Príklady na cvičenia Neprocedurálne programovanie PRG005.

Jan Hric
KTIML MFF UK
e-mail: Jan.Hric@mff.cuni.cz
http://ktiml.ms.mff.cuni.cz/~hric/vyuka/cvneproc.{pdf,ps}

15. dubna 2009

Programy pre zápočet kombinovaných študentov. Naprogramujte, odladťe, pripravte testovacie data (aspoň 5, aspoň 2 netriviálne). 5x Prolog, 3x Haskell.

1 Transpozícia matice *Prolog: Matica ako zoznam zoznamov. Skontrolujte, že je správne zadaná (tj. obdĺžnik m x n) a transponujte ju.*

2 Izomorfizmus stromov *(obecne n-árnych, pre binárne jednoduché) Zistite, či existuje izomorfizmus dvoch daných (zakorenených) stromov, tj, či je možné prehadzovaním vetiev previesť jeden na druhý.* Rozšírenie: ... a vydať popis zmien $T1$ vs. $T2$ v dat. štruktúre: Pre bin. Id a $Swap$ (L a R), pre $NTree$ nové poradie - permutácia

3 Zjednotenie n-árnych stromov *Štruktúra $t(kluc, hodnota, [podstromy])$. Zlievame stromy $T1$ a $T2$, v tomto poradí. Ak majú dva zlievané stromy v zozname podstromy s rovnakým kľúčom, potom ich rekurzívne zlejem a hodnotu nechám z $T1$. Ináč prenesiem celý podstrom do výsledku. Korene zlievame vždy.* Je možná usporiadaná alebo neusporiadaná implementácia (zoznamu podstromov), programátorsky jednoduchšia je usporiadaná.

4 Prevod výrokej formule do CNF *Vstup formula s konštantami (true a false), premennými, spojkami not, or, and, imp, ekv (prípadne xor, nand, nor). Zbavte sa konštánt a zvyšok preveďte do CNF, konjunktívnej normálnej formy. Nemusíte optimalizovať.* V jednej fáze alebo v niekoľkých (jednoduchšie). "*not*" poruší štruktúru CNF, preto alebo odstrániť najprv, alebo vzájomne rek. DNF pod *not*.

5 Topologické usporiadanie grafu *Pre orientované grafy.* Rozšírenie: Vypisovať priebežné stavy algoritmu: Aktuálna cesta/stack a aktuálne uzavreté vrcholy (v správnom poradí).

6 Suma čísel *Hs: Daný zoznam (kladných) čísel xs a požadovaná suma s. Zistite, či je možné vyjadriť číslo s ako súčet nejakej podmnožiny čísel z xs. Vydajte aspoň jednu "dosvedčujúcu" podnožinu.*

Zvoľte si ľub. implementáciu (okrem 3), ale pre ostatné možné typy napíšte prevádzajúci ("jednoriadkový") interface. 1) `Num a => a -> [a] -> Maybe a`, 2) ... `a -> [a] -> (Bool,[a])`, 3) ... `a -> [a] -> Bool` : len existencia riešenia, 4) ... `a -> [a] -> [[a]]` : Všetky možnosti súčtu.

7 Výber vrcholov s n-árneho stromu *Hs: Podľa danej podmienky vyberte z n-árneho stromu všetky vrcholy, ktoré ju spĺňujú. Typ* `selectNT :: (NTree a -> Bool) -> NTree a -> [NTree a]` Rozšírenie: Ďalším parametrom dovoľte zadávať, či chcete najvrchnejšie, najspodnejšie, vnútorné alebo všetky vrcholy (alebo nejaké iné).

8 Minimálne cesty *Hs: Dijkstra: Všetky min. cesty z daného vrcholu, v orientovanom grafe. Vydajte pre dosažitelné vrcholy dĺžku cesty a predchodcu na ceste.*

1 Datové štruktúry

9 Rodokmeň *Dané rodic/2, muz/1, zena/1. Definujte mama/2, prarodic/2, isRodic/1, surodenec/2, vlastnySurodenec/2, teta/2, bratranec/2, predok/2, starsieSestry/3. Vysvetlite vhodnosť (efektivitu) definície pre otázky pp(adam,X) vs. pp(X,adam) vs. pp(eva, adam) vs. pp(X,Z) a prípadne navrhnite lepšie procedúry.*

predok/2: varianty a cyklenie, sirota, polosirota – potreba negácie.

10 Dátum *Porovnajte výhody a nevýhody rôznych reprezentácií dátumu (vymyslíte* 10 *rôznych?)*

Operácie: get, set, make, zajtra, za-rok, korektný-dátum ...

11 Binárne stromy. *Navrhnite štruktúru pre reprezentáciu binárnych stromov.*

Rozšírte/upravte pre varianty b.s.

12 Obecné stromy *Navrhnite štruktúru pre reprezentáciu n-árnych stromov. Zahrňuje definícia B-stromy? V čom sa líši?*

Upravte štruktúru pre reprezentáciu XML dokumentov s vnorenými elementami. Umožnite reprezentovať atribúty elementov.

13 Log. výrazy a formule *Bez premenných, s premennými (logické premenné (tj. doménové) vs. prologovské premenné). So zvláštnymi spojkami (nand, nor), ...*

14 Polynomy *Hm.*

Viac premenných.

15 Obrázky *Navrhnúť štruktúru pre popis (2D) obrázkov vznikajúcich skladaním základných geometrických obrazcov a ich následnou transformáciou (posúvanie, otáčanie, zmena mierky/rozmerov, neproporcionálne zmeny mierky).*

Zaveďte prekrývanie a maskovanie

Zaveďte farby (napr. RGB), farebné operácie (kreslenie farbou, vyplňovanie farbou), transformácie farieb ...

Rozšírenie:

Upravte pre popis 3D scény

Animácie

(idea z: VRML)

16 Vnorené menu *Popis vnorených menu*

Podpora fíčur: oddelovacie riadky, hotkey, select-key, návratová akcia, dočasné zakázanie, selekcie alebo on/off alebo multi ... (Bez buttonov ...)

Rozširujúce: Prevod na riadkové parametry.

Varianta (samostatne?): štruktúra pre zadávanie options. Reprezentácia aktuálneho stavu. Rozbaľovanie, buttony (radio, check), popisy, listboxy, (záložky,) ...

17 Reprezentácia imperatívnych programov *Reprezentácia výrazov (rôznych typov), reprezentácia príkazov, ... deklarácií.*

Možné účely: interpretácia v Prologu (HS) viz., výpis do klasického tvaru (pre kompilátor).

Štrukturované: Pascal, ...

Základy rekurzie

18 Unárna repr. čísel *Čísla sú reprezentované: 0 pomocou z (tj. z/0), následník N+1 pomocou s(N). Napr. 3 je s(s(s(z))).*
a) Pre túto reprezentáciu naprogramujte sčítanie addS/3, primárne pre mod (+,+,-). (Štrukturálnou rekurziou.)
b) dtto násobenie multS/3
c) (omedzené) odčítanie minusS: ak by vyšiel výsledok záporný, výstupná hodnota je 0.
d) rozširujúce: zavedieme naviac p/1 pre predchodcu celého čísla.
e) normalizácia čísla: neobsahuje súčasne s/1 a p/1.
f) addSP/3, multSP/3, minusSP/3, na normalizovaných číslach (a výsledok je nomalizovaný), f2) na obecných číslach

19 Aritmetika pre n-vektory *Vektory s dĺžkou, pri operáciách sa kontroluje.*
(nemáme dependent typy :-))

20 Štrukturálna rekurzia vs. akumulátor *Popíšte a porovnajte tieto dve metody pre: a) sort (insertsort vs. selectsort), b) permutácie, c) reverse, d) sks/3 skalárny súčin, e) transpozícia matice 2) aký to má vplyv na podriadené procedúry?*

21 Binárne stromy *Štruktúra pomocou void/0, t/4 (kľúče a data a dva podstromy). a) vyhľadávanie, b) vkladanie c) rozsir: vypúšťanie d) AVL stromy - operácie, spätná propagácia e) ...*

22 Formule *Štruktúra: spojky (and, or, not, imp, ekv, ...), konštanty (tu bez const/1), premenné (pomocou p/1)*
a) vyhodnotenie eval/2 bez premenných, b) rozne možnosti zadania spojek (úplné a bez opakovania), c) vhodné ošetrenie imp a ekv d) vyhodnotenie eval/3 s premennými (ohodnotenie je 1. vstup)
e) transformácie: zbavenie sa imp a ekv, f) prevod not/1 dole k listom, g) "roznásobenie" do CNF, h) všetko e)-g) naraz
rozšir/add: (kontrola dokazu)

23

... a dalšie domény: (DSEL) pretty printing, algebra, ...

2.cv. (2009) štruktury a úvod zoznamov

- štruktúry (datum + varianty)
- concat: spojenie zoznamov do jednoho: concat(
- progr. idiom: štrukturálna rekurzia
- da sa concat použiť v opačnom smere, tj. v mode (-,+)? Ak nie upravte, aby sa dal použiť na rozdelovanie 1 zoznamu na ľub. počet zoznamov
- regularita matice, ak mate length/2
- transpozícia matice: najprv vybratie 1. stlpca, potom odobratie 1. stlpca, potom transpozícia. Ako končí rekurzia pre odoberanie stlpcov?
- Sú iné sposoby transpozicie?
- Rozš: suffix/2 backtrackingom, suffixy/2 zoznam výsledkov, prefix/2 backtrackingom, prefixy/2 zoznam výsledkov
- Progr. idiom: zoznam výsledkov

3. cv. (2009) zoznamy

- násobenie matíc, máte skalárny súčin sks/3.
- ako vylepšiť: idea map, ...
- sks rekurziou, akumulátorom - progr. idiomy, a úvodný interface predikát (typicky pre akumulátor - zadanie poč. hodnoty)
- reverse: klasický príklad na akumulátor - len zopakovať, ak je potreba
- prienik množín (lepšie po not, if a !)
- Q: usporiadane vs neusporiadane množiny, Q: ak mame na vstupe ordset, chceme ordset na výstupe?
- DC: prienik (a/alebo zjednostenie) pre usporiadané množiny
- isSet/2, progr. idiom interpret. Čo získame interpretom? (operátory neskor)
- Q: koncová podmienka: test zoznamu vs. prevod list2set, z hladiska programu, z hladiska užívateľa
- rozsirujúce: list2set, list2oset
- dtto: bag, alias multiset (a histogramy)

4.cv. stromy, d.š., n-árne stromy (+ písomka?)

Add: Hs - rozširujúce: Zipper + kontext, GADT

Zap: overenie dokazu vo vyrokovej logike, ocislovat a dohladat predpoklady a určiť správne pravidlo. (Motovacia: JAPE.) Dtto: v rovnostnej teórii - postup prepisovania: určiť ktoré pravidlo, v akom smere a na ktoré miesto bolo aplikované. Rozšírenie: Včetne hľadania medzi lemami.

2005 neusporiadane dodatky:

... kreslenie obrázkov - návrh operátorov, ... vymysliet dalsie domeny

- -/do predn/ parametricky dane grafy - namiesto explicitneho dátového popisu grafy zadane vypoctom susedov„ alebo char. funkciou hrany :-(
- viac reprezentácií =: preťažené procedúry, a) viac implementácií, b) prevod do zvolenej reprezentácie

– explicitný vstup a výstup je vlastne dávkový, tj. ladenie do súboru a vstup zo zdrojáku, tj. prístup: malé jazyky a interpretácia d.š.

- reprezentácia: nie je nutne jednoznačná r. objektu, rozdiel syntaktického a sémantického porovnávania, pr. neusporiadané množiny (v zozname), vs. usporiadané, ale usporiadanie sa typicky týka zápisu /tj. repr./ ...,
- doména: relačné počítanie, plus operátory
- interprety, abstraktné stroje a interpretačný cyklus

– náhrady podtermov ??

- multimnožinové operácie (tj. histogramy)„ operácie, vytvorenie multimnožiny, interpret výrazov
- grafové operácie: (analógia kreslenia obrázkov) - tvorba pravidelných grafov

5.cv.

- interpret intervalových výrazov, spravne ošetriť opakované spájanie intervalov
- DC: zlievanie FS

6.cv.

- indentovaný pretty print (rekurzívnej) FS /feature structure, štruktúra rysov/, varianta: n-arneho stromy, XML-dáta, programy
- - výstup a) priamo, b) ako reťazec, c) instrukcie pre výpis

— spravne osetrit odriadkovanie výstupu (na konci riadku), oddelovace prvkov (okrem za poslednym)„ vhodné vypisovat v znovunacitatelnej podobe (Prolog, XML)„ ak odriadkujeme (len) na zaciatku riadku, prilis neusetrime, pretoze musime zvlastne osetrit prvy (a posledny) riadok.

- overenie izomorfizmu grafov, podla daneho izomorfizmu

7.cv.

- prevod grafov z n-arnej repr. relácií na binárne relácie (hypergrafy na grafy) – idea z RDF, dosť jednoduché :-(
- najdenie izomorfizmu grafov ?? asi ťažké
- ES, var: botanický kľúč„ prechod AND/OR stromom

2 Zoznamy

24 Progr. techniky *iterácia, akumulátor, delenie na polovíny (O(log n)) ...*

list2set/2 (akum), prostredni/2 (predikát),

Súčtové zoznamy: A+B reprezentuje zoznam A spojený s obrátením B. pridaj/3, uber/3. append/3, member/2. nf/2 (nejaká normálna forma).

25 Sekvencie *Možnosti: Testy a generovanie backtrackingom vs. generovanie zoznamu výsledkov.*

Vypustenie prvku, vypustenie všetkých, prevod na množinu (vypustenie opakujúcich sa), member, append/3, reverse/2, nth/3

Prefix, suffix, last, first, párne (a nepárne) prvky, prvá (a druhá) polovica.

Všetky suffixy (v zozname), prefixy (2 sposoby). V akom poradí budú výsledky?

Súvislá podpostupnosť, vybraná podpostupnosť (podmnožina),

Maximum, súčet. Kartézsky súčin (pre 2, n), concat/2. (filter/3)

Synchronizovaný prechod dvoch (n) zoznamov (zip.hs, transpozícia): vydanie indexov so zhodnými hodnotami, vydanie cmp-zoznamu – obsahuje lt, eq, gt.

26 Triedenie *Zľava, sprava, ... Heapsort, quicksort, mergesort (zhora, zdola), insertsort, sort výberom, bubblesort (+ var.), ...*

Ostane algoritmom časová a pamäťová náročnosť, ak nemáte k dispozícii pole?

Lexikografické a maximolexikografické porovnávanie (zoznamov), bez dĺžky, s dĺžkou (HS).

27 Množiny *a1) Prienik, a2) zjednotenie, a3) rozdiel, a4) symetrický rozdiel a5) test podmnožiny, a6) suma množiny (concat/2)*

Overte, že váš program vydáva pri backtrackingu len jedno (správne) riešenie. Čo sposobuje vydávanie ďalších riešení?

Navrhnite usporiadanú a neusporiadanú reprezentáciu množiny a porovnajte časovú zložitosť operácií.

b) Multimnožiny, operácie. V multimnožine (bag) záleží na počte prvkov a nezáleží na poradí. nf/2

c) Selekcia podľa podmienky, pomocou copy_term. (tj. filter.hs)

d) Rozklad na triedy ekvivalencií; parametricky a-¿a-¿Bool

28 Zoznam deliteľov

29 Eratostenovo sito *Hm.*

30 Pascalov trojuholník *Hm.*

31 Filtry 1D *Jednodimenzionálne. Dĺžky 3, n. Navrhnite reprezentáciu filtru a výstupu.*

(Zobecnite (rozšírte) pre zadávanie špeciálnych okrajových podmienok.)

Navrhnite filter pre kĺzavý aritmetický priemer, pre zostupnú a vzostupnú hranu (prudkú)

32 Filtry 2D *Dvojdimenzionálne (maticové, grafické).* 3×3, $n \times n$. *Navrhnite sadu procedúr, ktorá umožní ich jednoduché naprogramovanie. Pridanie okraja. (Haskell: map, zip, take, drop ...).*

Odborné rozšírenie: konkrétne filtry pre
– odstraňovanie bodového šumu (pepř a sůl)
– detekcia hrán (smerová)
– life
... atp.

33 Analýza časových radov *Hm. Pravidelné, nepravidelné. Hľadanie vzorov, periódy, ...*

34 Histogram *Pre zadaný zoznam prvkov na vstupe vyrobte histogram, tj. zoznam dvojíc /prvok,násobnosť/*

a) Navrhnite (aspoň) tri odlišné stratégie výpočtu

b) Porovnajte ich časovú zložitosť (pri vhodných datových štruktúrach) a ich pamäťovú zložitosť.

c) Histogram s kvantizáciou

d) Vzdialenosť hist., (metrika euklid., súčtová, max)

e) k danému histogramu H na 1..m nájsť frekvencie 1..k a 1..l, tž. frekvencie súčtov odpovedajú H. Špeciálne $k = l$ *a* $m = k^2$ *a frekv. sú rovnaké (apl. kryptografia). e2) hist. H je daný pre súčty modulo* $k(= l)$

35 Kvantizácia *Hm.*

36 Vyhľadávanie vzorkov *Aho-Corasicková, ... (výstavba tabuliek - trie, interpretácia tabuliek).*

Varianty: naivné, Knuth-Morris-Pratt, Boyer-Moore (heur. zlý znak, dobrá podpost.), Rabin-Karp (s falošnými hitmi) a iné. Hm.

37 Hammingova vzdialenosť *Počet rozdielov medzi dvomi reťazcami, pri operáciách vloženie (I.nsert), vypustenie (D.elete) a zmena (e.X.change) znaku, výmena (S.wap) susedných znakov.*

Aplikácia dynamického programovania. :-)

Varianty: rôzne ohodnotené operácie, 3swap (výnema → výmena)

38 Rubikova kocka *Reprezentácia (2 spôsoby: steny vs. kocky). Operácie (natvrdo). Podľa popisu postupnosti operácií previesť jeden stav na iný. (Krásny príklad na akumulátor.)*

39 Permutácie a cykly

a) Skladanie dvoch premutácií a2) Skladanie zoznamu permutácií

b) Rozklad permutácie na cykly. (Aj jednobodové cykly.) b2) Var: bez jednobodových cyklov.

c) K zoznamu cyklov vyrobiť permutáciu.

d) Hs: k zoznamu prvkov vyrobiť usporiadavajúcu permutáciu.

2.1 Postupnosti

40 Kombinatorika *Generovanie a testovanie, generovanie všetkých :-(2x, generovanie nasledujúcej:*

a) permutácie

b) kombinácie, s a bez opakovania

c) variácie, dtto c2) lexikografické c3) maximolexikografické poradie

d) spočítanie kombinačných čísel, ...

... vstupný zoznam prvkov, reprezentácia čísel pomocou zoznamov s _ (pre backtracking)

41 Rady *(2x) a) Fibonacciho čísla. Logaritmicky; s inými počiatočnými podmienkami.*

b) Rady z konštantného počtu členov. (Pascalov trojuh.)

c) Z všetkých minulých členov. Viz. (Dyn. prog. v 1 rozmere)

42 Počty stromov *PL,HS a) Spočítajte počet (neizomorfných) zakorenených stromov na n (nerozlíšených) prvkoch.*

b) # neizomorfných stromov na n pomenovaných prvkoch (taxonomické stromy)

c) popíšte túto úlohu ako aplikáciu dynamického programovania (Počítanie radu s použitím všetkých minulých členov)

d) (Vytvorenie najlepšieho stromu vzhľadom k danému popisu a (Hammingovým) vzdialenostiam. Kritérium je minimalizovať súčet vzdialeností v strome.)

2.2 Maticové operácie

43 Maticové operácie *Hm.*

a) Násobenie konštantou (map),

b) súčet matíc (zip)

c) súčin

d) transpozícia (2x), otocenie ...

e) kontrola zarovnania matice

f) vybrana podmatica, f2) vybranie diagonály

g) jednotkova matica danej veľkosti

h) slupka, olúpanie/spirala

44 Šifrovacia mriežka *Zrkadlová (alebo otáčacia). Šifrovanie a dešifrovanie. Dlhá postupnosť.*

Rôzne prístupy: predpočítanie polôh, on-line zapisovanie, ... Navrhnite univerzálne podprocedúry.

45 Life *Daná matica s plnými a prázdnymi políčkami. Tvorí sa nová matica, v ktorej sa zaplní políčko, ak malo povodne 2-3 (?) plných susedov (z 8).*

Zobecnenia: predpis pre zaplnenie políčka daný parametrom. Niekoľkostavové políčka a predpis...

46 Hľadanie vzoru 2D *Bez otáčania, s otáčaním. Presného (úplne zadaného) vzoru, vzoru s žolíkmi (na niektorých hodnotách nezáleží).*

Aplikácia na piškvorky.

2.3 Generuj a testuj

Možnosti riešení: Generuj a potom testuj, generuj a priebežne testuj, inteligentne generuj (v pevnom poradí), dynamické generovanie (výber z najmenej možností, i heuristicky), dopredné testovanie. Výber z nezakázaných možností, (priebežné) testovanie prípustnosti, dopočítanie posledných prvkov (u štvorcov). Dosadzovanie konštánt do voľných premenných - prologovský programátorský trik. Backtraking vs. zoznamy výsledkov, použitie setof (ak je málo riešení). V čom spočíva rýchle vylúčenie nevhodných čiastočných riešení oproti iným prístupom?

47 Rozklad čísla *Spočítať možnosti rozkladu. Vydať rozklady, neizomorfné. Rozklady na dané čísla (mince).*

Rozklad na daný počet čísel, nenulových al. nulových. Kopky v NIMe s daným počtom zápaliek.

48 Dámy na šachovnici *Hm.*

49 Latinské štvorce *Hm. ... a obdĺžniky.*

Ortogonálne latinské štvorce.

50 Magické štvorce *Bez diagonál, s hlavnými diagonálami, s všetkými (aj prerušenými) diagonálami.*

51 Křížovky *Reprezentácia pomocou voľných premenných, metoda generuj a testuj.*

a) Maľované křížovky. Hm.

b) Obyčajné křížovky - preklad.

52 Zebra *Hm.*

53 Prehľadávanie omedzení *Hm. A ďalšie ... Farbenie grafu, viz. Algebrogramy.*

S cenou: obecné Branch and bound, prerezávanie.

Vhodné: chceme jedno riešenie, alebo je riešení málo a chceme všetky (setof).

3 Spracovanie rek. štruktúr

54 Unárna aritmetika *Hm. Plus, minus, krát, ... Rozne sposoby.*

55 Testovanie vlastností stromov *Testovanie usporiadanosti binárneho stromu, haldovitosti.*

Vyváženosť hĺbky presne, približne (neúplná vrstva), AVL, vyváženosť počtu vrcholov približne (+-1).

Stavba stromu, vyváženého s., ..., haldy.

56 Vyhodnocovanie log. formule *a) Bez premenných, len s konštantami true, false*

b) S daným ohodnotením premených (v zozname).

c) Hľadanie pravdivého ohodnotenia (jedného, všetkých)

d) Vyhodnotenie formule s hodnotou unknown (na true, false, unknown)

Popíšte použité netriviálne obraty (programátorské triky): ...

57 Vyhodnocovanie log. formule *a) Bez premenných, len s konštantami true, false*

b) S daným ohodnotením premených (v zozname).

c) Hľadanie pravdivého ohodnotenia (jedného, všetkých)

d) Vyhodnotenie formule s hodnotou unknown (na true, false, unknown)

Popíšte použité netriviálne obraty (programátorské triky): ...

58 Iné spracovania formule *Hĺbka, počet premenných, kontrola premenných voči zoznamu ohodnotení, ...*

Zjednodušovanie formule, prevod do CNF/DNF (konjunktívnej/disjunktívnej normálnej formy).

...

59 Selekcia podtermu *Navrhnúť d.š. pre popis pozície v n-árnom strome, (v Progovskom terme)*

Výber (selekcia) podtermu podľa popisu pozície.

60 Kontrola dôkazu formule *Zadané sú pravidlá pre nahradzovanie podformulí, postupnosť formulí, každá s popisom (pozície) nahradzovaného podtermu. Overiť, že dôkaz je platný.*

Zjednodušenie: u formulí v postupnosti je na vstupe dané aj použité pravidlo

Významná podúloha: selekcia podtermu, viz **??**.

Zjednodušená úloha: kontrola zjednodušovania formule, (s rovnakými dátami, ale iným významom)

Varianta: :-(Gramatiky, kontrola odvodenia (reťazca).

61 Preklad log. formule *Do zásobníkového stroja, tj. do postfixnej, poľskej, notácie.*

Rozširujúce: do prologovského programu (tj. cieľa). Ako ste vyriešili preklad premenných?

Interpret zásobníkového stroja.

62 Aritmetické výrazy *Reprezentácia, s premennými. Vyhodnocovanie.*
Preklad do zásobníkového stroja, bez/s prem.
Varianta: kalkulačka - aj vyhodnocovanie cifier
Iné analýzy: hĺbka, zložitosť, ...
Rozširujúce: do prologovského programu (tj. cieľa)
Interpret zásobníkového stroja.

63 Interpret množinových výrazov *Hm. a) Prienik, zjednotenie, rozdiel, symetrický rozdiel, ...*

b) Interpret množinových výrazov. Štruktúra výrazu, operácie z a), zadávanie základných množín, (reprezentácia intervalov a postupností z .HS). Interpret a hlavné volanie.

c) Varianta: c1) Multimnožiny, c2) intervaly, c3) postupnosti. c4) Prevod multimnožín na a z množín.

64 Vyhľadávacie stromy *Ku (každému) kľúči je priradená hodnota, ktorú chceme vydať. Operácie: find, insert, delete, deletemin, prevedenie na zoznam kľúčov (linearizácia), dávková výroba stromu ...*

Datové štruktúry:
Zoznam
Binárny strom
(2-3 strom, 2-4 strom)
(AVL strom alebo Red-Black (Č-Č) strom)
n-árny strom
Varianty: Rovnakých kľúčov je v strome viac, chceme všetky hodnoty.

65 Viaczložkové kľúče *Dvojzložkové kľúče v dvojrozmernom priestore (bod v mape). Môžu mať zložky rôzne typy (vo vašej štruktúre, v nejakej inej štruktúre)?*

Kľúče n-zložkové (n pevné).
Kľúče premennej dĺžky (reťazce).

66 Intervalové hľadanie *Hľadanie obecne vydáva niekoľko hodnôt, upravte výstupné d.š.*

Reprezentácia otázok pre IH: v každom kľúči môže byť hodnota, interval alebo "čokoľvek".

D.š. pre multidimenzionálne hľadanie: Vo vrcholoch stromu je index podkľúča a rozlišovacia hodnota.

Rozširujúce varianty úlohy: kľúče premennej dĺžky a prefixové hľadanie (trie), hľadanie ľub. (spojitého) podreťazca ...

67 Stavba vyhľadávacieho stromu *a) Jednodimenzionálny, ale vyvážený :-) Z usporiadaného zoznamu.*

b) Postavte multidimenzionálny strom. Navrhnite kriteriá pre výber podkľúča a jeho hodnoty.

68 Stavba haldy *Binárnej, vyváženej. (Binomiálnej.)*

Varianta: zhaldovatenie daného (vyváženého) stromu.

69 Stavba (binárneho) rozhodovacieho stromu *(Hm. Nepresné. Motiváciu a cieľ som zamlčal.) Všetky objekty majú rovnaké atribúty, ktorých hodnoty sú pre každý objekt zadané (odpovedá to MD-kľúču). Naviac má každý objekt danú výstupnú triedu (ktorá odpovedá hľadanej hodnote). Listy rozhodovacieho stromu obsahujú objekty len z jednej výstupnej triedy. (Pre niektoré aplikácie sa táto podmienka zmierňuje). Navrhnite stavbu stromu, ktorý určí správne výstupnú triedu každého objektu. Cieľom je čo najmenší strom.*

Navrhnite kriteriá pre výber podkľúča a jeho hodnoty. (Využíva sa teória informácie)

Varianta: Obvykle sú atribúty diskrétne, tj. majú konečný počet hodnôt. Výsledný strom je n-árny a vo vnútorných vrcholoch stromu sa rozdeľuje podľa hodnoty jednoho vybraného atribútu A. Počet podstromov je rovný počtu hodnôt atribútu A a každý podstrom obsahuje len objekty s rovnakou hodnotou A.

70 Dynamické programovanie *Hm. Viz. (podkapitola?)*

Batoh, stavba vyváženého vyhľadávacieho stromu s danými četnosťami, triangulácia mnohouholníka, najdlhšia zhodná podpostupnosť, minimálna pravoľavá cesta ...

Abstraktne – generický popis, lazy vyhodnocovanie, tabelácia (bez globálnych premenných netriviálna), čiastočné uvoľňovanie pamäti (hm) ...

71 Hľadanie vyhrávajúcej stratégie *NIM: Vybrať vyhrávajúce pozície z daného (usporiadaného) zoznamu. Ťahy natvrdo, parametrom (varianty), procedúrou (1:1, 1:n)*

Interpret hry.

72 Generovanie porovnávacieho zoznamu súborov *Hm. Viz pozn.*

Varianty a rozšírenia ...

73 Linearizácia "zvislých" menu. *Vstup je ozdobený n-árny strom, výstup je (ozdobený) zoznam. Aplikácie: podadresárová štruktúra, zvislé menu. Fíčury: rozbalené/zabalené, obrázky/ikony, indentácia (stačí úroveň), zvislé čiary (a ich ukončenie).*

74 Unifikácia Feature Structures *FS. Hm. b) Matchovanie sietí. c) Unifikácia*

75 Cesta v grafe *Hravovo ohodnotenom (XML). b) V sieti c) FS*

76 Obecné operácie pre štruktúry *map, zip, fold/foldr/foldl. fold1. kart/tenzor, flatten.*

77 Obecné spracovanie termu *Analogicky k formátovacím objektom v XML/XSL.*

4 Grafy

78 Prevod reprezentácií *1) zoznam vrcholov a hrán 2) zoznamy susedov. Prevody, aplikácia setof/3.*

79 Overenie izomorfizmus grafov *Testovanie izomorfizmu pri známom priradení, použitie v generuj a testuj.*

80 Hľadanie izomorfizmu grafov *Hľadanie izomorfizmu, všetkých izomorfizmov*

Zobecnite na iné datové a algebraické štruktúry (grupy, telesá, stromy ...)

81 Izomorfizmus rozkladom na triedy *Hm.*

82 Izomorfizmus zakorenených binárnych stromov *Dovolená výmena vrcholov.*

Rozširujúce: Nezakorenené; nebinárne

83 Normálna forma (NF) binárnych stromov *a) Vyrobte výmenami pravých a ľavých podstromov doprava zošikmený strom, v ktorom má v každom vrchole pravý podstrom aspoň toľko vrcholov ako ľavý. (Podstromy vrcholu považujte za zameniteľné)*

a1) je toto kritérium jednoznačné pre preusporiadanie izomorfných stromov?

a2) prevod naprogramujte ako jednoprechodový.

b) navrhnite jednoznačné kritérium pre preusporiadanie binárnych stromov a naprogramujte prevod do tejto normálnej formy.

84 Farbenie grafu *Daný graf a pevná množina farieb. Generuj a potom testuj, priebežne testuj, forward checking.*
Varianta: každý vrchol má svoje prípustné farby (apriorné podmienky)

85 Dijkstrov alg. *Hm.*

86 Klasické grafové alg. *Min. kostra 2x, Floyd-Warshall :-) (a výstavba reg. výrazov), PERT, ... (datové štruktúry)*

87 Prehľadávanie stavového priestoru *Hm. Do hĺbky, do šírky, podľa vzdialenosti, s heuristikou ...*
Aplikácie. Prelievanie, Lloydova 15, Hammingova vzdialenosť
(Implicitné zadanie (veľkého) grafu, iteratívne prehlbovanie, férová stratégia)

88 AND-OR stromy *Hm. ... graf. Nájsť riešiaci podgraf, jeden, všetky, najlepší.*

89 Hradlové siete *Prevod na term a z termu. Nájdenie a odstránenie ekvivalentných hradiel. Topologické usporiadanie.*

5 A ďalšie ...

Idey: Návrh dátových štruktúr (typy); symbolické (a štrukturované) data (mená, tagy); rekurzia, štrukturálna rekurzia pre spracovanie vstupu, akumulátor a skladanie medzivýsledkov pre vytváranie výstupu; backtracking vs. zoznam výsledkov; (symbolické) stavové dáta (napr. cmp) a nezlyhávajúce programy; (meta) dokazovanie vlastností programov (i formálne); obecné proc. (vyššieho rádu) pre spracovanie štruktúry (nie obsahu) – parametrický polymorfizmus, parametrizácia funkciou vs. statický rozskok, obecný prechod štruktúrou - fold (a unfold - obecné vytváranie štruktúry); abstrakcia a explicitné predávanie parametrov (1 NIM, 2 grafy), anonymné funkcie (FP) a datové štruktúry (napr. dvojice); transformácia funkcií (FP), špeciálne continuations, tj. pokračovania; viacprechodové spracovanie pomocou hodnôt vs. jednoprechodové spracovanie pomocou funkcií (FP); kompozičné programovanie (a návrhové vzory) – práca s celými štruktúrami, (kombinátory), rozširovanie syntaxe (trochu netradičný pohľad), prispôsobenie sa užívateľovi - DSL a DSEL, operátory; ; generovanie dát (napr. testovacích), generovanie programov - interpretácia a kompilácia (v prostredí interpretačného charakteru), (syntaktická) jednota programu a dát (a jednoduchá sémantika) - (meta)interprety; špecifické triky (LP: anon. prem., rozdielové zoznamy) a programátorské štýly (CPS, constraints - omedzenia, kompozičné, transformácie f., kombinátory ...), lenivé vyhodnocovanie; vzťah k OOP a triky z OOP - interface, selektory, konštruktory (neformálne v LP), (implicitné) pointry a spol., dispatch a virt. metody, simulácia implementácie (FP); programovanie v malom a veľkom (dlhodobé hľadisko - údržba, úpravy) ... (... monády, polytypické programy, staged programming - stupňovité, PE, implementácia, ...)

90 Grep *Štruktúra pre popis vyhľadávaného výrazu.*
Vyhľadávanie.

91 Konečné automaty *Simulátor.*
Varianty, KA, NDKA, NDKA-λ, D!KA-λ = Aho-Corasick.

92 Generovanie náh. byte-streamu *Popis: tabulka S 256 bytov (po 8 bitov), i, j. Krok: i:=(i+1)mod 256; j:=(j+S[i])mod 256; swap(S[i],S[j]); t:=(S[i]+S[j])mod 256; output(t)*
a) Generovanie. b) Hľadanie poč. pozície (dane/neznáme i,j).

93 Interpret imperatívnych programov *Hm. Z textov. !!!*

94 Interpret dataflow programov *Hm.*

95 Datamining *Hm. Z textových dát.*

96 Petriho siete *Výpočet, dosažiteľnosť stavu (do určitej hĺbky), hm. Zistenie pripravených prechodov pre daný stav.*

97 Plánovanie procesov *Hm. Jeden/viac strojov. Kriteriá a penalizácie.*

98 Hlavolamy *Hm. Ťažké. Had-*27*, drôtové hlavolamy, ...*

99 Výpis dynamickej pamäti *výpis a načítanie. Do termu, XML, uzátvorkovanej štruktúry. b) Linearizácia a delinearizácia.*

100 Preklad do štvoríc *Hm. Reprezentácia štvorica je s4(menoOperácie, 1Arg, 2Arg, menoVýstupu), pričom mená argumentov sú mená výstupov z iných (predchádzajúcich) štvoríc, v tvare s(číslo). Preložte aritmetický výraz do štvoríc.*

```
comp4(A+B, S4ice, VstBod, N1, N0) :-
  comp4(A, SA, VBA, N1, N2),
  comp4(B, SB, VBB, N2, N3),
  N0 is N3+1,
  VstBod=s(N0),
  append([s4(plus,VBA,VBB,Vstbod)|SB], SA, S4ice).
comp4(prom(A),[],prom(A),N0,N0).
```

101 xx Rubikova kocka *Reprezentácia (*2 *spôsoby). Operácie (natvrdo). Podľa popisu postupnosti operácií previesť jeden stav na iný. (Krásny príklad na akumulátor.)*

102 Príklady DC
postavenie vyváženého stromu
postavenie vyváženej haldy
hľadanie podstromu v n-árnom strome, ktorý spĺňuje podmienku podm/1.

hs: stavba vyhľ. stromu
rozklad podľa ekvivalencie

6 Scheme

103 Triedenie (čísel)
a) Zatrieďovaním (insert a insertsort)
b) Vyberaním. b1) S použitím `let` *vyberte minimum a vytvorte zvyšný zoznam v jednom prechode.*
(cieľ príkladu: stavba d.š., let, vydávanie a spracovanie dvojíc)

104 Malé úlohy
a) Minimum/maximum (cond) rekurzia/akumulator
b) Priemer 2x. Dva prechody vs. let a páry
c) Rýchle umocňovanie
c1)(a rýchla opakovaná aplikácia) - twice
d) mergesort ?
e) suma štvorcov - funkcionálne: suma, map
e1) metoda najmenších štvocov: f a dvojice (x,y)
f) rady, ako stavové programy - dvojice (cislo, hodnota)
g) DC: FFT 3x - výpočet; gen. funkcie (pre n); gen. zdrojáku
h) zarovnanie matice na najdlhsi riadok

105 Lookup
a) funkcia `lookup` *k danému kľúču a zoznamu dvojíc kľúč.hodnota vráti hodnotu odpovedajúceho kľúča alebo* `nil`*, ak kľúč nie je v zozname.*
(symbolické informácie)

106 Stavba haldy
a) K danému zoznamu čísel postavte vyváženú haldu
a1) výber minima, rozdelenie zvyšku a rekurzia
a2) výber prvku p, rozdelenie, postavenie a zatriedenie p
(tvorba d.š., rekurzia)

107 Stavba vyváž. binárneho stromu
a) porovnajte reprezentáciu pomocou trojíc a pomocou trojprvkových zoznamov z hľadiska pamäťovej náročnosti. (A uvážte, že čím viac pamäte naalokujete, tým viac pamäte musíte prechádzať a udržiavať.)
b) definujte prístupové procedúry (selektory) pre jednotlivé časti a funkciu `mkBT` *pre vytvorenie nového stromu z podčastí*
b1) Vysvetlite, prečo je výhodné používať (len) túto funkciu v aplikáciách namiesto priamej tvorby d.š.
c) Postavte vyvážený strom z usporiadaného zoznamu.
(tvorba d.š., rekurzia, pomocné procedúry a interface)

108 Hornerovo schéma
a) výpočet hodnoty polynomu podľa Hornerovej schémy. Aké poradie koeficienov je vhodné?
b) (hi-ord) prevod koeficientov na funkciu
c) preklad do symbolického tvaru
d) Popíšte analogické procedúry pre koeficienty zadané v opačnom poradí (bez `reverze`*)*
(možnosti FP, hi-ord funkcie)

109 Preklad výrazu
a) Výpočet výrazu pri danom ohodnotení premenných (použitie `lookup`*)*
b) (hi-ord) Preklad výrazu na funkciu, ktorá k danému ohodnoteniu vracia výsledok.
c) preklad do symbolického tvaru, vyhodnotiteľné pomocou `eval`*.*
(dtto)

110 Funkcie vyššieho rádu
(higher order functions, funkcie, ktoré ako argumenty alebo výsledok majú iné funkcie)
a) funkcia `mkAddConst` *k danému číslu n vráti jednoargumentovú funkciu, ktorá k svojmu argumentu pričíta konštantu n. Napíšte príklad volania funkcie a použitia výsledku.*
b) funkcia `comp` *skladá dve jednoargumentové funkcie.*
c) opakovaná aplikácia c1) funkcia `opak` *k číslu n, funkcii f a argumentu x vráti hodnotu n-násobného použitia f na x.*
c2) analogicky funkcia `mkOpak` *vráti k n a f funkciu, ktorá na svoj jediný argument aplikuje f n-krát.* (Transformácia funkcií.)
c3) použite `opak` *a* `mkOpak` *pre (efektívny) výpočet n-tého Fibonacciho čísla.*
(hi-ord funkcie, rozdiel medzi interpretom a kompilátorom)

111 Map
a) map pre zoznamy, príklad volania: `(map f '(1 2 3))`
b) použite definíciu map z a) pre definovanie map na maticiach
c) napíšte volanie map, ktoré k danému zoznamu vráti zoznam jednoargumentových funkcií, v ktorom každá pričíta k svojmu argumentu konštantu zo zoznamu.
d) Vysvetlite (parametrický) "polymorfizmus" funkcie `map` *a porovnajte ho s prístupom, resp. možnosťami OOP.*
e) Navrhnite ďalšie aplikácie map.
f) Rozširujúce: map pre iné d.š.
(hi-ord f.)

112 Prechod bin. stromu
a) Navrhnite a1) interface funkcie a a2) implementáciu funkcie, ktorá bude podľa daného argumentu (-ých -ov) prechádzať binárny strom preorder, inorder, postorder, prepostorder, ... a vytvárať pre jednoduchosť zoznam listov.
b) Implementácia navrhnutá ako rozskok podla daného druhu prechodu (predávaného samozrejme symbolicky) nie je dostatočne obecná. Spôsob prechodu zadávajte funkciou a napíšte volania pre typické prechody stromu.
(hi-ord f., (dynamická) parametrizácia)

113 Dispatch
a) Hm.
a)
a) - riešenie problému s rôznym počtom argumentov funkcií (statický rozskok (nepoužitelný v HS), `apply`*)*
a) - TVM, kontrola tagu-reprezentácie, typecase (pre jednopredkové hierarchie)
a) - kontrola existencie metod podľa daného stromu hierarchie
(funkcie v d.š., symb. prog., použitie tagov, hi-ord. f., OOP)

114 (Syntax symbolických dát)
a) Porovnajte syntaktické rozlíšenie medzi symbolickými datami a premennými v Prologu, Scheme, Haskelli a C/Pascale/Jave (a XML).
a1) Rozoberte to na príklade práce s polynomami, kde máte symbolické (doménové) premenné v polynome.
a2) Nájdite ďalšie domény, kde sa pracuje s premennými. (L,C/I)
b) Vysvetlite rozdiel a prevod do prirodzeného jazyka pre volania v Scheme `(print heslo)` *a* `(print 'heslo)`*.*
c) (Z Pána prsteňov I: Gandalf a spol. pred vstupom do Morie) Řekni příteli a vejdi. Vysvetlite dve možné čítania.
d) Hm. Porovnanie spôsobov citovania: lexikálne konvencie, quote, ", escape znaky (a 16 apostrofov), ... backquote, zložené datá, (pointry).
d1) Generovanie zdrojákov, i v inom jazyku.

115 Vzdialenosť
a) Euklidovská vzdialenosť bodov (nerekurzívna proc.)
b) Euklidovská vzdialenosť vektorov. (Nevhodné je opakovane odmocňovať a umocňovať)
c) Vzdialenosť sa počíta roznymi sposobmi. Rozoberte možnosti predávania parametrov. (Dispatch: podľa mena/reťazca alebo funkcia)
d) Využite funkcie pre rozdiel vektorov a normu.

116 Numerická derivácia
a) Spočítajte numericky deriváciu danej funkcie f v danom bode x. Zvolte si požadovanú presnosť a prvotný odhad pre "blízky" bod.

117 Update datovej štruktúry
a) Je daný počiatočný stav datovej štruktúry a zoznam updatovacích procedúr ako zoznam dvojíc procedúra a jej argumenty. Napište interpret, ktorý vykoná požadované zmeny na datovej štruktúre.
(Toto je jedna z aplikácií, ktoré nejdú priamočiaro prepísať do Haskellu, pretože typy funkcií v zozname sú obecne rozne. Dala by sa predávať TVM (tabuľka virtuálnych metód) a reprezentácia typu pre každú funkciu.)

118 Vzdialenosť

119 Rezerva

7 Haskell

Všetky príklady zo Scheme sú naprogramovatelné v Haskeli, ale 1) už to je "len" opakovanie (líši sa to len syntaxou, nie myšlienkami), teda vhodné na DC.
2) niektoré operácie sú schované (napr. vďaka curryfikácii) a niektoré obraty nepoužiteľné (napr. vďaka typom)

120 Minulé príklady (P,S).
a) transpozícia matice, otočenie
b)+ filtrácia 1D, 2D
c0)+ f. `ekvRozklad` *pre rozklad zoznamu podľa (f.) parametrom zadanej ekvivalencie (používané ďalej)*
c)+ postavenie dopredného vyhľadávacieho stromu pre alg. Aho-Corasicková. (využíva `filter`*, delenie na triedy ekvivalencie* `ekvRozklad` *)*
d) stavba vyhľ. stromu, alebo haldy

121 Datové štruktúry *pre:*
a) Binárne stromy s vnútornými vrcholmi (BVS, haldy), bez vnútorných vrcholov (navrhnite aplikáciu), n-árne stromy (pre dopredné vyhľadávanie v Aho-Corasicková)
a1) Zaistite, aby vo vnútorných vrcholoch (vyhľadávacieho) stromu bola okrem kľúča aj (užitočná, informačná) hodnota.
b) Výrokové formule s klasickými spojkami (aj s menej klasickými), s konštantami (nad danou doménou, nie nutne Boolovskou), s premennými.
c) Programy vo vašom obľúbenom jazyku. Hm. Napište počítanie faktoriálu. :-)

122 Násobenie matíc
a) Skalárny súčin skalSouc :: ...
b) Násobenie matíc. Využite map, skalárny súčin a pomocnú procedúru.
```
multMat m1 m2 = map(ĺ->map(ŝ->skalSouc r1 r2)(trasp
m2))m1
```

123 Najdlhšia rastúca podpostupnosť
a) Dominancia postupnosti: a dominuje b, ak každé predĺženie b je aj platným predĺžením a. Postupnosť b si nemusíme pamätať, ak je dominovaná niečím.
b) Pomocou dynamického programovania nájdite (efektívne) najdlhšiu rastúcu (1) podpostupnosť. Vydajte dĺžku postupnosti.
c) Vydajte celú tabuľku a zrekonštruujte nájdenú postupnosť.
d) (1) varianty: d1) neklesajúca postupnosť, d2) klesajúca d3) nerastúca
(Metoda generuj a testuj všetky počiatočné úseky prejde skoro na dynamické programovanie, ak vyberáme len nedominované postupnosti. A z nich si stačí pamätať len "konečnú" informáciu - dĺžku a posledný prvok)
(Hm.)

124 Cyclic Redundancy Check
a) De facto počítanie zvyšku pri delení polynómov v Z_2, tj. bez prenosov. Prúdovo, vystupujúce "1" xor-ujeme s určenými bitmy stavu. Implementačné varianty: stav v n-tici, stav v zozname.

125 Hladový algoritmus
a) Do prednášok: hladový algoritmus, napr. pre farbenie grafu, s návrhovým vzorom Stratégia.

126 Šifrovanie kľúčovým slovom
a) aplikácia zipWith, cycle (alebo iterate - menej vhodne)
(b) ...
(pekná aplikácia)

127 Usporiadavajúca permutácia *K zoznamu prvkov vyrobiť usporiadavajúcu permutáciu. (viz)*
(Skladanie bežných funkcií.)

128 Rezerva

129 Rezerva

130 Rezerva

131 Rezerva

132 Map
a) pre zoznamy
b) pre dvojice, Maybe, Either (nerekurzívne d.š.)
c) pre binárne stromy, 2-3 stromy
d) pre n-árne stromy
e) pre logické výrazy, e2) apl: nahradzovanie premenných hodnotami.

133 Fold
a)
*a) aplikácia fold: kalkulačka. Hm. Popis. Vysvetlite, prečo sú vstupné štruktúry "zbytočne" zložité a nie je možné predať len *zoznam* tlačítok, ako v Prologu alebo Scheme.*
a) fold pre iné d.š.
a)
(volanie fold s hi-ord funkciou)

134 Rady
a0) úlohy b)-f) z nasledujúceho príkladu č. **??** *riešte pre (parametrom) dané n.*

135 Nekonečné d.š.
a) funkcia `repeat` *k danej hodnote x tvorí nekonečný zoznam hodnôt x.*
b) funkcia `iterate` *k danej hodnote x tvorí zoznam hodnôt $f^i(x)$ pre $i = 0 \ldots \infty$*
c) funkcia `cycle` *k danému zoznamu tvorí nekonečný zoznam cyklickým výberom hodnôt.*
d) Vytvorte nekonečnú jednotkovú maticu.
e) Rozširujúce: Vytvorte nekonečný zoznam Fibonacciho čísel
f) Eratostenovo síto, f2) Pascalov trojuholník.

136 Postupový kľúč
V n kvalifikačných turnajoch sa získavajú body pre postup do turnaja "Masters" s omedzeným počtom miest p. Každému účastníkovi sa sčítajú body z najviac m turnajov a postupuje p najlepších. Počet bodov za určité miesto je daný pevne.
a) Ak je daný zoznam výsledkov kvalifikačných turnajov, určite, kto postúpil.
b) Pre dané parametry určite minimálny počet bodov, ktoré zaručujú postup, tj. prvý nepostupujúci má (ostro) menej bodov než posledný postupujúci.
c) Riešte pre $n = 3$, $m = 2$, $p = 9$ a počty bodov za umiestnenie na prvých desiatich miestach $[100, 77, 60, 46, 36, 28, 22, 17, 13, 10]$.
(stručné zoznamy, generuj a testuj)

137 Multidimenzionálne vyhľadávanie
a) Postavte z daných multidimenzionálnych dát multidimenzionálny vyhľadávací strom s podporou neúplných dotazov. Návod: v strome sa (cyklicky) strieda, podľa ktorej dimenzie sa rozhoduje v danom vrchole.
a0) Bez neúplných dotazov.
a1)- Rozhodovanie o dimenzii a hodnote je dynamické, pomocou funkcie a podľa aktuálnych dát (rozhodovacie stromy).

a2)- (Ako) Je možné podporovať heterogénne multidimenzionálne dáta? (Reprezentácia kľúčov s `Either`*, selektorové funkcie, ! a ich update.)*
b) Vyhľadávanie podľa neúplného dotazu na navrhnutej štruktúre. b1) Ako vyriešite výstup, keď výsledkov môže byť viac?
c) Vyhľadávanie intervalových dotazov v navrhnutej štruktúre. V každej dimenzii je možné zadať (jeden) interval, v ktorom musia ležať všetky výsledky.
(d.š.)

138 Rez1.
a)
a)
()

139 Rez1.