

Komputeralgebra rendszerek

XV. Összetett adatszerkezetek a Maple-ben

Czirbusz Sándor
czirbusz@gmail.com

Komputeralgebra Tanszék
ELTE Informatika Kar

2010-2011 őszi

Index

- 1 Sorozatok
- 2 Halmazok
- 3 Listák
- 4 Vermek, sorok
 - Vermek
 - Sorok
 - Prioritásos adatszerkezetek
- 5 Rekordok
- 6 Tömbök
 - Az array típus
 - Az Array típus
- 7 Mátrixfüggvények
- 8 A kétféle tömbszerkezet összehasonlítása

Sorozatok

exprseq

- Vesszővel elválasztott kifejezéssorozat
- A függvényhívások argumentumlistája : **args, nargs**
- Típusa : `exprseq`

exprseq	expr1	expr2
---------	-------	-------	-------

- Üres sorozat : `NULL`
- Generálás : **seq**(`expr`, `range`), `x$n` (leggyakrabban a **diff**-el)
- Kiválasztás : **[n]**, visszafelé - **n**

Halmazok

set

- Sorozat kapcsolószerűjelek között : seq. Üres halmaz {}
- Egy elem csak egyszer fordul elő, nincs rendezés
- Alapműveletek : **union, minus, intersect**. Használhatók a műveleti jelek is!
- Lekérdezés : **member**
- Kiválasztás : **select, remove, selectremove**
select(crit, set, extra)
- Egyéb : **combinat** csomag, pl. **powerset**

Listák

list, listlist

- Sorozat szögletes zárójelek között :[**seq**]
- Egy elem többször is előfordul, rendezéstartó, könnyen adható új érték elemnek.
- Rendezés : **sort**(list,mode). Helyben történik !
- Eleme **member** eleme - vizsgálat
- Szelektorok :T[],T[i], T[-i], T[i..j],T[i,j,...],T[i..j,k,...]
- A `ListTools` csomag
 - **Flatten**
 - **Rotate, Reverse**
 - **Categorize**

Vermek I

A régi konstrukció :

stack

- **stack[new](x1, ..., xn)** – Új létrehozása, opcionálisan elemekkel
- **stack[push](x, s)**
- **stack[pop](s)**
- Információk : **stack[empty](s)**, **stack[top](s)**, **stack[depth](s)**

Vermek II

Az objektumorientált konstrukciók:

Stack

- Konstruktorok :
 - SimpleStack() – a szokásos verem, objektumorientált köntösben
 - BoundedStack(bound::posint) – a stack legfeljebb a megadott méretű lehet
 - MeteredStack() – a „naplózó” verem
- Műveletek : s:-push(e::anything), s:-pop(), s:-empty(), s:-top(),s:-depth()
- A MeteredStack() többlete : s:-stats()

Sorok

A régi :

queue

- Létrehozás : hasonlóan a stack-hez
- Lehetőségek : new, empty, enqueue – sor végére szúrás, dequeue – utolsó elem
- Továbbá : front – első elem törlés nélkül, lengt, clear, reverse
- Használat : with(queue)

Az új :

Queue

A Simplestack és a queue-ból kitalálható a szintaxis

Prioritásos adatszerkezetek

heap

- **heap[new](f,x1,x2,..xn)** f: a rendezést megvalósító logikai fv, pl lexorder
- insert, extract, empty, max, size

priority queue

- Konstruálás : **initialize(pq)**
- Műveletek : **insert(NewRecord, pq), extract(pq)**
- Használata : **with(priqueue)**

Rekordok

- Pascal stílusú rekordmegvalósítása
- Konstruktor : **Record**(nev1=ert1,nev2=ert2,...)
- Implementáció : modulokkal
- Hivatkozás: rec_name:-
- Egyenlőség tesztelése : **verify(expr1, expr2, record)**

Az array típus I

- A linalg csomag tartalmazza
- A `table` adattípuson alapszik (Hash-tábla).
- Szintaxis : **array**(indexfcn, bounds, list)
- **Indexelő függvények** : antisymmetric, diagonal, identity, sparse, symmetric.
- **table**(indexfcn,list)

table	↑ indexfv	↑ méretek	↑ hash tábla
-------	-----------	-----------	--------------

- **matrix** – kétdimenziós tömb, indexei 1-től indulnak.

Az array típus II

- **Speciális mátrixok** : bezout, fibonacci, hilbert, jacobian, sylvester, toeplitz, vandermonde
- **vector** – sorvektor, indexei 1-től indulnak.
- Nem teljes kiértékelés történik, hanem utolsó név szerinti
- Teljes kiértékelés kikényszerítése : **map(eval,array)**
- A **copy** utasítással készíthetünk még egy példányt. (Az összeláncolás itt megszűnik)

Az Array típus I

Az `rtable` adattípuson alapszik (Téglalap alakú tábla). A `LinearAlgebra` csomag használja őket.

- **Array**(indfncs, dims, init, opts)
- Az opciók :
 - **datatype**= : minden, amit a Maple típusspecifikációi megengednek.
 - **datatype**=integer[n], float[n], complex[n] – hány bites (1,2,4,8, illetve csak 8)
 - **storage**=sparse, rectangular,....
 - **order**=C_order,Fortran_order

Az Array típus II

- **Matrix**(r, c, init, ro, sym, sc, sh, st, o, dt, f, a)
 - A paraméterek opcionálisak
 - r,c : sor, oszlop
 - init – kezdeti értékek: Maple procedúra, table, Array, vector, list, stb
 - ro : **readonly**=true|false
 - sym : **symbol**=name – a mátrixelemek szimbolikus neve
 - sc : **scan**=name|list – a kezdőértékek interpretálása
 - sh : **shape**=name|list – indexelő függvény
 - st : **storage**=name – tárolási mód
 - o : **order**=name – C vagy Fortran típusú tárolás
 - dt : **datatype**=name – adattípus
 - f : **fill**=value
 - a : **attributes**=list – attribútumok

Az Array típus III

- Kiegészítések
 - Az indexelők lehetnek symmetric, identity,....
 - A tárolás lehet pl triangular, sparse,...
 - Az attribútum lényegében bármi lehet, azonban az attributes=[positive_definite] beállítás esetén szabályozhatók a LienarAlgebra csomag bizonyos procedúrái : LinearSolve, Eigenvalue,...

Az Array típus IV

- **Vector**[o](d, init, ro, sym, sh, st, dt, f, a, o)
 - o : row/column
 - Lényegében mint a **Matrix** esetén.
 - indexelő függvények : **constant, scalar, zero**
 - Alapértelmezett az oszlopvektor

Az Array típus V

- **rtable**(indfcns, dims, init, opts)
 A tömbkonstruálás alacsony szintű procedúrája.
 Különleges opciók :
 - **subtype**=value : **Array, Matrix, Vector**[row|column]
 Default az **Array**
 - **transpose**=true

Mátrixfüggvények

- Szintaxis : **MatrixFunction(A, F, x, outputopt=[outopts])**
- Paraméterek :
 - A : a tömb
 - F : analitikus függvény
 - Outputopciók : mint a mátrixkonstruktornál
- Az „F” analitikus függvény hatványsorába behelyettesíti a mátrixot, az eredmény a határérték.
- Külön függvény van az exponenciálisra :
MatrixExponential(A, t, outputopt=[outopts])

A kétféle tömbszerkezet összehasonlítása

feladat	array	matrix	Array	Matrix
tárolás	ritka	ritka	sűrű	sűrű
speciális tárolás	nincs	nincs	van	van
indexelő fv	van	nincs	van	van
index-start	bármilyen	1	bármilyen	1
default elem	szimbolikus	szimbolikus	0 (állítható)	0 (állítható)
vált. struktúra	nem	nem	igen	igen
import	readdata	readdata	ImportMatrix	ImportMatrix
export	writedata	readdata	ExportMatrix	ExportMatrix
megjelenítés	teljes	teljes	teljes v. részleges	teljes v. részleges
szorzás	&*, evalm	&*, evalm	.	. , Multiply
kiértékelés	utolsó név	utolsó név	teljes	teljes

Példák

Lásd a munkalapokon