_{i=x.lob}^{k+1} (-1)^i \cdot x_i \wedge t = t_0) \checkmark$$

$$Q'' = lf(k := k + 1, P) \wedge t = t_0 \stackrel{?}{\Rightarrow} lf(k := k + 1, P \wedge t < t_0) = (lf(k := k + 1, P) \wedge lf(k := k + 1, t < t_0)) \checkmark$$

$k, r := x.lob - 1, 0$
$k \neq x.hib$
$r := r + ((-1)^{k+1} \cdot x_{k+1})$
$k := k + 1$

9.

*Specifikáció:* $V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$ 

$$A = V \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{L} \times \mathbb{L} \times \{1, 2\} \times \mathbb{Z}$$

$$v \quad z_1 \quad z_2 \quad l_1 \quad l_2 \quad r \quad rsz$$

$$B = V \times \mathbb{Z} \times \mathbb{Z}$$

$$v' \quad z'_1 \quad z'_2$$
 $Q : (v = v' \wedge z_1 = z'_1 \wedge z_2 = z'_2 \wedge z_2 \neq z_1)$ 

$$R : (Q \wedge \forall i \in [1..2] : l_i = (\exists j \in [v.lob..v.hib] : z_i = v_j) \wedge (l_1 \wedge l_2) \rightarrow (r = 1 \leftrightarrow g(z_1) < g(z_2)) \wedge (l_1 \wedge \neg l_2) \rightarrow r = 1 \wedge (l_2 \wedge \neg l_1) \rightarrow r = 2 \wedge (l_1 \vee l_2) \rightarrow rsz = z_r), \text{ ahol}$$

$$\mathcal{D}_g = \{x \in \mathbb{Z} \mid \exists i \in [v.lob..v.hib] : v_i = x\}, \forall x \in \mathcal{D}_g : g(x) := i, \text{ ha}$$

$$i \in [v.lob..v.hib] \wedge v_i = x \wedge \forall j \in [v.lob..i - 1] : v_j \neq x$$

Informálisan:  $g(x)$  első előfordulásának indexét adja meg a vektorban. A specifikáció tehát azt írja le, hogy a két logikai változó külön-külön mutatja a program lefutása végén, hogy előfordulnak-e az adott egész számok. Ha mindkettő előfordul, akkor a korábban előforduló számot tartalmazza  $rsz$  és a hozzá tartozó „indexet”  $r$ . Amennyiben csak az egyik szám fordul elő, akkor az  $rsz$  ezt a számot, míg az  $r$  az „indexét” tartalmazza.

*Megoldás:*

Használjuk az intervallumon megadott programozási tételeknél megismert módszert azzal a megjegyzéssel, hogy most  $m - 1, n$  szerepét  $v.lob - 1, v.hib$  fogja átvenni.

$$\varphi(m, n, z_1, z_2, l_1, l_2, r, rsz) = (\forall i \in [1..2] : l_i = (\exists j \in [m, n] : z_i = v_j) \wedge (l_1 \wedge l_2) \rightarrow (r = 1 \leftrightarrow g(z_1) < g(z_2)) \wedge (l_1 \wedge \neg l_2) \rightarrow r = 1 \wedge (l_2 \wedge \neg l_1) \rightarrow r = 2 \wedge (l_1 \vee l_2) \rightarrow rsz = z_r)$$

Ennek felhasználásával:

 $R = (Q \wedge \varphi(v.lob, v.hib, z_1, z_2, l_1, l_2, r, rsz))$ 

$$P = (Q \wedge k \in [v.lob - 1..v.hib] \wedge \varphi(v.lob, k, \dots)) = (Q \wedge k \in [v.lob - 1..v.hib] \wedge \forall i \in [1..2] : l_i = (\exists j \in [v.lob..k] : z_i = v_j) \wedge (l_1 \wedge l_2) \rightarrow (r = 1 \leftrightarrow g(z_1) < g(z_2)) \wedge (l_1 \wedge \neg l_2) \rightarrow r = 1 \wedge (l_2 \wedge \neg l_1) \rightarrow r = 2 \wedge (l_1 \vee l_2) \rightarrow rsz = z_r)$$

$$Q' = (Q \wedge k = v.lob - 1 \wedge \varphi(v.lob, k, \dots)) = (Q \wedge k = v.lob - 1 \wedge l_1 = \text{hamis} \wedge l_2 = \text{hamis})$$

$$Q'' = (lf(k := k + 1, P) \wedge t = t_0) = (Q \wedge k + 1 \in [v.lob - 1..v.hib] \wedge \forall i \in [1..2] : l_i = (\exists j \in [v.lob..k + 1] : z_i = v_j) \wedge (l_1 \wedge l_2) \rightarrow (r = 1 \leftrightarrow g(z_1) < g(z_2)) \wedge (l_1 \wedge \neg l_2) \rightarrow r = 1 \wedge (l_2 \wedge \neg l_1) \rightarrow r = 2 \wedge (l_1 \vee l_2) \rightarrow rsz = z_r \wedge t = t_0)$$

A kezdeti értékadás  $k, l_1, l_2 := v.lob - 1, hamis, hamis$ .

$k, l_1, l_2 := v.lob - 1, hamis, hamis$
$k \neq v.hib$
$\dots$
$k := k + 1$

A ciklus mag első részprogramjának (a ciklusváltozó növelése előtt) egy olyan  $S_2$  elágazásnak kell lennie, ami teljesíti  $P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_2, Q'')$  feltételt.

Technikai okokból a szokásos stuktogram formától eltérünk:

- $l_1 \wedge l_2$ , akkor *SKIP*,
- $z_1 \neq v_{k+1} \wedge z_2 \neq v_{k+1}$ , akkor *SKIP*,
- $\neg l_1 \wedge \neg l_2 \wedge v_{k+1} = z_1$ , akkor  $l_1, r, rsz := igaz, 1, z_1$ ,
- $\neg l_1 \wedge \neg l_2 \wedge v_{k+1} = z_2$ , akkor  $l_2, r, rsz := igaz, 2, z_2$ ,
- $\neg l_1 \wedge l_2 \wedge v_{k+1} = z_1$ , akkor  $l_1 := igaz$ ,
- $\neg l_1 \wedge l_2 \wedge v_{k+1} \neq z_1$ , akkor *SKIP*,
- $\neg l_2 \wedge l_1 \wedge v_{k+1} = z_2$ , akkor  $l_2 := igaz$ ,
- $\neg l_2 \wedge l_1 \wedge v_{k+1} \neq z_2$ , akkor *SKIP*.

$$1. P \wedge \pi \wedge t = t_0 \Rightarrow \left( \bigvee_{i=1}^n \pi_i \right)$$

A feltételek közös diszjunkciójának mindig igaz.

$$2. \forall i \in [1..n] : P \wedge \pi \wedge t = t_0 \wedge \pi_i \Rightarrow lf(S_i, Q'')$$

(a)  $l_1 \wedge l_2$

$$\begin{aligned} & Q \wedge k \in [v.lob - 1..v.hib - 1] \wedge \forall i \in [1..2] : l_i = (\exists j \in [v.lob..k] : z_i = v_j) \wedge (l_1 \wedge l_2) \rightarrow (r = 1 \leftrightarrow g(z_1) < g(z_2)) \wedge (l_1 \wedge \neg l_2) \rightarrow r = 1 \wedge (l_2 \wedge \neg l_1) \rightarrow r = 2 \wedge (l_1 \vee l_2) \rightarrow rsz = z_r \wedge (l_1 \wedge l_2) \wedge t = t_0 \\ & \stackrel{?}{\Rightarrow} lf(SKIP, Q'') = (Q \wedge k + 1 \in [v.lob - 1..v.hib] \wedge \forall i \in [1..2] : l_i = (\exists j \in [v.lob..k + 1] : z_i = v_j) \wedge (l_1 \wedge l_2) \rightarrow (r = 1 \leftrightarrow g(z_1) < g(z_2)) \wedge (l_1 \wedge \neg l_2) \rightarrow r = 1 \wedge (l_2 \wedge \neg l_1) \rightarrow r = 2 \wedge (l_1 \vee l_2) \rightarrow rsz = z_r \wedge t = t_0) \checkmark \end{aligned}$$

- (b)  $z_1 \neq v_{k+1} \wedge z_2 \neq v_{k+1}$   
 $Q \wedge k \in [v.lob - 1..v.hib - 1] \wedge \forall i \in [1..2] : l_i = (\exists j \in [v.lob..k] : z_i = v_j) \wedge (l_1 \wedge l_2) \rightarrow (r = 1 \leftrightarrow g(z_1) < g(z_2)) \wedge (l_1 \wedge \neg l_2) \rightarrow r = 1 \wedge (l_2 \wedge \neg l_1) \rightarrow r = 2 \wedge (l_1 \vee l_2) \rightarrow rsz = z_r \wedge z_1 \neq v_{k+1} \wedge z_2 \neq v_{k+1} \wedge t = t_0$   
 $\stackrel{?}{\Rightarrow} lf(SKIP, Q'') = (Q \wedge k + 1 \in [v.lob - 1..v.hib] \wedge \forall i \in [1..2] : l_i = (\exists j \in [v.lob..k + 1] : z_i = v_j) \wedge (l_1 \wedge l_2) \rightarrow (r = 1 \leftrightarrow g(z_1) < g(z_2)) \wedge (l_1 \wedge \neg l_2) \rightarrow r = 1 \wedge (l_2 \wedge \neg l_1) \rightarrow r = 2 \wedge (l_1 \vee l_2) \rightarrow rsz = z_r \wedge t = t_0) \checkmark$
- (c)  $\neg l_1 \wedge \neg l_2 \wedge v_{k+1} = z_1$   
 $Q \wedge k \in [v.lob - 1..v.hib - 1] \wedge \forall i \in [1..2] : l_i = (\exists j \in [v.lob..k] : z_i = v_j) \wedge (l_1 \wedge l_2) \rightarrow (r = 1 \leftrightarrow g(z_1) < g(z_2)) \wedge (l_1 \wedge \neg l_2) \rightarrow r = 1 \wedge (l_2 \wedge \neg l_1) \rightarrow r = 2 \wedge (l_1 \vee l_2) \rightarrow rsz = z_r \wedge v_{k+1} = z_1 \wedge \neg l_1 \wedge \neg l_2 \wedge t = t_0$   
 $\stackrel{?}{\Rightarrow} lf(l_1, r, rsz := igaz, 1, z_1, Q'') = (Q \wedge k + 1 \in [v.lob - 1..v.hib] \wedge igaz = (\exists j \in [v.lob..k + 1] : z_1 = v_j) \wedge l_2 = (\exists j \in [v.lob..k + 1] : z_2 = v_j) \wedge (igaz \wedge l_2) \rightarrow (1 = 1 \leftrightarrow g(z_1) < g(z_2)) \wedge (igaz \wedge \neg l_2) \rightarrow 1 = 1 \wedge (l_2 \wedge \neg igaz) \rightarrow 1 = 2 \wedge (igaz \vee l_2) \rightarrow z_1 = z_1 \wedge t = t_0) \checkmark$
- (d)  $\neg l_1 \wedge \neg l_2 \wedge v_{k+1} = z_2$   
 hasonlón
- (e)  $\neg l_1 \wedge l_2 \wedge v_{k+1} = z_1$   
 $Q \wedge k \in [v.lob - 1..v.hib - 1] \wedge \forall i \in [1..2] : l_i = (\exists j \in [v.lob..k] : z_i = v_j) \wedge (l_1 \wedge l_2) \rightarrow (r = 1 \leftrightarrow g(z_1) < g(z_2)) \wedge (l_1 \wedge \neg l_2) \rightarrow r = 1 \wedge (l_2 \wedge \neg l_1) \rightarrow r = 2 \wedge (l_1 \vee l_2) \rightarrow rsz = z_r \wedge z_1 = v_{k+1} \wedge l_2 \wedge \neg l_1 \wedge t = t_0$   
 $\stackrel{?}{\Rightarrow} lf(l_1 := igaz, Q'') = (Q \wedge k + 1 \in [v.lob - 1..v.hib] \wedge igaz = (\exists j \in [v.lob..k + 1] : z_1 = v_j) \wedge l_2 = (\exists j \in [v.lob..k + 1] : z_2 = v_j) \wedge (igaz \wedge l_2) \rightarrow (r = 1 \leftrightarrow g(z_1) < g(z_2)) \wedge (igaz \wedge \neg l_2) \rightarrow r = 1 \wedge (l_2 \wedge \neg igaz) \rightarrow r = 2 \wedge (igaz \vee l_2) \rightarrow rsz = z_r \wedge t = t_0) \checkmark$
- (f)  $\neg l_1 \wedge l_2 \wedge v_{k+1} \neq z_1$   
 $Q \wedge k \in [v.lob - 1..v.hib - 1] \wedge \forall i \in [1..2] : l_i = (\exists j \in [v.lob..k] : z_i = v_j) \wedge (l_1 \wedge l_2) \rightarrow (r = 1 \leftrightarrow g(z_1) < g(z_2)) \wedge (l_1 \wedge \neg l_2) \rightarrow r = 1 \wedge (l_2 \wedge \neg l_1) \rightarrow r = 2 \wedge (l_1 \vee l_2) \rightarrow rsz = z_r \wedge z_1 \neq v_{k+1} \wedge l_2 \wedge \neg l_1 \wedge t = t_0$   
 $\stackrel{?}{\Rightarrow} lf(SKIP, Q'') = (Q \wedge k + 1 \in [v.lob - 1..v.hib] \wedge \forall i \in [1..2] : l_i = (\exists j \in [v.lob..k + 1] : z_i = v_j) \wedge (l_1 \wedge l_2) \rightarrow (r = 1 \leftrightarrow g(z_1) < g(z_2)) \wedge (l_1 \wedge \neg l_2) \rightarrow r = 1 \wedge (l_2 \wedge \neg l_1) \rightarrow r = 2 \wedge (l_1 \vee l_2) \rightarrow rsz = z_r \wedge t = t_0) \checkmark$
- (g)  $\neg l_2 \wedge l_1 \wedge v_{k+1} = z_2$   
 hasonlón, mint (e).

- (h)  $\neg l_2 \wedge l_1 \wedge v_{k+1} \neq z_2$   
 hasonlóan, mint (f).

## 10.

Először egy kicsit egyszerűbb feladatot oldunk meg: a vektor egy konkrét elemét visszük a helyére.

*Megjegyzés:* Ezt a feladatot és első megoldását eredetileg Hoare vezette be, a gyorsrendezés részeként. Ő az alábbi megoldást adta (aminek helyességét azonban nem bizonyítjuk, csak egy állapottér magyarázó táblázattal szolgálunk):

A	a rendezendő vektor
i	balról lépkedő mutató
j	jobbról lépkedő mutató (a végén mutatja a pivot elem új indexét)
p	a vektor alsó indexe, egyben a pivot elem indexe
r	a vektor felső indexe

$x, i, j := A[p], p + 1, r$	
$i < j$	
$A[i] \leq x \wedge i < j$	
$i := i + 1$	
$A[j] \geq x \wedge i \leq j$	
$j := j - 1$	
$i < j$	
$A[i], A[j] := A[j], A[i]$	<i>SKIP</i>
$i, j := i + 1, j - 1$	
$A[j], A[p] := A[p], A[j]$	

Mi egy másik megoldást, nevezetesen Lomu megoldását vesszük alaposan szemügyre. Ez annyiban is eltér Hoare megoldásától, hogy az első helyett az utolsó elemet teszi helyre.

*Specifikáció:*

$$A = V \times \mathbb{Z}$$

$$x \quad i$$

$$V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$$

$$B = V$$

$$x'$$

$$Q : (x = x')$$

$R : (x \in \text{perm}(x') \wedge i \in [x.\text{lob}..x.\text{hib}] \wedge x[i] = x'.\text{hiv} \wedge \forall k \in [x.\text{lob}..i-1] : x_k \leq x_i \wedge \forall k \in [i+1..x.\text{hib}] : x_k > x_i)$

$x \in \text{perm}(x')$  azt írja le, hogy az  $x$  vektor elemei ugyanazok, mint az  $x'$ -é, csupán a sorrendjük különbözik.

*Megoldás:*

A megoldást könnyedén (egy megengedett szimultán értékadással elérhetnénk), ha már teljesülne  $R'$  állapot:

$R' = (x \in \text{perm}(x') \wedge i \in [x.\text{lob} - 1..x.\text{hib} - 1] \wedge x.\text{hiv} = x'.\text{hiv} \wedge \forall k \in [x.\text{lob}..i] : x_k \leq x.\text{hiv} \wedge \forall k \in [i+1..x.\text{hib} - 1] : x_k > x.\text{hiv})$

Tűzzük ki tehát új utófeltételnek  $R'$ -t és ha oda eljutunk, akkor a szekvencia levezetési szabályát alkalmazzuk majd a meglévő programra és az  $R$ -be vezető  $i, x_{i+1}, x.\text{hiv} := i+1, x.\text{hiv}, x_{i+1}$  értékadásra, hogy a teljes megoldó programot kapjuk.

Most írjuk fel az új utófeltételhez elvezető ciklus invariánsát:  $P = (x \in \text{perm}(x') \wedge j \in [x.\text{lob}..x.\text{hib}] \wedge i \in [x.\text{lob} - 1..j - 1] \wedge x.\text{hiv} = x'.\text{hiv} \wedge \forall k \in [x.\text{lob}..i] : x_k \leq x.\text{hiv} \wedge \forall k \in [i+1..j - 1] : x_k > x.\text{hiv})$

Ellenőrizzük le a ciklus feltételeit:

1.  $Q \Rightarrow P$

$Q' = (Q \wedge i = x.\text{lob} - 1 \wedge j = x.\text{lob})$

Látható, hogy  $Q' \Rightarrow P$  és  $Q \Rightarrow lf(i, j := x.\text{lob} - 1, x.\text{lob}, Q') = Q$ .

2.  $P \wedge \neg\pi \Rightarrow R'$

Ebből a pontból kapjuk meg a ciklusfeltételt, hiszen  $P$  és  $R'$  összehasonlításából  $\neg\pi$ -re  $x.\text{hib} = j$  adódik. Tehát  $\pi = (j \neq x.\text{hib})$ .

$i, j := x.\text{lob} - 1, x.\text{lob}$
$j \neq x.\text{hib}$
$\dots$
$i, x_{i+1}, x.\text{hiv} := i+1, x.\text{hiv}, x_{i+1}$

3.  $P \wedge \pi \Rightarrow t > 0$

Jelen esetben ez:  $j \in [x.\text{lob}..x.\text{hib} - 1] \wedge \dots \Rightarrow t > 0$ . Tehát  $t := x.\text{hib} - j$  egy megfelelő terminálófüggvény.

4.  $P \wedge \pi \Rightarrow lf(S_0, P)$

Nézzük az alábbi struktogramot  $S_0$ -ként:



$x_j \leq x.hiv$	
$i, j, x_{i+1}, x_j :=$ $i + 1, j + 1, x_j, x_{i+1}$	$j := j + 1$

Annak bizonyítását, hogy ez az elágazás mindig növeli  $j$  értékét (és így csökkenti a termináló függvényt), az olvasóra bízom.

4.  $P \wedge \pi \Rightarrow lf(S_0, P)$

Ezen állítás igazolásához az elágazás levezetési szabályának feltételeit kell ellenőriznünk:

$$4/1. P \wedge \pi \Rightarrow \left( \bigvee_{i=1}^n \pi_i \right) \checkmark$$

4/2.  $\forall i \in [1..n] : P \wedge \pi \wedge \pi_i \Rightarrow lf(S_i, P)$

(a)  $x_j \leq x.hiv$ :

$$x \in perm(x') \wedge j \in [x.lob..x.hib - 1] \wedge i \in [x.lob - 1..j - 1] \wedge x.hiv = x'.hiv \wedge \forall k \in [x.lob..i] : x_k \leq x.hiv \wedge \forall k \in [i + 1..j - 1] : x_k > x.hiv \wedge x_j \leq x.hiv$$

$\stackrel{?}{\Rightarrow}$

$$lf(i, j, x_{i+1}, x_j := i + 1, j + 1, x_j, x_{i+1}, P) = (y \in perm(x') \wedge j + 1 \in [y.lob..y.hib] \wedge i + 1 \in [y.lob - 1..j] \wedge y.hiv = x'.hiv \wedge \forall k \in [y.lob..i + 1] : y_k \leq y.hiv \wedge \forall k \in [i + 2..j] : y_k > y.hiv, ahol y.hib = x.hib \wedge y.lob = x.lob \wedge \forall h \in [x.lob, x.hib] \setminus \{i + 1, j\} : y_h = x_h \wedge y_{i+1} = x_j \wedge y_j = x_{i+1}. \checkmark$$

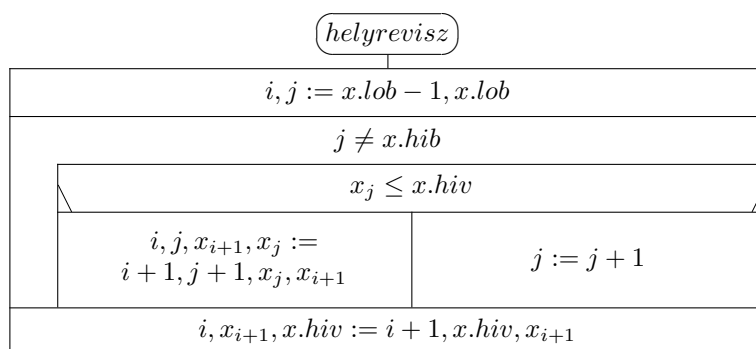
(b)  $x_j > x.hiv$ :

$$x \in perm(x') \wedge j \in [x.lob..x.hib - 1] \wedge i \in [x.lob - 1..j - 1] \wedge x.hiv = x'.hiv \wedge \forall k \in [x.lob..i] : x_k \leq x.hiv \wedge \forall k \in [i + 1..j - 1] : x_k > x.hiv \wedge x_j > x.hiv$$

$\stackrel{?}{\Rightarrow}$

$$lf(j := j + 1, P) = (x \in perm(x') \wedge j + 1 \in [x.lob..x.hib] \wedge i \in [x.lob - 1..j] \wedge x.hiv = x'.hiv \wedge \forall k \in [x.lob..i] : x_k \leq x.hiv \wedge \forall k \in [i + 1..j] : x_k > x.hiv) \checkmark$$

Ezzel megkaptuk a szakmában helyrevisznek becézett feladat megoldását (ez volt a mi egyszerűsített feladatunk).



Ám nekünk az volt a feladatunk, hogy egy adott elemet vigyünk a helyére. Egészítsük ki a specifikációt ennek megfelelően:

$$A = V \times \mathbb{Z} \times \mathbb{Z}$$

$$\begin{array}{ccc} x & n & i \end{array}$$

$$V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$$

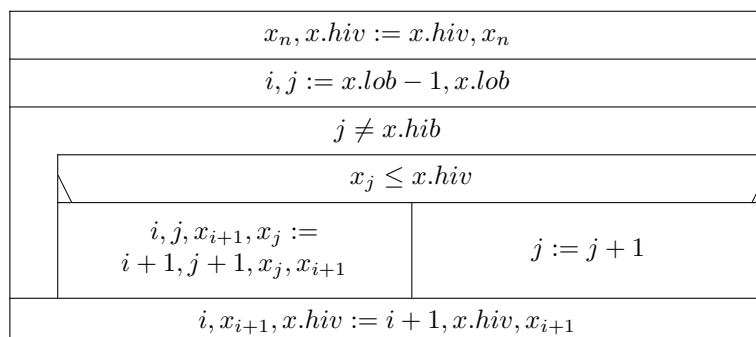
$$B = \mathbb{Z} \times \mathbb{Z}$$

$$\begin{array}{cc} x' & n' \end{array}$$

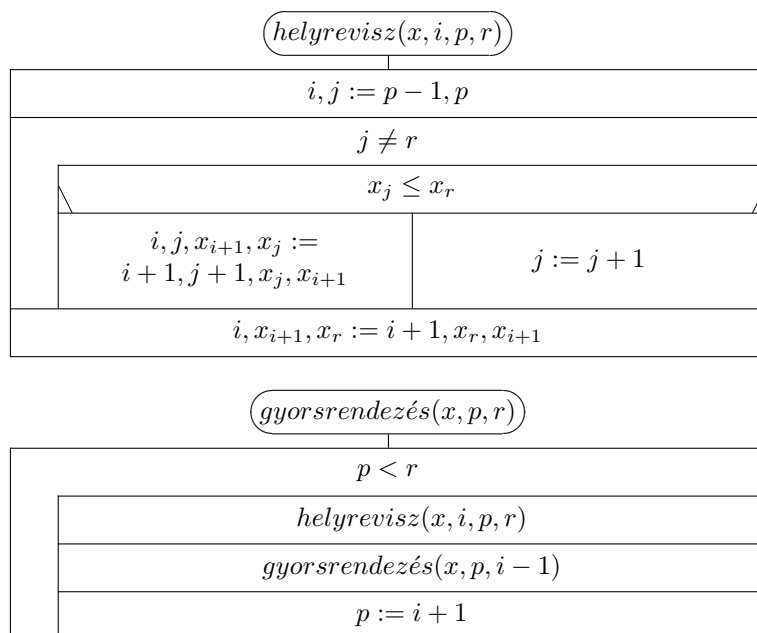
$$Q : (x = x' \wedge n = n' \wedge n \in [x.lob..x.hib])$$

$$R : (n = n' \wedge x \in \text{perm}(x') \wedge i \in [x.lob..x.hib] \wedge x_i = x'_i \wedge \forall k \in [x.lob..i - 1] : x_k \leq x_i \wedge \forall k \in [i + 1..x.hib] : x_k > x_i)$$

A szekvencia levezetési szabályával bizonyítható, hogy ezt a feladatot az alábbi program oldja meg:



A most megismert algoritmus legnagyobb jelentősége, hogy a jelenleg ismert leggyorsabb (és egyben „leghóbortosabb”) rendező algoritmus szerves része. Az érdekesség kedvéért itt közöljük egy olyan verzióját ennek a rendezőnek, ami a modell kereteibe igazából (például a rekurzió miatt) nem illik, de intuitíven megérthető. A rendező mély tárgyalását az Algoritmusok és adatszerkezetek című tárgy keretei között hallgathatjuk meg.

**11.**

*Specifikáció:*

$M = \text{vekt}(\mathbb{Z}, \text{vekt}(\mathbb{Z}, \mathbb{Z}))$ , invariánsa:  $I(x) = (\forall i \in [x.lob, x.hib] : x_i.lob = x.lob \wedge x_i.hib = x.hib)$

$$A = \begin{matrix} M & \times & \mathbb{Z} \\ t & & r \end{matrix}$$

$$B = \begin{matrix} M \\ t' \end{matrix}$$

$$Q : (t = t')$$

$$R : \left( Q \wedge r = \sum_{i=t.lob}^{t.hib} \sum_{j=t.lob}^i t_{ij} \right)$$

*Megoldás:*

A megoldó ciklus invariánsa:

$$P = \left( Q \wedge k \in [t.lob - 1..t.hib] \wedge r = \sum_{i=t.lob}^k \sum_{j=t.lob}^i t_{ij} \right)$$

Ellenőrizzük le a ciklus feltételeit:

$$1. Q \Rightarrow P$$

$$Q' = (Q \wedge k = t.lob - 1 \wedge r = 0)$$

Látható, hogy  $Q' \Rightarrow P$  és  $Q \Rightarrow lf(k, r := t.lob - 1, 0, Q') = Q$ .

2.  $P \wedge \neg\pi \Rightarrow R$

$P$  és  $R$  összehasonlításából  $\neg\pi$ -re  $k = t.hib$  adódik. Tehát  $\pi = (k \neq t.hib)$ .

$k, r := t.lob - 1, 0$
$k \neq t.hib$
$\dots$

3.  $P \wedge \pi \Rightarrow t > 0$

Jelen esetben ez:  $(Q \wedge k \in [t.lob - 1..t.hib - 1] \wedge r = \dots) \stackrel{?}{\Rightarrow} t > 0$ , tehát a termináló függvény:  $t := t.hib - k$ . Ez a függvény  $P \wedge \pi$  esetén nyilván pozitív.

4./5.  $P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_0, P \wedge t < t_0)$

Ezt a pontot a szokásos alakú ciklusmaggal biztosítjuk, ami egy szekvencia. A szekvencia második utasítása a ciklusváltozót növeli.

$Q'' = (lf(k := k + 1, P) \wedge t = t_0) =$

$$\left( Q \wedge k + 1 \in [t.lob - 1..t.hib] \wedge r = \sum_{i=t.lob}^{k+1} \sum_{j=t.lob}^i t_{ij} \right) \wedge t = t_0$$

A szekvencia első része a  $P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_{01}, Q'')$  feltételt kell teljesítse! Nézzük közelebbről ezt a feltételt ( $S_{01}$ -et egyelőre SKIP-nek tekintve):

$$\begin{aligned} & \left( Q \wedge k \in [t.lob - 1..t.hib - 1] \wedge r = \sum_{i=t.lob}^k \sum_{j=t.lob}^i t_{ij} \right) \wedge t = t_0 \stackrel{?}{\Rightarrow} \\ & \left( Q \wedge k + 1 \in [t.lob - 1..t.hib] \wedge r = \sum_{i=t.lob}^{k+1} \sum_{j=t.lob}^i t_{ij} \right) \wedge t = t_0 \Leftrightarrow \\ & \Leftrightarrow \left( Q \wedge k + 1 \in [t.lob - 1..t.hib] \wedge r = \left( \sum_{i=t.lob}^k \sum_{j=t.lob}^i t_{ij} \right) + \sum_{j=t.lob}^{k+1} t_{k+1j} \right) \wedge \\ & t = t_0 \end{aligned}$$

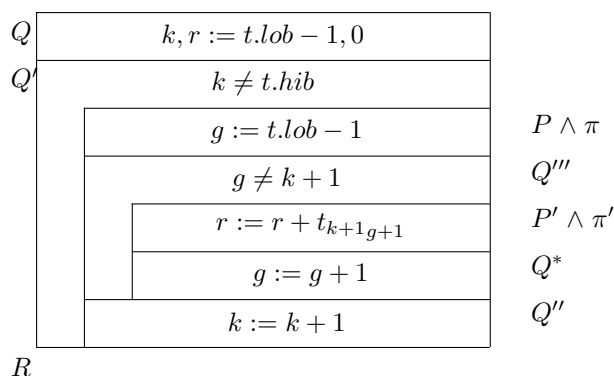
$S_{01}$  nem lehet egyetlen értékadás, mert akkor a nem megengedett szummát kellene használni a jobb oldalán. A  $P \wedge \pi \wedge t = t_0$  és a  $Q''$  állapot összekötésére egy ciklus alkalmas. Az invariánsát úgy kapjuk, hogy a  $Q''$ -ben a második szumma hatáskörét csökkentjük:

$$\begin{aligned} P' &= (Q \wedge k + 1 \in [t.lob - 1..t.hib] \wedge g \in [t.lob - 1..k + 1] \wedge r = \\ & \left( \sum_{i=t.lob}^k \sum_{j=t.lob}^i t_{ij} \right) + \sum_{j=t.lob}^g t_{k+1j} \wedge t = t_0) \end{aligned}$$

A belső ciklus levezetéséhez szükséges további állapotok:

$$Q''' = (Q \wedge k + 1 \in [t.lob - 1..t.hib] \wedge g = t.lob - 1 \wedge$$

$$\begin{aligned}
r &= \left( \sum_{i=t.lob}^k \sum_{j=t.lob}^i t_{ij} \right) \wedge t = t_0, \\
Q^* &= (Q \wedge k + 1 \in [t.lob - 1..t.hib] \wedge g + 1 \in [t.lob - 1, k + 1] \wedge r = \\
&\left( \sum_{t.lob=1}^k \sum_{j=t.lob}^i t_{ij} \right) + \sum_{j=t.lob}^{g+1} t_{k+1j}).
\end{aligned}$$



16.

$$V = \text{vekt}(\mathbb{Z}, \{0, 1, \dots, 9\})$$

A feladat megoldása során használni fogjuk a következő két függvényt:

$$f(x) ::= \sum_{i=0}^{x.dom-1} x_{x.hib-i} \cdot 10^i \quad (\forall x \in V)$$

$$g(x, l) ::= \sum_{i=0}^{l-1} x_{x.lob+l-i-1} \cdot 10^i \quad (\forall x \in V : \forall l \in [0, x.dom])$$

Az  $f$  függvény egy vektorban ábrázolt szám értékét számítja ki, míg  $g$  ugyan-  
ezt teszi, de csak az első  $l$  számjegyet veszi figyelembe. Vegyük észre, hogy  
 $g(x, x.dom) = f(x)$ .

*Specifikáció:*

$$A = \begin{array}{cc} V & \times & \mathbb{N}_0 \\ x & & d \end{array}$$

$$B = \begin{array}{c} V \\ x' \end{array}$$

$$Q : (x = x')$$

$$R : (Q \wedge d = f(x))$$

*Megoldás:*

$$P = (Q \wedge k \in [0, x.dom] \wedge d = g(x, k))$$

$$Q' = (Q \wedge k = 0 \wedge d = 0)$$

Ellenőrizzük le a ciklus feltételeit:

$$1. Q \Rightarrow P$$

$$Q \Rightarrow lf(k, d := 0, 0, Q') = Q.$$

$$2. P \wedge \neg\pi \Rightarrow R$$

$P$  és  $R$  összehasonlításából  $\neg\pi$ -re  $k = x.dom$  adódik. Tehát  $\pi = (k \neq x.dom)$ .

$k, d := 0, 0$
$k \neq x.dom$
$\dots$

$$3. P \wedge \pi \Rightarrow t > 0$$

$t := x.dom - k$  választással ez az állítás triviálisan teljesül.

$$4./5. P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_0, P \wedge t < t_0)$$

Egy szekvenciával, aminek közbülső feltétele:

$$Q'' = (lf(k := k + 1, P) \wedge t = t_0) = (Q \wedge k + 1 \in [0, x.dom] \wedge d = g(x, k + 1) \wedge t = t_0)$$

Ez azért jó választás, mert ebből az állapotból a  $k := k + 1$  utasítással eljuthatunk  $P \wedge t < t_0$ -ba.

A szekvencia levezetési szabálya szerint szükséges  $P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_1, Q'')$  állítást pedig az  $S_1 = (d := d \cdot 10 + x_{x.lob+k})$  programmal teljesíthetjük, hiszen:

$$\begin{aligned} (P \wedge \pi \wedge t = t_0) &= (Q \wedge k \in [0, x.dom - 1] \wedge d = g(x, k) \wedge t = t_0) \\ &\stackrel{?}{\Rightarrow} lf(S_1, Q'') = (Q \wedge k + 1 \in [0, x.dom] \wedge d \cdot 10 + x_{x.lob+k} = g(x, k + 1) \wedge t = t_0) \checkmark. \end{aligned}$$

$$\begin{aligned} \text{Ugyanis, } g(x, k) \cdot 10 + x_{x.lob+k} &= \left( \sum_{i=0}^{k-1} x_{x.lob+k-i-1} \cdot 10^i \right) \cdot 10 + x_{x.lob+k} = \\ &= \left( \sum_{i=1}^k x_{x.lob+k-i} \cdot 10^{i-1} \right) \cdot 10 + x_{x.lob+k} \cdot 10^0 = \left( \sum_{i=0}^k x_{x.lob+k-i} \cdot 10^i \right) = g(x, k+1). \end{aligned}$$

$k, d := 0, 0$
$k \neq x.dom$
$d := d \cdot 10 + x_{x.lob+k}$
$k := k + 1$

20.

$$V = \text{vekt}(\mathbb{Z}, \{0, 1, \dots, 9\})$$

A feladat megoldása során használni fogjuk a következő két függvényt:

$$f(x) ::= \sum_{i=0}^{x.\text{dom}-1} x_{x.\text{hib}-i} \cdot 10^i \quad (\forall x \in V)$$

$$g(x, l) ::= \sum_{i=0}^{l-1} x_{x.\text{hib}-i} \cdot 10^i \quad (\forall x \in V : \forall l \in [0, x.\text{dom}])$$

Az  $f$  függvény egy vektorban ábrázolt szám értékét számítja ki, míg  $g$  ugyanezt teszi, de csak az utolsó  $l$  számjegyet veszi figyelembe.

Ha  $x$  és  $y$  azonos értelmezési tartományú vektorok, akkor  $x \oplus y$  vektoron egy olyan az említettekkel azonos értelmezési tartományú vektort fogunk érteni, amelynek minden eleme a két vektor megfelelő elemének összege. Formálisan:  $\forall i \in [x.\text{lob}, x.\text{hib}] : (x \oplus y)_i ::= x_i + y_i$ .

*Specifikáció:*

$$A = V \times V \times V \times \mathbb{Z} \times \mathbb{L}$$

$$B = V \times V$$

$$Q : (x = x' \wedge y = y' \wedge x.\text{dom} = y.\text{dom} = z.\text{dom} \wedge x.\text{hib} = y.\text{hib} = z.\text{hib})$$

$$R : (Q \wedge f(x \oplus y) = 10^{x.\text{dom}} \cdot \chi(c) + f(z))$$

*Megoldás:*

Vegyük észre, hogy  $f(w) = g(w, w.\text{dom})$  minden  $w$  vektorra. Ezt kihasználva a ciklusinvariáns:

$$P = (Q \wedge k \in [0..x.\text{dom}] \wedge g(x \oplus y, k) = 10^k \cdot \chi(c) + g(z, k))$$

Ellenőrizzük le a ciklus feltételeit:

$$1. Q \Rightarrow P$$

$$Q' = (Q \wedge o = \text{hamis} \wedge k = 0)$$

$$\text{Látható, hogy } Q' \Rightarrow P \text{ és } Q \Rightarrow \text{If}(o, k := \text{hamis}, 0, Q') = Q.$$

$$2. P \wedge \neg\pi \Rightarrow R$$

Ez a feltétel kezünkbe adja a ciklusfeltételt, hiszen  $P$  és  $R$  összehasonlításából  $\neg\pi$ -re  $k = x.\text{dom}$  adódik. Tehát  $\pi = (k \neq x.\text{dom})$ .

$c, k := \text{hamis}, 0$
$k \neq x.\text{dom}$
$S_2$

3.  $P \wedge \pi \Rightarrow t > 0$

$t := x.dom - k$  választással ez az állítás triviálisan teljesül.

4./5.  $P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_0, P \wedge t < t_0)$

A ciklusmagot egy olyan szekvenciaként írjuk fel ismét, amiben a második utasítás csupán a ciklusváltozó növeléséről gondoskodik, és így a  $Q''$  közbülső állítás a következő:

$$Q'' = lf(k := k + 1, P) \wedge t = t_0 = (Q \wedge k + 1 \in [0..x.dom]) \wedge g(x \oplus y, k + 1) = 10^{k+1} \cdot \chi(c) + g(z, k + 1) \wedge t = t_0$$

Ebből az állapotból a  $k := k + 1$  utasítás eljut  $P \wedge t < t_0$ -ba.

A nehezebb kérdés az, hogy a ciklusmag elején fennálló  $P \wedge \pi \wedge t = t_0$  állapotból milyen programmal jutunk el  $Q''$ -be. Ennek megoldásához nézzük meg, hogy hogyan írható fel  $g(w, j + 1)$  valamilyen értelmes  $j$  esetén:

$$g(w, j + 1) = \sum_{i=0}^{j+1-1} x_{x.hib-i} \cdot 10^i = \sum_{i=0}^{j-1} x_{x.hib-i} \cdot 10^i + x_{x.hib-j} \cdot 10^j \\ = g(w, j) + x_{x.hib-j} \cdot 10^j$$

Ezt használjuk fel  $Q''$  egy újabb alakjának felírásához:

$$Q'' = (Q \wedge k + 1 \in [0..x.dom]) \wedge g(x \oplus y, k) + x \oplus y_{x \oplus y.hib-k} \cdot 10^k = 10^{k+1} \cdot \chi(c) + g(z, k) + z_{z.hib-k} \cdot 10^k \wedge t = t_0$$

Így már felismehető, hogy az alábbi programmal kell elérni  $P \wedge \pi$ -ből  $Q''$ -t:

$c$	
$c, z_{x.hib-k} :=$ $1 + x_{x.hib-k} + y_{y.hib-k} \geq 10,$ $1 + x_{x.hib-k} + y_{y.hib-k} \bmod 10$	$c, z_{x.hib-k} :=$ $x_{x.hib-k} + y_{y.hib-k} \geq 10,$ $x_{x.hib-k} + y_{y.hib-k} \bmod 10$

$$P = (Q \wedge k \in [0..x.dom]) \wedge g(x \oplus y, k) = 10^k \cdot \chi(c) + g(z, k)$$

Hiszen:

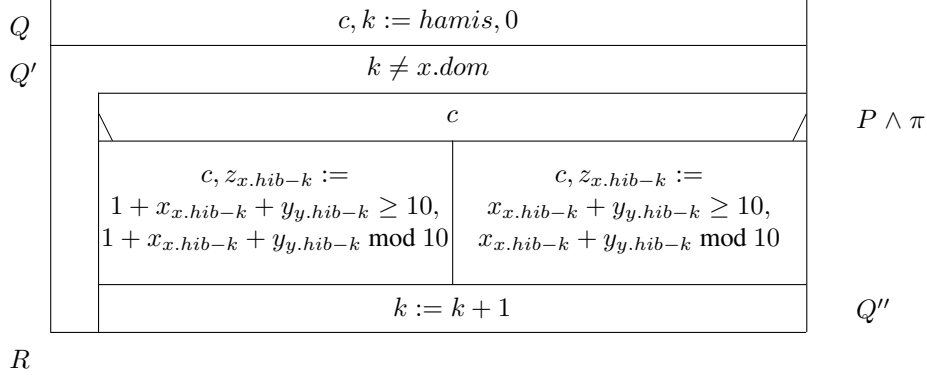
1.  $P \wedge \pi \Rightarrow \left( \bigvee_{i=1}^n \pi_i \right)$  Nyilván teljesül.

2.  $\forall i \in [1..n] : P \wedge \pi \wedge t = t_0 \wedge \pi_i \Rightarrow lf(S_i, Q'')$

(a)  $c: P \wedge \pi \wedge t = t_0 \wedge c \stackrel{?}{\Rightarrow} (Q \wedge k + 1 \in [0..x.dom]) \wedge g(x \oplus y, k) + x \oplus y_{x.hib-k} \cdot 10^k = 10^{k+1} \cdot \chi(1 + x_{x.hib-k} + y_{y.hib-k} \geq 10) + g(z, k) + (1 + x_{x.hib-k} + y_{y.hib-k} \bmod 10) \cdot 10^k \wedge t = t_0) \checkmark$



$$(b) \neg c: P \wedge \pi \wedge \neg c \stackrel{2}{\Rightarrow} (Q \wedge k+1 \in [0..x.dom] \wedge g(x \oplus y, k) + x \oplus y_{x.hib-k} \cdot 10^k = 10^{k+1} \cdot \chi(x_{x.hib-k} + y_{x.hib-k} \geq 10) + g(z, k) + (x_{x.hib-k} + y_{x.hib-k} \bmod 10) \cdot 10^k \wedge t = t_0) \checkmark$$



**26. Specifikáció:**

$$V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$$

$$A = V$$

$$x$$

$$B = V$$

$$x'$$

$$Q : (x = x' \wedge \text{dom}(x) \neq 0)$$

$$R : (x \in \text{perm}(x') \wedge x \text{ csökkenően rendezett})$$

Definiáljuk a *csökkenően rendezett* fogalmát:  $x$  vektor adott  $k$  indexéig csökkenően rendezett, ha  $(\forall i \in [x.lob..k-1] : x_i \geq x_{i+1}) \wedge (\forall i \in [k+1..x.hib] : x_k \geq x_i)$ . Továbbá  $x$  vektor csökkenően rendezett, ha  $x$   $x.hib$ -ig csökkenően rendezett. A definíciókból következik, hogy  $x$  vektor csökkenően rendezett akkor is, ha  $x$   $x.hib-1$ -ig csökkenően rendezett. Ezért mostantól ezt értjük ezalatt. Minden  $x$  vektor legyen definíció szerint  $x.lob-1$ -ig csökkenően rendezett.

*Megoldás:*

$$P = (x \in \text{perm}(x') \wedge k \in [x.lob-1..x.hib-1] \wedge x \text{ } k\text{-ig csökkenően rendezett})$$

Ellenőrizzük le a ciklus feltételeit:

$$1. Q \Rightarrow P$$

$$Q' = (Q \wedge k = x.lob - 1)$$

$$Q' \Rightarrow P \text{ (hiszen } x \text{ } x.lob - 1\text{-ig mindig csökkenően rendezett)}$$

$$\text{és } Q \Rightarrow lf(k := x.lob - 1, Q') = Q.$$

2.  $P \wedge \neg\pi \Rightarrow R$

Ez a feltétel adja a ciklusfeltételt,  $\neg\pi$ -re  $k = x.hib - 1$  adódik. Tehát  $\pi = (k \neq x.hib - 1)$ .

$k := x.lob - 1$
$k \neq x.hib - 1$
$\dots$

3.  $P \wedge \pi \Rightarrow t > 0$

Jelen esetben ez:  $(Q \wedge k \in [x.lob - 1..x.hib - 2] \wedge \dots) \stackrel{?}{\Rightarrow} t > 0$ , tehát a termináló függvény:  $t := x.hib - k$ . Ez a függvény  $P \wedge \pi$  esetén nyilván pozitív.

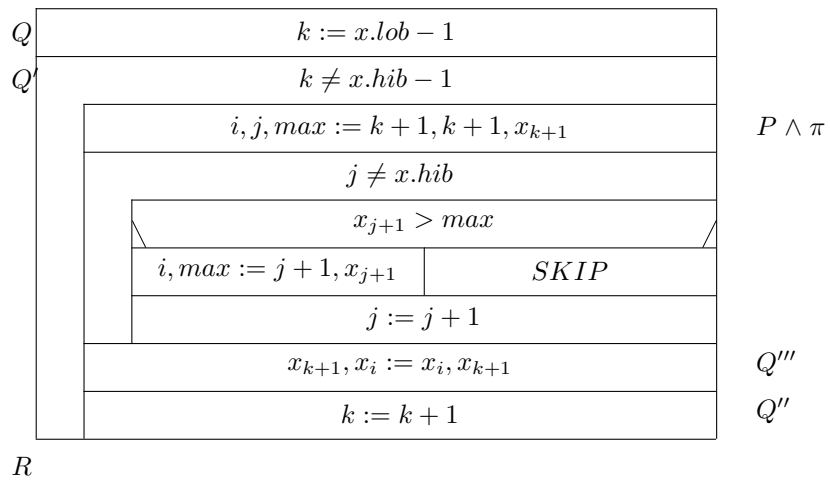
4./5.  $P \wedge \pi \wedge t = t_0 \Rightarrow lf(S_0, P \wedge t < t_0)$

A ciklusmagban szereplő szekvencia közbülső  $Q''$  állapota legyen a következő:  $Q'' = lf(k := k + 1, P) \wedge t = t_0 = (x \in perm(x') \wedge k + 1 \in [x.lob - 1..x.hib - 1] \wedge x \text{ } k + 1\text{-ig csökkenően rendezett} \wedge t = t_0)$

Milyen programmal juthatunk el a  $Q''$  állapotba  $P \wedge \pi \wedge t = t_0$  állapotból? Egy olyan szekvenciával, aminek első fele megkeresi a vektor hátra lévő részéből a maximális értékű elemet, a második fele pedig felcseréli ezt a  $k + 1$ -edik elemmel.

Tehát  $Q''' = (x \in perm(x') \wedge k + 1 \in [x.lob - 1..x.hib - 1] \wedge x \text{ } k\text{-ig csökkenően rendezett} \wedge t = t_0 \wedge i \in [k + 1..x.hib] \wedge \forall j \in [k + 1..x.hib] : x_j \leq x_i)$ . Vegyük észre, hogy  $Q'''$ -ben szereplő  $i$  azontúl, hogy a legnagyobb elem indexe a vektor  $k + 1$  mögötti részében, egy olyan elem indexe is, ami a csökkenően rendezettség definíciója szerint szükségeszerűen kisebb, mint bármelyik elem a vektor  $k$  előtti részében. Ezért igaz az az állítás, hogy  $Q''' \Rightarrow lf(x_{k+1}, x_i := x_i, x_{k+1}, Q'')$ .

Már csak azt az  $S_{010}$  programot kell megtalálnunk, amivel  $P \wedge \pi \wedge t = t_0 \Rightarrow Q'''$ . Ez a maximumkereséshez nagyon hasonló (de vektorrészleten működő) program, aminek levezetése a maximumkeresés programozási tételének levezetése ismeretében triviális, ezért itt csak a stuktogramot közöljük a kész program részeként:



## 15.10. Visszavezetés

1.

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{N}_0$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R : \left( Q \wedge d = \sum_{j=m+1}^n \chi(0 < f(j) \wedge f(j-1) < 0) \right)$$

1. Visszavezetés: számlálás

Feltétel:  $m \leq n$  | Paraméter: - | Szigorítás: -

Megfeleltetés:

$$m \leftarrow m + 1$$

$$n \leftarrow n$$

$$i \leftarrow i$$

$$d \leftarrow d$$

$$\beta(j) \leftarrow 0 < f(j) \wedge f(j-1) < 0$$

$i, d := m, 0$	
$i \neq n$	
$0 < f(i+1) \wedge f(i) < 0$	
$d := d + 1$	<i>SKIP</i>
$i := i + 1$	

2.

$$A = \mathbb{N} \times \mathbb{L}$$

$$B = \mathbb{N}$$

$$Q : (n = n')$$

$$R : \left( Q \wedge l = \exists j \in [2, n-1] : (2 \nmid j \wedge j \mid n) \wedge \right. \\ \left. l \rightarrow (i \in [2, n-1] \wedge (2 \nmid i \wedge i \mid n)) \right)$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter: - | Szigorítás: ha van, keressük meg, melyik az

Megfeleltetés:

$$m \leftarrow 2$$

$$n \leftarrow n - 1$$

$$l \leftarrow l$$

$$i \leftarrow i$$

$$\beta(j) \leftarrow (2 \nmid j \wedge j \mid n)$$

$i, l := 1, \textit{hamis}$	
$\neg l \wedge i \neq n - 1$	
$l := (2 \nmid i + 1 \wedge i + 1 \mid n)$	
$i := i + 1$	

3.

$$A = \mathbb{N} \times \mathbb{N} \times \mathbb{L}$$

$$B = \mathbb{N}$$

$$Q : (n = n' \wedge 2 < n)$$

$$R : \left( Q \wedge l = \exists j \in [2, n-1] : (2 \nmid j \wedge j \mid n) \wedge \right. \\ \left. l \rightarrow (i \in [2, n-1] \wedge (2 \nmid i \wedge i \mid n) \wedge \forall j \in [2, i-1] : (2 \mid j \vee j \nmid n)) \right)$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter: - | Szigorítás: -  
 Megfeleltetés:

$m \leftarrow 2$   
 $n \leftarrow n - 1$   
 $l \leftarrow l$   
 $i \leftarrow i$   
 $\beta(j) \leftarrow (2 \nmid j \wedge j \mid n)$

$i, l := 1, \text{hamis}$
$\neg l \wedge i \neq n - 1$
$l := (2 \nmid i + 1 \wedge i + 1 \mid n)$
$i := i + 1$

**4.**

$V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$

$A = V \times V \times \mathbb{Z} \times \mathbb{Z}$   
 $x \quad y \quad \text{max} \quad i$

$B = V \times V$   
 $x' \quad y'$

$Q : \left( x = x' \wedge y = y' \wedge \left( \forall j \in [y.\text{lob}, y.\text{hib}] : y_j \in [x.\text{lob}, x.\text{hib}] \wedge 0 < x.\text{dom} \wedge 0 < y.\text{dom} \right) \right)$

$R : (Q \wedge i \in [y.\text{lob}, y.\text{hib}] \wedge \text{max} = x_{y_i} \wedge \forall j \in [y.\text{lob}, y.\text{hib}] : x_{y_j} \leq \text{max})$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter:  $x, y$  | Szigorítás: -  
 Megfeleltetés:

$m \leftarrow y.\text{lob}$   
 $n \leftarrow y.\text{hib}$   
 $\text{max} \leftarrow \text{max}$   
 $i \leftarrow i$   
 $k \leftarrow k$   
 $a \leq b \leftarrow a \leq b$   
 $f(j) \leftarrow x_{y_j}$

$\text{max}, i, k := x_{y.y.\text{lob}}, y.\text{lob}, y.\text{lob}$
$k \neq y.\text{hib}$
$\text{max} \leq x_{y_{k+1}}$
$\text{max}, i := x_{y_{k+1}}, k + 1$   <b>SKIP</b>
$k := k + 1$

**5.**

$V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$

$A = V \times \mathbb{N}_0$   
 $x \quad d$

$B = V$   
 $x'$

$Q : (x = x')$

$R : \left( Q \wedge d = \sum_{j=v.\text{lob}}^{v.\text{hib}} \chi(x_j < j) \right)$

1. Visszavezetés: számlálás

Feltétel: - | Paraméter:  $x$  | Szigorítás: -

Megfeleltetés:

$$\begin{aligned} m &\leftarrow v.lob \\ n &\leftarrow v.hib \\ i &\leftarrow i \\ d &\leftarrow d \\ \beta(j) &\leftarrow x_j < j \end{aligned}$$

$i, d := v.lob - 1, 0$	
$i \neq v.hib$	
$x_{i+1} < i + 1$	
$d := d + 1$	<i>SKIP</i>
$i := i + 1$	

**6.**

$$V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$$

$$A = \mathbb{N} \times \mathbb{N}$$

$$\begin{matrix} n & i \end{matrix}$$

$$B = \mathbb{N}$$

$$n'$$

$$Q : (n = n' \wedge 2 \leq n)$$

$$R : \left( Q \wedge 2 \leq i \wedge i \mid n \wedge i \nmid \frac{n}{i} \wedge \forall j \in [2, i-1] : (j \nmid n \vee i \mid \frac{n}{j}) \right)$$

1. Visszavezetés: lineáris keresés 2.

Feltétel: - | Paraméter: - | Szigorítás: -

Megfeleltetés:

$$\begin{aligned} m &\leftarrow 2 \\ l &\leftarrow l \\ i &\leftarrow i \\ \beta(j) &\leftarrow j \mid n \wedge j \nmid \frac{n}{j} \end{aligned}$$

$i, l := 1, \textit{hamis}$	
$\neg l$	
$l := i + 1 \mid n \wedge i + 1 \nmid \frac{n}{i+1}$	
$i := i + 1$	

**7.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{N}_0$$

$$\begin{matrix} m & n & d \end{matrix}$$

$$B = \mathbb{Z} \times \mathbb{Z} \times$$

$$\begin{matrix} m' & n' \end{matrix}$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R : \left( Q \wedge d = \sum_{j=m}^n \chi(2 \mid f(j) \wedge g(j) = 1) \right)$$

1. Visszavezetés: számlálás

Feltétel: - | Paraméter: - | Szigorítás: -

Megfeleltetés:

$$\begin{aligned} m &\leftarrow m \\ n &\leftarrow n \\ i &\leftarrow i \\ d &\leftarrow d \\ \beta(j) &\leftarrow 2 \mid f(j) \wedge g(j) = 1 \end{aligned}$$

$i, d := m - 1, 0$	
$i \neq n$	
$2 \mid f(i+1) \wedge g(i+1) = 1$	
$d := d + 1$	<i>SKIP</i>
$i := i + 1$	

**8.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{L}$$

$$m \quad n \quad l$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R : (Q \wedge i \in [m, n] \wedge \forall j \in [m, n] : f(j) \bmod 10 \leq f(i) \bmod 10)$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter: - | Szigorítás: adjuk meg a számjegy értékét is

Megfeleltetés:

$$m \leftarrow m$$

$$n \leftarrow n$$

$$max \leftarrow max$$

$$i \leftarrow i$$

$$k \leftarrow k$$

$$a \leq b \leftarrow a \leq b$$

$$f(j) \leftarrow f(j) \bmod 10$$

$max, i, k := f(m) \bmod 10, m, m$	
$k \neq n$	
$max \leq f(k + 1) \bmod 10$	
$max, i := f(k + 1) \bmod 10, k + 1$	<i>SKIP</i>
$k := k + 1$	

**9.**

$$V = \text{vekt}(\mathbb{Z}, C)$$

$$A = V \times \mathbb{L}$$

$$x \quad l$$

$$B = V$$

$$x'$$

$$Q : (x = x')$$

$$R : (Q \wedge l = \forall j \in [0, x.dom - 1] : x_{x.lob+j} = x_{x.hib-j})$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $x$  | Szigorítás: adjuk meg az első eltérés helyét is

Megfeleltetés:

$$m \leftarrow 0$$

$$n \leftarrow x.dom - 1$$

$$l \leftarrow \neg l$$

$$i \leftarrow i$$

$$\beta(j) \leftarrow x_{x.lob+j} \neq x_{x.hib-j}$$

$i, l := -1, igaz$	
$l \wedge i \neq x.dom - 1$	
$l := x_{x.lob+i+1} = x_{x.hib-i-1}$	
$i := i + 1$	

**10.**

$$NNM = \text{vekt}(\mathbb{Z}, V)$$

$$V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$$

$$I_{NNM}(x) = \forall i \in [x.lob, x.hib] : (x_i.dom = x.dom \wedge x_i.lob = x.lob)$$

$$A = NNM \times \mathbb{N} \times \mathbb{Z}$$

$$t \quad k \quad d$$

$$B = NNM \times \mathbb{N}$$

$$t' \quad k'$$

$$Q : (t = t' \wedge k = k' \wedge k \leq t.dom \wedge k \leq t.dom)$$

$$R : \left( Q \wedge d = \sum_{j=t.lob}^{t.hib} t_{k,j} \right)$$

1. Visszavezetés: összegzés

Felt: - | Paraméter:  $t, k$  | Szigorítás: -

Megfeleltetés:

$$m \leftarrow t.lob$$

$$n \leftarrow t.hib$$

$$d \leftarrow d$$

$$i \leftarrow i$$

$$f(j) \leftarrow t_{k,j}$$

$i, d := t.lob - 1, 0$
$i \neq t.hib$
$d := d + t_{k,i+1}$
$i := i + 1$

**11.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{N} \times \mathbb{L} \times \mathbb{Z} \times \mathbb{Z}$$

$$m \quad n \quad k \quad l \quad max \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \mathbb{N}$$

$$m' \quad n' \quad k'$$

$$Q : (m = m' \wedge n = n' \wedge k = k' \wedge m \leq n + 1)$$

$$R : \left( Q \wedge l = \exists j \in [m, n] : k \mid f(j) \wedge \right.$$

$$\left. l \rightarrow \left( i \in [m, n] \wedge max = f(i) \wedge k \mid f(i) \wedge \right. \right.$$

$$\left. \left. \forall j \in [m, n] : k \mid f(j) \rightarrow f(j) \leq max \right) \right)$$

1. Visszavezetés: feltételes maximum keresés

Feltétel: - | Paraméter:  $k$  | Szigorítás: -

Megfeleltetés:



$m \leftarrow m$	$n \leftarrow n$	$l \leftarrow l$
$max \leftarrow max$	$i \leftarrow i$	$k \leftarrow p$
$a \leq b \leftarrow a \leq b$	$f(j) \leftarrow f(j)$	$\beta(j) \leftarrow k \mid f(j)$
$l, p := hamis, m - 1$		
$p \neq n$		
$k \nmid f(p+1)$	$k \mid f(p+1) \wedge \neg l$	$k \mid f(p+1) \wedge l$
<i>SKIP</i>	$l, i, max :=$ $igaz, p+1, f(p+1)$	$max \leq f(p+1)$
	$i, max := p+1, f(p+1)$	<i>SKIP</i>
$p := p+1$		

**12.**

$$A = \mathbb{N} \times \mathbb{N}_0$$

$$n \quad d$$

$$B = \mathbb{N}$$

$$n'$$

$$Q : (n = n')$$

$$R : \left( Q \wedge d = \sum_{j=2}^{n-1} \chi(2 \mid j \wedge j \mid n) \right)$$

1. Visszavezetés: számlálás

Feltétel: - | Paraméter: - | Szigorítás: -

$$m \leftarrow 2 \mid n \leftarrow n - 1 \mid i \leftarrow i \mid d \leftarrow d \mid \beta(j) \leftarrow 2 \mid j \wedge j \mid n$$

**13.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{L} \times \mathbb{Z} \times \mathbb{Z}$$

$$m \quad n \quad l \quad i \quad max$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m + 1 \leq n - 1)$$

$$R : \left( Q \wedge l = \exists j \in [m+1, n-1] : \left( \begin{array}{l} f(j) < f(j-1) \wedge \\ f(j) < f(j+1) \end{array} \right) \wedge \right. \\ \left. l \rightarrow \left( \begin{array}{l} i \in [m+1, n-1] \wedge max = f(i) \wedge \left( \begin{array}{l} f(i) < f(i-1) \wedge \\ f(i) < f(i+1) \end{array} \right) \wedge \\ \forall j \in [m+1, n-1] : \left( \begin{array}{l} f(j) < f(j-1) \wedge \\ f(j) < f(j+1) \end{array} \right) \rightarrow f(j) \leq max \end{array} \right) \right) \right)$$

1. Visszavezetés: feltételes maximum keresés

Feltétel: - | Paraméter: - | Szigorítás: -

$$m \leftarrow m+1 \mid n \leftarrow n-1 \mid l \leftarrow l$$

$$max \leftarrow max \mid i \leftarrow i \mid k \leftarrow k$$

$$a \leq b \leftarrow a \leq b \mid f(j) \leftarrow f(j) \mid \beta(j) \leftarrow \left( \begin{array}{l} f(j) < f(j-1) \wedge \\ f(j) < f(j+1) \end{array} \right)$$

**14.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{L} \times \mathbb{Z}$$

$$m \quad n \quad k \quad l \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n' \quad k'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1 \wedge k = k')$$

$$R : (Q \wedge l = \exists j \in [m, n] : k \mid f(j) \wedge l \rightarrow (i \in [m, n] \wedge k \mid f(i)))$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $k$  | Szigorítás: első ilyen argumentumot keressük meg  
 $m \leftarrow m \mid n \leftarrow n \mid l \leftarrow l \mid i \leftarrow i \mid \beta(j) \leftarrow k \mid f(j)$

**15.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{L} \times \mathbb{Z} \times \mathbb{Z}$$

$$m \quad n \quad k \quad l \quad \max \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n' \quad k'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1 \wedge k = k')$$

$$R : \left( Q \wedge l = \exists j \in [m, n] : f(j) < k \wedge \right. \\ \left. l \rightarrow (i \in [m, n] \wedge \max = f(i) \wedge f(i) < k \wedge \right. \\ \left. \forall j \in [m, n] : f(j) < k \rightarrow f(j) \leq \max) \right)$$

1. Visszavezetés: feltételes maximum keresés

Feltétel: - | Paraméter:  $k$  | Szigorítás: -

$$\begin{array}{c|c|c} m \leftarrow m & n \leftarrow n & l \leftarrow l \\ \max \leftarrow \max & i \leftarrow i & k \leftarrow p \\ a \leq b \leftarrow a \leq b & f(j) \leftarrow f(j) & \beta(j) \leftarrow f(j) < k \end{array}$$

**16.**

$$V = \text{vekt}(\mathbb{Z}, P)$$

$$P = (x : \mathbb{Z}, y : \mathbb{Z})$$

$$A = V \times P \times \mathbb{Z} \times \mathbb{L} \times \mathbb{Z}$$

$$v \quad c \quad r \quad l \quad i$$

$$B = V \times P \times \mathbb{Z}$$

$$v' \quad c' \quad r'$$

$$Q : (v = v' \wedge c = c' \wedge r = r')$$

$$R : \left( Q \wedge l = \exists j \in [v.\text{lob}, v.\text{hib}] : (v_j.x - c.x)^2 + (v_j.y - c.y)^2 \leq r^2 \wedge \right. \\ \left. l \rightarrow (i \in [v.\text{lob}, v.\text{hib}] \wedge (v_i.x - c.x)^2 + (v_i.y - c.y)^2 \leq r^2) \right)$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $v, c, r$  | Szigorítás: az első ilyen pontot keressük meg  
 $m \leftarrow v.\text{lob} \mid n \leftarrow v.\text{hib} \mid l \leftarrow l \mid i \leftarrow i \mid \beta(j) \leftarrow (v_j.x - c.x)^2 + (v_j.y - c.y)^2 \leq r^2$

**17.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{L} \times \mathbb{Z} \times \mathbb{Z}$$

$$m \quad n \quad a \quad b \quad l \quad \max \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n' \quad a' \quad b'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1 \wedge a = a' \wedge b = b')$$

$$R : \left( Q \wedge l = \exists j \in [m, n] : a \leq f(j) \leq b \wedge \right. \\ \left. l \rightarrow \left( i \in [m, n] \wedge \max = f(i) \wedge a \leq f(i) \leq b \wedge \right. \right. \\ \left. \left. \forall j \in [m, n] : a \leq f(j) \leq b \rightarrow f(j) \leq \max \right) \right)$$

1. Visszavezetés: feltételes maximum keresés

Feltétel: - | Paraméter:  $a, b$  | Szigorítás: -

$$\begin{array}{ccc|ccc} m & \leftarrow & m & n & \leftarrow & n & l & \leftarrow & l \\ \max & \leftarrow & \max & i & \leftarrow & i & k & \leftarrow & k \\ a \leq b & \leftarrow & a \leq b & f(j) & \leftarrow & f(j) & \beta(j) & \leftarrow & a \leq f(j) \leq b \end{array}$$

**18.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{N}_0$$

$$m \quad n \quad a \quad b \quad c \quad d \quad p$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n' \quad a' \quad b' \quad c' \quad d'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1 \wedge a = a' \wedge b = b' \wedge c = c' \wedge d = d')$$

$$R : \left( Q \wedge p = \sum_{j=m}^n \chi(a \leq f(j) \leq b \vee c \leq f(j) \leq d) \right)$$

1. Visszavezetés: számlálás

Feltétel: - | Paraméter:  $a, b, c, d$  | Szigorítás: -

$$m \leftarrow m \mid n \leftarrow n \mid i \leftarrow i \mid d \leftarrow p \mid \beta(j) \leftarrow a \leq f(j) \leq b \vee c \leq f(j) \leq d$$

**19.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{L} \times \mathbb{Z} \times \mathbb{Z}$$

$$m \quad n \quad k \quad l \quad \max \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n' \quad k'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1 \wedge k = k')$$

$$R : \left( Q \wedge l = \exists j \in [m, n] : k \mid f(j) - 1 \wedge \right. \\ \left. l \rightarrow \left( i \in [m, n] \wedge \max = f(i) \wedge k \mid f(i) - 1 \wedge \right. \right. \\ \left. \left. \forall j \in [m, n] : k \mid f(j) - 1 \rightarrow f(j) \leq \max \right) \right)$$

1. Visszavezetés: feltételes maximum keresés

Feltétel: - | Paraméter:  $k$  | Szigorítás: -

$$\begin{array}{ccc|ccc} m & \leftarrow & m & n & \leftarrow & n & l & \leftarrow & l \\ \max & \leftarrow & \max & i & \leftarrow & i & k & \leftarrow & k \\ a \leq b & \leftarrow & a \leq b & f(j) & \leftarrow & f(j) & \beta(j) & \leftarrow & k \mid f(j) - 1 \end{array}$$

**20.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$m \quad n \quad N \quad \max \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n' \quad N'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n \wedge N = N')$$

$$R : \left( Q \wedge i \in [m, n] \wedge \max = f(i) \wedge \right. \\ \left. \forall j \in [m, n] : f(j) \bmod N \leq \max \bmod N \right)$$

1. **Visszavezetés: maximum keresés**Feltétel: - | Paraméter:  $N$  | Szigorítás: -

$$\begin{array}{l|l|l} m \leftarrow m & n \leftarrow n & \max \leftarrow \max \\ i \leftarrow i & k \leftarrow k & a \leq b \leftarrow a \bmod N \leq b \bmod N \\ f(j) \leftarrow f(j) & & \end{array}$$

**21.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{N}_0$$

$$m \quad n \quad x_0 \quad y_0 \quad r \quad d$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n' \quad x'_0 \quad y'_0 \quad r'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1 \wedge x_0 = x'_0 \wedge y_0 = y'_0 \wedge r = r')$$

$$R : \left( Q \wedge d = \sum_{j=m}^n \chi((f(j) - x_0)^2 + (g(j) - y_0)^2 \leq r^2) \right)$$

1. **Visszavezetés: számlálás**Feltétel: - | Paraméter:  $x_0, y_0, r$  | Szigorítás: -

$$m \leftarrow m \mid n \leftarrow n \mid i \leftarrow i \mid d \leftarrow d \mid \beta(j) \leftarrow (f(j) - x_0)^2 + (g(j) - y_0)^2 \leq r^2$$

**22.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{L}$$

$$a \quad b \quad n \quad i \quad l$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$a' \quad b' \quad n'$$

$$Q : (a = a' \wedge b = b' \wedge a \leq b \wedge n = n')$$

$$R : \left( Q \wedge l = \exists j \in [a, b] : (f(j) = 0 \wedge f(j) < n) \wedge \right. \\ \left. l \rightarrow \left( i \in [a, b] \wedge (f(i) = 0 \wedge f(i) < n) \wedge \right. \right. \\ \left. \left. \forall j \in [a, i-1] : (f(j) \neq 0 \vee f(j) \geq n) \right) \right)$$

1. **Visszavezetés: lineáris keresés 2.8.**Feltétel: - | Paraméter:  $n$  | Szigorítás: -

$$m \leftarrow a \mid n \leftarrow b \mid l \leftarrow l \mid i \leftarrow i \mid \beta(j) \leftarrow (f(j) = 0 \wedge f(j) < n)$$

**23.**

$$V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$$

$$A = V \times V \times \mathbb{Z} \times \mathbb{N}_0$$

$$x \quad y \quad k \quad d$$

$$B = V \times V \times \mathbb{Z}$$

$$x' \quad y' \quad k'$$

$$Q : \left( \begin{array}{l} x = x' \wedge y = y' \wedge k = k' \wedge \forall j \in [y.lob, y.hib] : y_j \in [x.lob, x.hib] \wedge \\ \forall a, b \in [y.lob, y.hib] : (y_a = y_b \rightarrow a = b) \end{array} \right)$$

$$R : \left( Q \wedge d = \sum_{j=y.lob}^{y.hib} \chi(k | x_{y_j}) \right)$$

1. Visszavezetés: számlálás

Feltétel: - | Paraméter:  $x, y, z$  | Szigorítás: -
 $m \leftarrow y.lob \mid n \leftarrow y.hib \mid i \leftarrow i \mid d \leftarrow d \mid \beta(j) \leftarrow k \mid x_{y_j}$ 
**24.**

$$V = \text{vekt}(\mathbb{Z}, C)$$

$$A = V \times \mathbb{L}$$

$$x \quad l$$

$$B = V$$

$$x'$$

$$Q : (x = x')$$

$$R : (Q \wedge l = \exists j \in [x.lob, x.hib] : x_j \in ['a', 'e', 'i', 'o', 'u'])$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $x$  | Szigorítás: keressük az első magánhangzó helyét is
 $m \leftarrow x.lob \mid n \leftarrow x.hib \mid l \leftarrow l \mid i \leftarrow i \mid \beta(j) \leftarrow x_j \in ['a', 'e', 'i', 'o', 'u']$ 
**25.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{L} \times \mathbb{Z}$$

$$m \quad n \quad a \quad b \quad c \quad d \quad l \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n' \quad a' \quad b' \quad c' \quad d'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1 \wedge a = a' \wedge b = b' \wedge c = c' \wedge d = d')$$

$$R : \left( \begin{array}{l} Q \wedge l = \exists j \in [m, n] : a \leq f(j) \leq b \wedge c \leq f(j) \leq d \wedge \\ l \rightarrow (i \in [m, n] \wedge a \leq f(i) \leq b \wedge c \leq f(i) \leq d) \end{array} \right)$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $a, b, c, d$  | Szigorítás: első ilyen elemet keressük meg
 $m \leftarrow m \mid n \leftarrow n \mid l \leftarrow l \mid i \leftarrow i \mid \beta(j) \leftarrow a \leq f(j) \leq b \wedge c \leq f(j) \leq d$

**26.**

$$V = \text{vekt}(\mathbb{Z}, C)$$

$$A = V \times \mathbb{N}_0$$

$$x \quad d$$

$$B = V$$

$$x'$$

$$Q : (x = x')$$

$$R : \left( Q \wedge d = \sum_{j=x.lob}^{x.hib} \chi(x_j \in ['a', 'e', 'i', 'o', 'u']) \right)$$

1. Visszavezetés: számlálás

Feltétel: - | Paraméter:  $x$  | Szigorítás: -

$$m \leftarrow x.lob \mid n \leftarrow x.hib \mid i \leftarrow i \mid d \leftarrow d \mid \beta(j) \leftarrow x_j \in ['a', 'e', 'i', 'o', 'u']$$

**27.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$m \quad n \quad k$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m + 1 \leq n)$$

$$R : (Q \wedge k \in [m + 1, n] \wedge \forall j \in [m + 1, n] : f(j) - f(j - 1) \leq f(k) - f(k - 1))$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter: - | Szigorítás: mondjuk meg a különbség mértékét is

$$m \leftarrow m + 1 \mid n \leftarrow n \mid \max \leftarrow \max$$

$$i \leftarrow k \mid k \leftarrow p \mid a \leq b \leftarrow a \leq b$$

$$f(j) \leftarrow f(j) - f(j - 1) \mid$$

**28.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{L} \times \mathbb{Z}$$

$$m \quad n \quad l \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R : \left( Q \wedge l = \exists j \in [m, n] : 2 \mid f(j) \wedge l \rightarrow (i \in [m, n] \wedge 2 \mid f(i) \wedge \forall j \in [m, n] : 2 \mid f(j) \rightarrow f(j) \leq f(i)) \right)$$

1. Visszavezetés: feltételes maximum keresés

Feltétel: - | Paraméter: - | Szigorítás: a legnagyobb értéket is adjuk meg

$$m \leftarrow m \mid n \leftarrow n \mid l \leftarrow l$$

$$\max \leftarrow \max \mid i \leftarrow i \mid k \leftarrow k$$

$$a \leq b \leftarrow a \leq b \mid f(j) \leftarrow f(j) \mid \beta(j) \leftarrow 2 \mid f(j)$$

**29.**

$$V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$$

$$A = V \times \mathbb{N} \times \mathbb{L}$$

$$\begin{array}{ccc} x & i & l \end{array}$$

$$B = V$$

$$\begin{array}{c} x' \end{array}$$

$$Q : (x = x')$$

$$R : \left( Q \wedge l = \exists j \in [x.\text{lob}, x.\text{hib} - 1] : x_j x_{j+1} < 0 \wedge \right.$$

$$\left. l \rightarrow (i \in [x.\text{lob}, x.\text{hib} - 1] \wedge x_i x_{i+1} < 0) \right)$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel:  $0 < x.\text{dom}$  | Paraméter:  $x$  | Szigorítás: első ilyen keressük meg  
 $m \leftarrow x.\text{lob}$  |  $n \leftarrow x.\text{hib} - 1$  |  $l \leftarrow l$  |  $i \leftarrow i$  |  $\beta(j) \leftarrow x_j x_{j+1} < 0$

**30.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$\begin{array}{ccc} m & n & d \end{array}$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$\begin{array}{cc} m' & n' \end{array}$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R : \left( Q \wedge d = \sum_{j=m}^n |f(j) - g(j)| \right)$$

1. Visszavezetés: összegzés

Feltétel: - | Paraméter: - | Szigorítás: -  
 $m \leftarrow m$  |  $n \leftarrow n$  |  $d \leftarrow d$  |  $i \leftarrow i$  |  $f(j) \leftarrow |f(j) - g(j)|$

**31.**

$$V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$$

$$A = V \times V \times \mathbb{N}_0$$

$$\begin{array}{ccc} x & b & d \end{array}$$

$$B = V \times V$$

$$\begin{array}{cc} x' & b' \end{array}$$

$$Q : (x = x' \wedge b = b')$$

$$R : \left( Q \wedge d = \sum_{j=0}^{x.\text{dom}-1} \chi(x_{x.\text{lob}+j} = b_{b.\text{lob}+j \bmod b.\text{dom}}) \right)$$

1. Visszavezetés: számlálás

Feltétel: - | Paraméter: - | Szigorítás: -  
 $m \leftarrow 0$  |  $n \leftarrow x.\text{dom} - 1$  |  $i \leftarrow i$  |  $d \leftarrow d$  |  $\beta(j) \leftarrow x_{x.\text{lob}+j} = b_{b.\text{lob}+j \bmod b.\text{dom}}$

**32.**

$$A = \mathbb{N} \times \mathbb{N} \times \mathbb{L}$$

$$\quad \quad \quad n \quad i \quad l$$

$$B = \mathbb{N}$$

$$\quad \quad \quad n'$$

$$Q : (n = n')$$

$$R : (Q \wedge l = \exists j \in [2, n-1] : j \mid n \wedge l \rightarrow (i \in [2, n-1] \wedge i \mid n))$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter: - | Szigorítás: legkisebb valódi osztót keressük

$$m \leftarrow 2 \mid n \leftarrow n-1 \mid l \leftarrow l \mid i \leftarrow i \mid \beta(j) \leftarrow j \mid n$$

**33.**

$$g(x) = \left| f(x) - \frac{f(x-1)+f(x+1)}{2} \right|$$

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$\quad \quad \quad m \quad n \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$\quad \quad \quad m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m+1 \leq n-1)$$

$$R : (Q \wedge i \in [m+1, n-1] \wedge \forall j \in [m+1, n-1] : g(j) \leq g(i))$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter: - | Szigorítás: adjuk meg az eltérés mértékét is

$$m \leftarrow m+1 \mid n \leftarrow n-1 \mid \begin{array}{l} \max \leftarrow \max \\ a \leq b \leftarrow a \leq b \end{array}$$

$$i \leftarrow i \mid k \leftarrow k$$

$$f(j) \leftarrow g(j)$$

**34.**

$$V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$$

$$A = V \times \mathbb{N} \times \mathbb{L}$$

$$\quad \quad \quad x \quad i \quad l$$

$$B = V$$

$$\quad \quad \quad x'$$

$$Q : (x = x')$$

$$R : (Q \wedge l = \exists j \in [x.lob, x.hib] : j \mid x_j \wedge l \rightarrow (i \in [x.lob, x.hib] \wedge i \mid x_i))$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $x$  | Szigorítás: első ilyen elemet keressük meg

$$m \leftarrow x.lob \mid n \leftarrow x.hib \mid l \leftarrow l \mid i \leftarrow i \mid \beta(j) \leftarrow j \mid x_j$$



35.

$$V = \text{vekt}(\mathbb{Z}, \text{vekt}(\mathbb{Z}, \mathbb{Z}))$$

$$I_V(t) = \left( \forall j \in [t.\text{lob}, t.\text{hib} - 1] : \left( \begin{array}{l} t_j.\text{dom} = t_{j+1}.\text{dom} \wedge \\ t_j.\text{lob} = t_{j+1}.\text{lob} \end{array} \right) \wedge t.\text{lob} = t_{t.\text{lob}}.\text{lob} \right)$$

$$g(i) = t.\text{lob} + \left\lfloor \frac{i}{t_{t.\text{lob}}.\text{dom}} \right\rfloor \quad h(i) = t.\text{lob} + i \bmod t_{t.\text{lob}}.\text{dom}$$

$$A = V \times \mathbb{Z} \times \mathbb{L}$$

$$t \quad n \quad l$$

$$B = V \times \mathbb{Z}$$

$$t' \quad n'$$

$$Q : \left( \begin{array}{l} t = t' \wedge n = n' \wedge t.\text{dom} > 0 \wedge \\ \forall j \in [0, x.\text{dom} \cdot x_{x.\text{lob}}.\text{dom} - 2] : t_{g(j),h(j)} \leq t_{g(j+1),h(j+1)} \end{array} \right)$$

$$R : \left( \begin{array}{l} Q \wedge l = \exists j \in [0, x.\text{dom} \cdot x_{x.\text{lob}}.\text{dom} - 1] : t_{g(j),h(j)} = n \wedge \\ l \rightarrow (i \in [0, x.\text{dom} \cdot x_{x.\text{lob}}.\text{dom} - 1] \wedge t_{g(i),h(i)} = n) \end{array} \right)$$

1. Visszavezetés: logaritmus keresés

Feltétel: - | Paraméter: - | Szigorítás: -

$$\begin{array}{l|l|l} m \leftarrow 0 & n \leftarrow x.\text{dom} \cdot x_{x.\text{lob}}.\text{dom} - 1 & i \leftarrow i \\ k \leftarrow n & u \leftarrow u & v \leftarrow v \\ l \leftarrow l & a < b \leftarrow a < b & f(j) \leftarrow t_{g(j),h(j)} \end{array}$$

$u, v, l := 0, x.\text{dom} \cdot x_{x.\text{lob}}.\text{dom} - 1, \text{hamis}$		
$\neg l \vee u \leq v$		
$i := \lfloor \frac{u+v}{2} \rfloor$		
$t_{g(i),h(i)} < n$	$t_{g(i),h(i)} = n$	$t_{g(i),h(i)} > n$
$u := i + 1$	$l := \text{igaz}$	$v := i - 1$

36.

$$V = \text{vekt}(\mathbb{Z}, S_z)$$

$$A = V \times \mathbb{N} \times \mathbb{L}$$

$$x \quad i \quad l$$

$$B = V$$

$$x'$$

$$Q : (x = x')$$

$$R : \left( \begin{array}{l} Q \wedge l = \exists j \in [x.\text{lob}, x.\text{hib}] : x_j = \text{'világoskék'} \wedge \\ l \rightarrow (i \in [x.\text{lob}, x.\text{hib}] \wedge x_i = \text{'világoskék'}) \end{array} \right)$$

1. Visszavezetés: logaritmus keresés

Feltétel: - | Paraméter: - | Szigorítás: -

$$\begin{array}{l|l|l} m \leftarrow x.\text{lob} & n \leftarrow x.\text{hib} & i \leftarrow i \\ k \leftarrow \text{'világoskék'} & u \leftarrow u & v \leftarrow v \\ l \leftarrow l & a < b \leftarrow a < b & f(j) \leftarrow x_j \end{array}$$

37.

$$V = \text{vekt}(\mathbb{Z}, P)$$

$$P = (x : \mathbb{Z}, y : \mathbb{Z})$$

$$A = V \times P$$

$$B = V$$

$$v$$

$$p$$

$$Q : (v = v')$$

$$R : \left( Q \wedge i \in [v.\text{lob}, v.\text{hib}] \wedge p = v_i \wedge \right. \\ \left. \forall j \in [v.\text{lob}, v.\text{hib}] : v_j.x^2 + v_j.y^2 \leq p.x^2 + p.y^2 \right)$$

1. Visszavezetés: maximum keresés

Feltétel: -	Paraméter: $v$	Szigorítás: keressük a vektorban elfoglalt helyet
$m \leftarrow v.\text{lob}$	$n \leftarrow v.\text{hib}$	$max \leftarrow p$
$i \leftarrow i$	$k \leftarrow k$	$a \leq b \leftarrow a.x^2 + a.y^2 \leq b.x^2 + b.y^2$
$f(j) \leftarrow v_j$		

38.

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{L} \times \mathbb{Z}$$

$$m \quad n \quad l \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R : \left( Q \wedge l = \exists j \in [m, n] : 0 < f(j) \wedge \right. \\ \left. l \rightarrow (i \in [m, n] \wedge 0 < f(i) \wedge \forall j \in [m, i - 1] : 0 \geq f(j)) \right)$$

1. Visszavezetés: lineáris keresés 2.8. ált. változat

Feltétel: -	Paraméter: -	Szigorítás: -
$m \leftarrow n$	$n \leftarrow m$	$l \leftarrow l$
$\text{succ}(i) \leftarrow i := i + 1$	$\text{pred}(i) \leftarrow i := i - 1$	$\beta(j) \leftarrow 0 < f(j) \mid i \leftarrow i$

39.

$$V = \text{vekt}(\mathbb{Z}, \{0, 1\})$$

$$A = V \times \mathbb{N}_0$$

$$x \quad d$$

$$B = V$$

$$x'$$

$$Q : (x = x')$$

$$R : \left( Q \wedge d = \sum_{j=t.\text{lob}}^{t.\text{hib}-3} \chi \left( \begin{array}{l} x_j = x_{j+2} = 0 \wedge \\ x_{j+1} = x_{j+3} = 1 \end{array} \right) \right)$$

1. Visszavezetés: számlálás

$$\text{Feltétel: } 3 < t.dom \mid \text{Paraméter: } x \mid \text{Szigorítás: -}$$

$$m \leftarrow t.lob \mid n \leftarrow t.hib - 3 \mid i \leftarrow i \mid d \leftarrow d \mid \beta(j) \leftarrow \begin{array}{l} x_j = x_{j+2} = 0 \wedge \\ x_{j+1} = x_{j+3} = 1 \end{array}$$

**40.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{L} \times \mathbb{Z}$$

$$\quad m \quad n \quad k \quad l \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$\quad m' \quad n' \quad k'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1 \wedge k = k')$$

$$R : (Q \wedge l = \exists j \in [m, n] : rprim(f(j), k) \wedge l \rightarrow (i \in [m, n] \wedge rprim(f(i), k)))$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter: - | Szigorítás: első ilyen értéket keressük meg

Megfeleltetés:

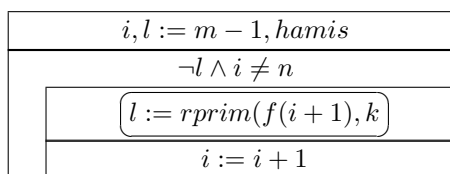
$$m \leftarrow m$$

$$n \leftarrow n$$

$$l \leftarrow l$$

$$i \leftarrow i$$

$$\beta(j) \leftarrow rprim(f(j), k)$$



2. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $k$  | Szigorítás: hol lesz „nem relatív prím”

Megfeleltetés:

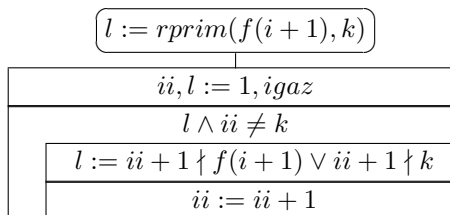
$$m \leftarrow 2$$

$$n \leftarrow k$$

$$l \leftarrow \neg l$$

$$i \leftarrow ii$$

$$\beta(j) \leftarrow j \mid f(i + 1) \wedge j \mid k$$

**41.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{L}$$

$$\quad n \quad i \quad l$$

$$B = \mathbb{Z}$$

$$\quad n'$$

$$Q : (n = n' \wedge 2 \leq n)$$

$$R : \left( \begin{array}{l} Q \wedge l = \exists j \in [2, n - 1] : (j \mid n \wedge \neg prim(j)) \wedge \\ l \rightarrow \left( \begin{array}{l} i \in [2, n - 1] \wedge (i \mid n \wedge \neg prim(i)) \wedge \\ \forall j \in [2, i - 1] : (j \nmid n \vee prim(j)) \end{array} \right) \end{array} \right)$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter: - | Szigorítás: -

$$m \leftarrow 2 \mid n \leftarrow n - 1 \mid l \leftarrow l \mid i \leftarrow i \mid \beta(j) \leftarrow (j \mid n \wedge \neg prim(j))$$

2. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $i, n$  | Szigorítás: hol lesz „nem prím”

$$m \leftarrow 2 \mid n \leftarrow i \mid l \leftarrow l \mid i \leftarrow ii \mid \beta(j) \leftarrow i + 1 \mid n \wedge j \mid i + 1$$

42.

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{L}$$

$$m \quad n \quad a \quad b \quad k \quad i \quad j \quad l$$

$$B = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n' \quad a' \quad b' \quad k'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1 \wedge a = a' \wedge b = b' \wedge k = k')$$

$$R : \left( Q \wedge l = \exists p \in [m, n] : \exists q \in [a, b] : f(p) + g(q) = k \wedge \right. \\ \left. l \Rightarrow (i \in [m, n] \wedge j \in [a, b] \wedge f(i) + g(j) = k) \right)$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $a, b, k$  | Szigorítás: első ilyen találjuk meg

$$m \leftarrow m \mid n \leftarrow n \mid l \leftarrow l \mid i \leftarrow i \mid \beta(p) \leftarrow \exists q \in [a, b] : f(p) + g(q) = k$$

2. Visszavezetés: logaritmus keresés

Feltétel: - | Paraméter:  $a, b, i, k$  | Szigorítás: -

$$m \leftarrow a \mid n \leftarrow b \mid i \leftarrow j$$

$$k \leftarrow k \mid u \leftarrow u \mid v \leftarrow v$$

$$l \leftarrow l \mid a < b \leftarrow a \leq b \mid f(j) \leftarrow f(i + 1) + g(j)$$

43.

$$V = \text{vekt}(\mathbb{Z}, \text{vekt}(\mathbb{Z}, \mathbb{Z}))$$

$$g : V \times \mathbb{Z} \rightarrow \mathbb{L}$$

$$g(t, j) = \forall k \in [t.j.lob, t.j.hib] : t_{j,k} \leq 0$$

$$A = V \times \mathbb{L} \times \mathbb{Z}$$

$$t \quad l \quad i$$

$$B = V$$

$$t'$$

$$Q : (t = t')$$

$$R : (Q \wedge l = \exists j \in [t.lob, t.hib] : g(t, j) \wedge l \rightarrow (i \in [t.lob, t.hib] \wedge g(t, i)))$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $t$  | Szigorítás: -

Megfeleltetés:

$$m \leftarrow t.lob$$

$$n \leftarrow t.hib$$

$$l \leftarrow l$$

$$i \leftarrow i$$

$$\beta(j) \leftarrow g(t, j)$$

2. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $t, i$  | Szigorítás: -

$i, l := t.lob - 1, \text{hamis}$
$\neg l \wedge i \neq t.hib$
$l := g(t, i + 1)$
$i := i + 1$

Megfeleltetés:

$$\begin{aligned} m &\leftarrow t_{i+1}.lob \\ n &\leftarrow t_{i+1}.hib \\ l &\leftarrow \neg l \\ i &\leftarrow ii \\ \beta(j) &\leftarrow t_{i+1,j} > 0 \end{aligned}$$

44.

$$\begin{aligned} V &= \text{vekt}(\mathbb{Z}, C) \\ g &: V \times V \times \mathbb{Z} \rightarrow \mathbb{L} \\ g(x, b, j) &= \forall k \in [0, b.dom - 1] : x_{j+k} = b_{b.lob+k} \end{aligned}$$

$$A = V \times V \times \mathbb{L}$$

$$x \quad b \quad l$$

$$B = V \times V$$

$$x' \quad b'$$

$$Q : (x = x' \wedge b = b' \wedge b.dom \leq x.dom)$$

$$R : (Q \wedge l = \exists j \in [x.lob, x.hib - b.dom] : g(x, b, j))$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $x, b$  | Szigorítás: adjuk meg az első előfordulás helyét

Megfeleltetés:

$$\begin{aligned} m &\leftarrow x.lob \\ n &\leftarrow x.hib - b.dom \\ l &\leftarrow l \\ i &\leftarrow i \\ \beta(j) &\leftarrow g(x, b, j) \end{aligned}$$

2. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $x, b, i$  | Szigorítás: -

Megfeleltetés:

$$\begin{aligned} m &\leftarrow 0 \\ n &\leftarrow b.dom - 1 \\ l &\leftarrow \neg l \\ i &\leftarrow ii \\ \beta(j) &\leftarrow x_{i+1+j} \neq b_{b.lob+j} \end{aligned}$$

45.

$$\begin{aligned} g &: \mathbb{Z} \rightarrow \mathbb{L} \\ g(j) &= \exists k \in [m, j - 1] : f(j) < f(k) \end{aligned}$$

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{L} \times \mathbb{Z}$$

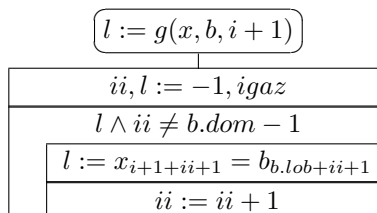
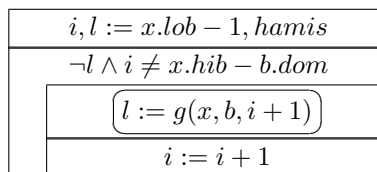
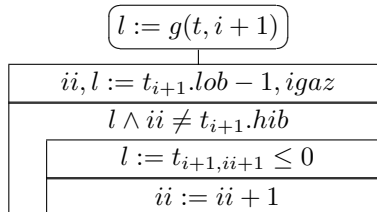
$$m \quad n \quad l \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R : (Q \wedge l = \exists j \in [m, n] : g(j) \wedge l \rightarrow (i \in [m, n] \wedge g(i)))$$



1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter: - | Szigorítás: első ilyen értéket adjuk meg

$$m \leftarrow m \mid n \leftarrow n \mid l \leftarrow l \mid i \leftarrow i \mid \beta(j) \leftarrow g(j)$$

2. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $i$  | Szigorítás: első ilyen értéket keressük

$$m \leftarrow m \mid n \leftarrow i \mid l \leftarrow l \mid i \leftarrow ii \mid \beta(j) \leftarrow f(i+1) < f(j)$$

**46.**

$$g : \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{L}$$

$$g(j, n) = \exists k \in [j+1, n] : f(j) = f(k)$$

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{L} \times \mathbb{Z}$$

$$m \quad n \quad l \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R : \left( Q \wedge l = \exists j \in [m, n-1] : g(j, n) \wedge \right. \\ \left. l \rightarrow (i \in [m, n-1] \wedge g(i, n) \wedge \forall j \in [m, i-1] : \neg g(j, n)) \right)$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel:  $m \leq n$  | Paraméter: - | Szigorítás: -

$$m \leftarrow m \mid n \leftarrow n-1 \mid l \leftarrow l \mid i \leftarrow i \mid \beta(j) \leftarrow g(j, n)$$

2. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $i, n$  | Szigorítás: adjuk meg, hol fordul elő az elem másodszor

$$m \leftarrow i+2 \mid n \leftarrow n \mid l \leftarrow l \mid i \leftarrow ii \mid \beta(j) \leftarrow f(i+1) = f(j)$$

**47.**

$$M = \text{vekt}(\mathbb{Z}, \text{vekt}(\mathbb{Z}, \mathbb{Z}))$$

$$g : M \times \mathbb{Z} \rightarrow \mathbb{L}$$

$$g(x, j) = \forall k \in [x_j.\text{lob}, x_j.\text{hib}] : t_{j,k} = 1$$

$$A = M \times \mathbb{Z} \times \mathbb{L}$$

$$x \quad i \quad l$$

$$B = M$$

$$x'$$

$$Q : (x = x')$$

$$R : (Q \wedge l = \exists j \in [x.\text{lob}, x.\text{hib}] : g(x, j) \wedge l \rightarrow (i \in [x.\text{lob}, x.\text{hib}] \wedge g(x, i)))$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $x$  | Szigorítás: első ilyen sort keressük meg

$$m \leftarrow x.\text{lob} \mid n \leftarrow x.\text{hib} \mid l \leftarrow l \mid i \leftarrow i \mid \beta(j) \leftarrow g(x, j)$$

2. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $x, i$  | Szigorítás: adjuk meg, hol lesz először nem 1

$$m \leftarrow x_{i+1}.\text{lob} \mid n \leftarrow x_{i+1}.\text{hib} \mid l \leftarrow \neg l \mid i \leftarrow ii \mid \beta(j) \leftarrow x_{i+1,j} \neq 1$$

**48.**

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{N}_0$$

$$\begin{array}{ccc} a & b & d \end{array}$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$\begin{array}{cc} a' & b' \end{array}$$

$$Q : (a = a' \wedge b = b' \wedge a \leq b + 1)$$

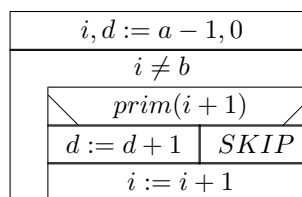
$$R : \left( Q \wedge d = \sum_{j=a}^b \chi(\text{prim}(j)) \right)$$

1. Visszavezetés: számlálás

Feltétel: - | Paraméter: - | Szigorítás: -

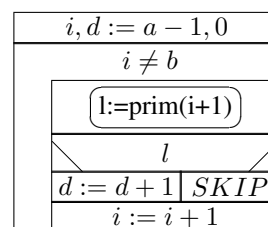
Megfeleltetés:

$$\begin{array}{l} m \leftarrow a \\ n \leftarrow b \\ i \leftarrow i \\ d \leftarrow d \\ \beta(j) \leftarrow \text{prim}(j) \end{array}$$



2. Nem megengedett elágazásfeltétel kitranszformálása

$$l := \text{prim}(i + 1)$$

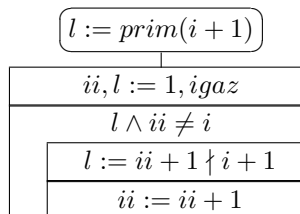


3. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $i$  | Szigorítás: hol lesz „nem prím”

Megfeleltetés:

$$\begin{array}{l} m \leftarrow 2 \\ n \leftarrow i \\ l \leftarrow \neg l \\ i \leftarrow ii \\ \beta(j) \leftarrow j \mid i + 1 \end{array}$$

**49.**

$$A = \mathbb{N} \times \mathbb{N}_0$$

$$\begin{array}{cc} n & d \end{array}$$

$$B = \mathbb{N}$$

$$n'$$

$$Q : (n = n')$$

$$R : \left( Q \wedge d = \sum_{j=2}^{n-1} \chi(\text{rprim}(j, n)) \right)$$

1. Visszavezetés: számlálás

Feltétel: - | Paraméter: - | Szigorítás: -

$$m \leftarrow 2 \mid n \leftarrow n - 1 \mid i \leftarrow i \mid d \leftarrow d \mid \beta(j) \leftarrow rprim(j, n)$$

2. Nem megengedett elágazásfeltétel kitranszformálása

$$l := rprim(i + 1, n)$$

3. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $i, n$  | Szigorítás: hol lesz először „nem relatív prím”

$$m \leftarrow 2 \mid n \leftarrow i + 1 \mid l \leftarrow \neg l \mid i \leftarrow ii \mid \beta(j) \leftarrow j \mid i + 1 \wedge j \mid n$$

50.

$$V = \text{vekt}(\mathbb{Z}, S_z)$$

$$g : V \times V \times \mathbb{N} \rightarrow \mathbb{N}$$

$$g(a, b, j) = \sum_{k=1}^{a.dom} \chi(a_{((j+k-1) \bmod a.dom)+1} = b_k)$$

$$A = V \times V \times \mathbb{N}_0$$

$$a \quad b \quad i$$

$$B = V \times V$$

$$a' \quad b'$$

$$Q : (a = a' \wedge b = b' \wedge a.lob = b.lob = 1 \wedge 1 < a.dom = b.dom)$$

$$R : (Q \wedge i \in [a.lob, a.hib] \wedge \forall j \in [a.lob, a.hib] : g(a, b, j) \leq g(a, b, i))$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter:  $a, b$  | Szigorítás: mondjuk meg, hány helyen egyeznek meg

Megfeleltetés:

$$m \leftarrow a.lob$$

$$n \leftarrow a.hib$$

$$max \leftarrow max$$

$$i \leftarrow i$$

$$k \leftarrow k$$

$$a \leq b \leftarrow a \leq b$$

$$f(j) \leftarrow g(a, b, j)$$

$max, i, k := g(a, b, a.lob), a.lob, a.lob$	
$k \neq a.hib$	
$max \leq g(a, b, k + 1)$	
$max, i := g(a, b, k + 1), k + 1$	SKIP
$k := k + 1$	

2. Függvény helyettesítése változóval

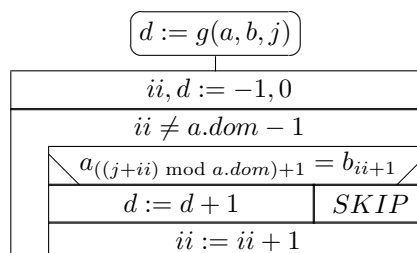
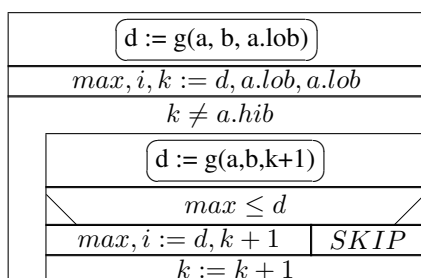
$$d := g(a, b, j)$$

3. Visszavezetés: számlálás

Feltétel: - | Paraméter:  $a, b, j$  | Szigorítás: -

$$m \leftarrow 0 \mid n \leftarrow a.dom - 1 \mid i \leftarrow ii \mid d \leftarrow d \mid \beta(j) \leftarrow a_{((j+i-1) \bmod a.dom)+1} = b_j$$



**51.**

$$g : \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}$$

$$g(m, n, j) = \sum_{k=m}^n \chi(f(j, k) = 0)$$

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$m \quad n \quad d$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n + 1)$$

$$R : \left( Q \wedge d = \sum_{j=m}^n g(m, n, j) \right)$$

1. **Visszavezetés: összegzés**

Feltétel: - | Paraméter: - | Szigorítás: -

$$m \leftarrow m \mid n \leftarrow n \mid d \leftarrow d \mid i \leftarrow i \mid f(j) \leftarrow g(m, n, j)$$

2. **Függvény helyettesítése változóval** ( $z := g(m, n, i + 1)$ )3. **Visszavezetés: számlálás**Feltétel: - | Paraméter:  $m, n, i$  | Szigorítás: -

$$m \leftarrow m \mid n \leftarrow n \mid i \leftarrow ii \mid d \leftarrow z \mid \beta(j) \leftarrow f(i + 1, j) = 0$$

**52.**

$$V = \text{vekt}(\mathbb{Z}, \text{vekt}(\mathbb{Z}, \mathbb{Z}))$$

$$g : V \times \mathbb{Z} \rightarrow \mathbb{Z}$$

$$g(t, j) = \sum_{k=t.lob}^{t.hib} \chi(t_{j,k} \neq 0)$$

$$A = V \times \mathbb{N}_0$$

$$t \quad d$$

$$B = V$$

$$t'$$

$$Q : (t = t')$$

$$R : \left( Q \wedge d = \sum_{j=t.lob}^{t.hib} \chi(g(t, j) = 1) \right)$$

1. **Visszavezetés: számlálás**

Feltétel: - | Paraméter:  $t$  | Szigorítás: -  
 $m \leftarrow t.lob \mid n \leftarrow t.hib \mid i \leftarrow i \mid d \leftarrow d \mid \beta(j) \leftarrow g(t, j) = 1$   
 2. Függvény helyettesítése változóval ( $z := g(t, i + 1)$ )  
 3. Visszavezetés: számlálás  
 Feltétel: - | Paraméter:  $t, i$  | Szigorítás: -  
 $m \leftarrow t_{i+1}.lob \mid n \leftarrow t_{i+1}.hib \mid i \leftarrow ii \mid d \leftarrow z \mid \beta(j) \leftarrow t_{i+1,j} \neq 0$

53.

$$g : \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}$$

$$g(j, m, n) = \sum_{k=m}^n \chi(f(k) = f(j))$$

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$m \quad n \quad max$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n)$$

$$R : (Q \wedge \exists i \in [m, n] : max = g(i, m, n) \wedge \forall j \in [m, n] : g(j, m, n) \leq max)$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter: - | Szigorítás: adjuk meg, hol veszi fel ezt az értéket

$$\begin{array}{l|l|l} m \leftarrow m & n \leftarrow n & max \leftarrow max \\ i \leftarrow i & k \leftarrow k & a \leq b \leftarrow a \leq b \end{array}$$

$$f(j) \leftarrow g(j, m, n)$$

2. Függvény helyettesítése változóval ( $z := g(k + 1, m, n)$ )

3. Visszavezetés: számlálás

Feltétel: - | Paraméter: - | Szigorítás: -

$$m \leftarrow m \mid n \leftarrow n \mid i \leftarrow ii \mid d \leftarrow z \mid \beta(j) \leftarrow f(j) = f(k + 1)$$

54.

$$g : \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}$$

$$g(m, j) = \sum_{k=m}^{j-1} \chi(f(j) < f(k))$$

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$m \quad n \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n)$$

$$R : (Q \wedge \exists i \in [m, n] \wedge \forall j \in [m, n] : g(m, j) \leq g(m, i))$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter: - | Szigorítás: adjuk meg, hány elem előzi meg

$$\begin{array}{l|l|l} m \leftarrow m & n \leftarrow n & max \leftarrow max \\ i \leftarrow i & k \leftarrow k & a \leq b \leftarrow a \leq b \end{array}$$

$$f(j) \leftarrow g(m, j)$$

2. Függvény helyettesítése változóval ( $z := g(m, k + 1)$ )

3. Visszavezetés: számlálás

Feltétel: - | Paraméter:  $k, m$  | Szigorítás: -

$m \leftarrow m$  |  $n \leftarrow k$  |  $i \leftarrow ii$  |  $d \leftarrow z$  |  $\beta(j) \leftarrow f(k + 1) < f(j)$

55.

$M = \text{vekt}(\mathbb{Z}, V)$     $V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$

$I_M(x) = \forall i \in [x.lob, x.hib] : (x_i.dom = x.dom \wedge x_i.lob = x.lob)$

$g : M \times \mathbb{Z} \rightarrow \mathbb{L}$

$g(t, j) = \sum_{m=t.lob}^{j-1} t_{m,j}$

$A = M \times \mathbb{Z} \times \mathbb{Z}$   
 $t \quad max \quad i$

$B = M$   
 $t'$

$Q : (t = t')$

$R : (Q \wedge i \in [t.lob, t.hib] \wedge max = g(t, i) \wedge \forall j \in [t.lob, t.hib] : g(t, j) \leq max)$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter:  $t$  | Szigorítás: -

$m \leftarrow t.lob$  |  $n \leftarrow t.hib$  |  $max \leftarrow max$   
 $i \leftarrow i$  |  $k \leftarrow k$  |  $a \leq b \leftarrow a \leq b$

$f(j) \leftarrow g(t, j)$

$max, i, k := g(t, t.lob), t.lob, t.lob$	
$k \neq t.hib$	
$max \leq g(t, k + 1)$	
$max, i := g(t, k + 1), k + 1$	<i>SKIP</i>
$k := k + 1$	

$z := g(t, t.lob)$
$max, i, k := z, t.lob, t.lob$
$k \neq t.hib$
$z := g(t, k + 1)$
$max \leq z$
$max, i := z, k + 1$   <i>SKIP</i>
$k := k + 1$

2. Függvény helyettesítése változóval

$z := g(t, k + 1)$

3. Visszavezetés: összegzés

Felt: - | Paraméter:  $t, k$  | Szigorítás: -

Megfeleltetés:

$m \leftarrow t.lob$

$n \leftarrow k$

$d \leftarrow z$

$i \leftarrow ii$

$f(j) \leftarrow t_{j,k+1}$

$z := g(t, k + 1)$
$ii, z := t.lob - 1, 0$
$ii \neq k$
$z := z + t_{ii+1,k+1}$
$ii := ii + 1$

56.

$$\begin{aligned}
M &= \text{vekt}(\mathbb{Z}, V) \\
V &= \text{vekt}(\mathbb{Z}, \mathbb{Z}) \\
I_M(x) &= \forall i \in [x.\text{lob}, x.\text{hib}] : (x_i.\text{dom} = x.\text{dom} \wedge x_i.\text{lob} = x.\text{lob}) \\
g &: M \times \mathbb{Z} \rightarrow \mathbb{Z} \\
g(t, j) &= \max \{t_{m,j} \mid m \in [j+1, t.\text{hib}]\}
\end{aligned}$$

$$A = M \times \mathbb{Z}$$

$$t \quad \text{max}$$

$$B = M$$

$$t'$$

$$Q : (t = t')$$

$$R : \left( Q \wedge \exists i \in [t.\text{lob}, t.\text{hib} - 1] : \text{max} = g(t, i) \wedge \right. \\ \left. \forall j \in [t.\text{lob}, t.\text{hib} - 1] : g(t, j) \leq \text{max} \right)$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter:  $t$  | Szigorítás: adjuk meg, melyik sorban van

$$\begin{array}{l|l|l}
m \leftarrow t.\text{lob} & n \leftarrow t.\text{hib} - 1 & \text{max} \leftarrow \text{max} \\
i \leftarrow i & k \leftarrow k & a \leq b \leftarrow a \leq b
\end{array}$$

$$f(j) \leftarrow g(t, j)$$

2. Függvény helyettesítése változóval ( $y := g(t, k+1)$ )

3. Visszavezetés: maximum keresés

Feltétel: - | Paraméter:  $k, t$  | Szigorítás: adjuk meg e helyét is

$$\begin{array}{l|l|l}
m \leftarrow k + 2 & n \leftarrow t.\text{hib} & \text{max} \leftarrow y \\
i \leftarrow ii & k \leftarrow kk & a \leq b \leftarrow a \leq b
\end{array}$$

$$f(j) \leftarrow t_{k+1, j}$$

57.

$$\begin{aligned}
V &= \text{vekt}(\mathbb{Z}, \text{vekt}(\mathbb{Z}, \mathbb{Z})) \\
g &: V \times \mathbb{Z} \rightarrow \mathbb{Z} \\
g(t, j) &= \sum_{k=t.\text{lob}}^{t.\text{hib}} \chi(t_{j,k} \neq 0)
\end{aligned}$$

$$A = V \times \mathbb{N}_0$$

$$t \quad d$$

$$B = V$$

$$t'$$

$$Q : (t = t')$$

$$R : \left( Q \wedge d = \sum_{j=t.\text{lob}}^{t.\text{hib}} \chi(g(t, j) = 1) \right)$$

1. Visszavezetés: számlálás

Feltétel: - | Paraméter:  $t$  | Szigorítás: -

$$m \leftarrow t.\text{lob} \mid n \leftarrow t.\text{hib} \mid i \leftarrow i \mid d \leftarrow d \mid \beta(j) \leftarrow g(t, j) = 1$$

2. Függvény helyettesítése változóval ( $z := g(t, i+1)$ )

3. Visszavezetés: számlálás

Feltétel: - | Paraméter:  $t, i$  | Szigorítás: -  
 $m \leftarrow t_{i+1}.lob$  |  $n \leftarrow t_{i+1}.hib$  |  $i \leftarrow ii$  |  $d \leftarrow z$  |  $\beta(j) \leftarrow t_{i+1,j} \neq 0$

58.

$$V = \text{vekt}(\mathbb{Z}, P)$$

$$P = (x : \mathbb{Z}, y : \mathbb{Z})$$

$$E = (p : P, q : P)$$

$$g : V \times \mathbb{Z} \rightarrow \mathbb{Z}$$

$$h : P \times P \rightarrow \mathbb{Z}$$

$$g(v, j) = m \in [v.lob, v.hib] : (\forall q \in [v.lob, v.hib] : h(v_j, v_q) \leq h(v_j, v_m))$$

$$h(p, q) = (p.x - q.x)^2 + (p.y - q.y)^2$$

$$A = V \times E$$

$$v \quad max$$

$$B = V$$

$$v'$$

$$Q : (v = v')$$

$$R : \left( Q \wedge \exists i \in [v.lob, v.hib] : max = (v_i, v_{g(v,i)}) \wedge \right. \\ \left. \forall j \in [v.lob, v.hib] : (v_j, v_{g(v,j)}) \leq max \right)$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter:  $v$  | Szigorítás: adjuk meg a vektorban elfoglalt helyet is

$$\begin{array}{l|l|l} m \leftarrow v.lob & n \leftarrow v.hib & max \leftarrow max \\ i \leftarrow i & k \leftarrow k & a \leq b \leftarrow h(a.p, a.q) \leq h(b.p, b.q) \end{array}$$

$$f(j) \leftarrow (v_j, v_{g(v,j)})$$

2. Függvény helyettesítése változóval

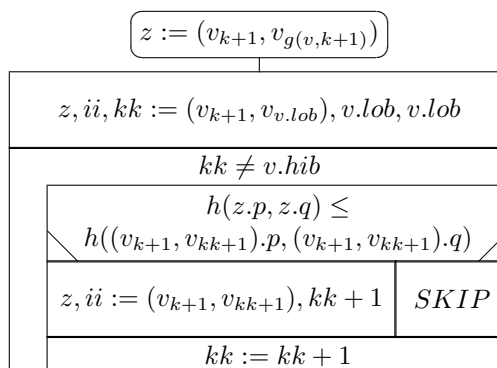
$$z := (v_{k+1}, v_{g(v,k+1)})$$

$max, i, k := (v_{v.lob}, v_{g(v,v.lob)}), v.lob, v.lob$	
$k \neq v.hib$	
$h(max.p, max.q) \leq$	
$h((v_{k+1}, v_{g(v,k+1)}) \cdot p, (v_{k+1}, v_{g(v,k+1)}) \cdot q)$	
$max, i := (v_{k+1}, v_{g(v,k+1)}), k + 1$	$SKIP$
$k := k + 1$	

$z := (v_{v.lob}, v_{g(v,v.lob)})$	
$max, i, k := z, t.lob, t.lob$	
$k \neq v.hib$	
$z := (v_{k+1}, v_{g(v,k+1)})$	
$max \leq h(z.p, z.q)$	
$max, i := z, k + 1$	$SKIP$
$k := k + 1$	

3. Visszavezetés: maximum keresés

Feltétel: - | Paraméter:  $v, k$  | Szigorítás: keressük a vektorban elfoglalt helyet is  
Megfeleltetés:

$$\begin{aligned}
m &\leftarrow v.lob \\
n &\leftarrow v.hib \\
max &\leftarrow z \\
i &\leftarrow ii \\
k &\leftarrow kk \\
a \leq b &\leftarrow h(a.p, a.q) \leq h(b.p, b.q) \\
f(j) &\leftarrow (v_{k+1}, v_j)
\end{aligned}$$


59.

$$\begin{aligned}
M &= \text{vekt}(\mathbb{Z}, V) \\
V &= \text{vekt}(\mathbb{Z}, \mathbb{Z}) \\
I_M(x) &= \forall i \in [x.lob, x.hib] : (x_i.dom = x.dom \wedge x_i.lob = x.lob) \\
g &: M \times \mathbb{Z} \rightarrow \mathbb{Z} \\
g(t, j) &= \sum_{k=t.lob}^{t.hib} t_{j,k}
\end{aligned}$$

$$A = M \times \mathbb{Z}$$

$$B = M$$

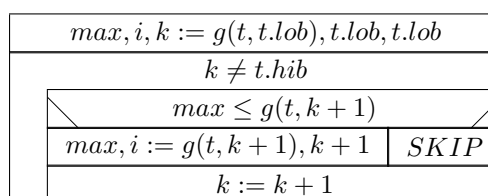
$$Q : (t = t')$$

$$R : (Q \wedge i \in [t.lob, t.hib] \wedge \forall j \in [t.lob, t.hib] : g(t, j) \leq g(t, i))$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter:  $t$  | Szigorítás: mondjuk meg, hány pontja van a nyertesnek

Megfeleltetés:

$$\begin{aligned}
m &\leftarrow t.lob \\
n &\leftarrow t.hib \\
max &\leftarrow max \\
i &\leftarrow i \\
k &\leftarrow k \\
a \leq b &\leftarrow a \leq b \\
f(j) &\leftarrow g(t, j)
\end{aligned}$$


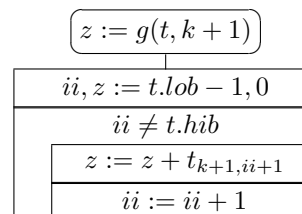
2. Függvény helyettesítése változóval

$$z := g(t, k + 1)$$

3. Visszavezetés: összegzés

Felt: - | Paraméter:  $t, k$  | Szigorítás: -

Megfeleltetés:

$$\begin{aligned}
m &\leftarrow t.lob \\
n &\leftarrow t.hib \\
d &\leftarrow z \\
i &\leftarrow ii \\
f(j) &\leftarrow t_{k+1, j}
\end{aligned}$$


**60.**

$$g : \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{L}$$

$$g(b, i) = \forall j \in [(i+1), b] : f(j) < f(i)$$

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{N}_0$$

$$\quad a \quad b \quad d$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$\quad a' \quad b'$$

$$Q : (a = a' \wedge b = b' \wedge a \leq b + 1)$$

$$R : \left( Q \wedge d = \sum_{j=a}^b \chi(g(b, j)) \right)$$

1. Visszavezetés: számlálás

Feltétel: - | Paraméter: - | Szigorítás: -

$$m \leftarrow a \mid n \leftarrow b \mid i \leftarrow i \mid d \leftarrow d \mid \beta(j) \leftarrow g(b, j)$$

2. Függvény helyettesítése változóval

$$l := g(b, i + 1)$$

3. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $i, b$  | Szigorítás: adjuk meg, hol tér el először

$$m \leftarrow i + 2 \mid n \leftarrow b \mid l \leftarrow \neg l \mid i \leftarrow ii \mid \beta(j) \leftarrow f(j) \geq f(i + 1)$$

**61.**

$$M = \text{vekt}(\mathbb{Z}, V)$$

$$V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$$

$$I_M(x) = \forall i \in [x.lob, x.hib] : (x_i.dom = x.dom \wedge x_i.lob = x.lob)$$

$$g : M \times \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{L}$$

$$g(t, j, k) = \begin{cases} 0, & \text{ha } j + k < 0 \vee t.dom \leq j + k; \\ t_{t.lob+j+k, t.lob+j}, & \text{ha } 0 \leq j + k < t.dom. \end{cases}$$

$$A = M \times \mathbb{Z} \times \mathbb{Z}$$

$$\quad t \quad max \quad i$$

$$B = M$$

$$\quad t'$$

$$Q : (t = t')$$

$$R : \left( Q \wedge i \in [-t.dom + 1, t.dom - 1] \wedge max = \sum_{p=0}^{t.dom-1} g(t, i, p) \wedge \right.$$

$$\left. \forall j \in [-t.dom + 1, t.dom - 1] : \sum_{p=0}^{t.dom-1} g(t, j, p) \leq max \right)$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter: - | Szigorítás: -

$$\begin{array}{l|l|l}
 m \leftarrow -t.dom + 1 & n \leftarrow t.dom - 1 & max \leftarrow max \\
 i \leftarrow i & k \leftarrow k & a \leq b \leftarrow a \leq b \\
 f(j) \leftarrow \sum_{p=0}^{t.dom-1} g(t, j, p) & & 
 \end{array}$$

$max, i, k := \sum_{p=0}^{t.dom-1} g(t, -t.dom + 1, p), -t.dom + 1, -t.dom + 1$	
$k \neq t.dom - 1$	
$max \leq \sum_{p=0}^{t.dom-1} g(t, k + 1, p)$	
$max, i := \sum_{p=0}^{t.dom-1} g(t, k + 1, p), k + 1$	<i>SKIP</i>
$k := k + 1$	

2. Függvény helyettesítése változóval ( $z := \sum_{p=0}^{t.dom-1} g(t, j, p)$ )

3. Visszavezetés: összegzés

Felt: - | Paraméter: - | Szigorítás:  $t, k$

Megfeleltetés:

$$\begin{array}{l}
 m \leftarrow 0 \\
 n \leftarrow t.dom - 1 \\
 d \leftarrow z \\
 i \leftarrow p \\
 f(j) \leftarrow g(t, k + 1, j)
 \end{array}$$

$$z := \sum_{p=0}^{t.dom-1} g(t, j, p)$$

$p, z := -1, 0$	
$p \neq t.dom - 1$	
$0 \leq t.lob + k + 1 + p + 1 < t.dom$	
$zz := t.lob + k + 1 + p + 1, t.lob + k + 1$	$zz := 0$
$z := z + zz$	
$p := p + 1$	

4. Függvény helyettesítése változóval

$$zz := g(t, k + 1, p + 1)$$

5. Visszavezetés: esetsztékválasztással definiált függvény helyettesítési értékének a kiszámolása ( $zz := g(t, k + 1, p + 1)$ )



62.

$$g : \mathbb{Z} \rightarrow (\mathbb{N}_0 \times \mathbb{Z} \times \mathbb{L})$$

$$g(m-1) = (0, m-1, \text{hamis})$$

$$g(i+1) = \begin{cases} (1, i+1, \text{igaz}), & \text{ha } f(i+1) = 1 \wedge g(i)_1 = 0; \\ (g(i)_1 + 1, g(i)_2, g(i)_3), & \text{ha } f(i+1) = 1 \wedge g(i)_1 > 0; \\ (0, g(i)_2, g(i)_3), & \text{ha } f(i+1) = 0. \end{cases}$$

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{L} \times \mathbb{Z}$$

$$m \quad n \quad l \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n)$$

$$R : \left( Q \wedge l = \exists j \in [m, n] : f(j) = 1 \wedge \right. \\ \left. l \rightarrow (i \in [m, n] \wedge \forall j \in [m, n] : g(j)_1 \leq g(i)_1) \right)$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter: - | Szigorítás: Adjuk meg mekkora a leghosszabb sorozat

Megfeleltetés:

$$m \leftarrow m-1$$

$$n \leftarrow n$$

$$\text{max} \leftarrow (\text{max}, i, l)$$

$$i \leftarrow j$$

$$k \leftarrow k$$

$$a \leq b \leftarrow a_1 \leq b_1$$

$$f(j) \leftarrow g(j)$$

$(\text{max}, i, l), j, k := g(m-1), m-1, m-1$	
$k \neq n$	
$(\text{max}, i, l)_1 \leq g(k+1)_1$	
$(\text{max}, i, l), j := g(k+1), k+1$	<i>SKIP</i>
$k := k+1$	

2. Rekurzív függvény helyettesítése változóval

$$z := g(k+1)$$

$z := (0, m-1, \text{hamis})$	
$(\text{max}, i, l), j, k := z, m-1, m-1$	
$k \neq n$	
$z := g(k+1)$	
$\text{max} \leq z_1$	
$(\text{max}, i, l), j := z, k+1$	<i>SKIP</i>
$k := k+1$	

3. Visszavezetés: Esetsztékválasztással definiált függvény helyettesítési értékének a kiszámítása

$$z := g(i + 1)$$

$z := (0, m - 1, hamis)$		
$(max, i, l), j, k := z, m - 1, m - 1$		
$k \neq n$		
$f(k + 1) = 1 \wedge z_1 = 0$	$f(k + 1) = 1 \wedge z_1 > 0$	$f(k + 1) = 0$
$z := (1, k + 1, igaz)$	$z_1 := z_1 + 1$	$z_1 := 0$
$max \leq z_1$		
$(max, i, l), j := z, k + 1$	<i>SKIP</i>	
$k := k + 1$		

**63.**

$$g : \mathbb{Z} \rightarrow \mathbb{N}_0$$

$$g(m - 1) = 0$$

$$g(i + 1) = g(i) + f(i + 1)$$

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{L}$$

$$m \quad n \quad l$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n)$$

$$R : (Q \wedge l = \exists j \in [m, n] : g(j) < 0)$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter: - | Szigorítás: adjuk meg, hol lesz először negatív

$$m \leftarrow m \mid n \leftarrow n \mid l \leftarrow l \mid i \leftarrow i \mid \beta(j) \leftarrow g(j) < 0$$

2. Rekurzív függvény helyettesítése változóval

$$z := g(i + 1)$$

**64.**

$$g : \mathbb{Z} \rightarrow \mathbb{N}_0$$

$$g(m - 1) = 0$$

$$g(i + 1) = \begin{cases} g(i) + 1, & \text{ha } f(i + 1) < 0; \\ 0, & \text{ha } f(i + 1) \geq 0. \end{cases}$$

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{N}_0$$

$$m \quad n \quad d$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n)$$

$$R : \left( Q \wedge d = \sum_{j=m}^n \chi(g(j) = 1) \right)$$

1. Visszavezetés: számlálás

Feltétel: - | Paraméter: - | Szigorítás: -

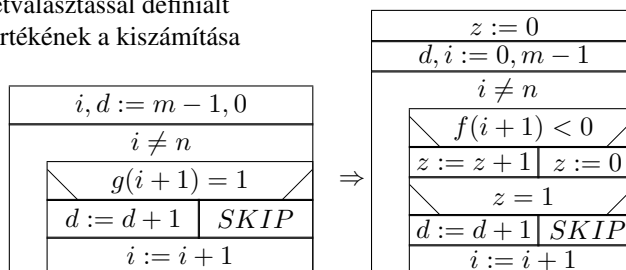
$$m \leftarrow m \mid n \leftarrow n \mid i \leftarrow i \mid d \leftarrow d \mid \beta(j) \leftarrow g(j) = 1$$

2. Rekurzív függvény helyettesítése változóval

$$z := g(i + 1)$$

3. Visszavezetés: Esetsztésválasztással definiált függvény helyettesítési értékének a kiszámítása

$$z := g(i + 1)$$



65.

$$g : \mathbb{N}_0 \rightarrow (\mathbb{N}_0, \mathbb{N}_0)$$

$$g(0) = (0, n)$$

$$g(i + 1) = \begin{cases} \left(1, \frac{g(i)_2 - 1}{2}\right), & \text{ha } 2 \nmid g(i)_2; \\ \left(0, \frac{g(i)_2}{2}\right), & \text{ha } 2 \mid g(i)_2. \end{cases}$$

$$A = \mathbb{N} \times \mathbb{N}$$

$$\begin{matrix} n & k \end{matrix}$$

$$B = \mathbb{N}$$

$$n'$$

$$Q : (n = n')$$

$$R : (Q \wedge g(k)_1 = 1)$$

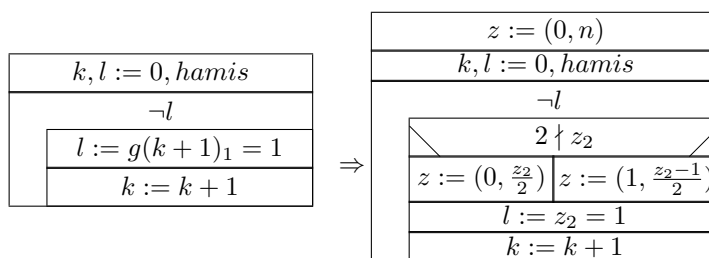
1. Visszavezetés: lineáris keresés 2.

Feltétel: - | Paraméter: - | Szigorítás: -

$$m \leftarrow 1 \mid l \leftarrow l \mid i \leftarrow k \mid \beta(j) \leftarrow g(j)_1 = 1$$

2. Rekurzív függvény helyettesítése változóval ( $z := g(k + 1)$ )

3. Visszavezetés: Esetsztésválasztással definiált függvény helyettesítési értékének a kiszámítása ( $z := g(k + 1)$ )



66.

$$\begin{aligned}
P &= (a : \mathbb{Z}, b : \mathbb{Z}) \\
g &: \mathbb{Z} \rightarrow (\mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{Z}) \\
g(m) &= (1, m, m) \\
g(i+1) &= \begin{cases} (g(i)_1 + 1, g(i)_2, i + 1), & \text{ha } f(i) < f(i+1); \\ (1, i + 1, i + 1), & \text{ha } f(i) \geq f(i+1). \end{cases}
\end{aligned}$$

$$A = \mathbb{Z} \times \mathbb{Z} \times P$$

$$m \quad n \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$m' \quad n'$$

$$Q : (m = m' \wedge n = n' \wedge m \leq n)$$

$$R : (Q \wedge i.a, i.b \in [m, n] \wedge i.a = g(i.b)_2 \wedge \forall k \in [m, n] : g(k)_1 \leq g(i.b)_1)$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter: - | Szigorítás: adjuk meg a leghosszabb szakasz hosszát

$$m \leftarrow m - 1 \quad \left| \quad n \leftarrow n \quad \right| \quad \text{max} \leftarrow (\text{max}, i_a, i_b)$$

$$i \leftarrow i \quad \left| \quad k \leftarrow k \quad \right| \quad a \leq b \leftarrow a_1 \leq b_1$$

$$f(j) \leftarrow g(j)$$

2. Rekurzív függvény helyettesítése változóval ( $z := g(k+1)$ )

3. Visszavezetés: Esetsztévválasztással definiált függvény helyettesítési értékének a kiszámítása ( $z := g(k+1)$ )

**67.**

$$g : \mathbb{N}_0 \rightarrow (\mathbb{N}_0, \mathbb{N}_0)$$

$$g(0) = (0, x)$$

$$g(i+1) = \begin{cases} \left(1, \frac{g(i)_2 - 1}{2}\right), & \text{ha } 2 \nmid g(i)_2; \\ \left(0, \frac{g(i)_2}{2}\right), & \text{ha } 2 \mid g(i)_2. \end{cases}$$

$$A = \mathbb{N} \times \mathbb{N}_0$$

$$x \quad d$$

$$B = \mathbb{N}$$

$$x'$$

$$Q : (x = x')$$

$$R : \left( Q \wedge d = \sum_{j=1}^{\lceil \log_2 n \rceil} \chi(g(j)_1 = 1) \right)$$

1. Visszavezetés: számlálás

Feltétel: - | Paraméter: - | Szigorítás: -

$$m \leftarrow 1 \quad \left| \quad n \leftarrow \lceil \log_2 n \rceil \quad \right| \quad i \leftarrow i \quad \left| \quad d \leftarrow d \quad \right| \quad \beta(j) \leftarrow g(j)_1 = 1$$

2. Rekurzív függvény helyettesítése változóval ( $z := g(i+1)$ )

3. Visszavezetés: Esetsztévválasztással definiált függvény helyettesítési értékének a kiszámítása ( $z := g(i+1)$ )

**68.**

$$\begin{aligned}
V &= \text{vekt}(\mathbb{Z}, C) \\
g &: \mathbb{Z} \rightarrow (\mathbb{N}_0 \times \mathbb{Z}) \\
g(x.lob - 1) &= (0, x.lob) \\
g(i + 1) &= \begin{cases} (0, i + 2), & \text{ha } x_{i+1} = ' '; \\ (g(i)_1 + 1, g(i)_2), & \text{ha } x_{i+1} \neq ' '. \end{cases}
\end{aligned}$$

$$A = V \times \mathbb{Z}$$

$$x \quad i$$

$$B = V$$

$$x'$$

$$Q : (x = x')$$

$$R : (Q \wedge i \in [x.lob, x.hib] \wedge \forall j \in [x.lob, x.hib] : g(j)_1 \leq g(i)_1)$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter:  $x$  | Szigorítás: adjuk meg, mekkora a leghosszabb szó

Megfeleltetés:

$$m \leftarrow x.lob - 1$$

$$n \leftarrow x.hib$$

$$max \leftarrow (max, i)$$

$$i \leftarrow j$$

$$k \leftarrow k$$

$$a \leq b \leftarrow a_1 \leq b_1$$

$$f(j) \leftarrow g(j)$$

$(max, i), j, k := g(x.lob - 1), x.lob - 1, x.lob - 1$	
$k \neq x.hib$	
$(max, i)_1 \leq g(k + 1)_1$	
$(max, i), j := g(k + 1), k + 1$	<i>SKIP</i>
$k := k + 1$	

2. Rekurzív függvény helyettesítése változóval ( $z := g(k + 1)$ )

3. Visszavezetés: Esetsztékválasztással definiált függvény helyettesítési értékének a kiszámítása ( $z := g(k + 1)$ )

$z := g(x.lob - 1)$	$z := (0, x.lob)$
$(max, i), j, k := z, x.lob - 1, x.lob - 1$	$(max, i), j, k := z, x.lob - 1, x.lob - 1$
$k \neq x.hib$	
$z := g(k + 1)$	
$max \leq z_1$	
$(max, i), j := z, k + 1$	<i>SKIP</i>
$k := k + 1$	

$x_{i+1} = ' '$	
$z := (0, i + 2)$	$z_1 := z_1 + 1$
$max \leq z_1$	
$(max, i), j := z, k + 1$	<i>SKIP</i>
$k := k + 1$	

69.

$$\begin{aligned}
g &: \mathbb{Z} \rightarrow (\mathbb{N}_0 \times \mathbb{Z}) \quad g(a) = (0, a) \\
g(i + 1) &= \begin{cases} (0, i + 1), & \text{ha } (f(i + 1) < f(i) \wedge f(i + 1) < f(i + 2)) \\ & \wedge (i + 1 \neq b) \\ (g(i)_1 + 1, g(i)_2), & \text{ha } (f(i + 1) \geq f(i) \vee f(i + 1) \geq f(i + 2)) \\ & \vee (i + 1 = b) \end{cases}
\end{aligned}$$

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$a \quad b \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$a' \quad b'$$

$$Q : (a = a' \wedge b = b' \wedge a \leq b + 1)$$

$$R : (Q \wedge i \in [a, b] \wedge \forall j \in [a, b] : g(j)_1 \leq g(i)_1)$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter: - | Szigorítás: adjuk meg, mekkora a legszélesebb hegy

Megfeleltetés:

$$\begin{aligned} m &\leftarrow a \\ n &\leftarrow b \\ max &\leftarrow (max, i) \\ i &\leftarrow i \\ k &\leftarrow k \\ a \leq b &\leftarrow a_1 \leq b_1 \\ f(j) &\leftarrow g(j) \end{aligned}$$

$(max, i), i, k := g(a), a, a$	
$k \neq b$	
$(max, i)_1 \leq g(k+1)_1$	
$(max, i), i :=$ $g(k+1), k+1$	<i>SKIP</i>
$k := k+1$	

2. Rekurzív függvény helyettesítése változóval

$$z := g(k+1)$$

3. Visszavezetés: Esetszétválasztással definiált függvény helyettesítési értékének a kiszámítása ( $z := g(k+1)$ )

$z := (0, a)$	
$(max, i), j, k := z, a, a$	
$k \neq b$	
$f(i+1) < f(i) \wedge$ $f(i+1) < f(i+2) \wedge$ $i+1 \neq b$	$f(i+1) \geq f(i) \vee$ $f(i+1) \geq f(i+2) \vee$ $i+1 = b$
$z := (0, k+1)$	$z_1 := z_1 + 1$
$max \leq z_1$	
$(max, i), j := z, k+1$	<i>SKIP</i>
$k := k+1$	

**70.**

$$\begin{aligned}
V &= \text{vekt}(\mathbb{Z}, \mathbb{Z}) \\
g &: \mathbb{Z} \rightarrow (\mathbb{N}_0 \times \mathbb{Z} \times \mathbb{L}) \\
g(x.\text{lob} - 1) &= (0, x.\text{lob} - 1, \text{hamis}) \\
g(i + 1) &= \begin{cases} (g(i)_1 + 1, g(i)_2, igaz), & \text{ha } x_{i+1} < 0; \\ (0, i + 2, g(i)_3), & \text{ha } x_{i+1} \geq 0. \end{cases}
\end{aligned}$$

$$A = V \times \mathbb{L} \times \mathbb{Z}$$

$$x \quad l \quad i$$

$$B = V$$

$$x'$$

$$Q : (x = x')$$

$$R : \left( Q \wedge l = \exists j \in [x.\text{lob}, x.\text{hib}] : x_j < 0 \wedge \right. \\ \left. l \rightarrow (i \in [x.\text{lob}, x.\text{hib}] \wedge \forall j \in [x.\text{lob}, x.\text{hib}] : g(j)_1 \leq g(i)_1) \right)$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter:  $x$  | Szigorítás: adjuk meg mekkora a leghosszabb sorozat

$$m \leftarrow x.\text{lob} - 1 \quad \left| \quad n \leftarrow x.\text{hib} \quad \left| \quad \begin{array}{l} \text{max} \leftarrow (\text{max}, i, l) \\ a \leq b \leftarrow a_1 \leq b_1 \end{array} \right. \right.$$

$$i \leftarrow j \quad \left| \quad k \leftarrow k \quad \left| \quad a \leq b \leftarrow a_1 \leq b_1 \right. \right.$$

$$f(j) \leftarrow g(j)$$

2. Rekurzív függvény helyettesítése változóval ( $z := g(k + 1)$ )3. Visszavezetés: Esetszétválasztással definiált függvény helyettesítési értékének a kiszámítása ( $z := g(k + 1)$ )**71.**

$$g : \mathbb{Z} \rightarrow \mathbb{N}$$

$$g(a - 1) = 0$$

$$g(i + 1) = g(i) + x(i) - y(i)$$

$$A = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$

$$a \quad b \quad i$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$a' \quad b'$$

$$Q : (a = a' \wedge b = b' \wedge a \leq b + 1)$$

$$R : (Q \wedge i \in [a - 1, b] \wedge \forall j \in [a - 1, b] : g(j) \leq g(i))$$

1. Visszavezetés: maximum keresés

Feltétel: - | Paraméter: - | Szigorítás: adjuk meg a látogatók számát is

$$m \leftarrow a - 1 \quad \left| \quad n \leftarrow b \quad \left| \quad \begin{array}{l} \text{max} \leftarrow \text{max} \\ a \leq b \leftarrow a \leq b \end{array} \right. \right.$$

$$i \leftarrow i \quad \left| \quad k \leftarrow k \quad \left| \quad a \leq b \leftarrow a \leq b \right. \right.$$

$$f(j) \leftarrow g(j)$$

2. Rekurzív függvény helyettesítése változóval ( $z := g(k + 1)$ )

72.

$$\begin{aligned}
V &= \text{vekt}(\mathbb{Z}, \mathbb{Z}) \\
g &: \mathbb{Z} \rightarrow (\mathbb{Z} \times \mathbb{L} \times \mathbb{L}) \\
g(t.lob - 1) &= (0, \text{hamis}, \text{hamis}) \\
g(i + 1) &= \begin{cases} (0, \text{hamis}, \text{hamis}), & \text{ha } \neg g(i)_2 \wedge t_{i+1} \neq a \wedge t_{i+1} \neq b \\ (t_{i+1}, \text{igaz}, \text{hamis}), & \text{ha } \neg g(i)_2 \wedge (t_{i+1} = a \vee t_{i+1} = b) \\ (g(i)_1, \text{igaz}, \text{hamis}), & \text{ha } \pi_1 \\ (g(i)_1, \text{igaz}, \text{igaz}), & \text{egyébként} \end{cases} \\
\pi_1 &= (g(i)_2 \wedge (t_{i+1} = g(i)_1 \vee (t_{i+1} \neq a \wedge t_{i+1} \neq b))) \wedge \neg g(i)_3
\end{aligned}$$

$$A = V \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{L}$$

$$\begin{array}{ccccc}
t & a & b & k & l
\end{array}$$

$$B = V \times \mathbb{Z} \times \mathbb{Z}$$

$$\begin{array}{ccc}
t' & a' & b'
\end{array}$$

$$Q : (a = a' \wedge b = b' \wedge t = t')$$

$$R : \left( Q \wedge l = \exists j \in [t.lob, t.hib] : g(j)_3 \wedge \right. \\ \left. l \rightarrow (\exists j \in [t.lob, t.hib] : (g(j)_3 \wedge k = g(j)_1)) \right)$$

1. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter: - | Szigorítás: -

$$m \leftarrow t.lob \mid n \leftarrow t.hib \mid l \leftarrow l \mid i \leftarrow i \mid \beta(j) \leftarrow g(j)_3$$

2. Rekurzív függvény helyettesítése változóval ( $z := g(i + 1)$ )3. Visszavezetés: Esetszétválasztással definiált függvény helyettesítési értékének a kiszámítása ( $z := g(i + 1)$ )

$k, l_1, l_2 := 0, \text{hamis}, \text{hamis}$			
$i, l := t.lob - 1, \text{hamis}$			
$\neg l \wedge i \neq t.hib$			
$\neg l_2 \wedge (t_{i+1} \neq a \wedge t_{i+1} \neq b)$	$\neg l_2 \wedge (t_{i+1} = a \vee t_{i+1} = b)$	$l_2 \wedge \neg l_3 \wedge t_{i+1} = k \vee (t_{i+1} \neq a \vee t_{i+1} \neq b)$	$l_3 \vee l_2 \wedge t_{i+1} \neq k \wedge (t_{i+1} = a \vee t_{i+1} = b)$
$k, l_1, l_2 := 0, \text{hamis}, \text{hamis}$	$k, l_1, l_2 := t_{i+1}, \text{igaz}, \text{hamis}$	$k, l_1, l_2 := k, \text{igaz}, \text{hamis}$	$k, l_1, l_2 := k, \text{igaz}, \text{igaz}$
$l := l_2$			
$i := i + 1$			



**73.**

$$\varrho_0 = \text{igaz}$$

$$\varrho_i = \exists(x_1, \dots, x_i) : \left( \begin{array}{l} \forall j \in [1, i] : (x_i \in [1, n] \wedge \\ \forall k \in [1, j-1] : x_j \neq x_k \wedge x_j \pm i - k \neq x_k) \end{array} \right)$$

$$\varrho = \varrho_n$$

$$\mathcal{N} = \times_{i=1}^n \{1 \dots n\}$$

$$A = \mathbb{N} \times \mathbb{L} \times \mathcal{N}$$

$$n \quad l \quad x$$

$$B = \mathbb{N}$$

$$n'$$

$$Q : (n = n')$$

$$R : (Q \wedge l = \exists y \in \mathcal{N} : \varrho(\varphi(y)) \wedge l \rightarrow (\varrho(\varphi(x))))$$

1. Visszavezetés: visszalépéses keresés

Feltétel: - | Paraméter: n | Szigorítás: Az első ilyen értéket adjuk meg

Megfeleltetés:

$$\nu_i \leftarrow x_i \mid \delta_i \leftarrow n \mid \varphi \leftarrow Id$$

2. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter: x | Szigorítás: -

Megfeleltetés:

$$m \leftarrow 1$$

$$n \leftarrow m$$

$$l \leftarrow \neg l$$

$$i \leftarrow i$$

$$\beta(j) \leftarrow x_{m+1} = x_j \vee$$

$$x_{m+1} \pm |m+1-j| = x_j$$

$$l := \varrho_{m+1}(x)$$

$$i, l := 0, \text{igaz}$$

$$l \wedge i \neq m$$

$$l := x_{m+1} \neq x_{i+1} \wedge x_{m+1} \pm \\ |m+1-j| \neq x_{i+1}$$

$$i := i + 1$$

**74.**

$$\varrho_0 = \text{igaz}$$

$$\varrho_i = \exists(x_1, \dots, x_i) : \left( \begin{array}{l} \forall j \in [1, i] : (x_i \in [1, n] \wedge \\ \forall k \in [1, j-1] : x_j \neq x_k \wedge x_j \pm i - k \neq x_k) \end{array} \right)$$

$$\varrho = \varrho_n$$

$$\mathcal{N} = \times_{i=1}^n \{1 \dots n\}$$

$$A = \mathbb{N} \times \mathbb{L} \times \mathbb{N}_0$$

$$n \quad l \quad d$$

$$B = \mathbb{N}$$

$$n'$$

$$Q : (n = n' \wedge x.\text{dom} = n \wedge x.\text{lob} = 1)$$

$$R : \left( Q \wedge d = \sum_{\mu < (\nu_0, \nu)} : \varrho(\varphi(\mu)) \right)$$

1. Visszavezetés: visszalépéses számlálás

Feltétel: - | Paraméter:  $n$  | Szigorítás: -  
 Megfeleltetés:

$$\nu_i \leftarrow \nu_i \mid \delta_i \leftarrow n \mid \varphi \leftarrow Id$$

2. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $\nu$  | Szigorítás: Adjuk meg a helyét is

$$m \leftarrow 1 \mid n \leftarrow m \mid l \leftarrow \neg l \mid i \leftarrow i \mid \beta(j) \leftarrow \nu_{m+1} = \nu_j \vee \nu_{m+1} \pm |m+1-j| = \nu_j$$

**76.**

$$\varrho_0 = \text{igaz}$$

$$\varrho_i = \exists(x_1, \dots, x_i) : (\forall j \in [1, i] : t_{j, x_j} \wedge \forall k \in [1, j-1] : x_j \neq x_k)$$

$$\varrho = \varrho_{t.dom}$$

$$\mathcal{N} = \times_{i=1}^n \{1 \dots n\}$$

$$NNM = \text{vekt}(\mathbb{Z}, \text{vekt}(\mathbb{Z}, \mathbb{L}))$$

$$I_{NNM}(x) = \forall i \in [x.lob, x.hib] : (x_i.dom = x.dom \wedge x_i.lob = x.lob)$$

$$V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$$

$$A = \begin{matrix} NNM & \times & \mathbb{L} & \times & V \\ t & & l & & x \end{matrix}$$

$$B = \begin{matrix} NNM \\ t \end{matrix}$$

$$Q : (t = t' \wedge x.lob = t.lob = 1 \wedge x.dom = t.dom = n)$$

$$R : (Q \wedge l = \exists y \in \mathcal{N} : \varrho(\varphi(y)) \wedge l \rightarrow (\varrho(\varphi(x))))$$

1. Visszavezetés: visszalépéses keresés

Feltétel: - | Paraméter:  $t$  | Szigorítás: Az első ilyen értéket adjuk meg

Megfeleltetés:

$$\nu_i \leftarrow x_i \mid \delta_i \leftarrow t.dom \mid \varphi \leftarrow Id$$

2. Visszavezetés: lineáris keresés 2.8.

Feltétel: - | Paraméter:  $x, t$  | Szigorítás: Adjuk meg a helyét is

$$m \leftarrow 1 \mid n \leftarrow m \mid l \leftarrow \neg l \mid i \leftarrow i \mid \beta(j) \leftarrow \neg t_{m+1, x_{m+1}} \vee x_{m+1} = x_j$$

**82.**

$$H = (x : \mathbb{N}, y : \mathbb{N})$$

$$T = (h : H, e : \mathbb{N}, t : \mathbb{N}, fazon : \mathbb{N}, j : \mathbb{L})$$

$$F = \text{file}(T)$$

$$A = \begin{matrix} F & \times & \text{seq}(H) & \times & \text{seq}(T) \\ x & & t & & z \end{matrix}$$

$$B = \begin{matrix} F \\ x' \end{matrix}$$

$$Q : (x = x')$$

$$R : (t = f_1(x') \wedge z = f_2(x'))$$

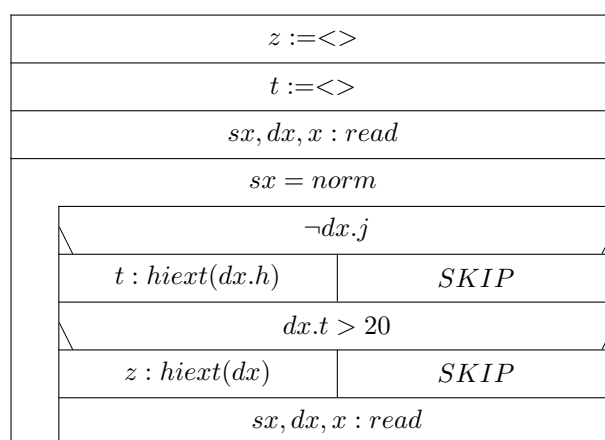
ahol  $f_1$  és  $f_2$  elemenként feldolgozható, és egy-egy elemet feldolgozó változtaik:

$e \in T$

$$f_1(\{e\}) = \begin{cases} \{e.h\}, & \text{ha } \neg e.j; \\ \emptyset & \text{egyébként.} \end{cases}$$

$$f_2(\{e\}) = \begin{cases} \{e\}, & \text{ha } e.t > 20; \\ \emptyset & \text{egyébként.} \end{cases}$$

Tehát ez egy egyváltozós kétértékű elemenkénti feldolgozás.



**83.**

$F = file(N)$

$A = \begin{matrix} F & \times & F & \times & F \\ k & & v & & z \end{matrix}$

$B = \begin{matrix} F & \times & F \\ k' & & v' \end{matrix}$

$Q : (k = k' \wedge v = v' \wedge k' \text{ növekvő} \wedge v' \text{ növekvő})$

$R : (z = f(k', v'))$

ahol  $f$  elemenként feldolgozható, és egy elemet feldolgozó változata:

$f(\{e\}, \emptyset) = \{e\}$

$f(\emptyset, \{e\}) = \emptyset$

$f(\{e\}, \{e\}) = \emptyset$

Tehát ez egy kétváltozós egyértékű elemenkénti feldolgozás. Méghozzá (a feladat betűzését használva és fájlokra átírva) az alábbi:

$z := \langle \rangle$		
$sk, dk, k : read$		
$sv, dv, v : read$		
$sk = norm \vee sv = norm$		
$sv \neq norm \vee$ $(sk = norm \wedge$ $sv = norm \wedge$ $dk < dv)$	$sk = norm \wedge$ $sv = norm \wedge$ $dk = dv$	$sk \neq norm \vee$ $(sk = norm \wedge$ $sv = norm \wedge$ $dk > dv)$
$z : hiezt(dk)$	$sk, dk, k : read$	$sv, dv, v : read$
$sk, dk, k : read$	$sv, dv, v : read$	

84.

Tekintsük az alábbi függvényt:  $H = \{0, 1\} \times \mathbb{N}_0 \times \mathbb{Z}, f : \mathbb{Z} \rightarrow H$ 

$$f(x) = \begin{cases} (1, x, x), & \text{ha } x > 0; \\ (0, -x, x), & \text{ha } x \leq 0. \end{cases}$$

$$F = file(\mathbb{Z})$$

$$A = F \times H$$

$$x \quad max$$

$$B = F$$

$$x'$$

$$Q : (x = x' \wedge x'.dom \geq 1)$$

$$R : (\exists i \in [1..x'.dom] : f(x'_i) = max \wedge \forall j \in [1..x'.dom] : f(x'_j) \tilde{\leq} max)$$

ahol az  $H$  típuson értelmezett rendezés a  $\tilde{\leq}$ , oly módon, hogy az első komponenset veszi figyelembe legerősebben, a másodikat „kevésbé”. A keresett számot  $max$  3. komponense tartalmazza. Látva ezt az utófeltételt, megpróbálhatjuk a feladatot maximumkeresésre visszavezetni, ehhez először írjuk fel a maximumkeresést sorozatokra:

$max := f(x.lov)$	
$x.dom \neq 0$	
$f(x.lov) \tilde{\geq} max$	
$max := f(x.lov)$	$SKIP$
$x : lorem$	

Innen már egyszerűbb az átírás fájlra. Tudjuk, hogy az  $x$  nem üres, ezért két előolvasást is végezhetünk. Valamint alkalmazzuk  $\leq$  kiszámításának módját, majd helyettesítsük  $f(dx)$ -et változóval (ekkor  $z$ -t már rekordnak is tekinthetjük):

$sx, dx, x : read$
$max := f(dx)$
$sx, dx, x : read$
$sx = norm$
$f(dx).1 > max.1 \vee f(dx).1 = max.1 \wedge f(dx).2 > max.2$
$max := f(dx)$
$SKIP$
$sx, dx, x : read$

$sx, dx, x : read$
$dx > 0$
$z := (1, dx, dx)$
$z := (0, -dx, dx)$
$max := z$
$sx, dx, x : read$
$sx = norm$
$dx > 0$
$z := (1, dx, dx)$
$z := (0, -dx, dx)$
$z.1 > max.1 \vee z.1 = max.1 \wedge z.2 > max.2$
$max := z$
$SKIP$
$sx, dx, x : read$

85.

$\mathcal{K} = (n : N, o : O, v : V, m : M)$

$F = file(\mathcal{K})$

$A = F \times F \times F$   
 $\quad \quad \quad x \quad y \quad z$

$B = F$   
 $\quad \quad \quad x'$

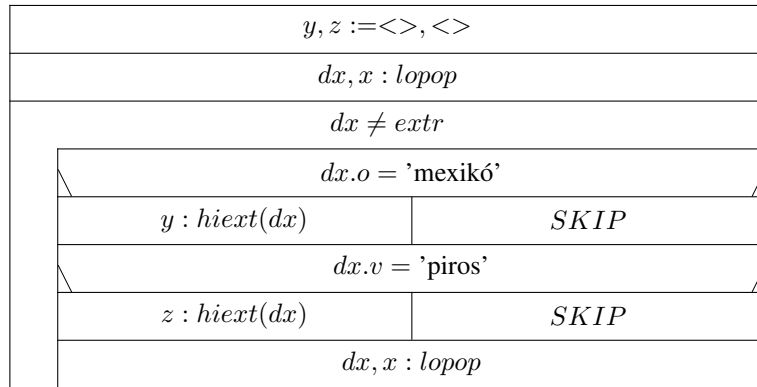
$Q : (x = x')$

$R : (y = f_1(x') \wedge z = f_2(x'))$

Egyváltozós kétértékű elemenkénti feldolgozásról van szó, ahol a két elemenként feldolgozható függvény egy-egy elemet feldolgozó változatai:

$$f_1(\{e\}) := \begin{cases} \emptyset, & \text{ha } e.o \neq \text{'mexikó'}; \\ \{e\}, & \text{ha } e.o = \text{'mexikó'}. \end{cases} \quad f_2(e) := \begin{cases} \emptyset, & \text{ha } e.v \neq \text{'piros'}; \\ \{e\}, & \text{ha } e.v = \text{'piros'}. \end{cases}$$

A megoldóprogram:



87.

$$NEV = \text{seq}(Ch)$$

$$F = \text{file}(\text{nev} : NEV, o : \mathbb{Z})$$

$$A = \begin{matrix} F & \times & NEV & \times & \mathbb{L} \\ x & & n & & l \end{matrix}$$

$$B = \begin{matrix} F \\ x' \end{matrix}$$

$$Q : (x = x')$$

$$R : (l = (\exists i \in [1..x'.dom] : x'_i.o \geq 0) \wedge l \rightarrow (\exists i \in [1..x'.dom] : (x'_i.o \geq 0 \wedge n = x'_i.nev \wedge \forall j \in [1..x'.dom] : (x'_j.o \geq 0 \rightarrow x'_i.o \leq x'_j.o))))$$

Feltételes maximumkeresésre vezethető vissza a feladat.

$l := hamis$			
$sx, dx, x : read$			
$sx = norm$			
$dx.o < 0$	$dx.o \geq 0$ $\wedge \neg l$	$dx.o \geq 0 \wedge l$	
$SKIP$	$l, max := igaz, dx.o$	$dx.o \geq max$	$dx.o \leq max$
	$n := dx.nev$	$SKIP$	$max := dx.o$
			$n := dx.nev$
$sx, dx, x : read$			

88.

 $F = file(\mathbb{Z})$  $V = vekt(\mathbb{Z}, \mathbb{N})$  $S = seq(\mathbb{Z})$  $A = V \times F \times S$  $B : V \times F$  $Q : (x = x' \wedge y = y' \wedge x' \text{ növekvő} \wedge y' \text{ növekvő} \wedge \forall j \in [y'.lob, y'.hib - 1] : y'_j > 0 \wedge y'.hib < 0)$  $R = (z = f(k', v'))$ ahol  $f$  elemenként feldolgozható, és egy elemet feldolgozó változata: $f(\{e\}, \emptyset) = \{e\}$  $f(\emptyset, \{e\}) = \{e\}$  $f(\{e\}, \{e\}) = \{e\}$ 

Tehát ez egy kétváltozós egyértékű elemenkénti feldolgozás.

$z := \langle \rangle$		
$dy, y : \text{lopop}$		
$k := x.\text{lob} - 1$		
$dy \geq 0 \vee k \neq x.\text{hib}$		
$dy < 0 \vee$ $(dy \geq 0 \wedge k \neq x.\text{hib}$ $\wedge x[k+1] < dy)$	$dy \geq 0 \wedge k \neq x.\text{hib}$ $\wedge x[k+1] = dy$	$k = x.\text{hib} \vee$ $(k \neq x.\text{hib} \wedge dy \geq 0$ $\wedge x[k+1] > dy)$
$z : \text{hiext}(x[k+1])$	$z : \text{hiext}(x[k+1])$	$z : \text{hiext}(dy)$
$k := k + 1$	$k := k + 1$	$dy, y : \text{lopop}$
	$dy, y : \text{lopop}$	

Megjegyzés: a középső ágba  $z : \text{hiext}(dy)$  is írható.



## 15.11. Absztrakciós stratégia

1.

Legyen

$$V = \text{vekt}(\mathbb{Z}, \mathbb{Z})$$

$$S = \text{seq}(\mathbb{Z})$$

$$A = \begin{matrix} V & \times & S \\ x & & y \end{matrix}$$

$$B = \begin{matrix} V \\ x' \end{matrix}$$

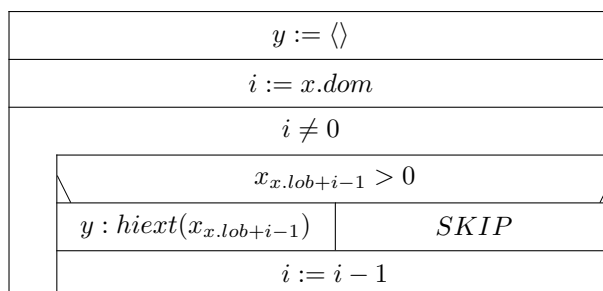
$$Q : (x = x')$$

$$R : (y = f(x'))$$

Tekintsük a vektor elemeit csupa különbözőnek. Ezt megtehetjük, hiszen minden eleme helyettesíthető egy (index, elem) párral. Legyen

$$f(x) = \bigcup_{e \in \{x\}} f(\{e\}) \text{ és } f(\{e\}) = \begin{cases} \emptyset, & \text{ha } e \leq 0; \\ \{e\}, & \text{ha } e > 0. \end{cases}$$

A 12.1 tétel alapján  $f$  elemenként feldolgozható függvény, így az egyváltozós elemenkénti feldolgozás vektorra és az esetszétválasztással definiált függvény kiszámításának tételét felhasználva:



3.

Legyen  $S = \text{seq}(D), T = \text{seq}(AZON), D = (\text{azon} : AZON, \text{sza} : SZA, \text{lc} : LC, \text{iv} : IV, \text{kezd} : \mathbb{N}, \text{beoszt} : BEOSZT, \text{fiz} : \mathbb{N}),$

$$A = \begin{matrix} S & \times & T & \times & \mathbb{N} & \times & S \\ x & & y & & ev & & z \end{matrix}$$

Transzformáljuk az állapotteret:

$$A' = S \times S \times \mathbb{N} \times S$$

$$\quad \quad \quad x \quad u \quad ev \quad z$$

$$B' = S \times S \times \mathbb{N}$$

$$\quad \quad \quad x' \quad u' \quad ev'$$

Az  $y$  és az  $u$  közötti kapcsolat:  $dom(u) = dom(y)$ , és  $\forall i \in [1..dom(u)] : u_i.azon = y_i.azon$ . Az  $u_i$  többi része megegyezik az  $x$   $u_i.azon$  azonosítójú elemének többi részével, ha van ilyen elem, bármi lehet egyébként.

$$Q : (ev = ev' \wedge x = x' \wedge u = u' \wedge \text{növekvő}(x') \wedge \text{növekvő}(u'))$$

$$R : (ev = ev' \wedge z = f(x', u'))$$

ahol  $\text{növekvő}(x') = \forall i \in [1..dom(x')] : x'_i.azon < x'_{i+1}.azon$ .

Az  $f$  függvény definíciója:

$$f(x, u) = \bigcup_{e \in \{x\} \cup \{u\}} f(s_1(e), s_2(e)), \text{ ahol}$$

$$s_1(e) = \begin{cases} \{e\}, & \text{ha } e \in \{x\}; \\ \emptyset, & \text{ha } e \notin \{x\}, \end{cases} \quad s_2(e) = \begin{cases} \{e\}, & \text{ha } e \in \{u\}; \\ \emptyset, & \text{ha } e \notin \{u\} \end{cases} \quad \text{és}$$

$$f(\{e\}, \emptyset) = \emptyset,$$

$$f(\emptyset, \{e\}) = \emptyset,$$

$$f(\{e\}, \{e\}) = \begin{cases} \{e\}, & \text{ha } e.kezd < ev; \\ \emptyset & \text{egyébként.} \end{cases}$$

Az egyszerűség kedvéért a dátumokat egy évszámnak tekintjük (azaz egy természetes számnak), és a hónapokat, napokat nem tároljuk.

Tehát ez egy kétváltozós egyértékű elemenkénti feldolgozás.

$z := \langle \rangle$		
$x.dom \neq 0 \vee u.dom \neq 0$		
$u.dom = 0 \vee$ $(u.dom \neq 0 \wedge$ $x.dom \neq 0 \wedge$ $x.loz.azon <$ $u.loz.azon)$	$x.dom \neq 0 \wedge$ $u.dom \neq 0 \wedge$ $x.loz.azon =$ $u.loz.azon$	$x.dom = 0 \vee$ $(x.dom \neq 0 \wedge$ $u.dom \neq 0 \wedge$ $x.loz.azon >$ $u.loz.azon)$
$x : lorem$	$x.loz.kezd < ev$	$u : lorem$
	$z : hiext(x.loz) SKIP$	
	$x : lorem$	
	$y : lorem$	

Mivel az  $u$  elemeinek csak az azonosító részei szerepelnek a programban, azok pedig éppen  $y$  elemei, az  $u.lov.azon \leftarrow y.lov$  helyettesítéssel kapjuk a megoldó programot az eredeti állapottéren.

## 4.

Első megoldás

Legyen  $X = seq(Ch)$ , és az állapottér

$$A = \begin{matrix} X & \times & X \\ x & & y \end{matrix} .$$

Transzformáljuk az állapottérrel.

Legyen  $U = seq(S)$ ,  $S = (Ch'; H)$ ,  $Ch' = Ch \setminus \{''\}$  és  $H = seq^+(\{''\})$ , azaz  $H$  szóközök nem üres sorozatai, és  $U$ -ra legyen igaz a következő invariáns tulajdonság:

$I_U(u) = \forall i \in [1..dom(u) - 1] : u_i.H \neq u_{i+1}.H$ , azaz  $u$  nem szóköz karakterek és szóköz sorozatok sorozata.

$$A' = \begin{matrix} U & \times & X \\ u & & y \end{matrix}$$

Az  $x$  és az  $u$  közötti kapcsolat:  $seq(x|Ch) = seq(u|Ch)$ .

Transzformáljuk tovább az állapottérrel. Legyen  $T = file(Ch)$ .

$$A'' = \begin{matrix} T & \times & X \\ t & & y \end{matrix}$$

$$B'' = \begin{matrix} T \\ t' \end{matrix}$$

A  $t$  és az  $u$  közötti kapcsolat:  $dom(t) = dom(u)$  és

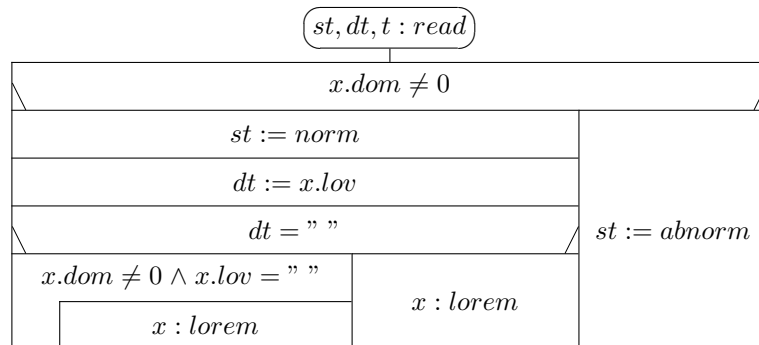
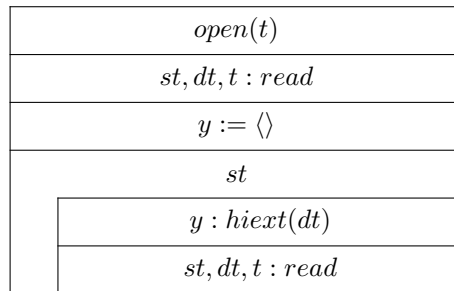
$$\forall i \in [1..dom(t)] : t_i = \begin{cases} u_i & , \text{ ha } u_i.Ch' \\ '' & , \text{ ha } u_i.H \end{cases} .$$

Ekkor a feladat:

$$Q : (t = t')$$

$$R : (y = t')$$

Ha valóban létezne egy ilyen  $t$  fájl, akkor a feladat egy egyszerű elemenkénti feldolgozás lenne az identikus leképezéssel, mint elemenként feldolgozható függvénnyel. Fölrírjuk az absztrakt másoló programot és az absztrakt  $t$  fájl műveleteit.



Az  $open(t)$  program ebben az esetben nem csinál semmit, azaz *Skip*.

Második megoldás

Legyen  $X = seq(Ch)$ . Definiáljuk a következő függvényt:  $f \in \mathbb{N}_0 \times X \rightarrow X \times \mathbb{L}$ ,

$$f(0) = (\langle \rangle, hamis)$$

$$f(i+1) = \begin{cases} (hiext(f(i)_1, x_{i+1}), hamis), & \text{ha } x_{i+1} \neq " "; \\ (hiext(f(i)_1, x_{i+1}), igaz), & \text{ha } x_{i+1} = " " \text{ és } \neg f(i)_2; \\ f(i), & \text{ha } x_{i+1} = " " \text{ és } f(i)_2 \end{cases} .$$

Ezután a feladat specifikációja:

$$A = X \times X .$$

$$x \quad y$$

$$B = X$$

$$x'$$

$$Q : (x = x')$$

$$R : (y = f(dom(x'))_1)$$

A megoldóprogram a rekurzív formulával adott függvény helyettesítési értékéről szóló tétel alapján:

$y, l := \langle \rangle, hamis$		
$x.dom \neq 0$		
$x.lov \neq " "$	$x.lov = " " \wedge \neg l$	$x.lov = " " \wedge l$
$y : hiert(x.lov)$	$y : hiert(x.lov)$	<i>Skip</i>
$l := hamis$	$l := igaz$	
$x : lorem$		

7.

Első megoldás

Legyen  $F = file(Ch)$ , és az állapotter

$$A = \begin{array}{c} F \times \mathbb{N}_0 \\ x \quad d \end{array}$$

Transzformáljuk az állapotteret, vezessük be a "szó" és az "elválasztó rész" fogalmát.  $Ch' = Ch \setminus \{ " " \}$ ,  $SZO = seq^+(Ch')$ ,  $ELV = seq^+(\{ " " \})$ . Legyen  $S = (SZO; ELV)$  és  $U = seq(S)$ , a már ismert invariáns tulajdonsággal:

$$I_U(u) = \forall i \in [1..dom(u) - 1] : u_i.SZO \neq u_{i+1}.SZO.$$

Ekkor az új állapotter

$$A' = \begin{array}{c} U \times \mathbb{N}_0 \\ u \quad d \end{array}$$

és az  $x$  és az  $u$  közötti kapcsolat  $seq(x|Ch) = seq(u|Ch)$  lesz.

Mivel az elválasztó részekre nincs szükségünk, az állapotter újabb transzformálásával el is hagyjuk azokat, azaz

$$A'' = \begin{array}{c} V \times \mathbb{N}_0 \\ v \quad d \end{array}, \text{ ahol } V = seq(SZO) \text{ és } seq(u|SZO) = seq(v|SZO).$$

Végül, mivel csak a szavak hosszára van szükségünk, legyen  $Y = file(\mathbb{N})$ ,

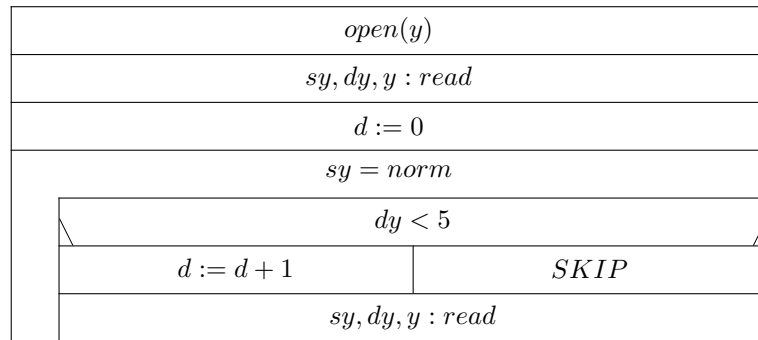
$$A''' = \begin{array}{c} Y \times \mathbb{N}_0 \\ y \quad d \end{array} \text{ és } dom(y) = dom(v) \text{ és } \forall i \in [1..dom(y)] : y_i = dom(v_i).$$

$$B = \begin{array}{c} Y \\ y' \end{array}$$

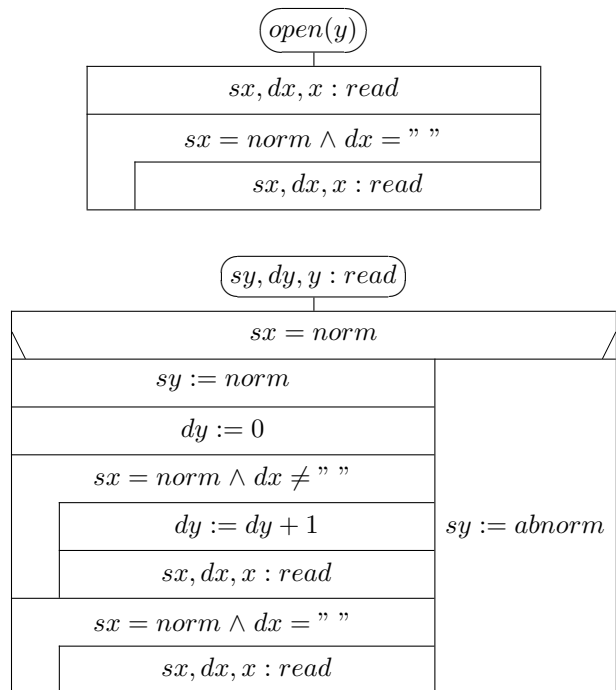
$$Q : (y = y')$$

$$R : (d = \sum_{i=1}^{dom(y')} \chi(y'_i > 5))$$

A megoldó program megszámlálás (absztrakt) szekvenciális fájlra.



Az absztrakt fájl műveletei:



Második megoldás

Függvényabsztrakciót használunk utolsó utáni elemmel. Legyen  $y = kon(x, \langle extr \rangle)$ ,  $ChE = Ch \cup \{extr\}$  és  $F = file(ChE)$ .

$$A' = \underset{y}{F} \times \underset{d}{\mathbb{N}_0}$$

$$\begin{aligned}
 B &= F \\
 &\quad y' \\
 Q &: (y = y') \\
 R &: (d = f(\text{dom}(y'))_1), \text{ ahol}
 \end{aligned}$$

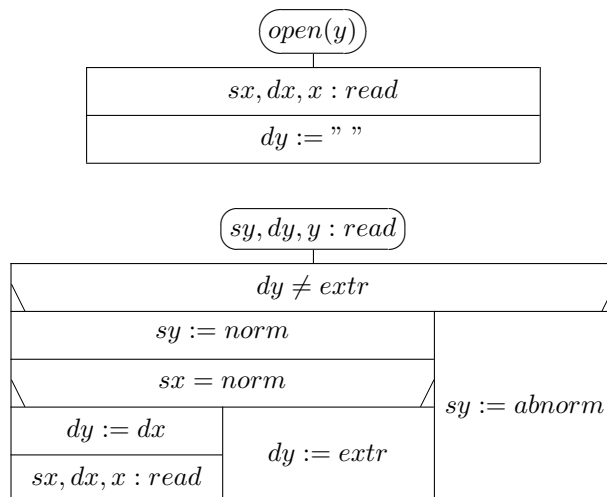
$$f(0) = (0, 0) \text{ és}$$

$$f(i+1) = \begin{cases} (f(i)_1, f(i)_2 + 1), & \text{ha } y_{i+1} \in Ch'; \\ (f(i)_1 + 1, 0), & \text{ha } y_{i+1} \notin Ch' \text{ és } 0 < f(i)_2 < 5; \\ f(i), & \text{ha } y_{i+1} \notin Ch' \text{ és } f(i)_2 = 0 \text{ vagy } f(i)_2 \geq 5. \end{cases}$$

Alkalmazva a rekurzív függvény tételt absztrakt szekvenciális fájlra:

$open(y)$		
$sy, dy, y : read$		
$d, h := 0, 0$		
$sy = norm$		
$dy \in Ch'$	$dy \notin Ch' \wedge h > 0 \wedge h < 5$	$dy \notin Ch' \wedge (h = 0 \vee h \geq 5)$
$h := h + 1$	$d, h := d + 1, 0$	<i>Skip</i>
$sy, dy, y : read$		

Az  $y$  absztrakt fájl műveletei:



8.

Először definiáljuk a feladatban szereplő típusokat.

$$U = (\text{azon} : AZON, \text{ossz} : \mathbb{Z})$$

$$T = (\text{tipus} : \{\text{kivét}, \text{betét}\}, \text{ossz} : \mathbb{N})$$

$$Y = (u : U; t : T)$$

$$F = \text{file}(Y)$$

$$F' = \text{file}(U)$$

$$A = \begin{array}{cc} F & \times & F' \\ x & & z \end{array}$$

$$B : \begin{array}{c} F \\ x' \end{array}$$

$$Q : (x = x' \wedge (x'.\text{lov}.u \vee x'.\text{dom} = 0))$$

Az előfeltételben szereplő  $x'.\text{lov}.u$ , mivel  $R$  egy egyesítés, azt fejezi ki, hogy a bemeneti fájl első rekordja ügyfél rekord kell, hogy legyen.

Első megoldás

Mielőtt az utófeltételt felírnánk, transzformáljuk az állapotteret. Legyen  $V = (u : U, s : S)$ , ahol  $S = \text{seq}(T)$  és  $H = \text{seq}(V)$ .

$$A' = \begin{array}{cc} H & \times & F' \\ h & & z \end{array}$$

Az  $x$  és a  $h$  közötti kapcsolat:  $\text{seq}(x|\{U, T\}) = \text{seq}(h|\{U, T\})$ .

Tovább transzformálva az állapotteret,

$$A'' = \begin{array}{cc} F' & \times & F' \\ t & & z \end{array}$$

A  $h$  és a  $t$  közötti kapcsolat:

$$\text{dom}(t) = \text{dom}(h) \text{ és } \forall i \in [1..\text{dom}(t)] : t_i.\text{azon} = h_i.\text{azon} \text{ és}$$

$$t_i.\text{ossz} = h_i.u.\text{ossz} + \sum_{j=1}^{h_i.s.\text{dom}} e(h_i.s_j.\text{tipus})h_i.s_j.\text{ossz}, \text{ ahol}$$

$$e(h_i.s_j.\text{tipus}) = \begin{cases} -1, & \text{ha } h_i.s_j.\text{tipus} = \text{kivét}; \\ 1, & \text{ha } h_i.s_j.\text{tipus} = \text{betét}. \end{cases}$$

Ezen az állapotéren a feladat már nagyon egyszerű, egy identikus egyváltozós-egyértékű elemenkénti feldolgozás.

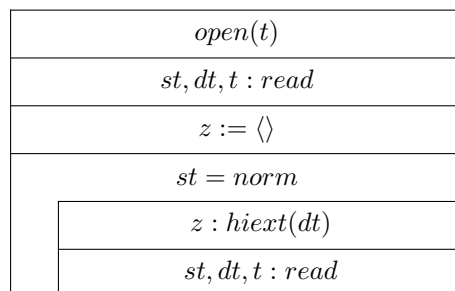
$$A'' = \begin{array}{cc} F' & \times & F' \\ t & & z \end{array}$$

$$B'' = \begin{array}{c} F' \\ t' \end{array}$$

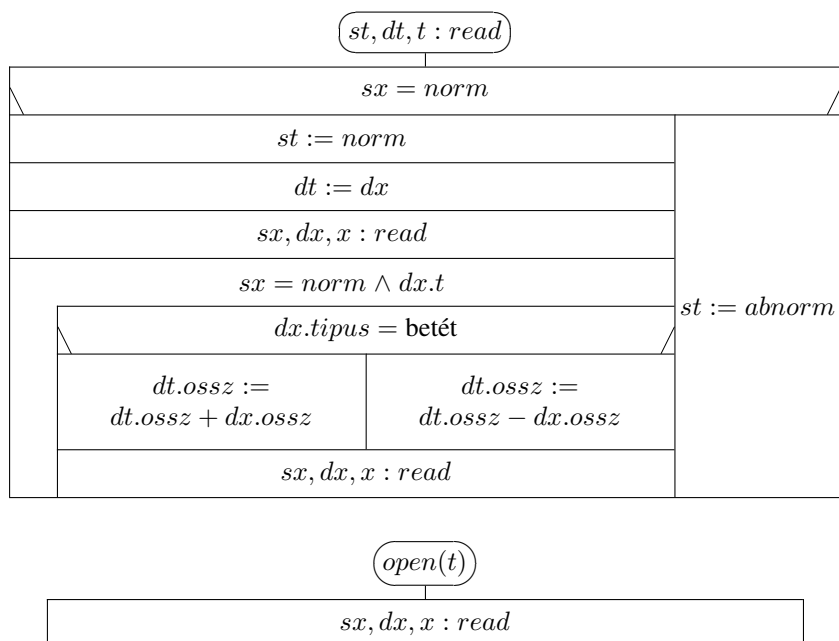


$$Q'' : (t = t')$$

$$R'' : (z = t')$$



A  $t$  absztrakt fájlra vonatkozó olvasás lényegében egy rekurzív függvény (összeg) helyettesítési értékének kiszámítása.



Ez a program garantálja, hogy  $dx = h_1.u$  mert föltettük, hogy  $x$  ügyfél rekorddal kezdődik.

Második megoldás

Függvényabsztrakciót használunk utolsó utáni elemmel.

Legyen  $y = \text{kon}(x, \langle \langle \text{extr}, 0 \rangle \rangle)$ .

$A' = F \times F'$

$y \quad z$

$B = F$

$y'$

$Q : (y = y' \wedge y'.\text{lov}.u)$

$R : (z = f(\text{dom}(y'))_1)$ ,

ahol az  $f$  függvény definíciója:

$$f(0) = (\langle \rangle, \text{"üres"}, 0) \text{ és}$$

$$f(i+1) =$$

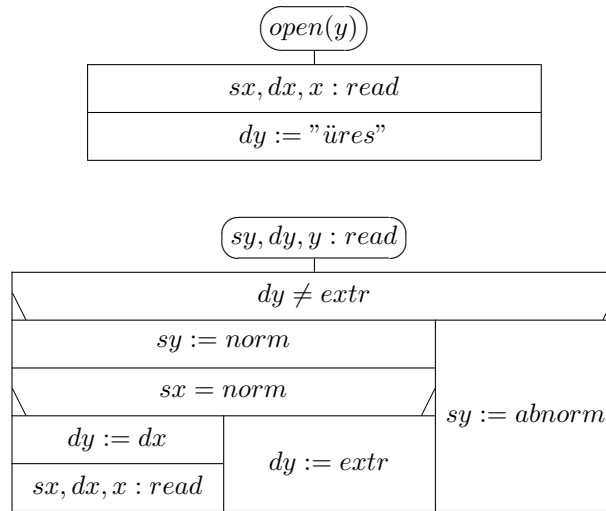
$$\begin{cases} (\text{hixt}(f(i)_1, (f(i)_2, f(i)_3), y_{i+1}.\text{azon}, y_{i+1}.\text{ossz}), & \text{ha } y_{i+1}.u \\ & \text{és } f(i)_2 \neq \text{"üres"}; \\ (f(i)_1, y_{i+1}.\text{azon}, y_{i+1}.\text{ossz}), & \text{ha } y_{i+1}.u \\ & \text{és } f(i)_2 = \text{"üres"}; \\ (f(i)_1, f(i)_2, f(i)_3 + e(y_{i+1}.\text{tipus})y_{i+1}.s), & \text{ha } y_{i+1}.t, \text{ ahol} \end{cases}$$

$$e(y_{i+1}.\text{tipus}) = \begin{cases} -1, & \text{ha } y_{i+1}.\text{tipus} = \text{kivét}; \\ 1, & \text{ha } y_{i+1}.\text{tipus} = \text{betét}. \end{cases}$$

Alkalmazva a rekurzív függvény tételt absztrakt szekvenciális fájlra:

$open(y)$			
$sy, dy, y : \text{read}$			
$z, a, s := \langle \rangle, \text{"üres"}, 0$			
$sy = \text{norm}$			
$dy.u \wedge a \neq \text{"üres"}$	$dy.u \wedge a = \text{"üres"}$	$dy.t$	
$z : \text{hixt}(a, s)$	$a, s :=$	$dy.\text{tipus} = \text{kivét}$	
$a, s :=$	$dy.\text{azon}, dy.\text{ossz}$	$s := s$	$s := s$
$dy.\text{azon}, dy.\text{ossz}$		$-dy.\text{ossz}$	$+dy.\text{ossz}$
$sy, dy, y : \text{read}$			

Az  $y$  absztrakt fájl műveletei:



9.

Specifikáció:

$$F = file(Ch)$$

$$A = F \times \mathbb{N}_0$$

$x$ 
 $d$

$$B = F$$

$x'$

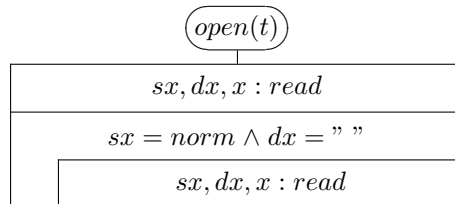
$$Q : (x = x')$$

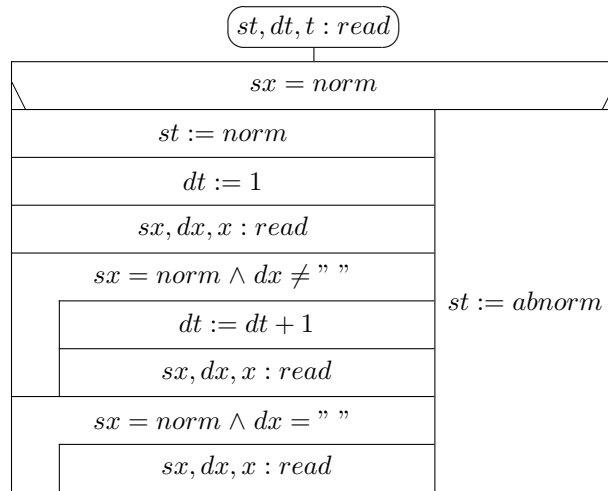
A feladat megoldásai nagyon hasonlóak a 7. feladat megoldásaihoz.

Első megoldás

A feladatot adatabsztrakcióval oldjuk meg, az absztrakt fájl természetes számokból áll, amelyek a konkrét fájl szóhosszainak felelnek meg.

Legyen  $SZÓ = seq^+(BETŰ)$ ,  $ELV = seq^+(BETŰ)$ ,  $E = (SZÓ; BETŰ)$  és  $S = seq(E)$ .





Ezzel az absztrakt fájlal a feladat már egyszerűen visszavezethető egy összegzésre (ahol aszerint a függvény szerint összegzünk, ami csak 12-nél nagyobb argumentumra 2, egyébként 1).

$$F' = file(\mathbb{N})$$

$$A' = F' \times \mathbb{N}_0$$

$t \quad d$

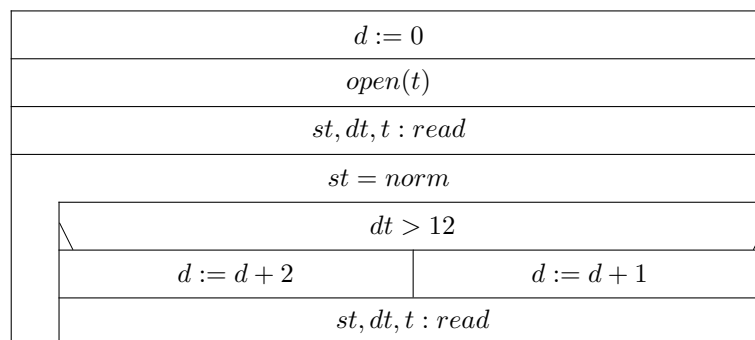
$$B' = F'$$

$t'$

$$Q' : (t = t')$$

$$R' : (d = \sum_{i=t'.lob}^{t'.hib} f(t'_i))$$

$$f(e) = \begin{cases} 2, & \text{ha } e > 12; \\ 1 & \text{egyébként} \end{cases}$$



Második megoldás

$$F = file(Ch).$$

$$A' = F \times \mathbb{N}_0$$

$$y \quad d$$

$$B = F$$

$$y'$$

$$Q : (y = y')$$

$$R : (d = f(dom(y'))_1), \text{ ahol}$$

$$f(0) = (0, 0) \text{ és}$$

$$f(i+1) = \begin{cases} (f(i)_1 + 1, f(i)_2 + 1), & \text{ha } y_{i+1} \in BET\check{U} \text{ és} \\ & (f(i)_2 = 0 \text{ vagy } f(i)_2 = 12); \\ (f(i)_1, 0), & \text{ha } y_{i+1} \notin BET\check{U}; \\ (f(i)_1, f(i)_2 + 1), & \text{ha } y_{i+1} \in BET\check{U} \text{ és} \\ & f(i)_2 \neq 0 \text{ és } f(i)_2 \neq 12. \end{cases}$$

Alkalmazva a rekurzív függvény tételt szekvenciális fájlra:

$sy, dy, y : read$		
$d, h := 0, 0$		
$sy = norm$		
$dy \in BET\check{U} \wedge$ $(h = 0 \vee h = 12)$	$dy \in BET\check{U} \vee$ $h \neq 0 \wedge h \neq 12$	$dy \notin BET\check{U}$
$d, h := d + 1, h + 1$	$h := h + 1$	$h := 0$
$sy, dy, y : read$		

10.

Első megoldás

Specifikáció:

$$F = file(Ch)$$

$$A = F \times \mathbb{N}_0$$

$$x \quad d$$

Transzformáljuk az állapotteret. Legyen  $Y = file(\mathbb{L})$  és

$$A' = Y \times \mathbb{N}_0$$

$$y \quad d$$

A kapcsolat  $x$  és  $y$  között:  $dom(y) = dom(x)$  és

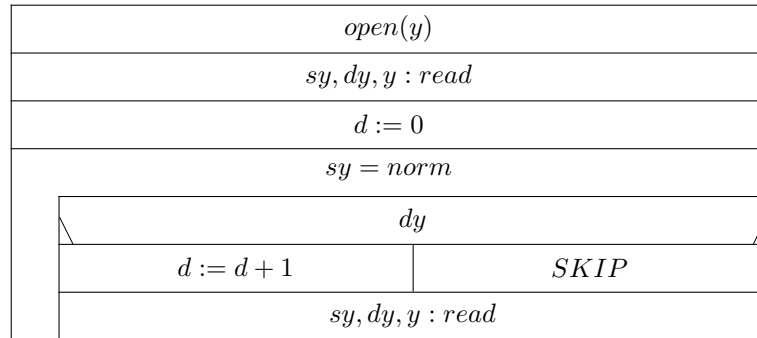
$$\forall i \in [1..dom(y)] : y_i = (x_i = "R").$$

$$B' = Y$$

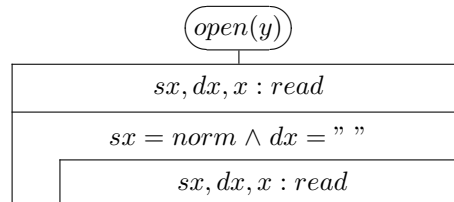
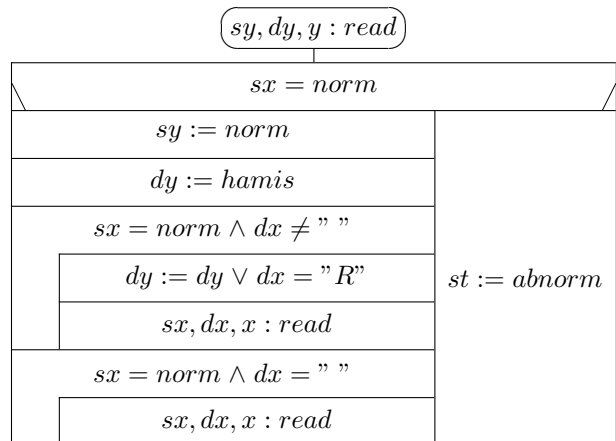
$$y'$$

$$Q : (y = y') \ R : (d = \sum_{i=1}^{dom(y')} \chi(y'_i))$$

A megoldóprogram számlálás:



Az  $y$  absztrakt fájl műveletei:



Második megoldás

$$F = file(Ch).$$

$$A = F \times \mathbb{N}_0$$

$$\quad \quad \quad y \quad \quad d$$

$$B = F$$

$$\quad \quad \quad y'$$

$$Q : (y = y')$$

$$R : (d = f(dom(y'))_1), \text{ ahol}$$

$$f(0) = (0, 0) \text{ és}$$

$$f(i+1) = \begin{cases} (f(i)_1 + 1, f(i)_2 + 1), & \text{ha } y_{i+1} = "R" \text{ és } f(i)_2 = 0; \\ (f(i)_1, 0), & \text{ha } y_{i+1} \notin BET\check{U}; \\ f(i), & \text{ha } y_{i+1} \in BET\check{U} \text{ és } f(i)_2 \neq 0. \end{cases}$$

Alkalmazva a rekurzív függvény tételt szekvenciális fájlra:

$sy, dy, y : read$		
$d, h := 0, 0$		
$sy = norm$		
$dy = "R" \wedge$ $h = 0$	$dy \notin BET\check{U}$	$dy \in BET\check{U} \wedge$ $(dy \neq "R" \vee h \neq 0)$
$d, h := d + 1, h + 1$	$h := 0$	$Skip$
$sy, dy, y : read$		

# Tárgymutató

<b>A</b>		függvény	146
állapottér	27	kétváltozós egyértékű	148
ekvivalens	62	előfeltétel	43
<b>B</b>		értékkadás	88
bővített identitás	52	egyszerű	88
		parciális	88
		szimultán	88, 155
<b>C</b>		értékkiválasztás	88
ciklus	68	parciális	88
levezetési szabálya	80	esetszétválasztás	144
programfüggvénye	72		
<b>D</b>		<b>F</b>	
direktszorzat	12, 16	függvény	13
altere	17	félkiterjesztés	53
ekvivalencia	17	feladat	28
kiegészítő altere	17	ekvivalens	104
<b>E</b>		finomítása	57
egyesítés	113, 115	feltételes maximumkeresés	126
ekvivalens		feltételig	131
állapottér	62	függvénytípus	116, 119, 157
direktszorzat	17	<b>G</b>	
feladat	104	gyenge igazsághalmaz	16, 63
program	51	<b>H</b>	
típusspecifikáció	104	hatásreláció	31
elágazás	66	<b>I</b>	
levezetési szabálya	79	igazsághalmaz	16
programfüggvénye	71	gyenge	16
elemenként feldolgozható		inverzió	165
általános változat	150	<b>K</b>	
egyváltozós egyértékű	147	kiterjesztés	
egyváltozós kétértékű	149		
elégészes feltétel	146		



feladaté	49	paraméterváltozó	41
megoldásé	61	parciális függvény	13
programé	50	primitív programok	33, 90
kiterjesztési tételek	51	primitív típus	114
kompozíció	14	program	29, 30
szigorú	14	programfüggvény	31
kompozíció kiszámítása	143	programozási feladat	33
konkatenáció	12	megoldása	33
		projekció	17
<b>L</b>		<b>R</b>	
leggyengébb előfeltétel	37	redukált	12
leszűkítés	14	rekord	112, 114, 117
levezetés	153	rekurzív függvény kiszámítása	144
lezárt	15	reláció	12
feltételre vonatkozó	16	értékkészlete	12
korlátos	15	értelmezési tartománya	12
lineáris keresés	127	determinisztikus	13
2.8	127	<b>S</b>	
első változat	131, 134	sorozat	113, 115, 118, 157–160
harmadik változat	133, 134	specifikáció tétele	40
második változat	132, 134	számlálás	124
logaritmikus keresés	137	feltételig	130
<b>M</b>		szekvenciális	
maximumkeresés	125	fájl	118, 158, 160, 162
feltételig	130	megfelelő	161
megengedett programok	33	szekvencia	66
megfelelés	103	levezetési szabálya	77
általánosítása	104	programfüggvénye	70
megoldás	31	szelektorfüggvény	114, 117, 119
$\varrho$ -n keresztül	102	szigorítás	32
általánosított	62	<b>T</b>	
átnevezéssel	62	típus	102
kiterjesztése	61	absztrakt	104
reláció szerint	63	ekvivalens	105
<b>O</b>		elemi	102
őskép	14	típusértékalmaz	27, 101, 102, 111–113,
összegzés	123	161	
feltételig	129	elemi	102
<b>P</b>		típus-specifikáció	101, 103
paramétertér	39	tétele	104

transzformáció	153
állapottér	161, 162
program	154
típus	157
<b>U</b>	
utófeltétel	43
<b>V</b>	
változó	17, 41
vektor	119
vetítéstartás	53
viisszalépéses	
keresés	138
számlálás	142
viisszavezetés	153
általánosítással	171
alteres	170, 171
konstanssal helyettesítés	170
paraméteres	154, 171
szigorítással	154, 169–171
természetes	168

# Irodalomjegyzék

- [Dij 68] Dijkstra, E. W.: Go to statement considered as harmful. *Communications of ACM* 11, 3 (1968) 147–148.
- [Mills 72] Mills, H. D.: Mathematical functions for structured programming. *SC 72-6012, IBM Gaithersburg* 1972.
- [Hoa 72] Hoare, C. A. R.: Proof of correctness of data representations. *Acta Informatica* 1, 4 (1972) 271–281.
- [Jac 75] Jackson, M. A.: *Principles of Programming Design*. Academic Press, 1975.
- [Dij 76] Dijkstra, E. W.: *A Discipline of Programming*. Prentice Hall, 1976.
- [San 80] Sanderson, J. G.: *A Relational Theory of Computing*. Springer-Verlag, 1980.
- [Gri 81] Gries, D.: *The Science of Programming*. Springer-Verlag, 1981.
- [Fót 83] Fóthi Á.: *Bevezetés a programozáshoz*. Egyetemi jegyzet. ELTE TTK, 1983.
- [Mal 90] Malcolm, G.: Data structures and program transformations. *Science of Computer Programming*, 14, 2-3 (1990) 255–279.
- [Bud 95] Budd, T.: *Multi-Paradigm Programming in Lede*. Addison–Wesley, 1995.

- [GHJV 95] Gamma, E. – Helm, R. – Johnson, R. – Vlissides, J.: *Design Patterns – Elements of Reusable Object-Oriented Software*. Addison–Wesley, 1995.
- [Hor 98] Horváth Z.: The formal specification of a problem solved by a parallel program – a relational model. *Annales Univ. Sci. Budapest. de R. Eötvös Nom. Sectio Computatorica* 17 (1998) 173–191.
- [Abr 96] Abrial, J.-R.: *B book*. Cambridge University Press, 1996.
- [Odi 98] Odifreddi, P.: *Classical Recursion Theory*. Elsevier Science Publ. BV., 1989.
- [Koz Var 03] Kozma L. – Varga L.: *A szoftvertechnológia elméleti kérdései*. ELTE Eötvös Kiadó, 2003.