Zápočtové príklady z Prologu

Jan Hric

16. listopadu 2004

Zápočtové príklady z prologu: v.10 16.11.2004 - ... v.9 21.11.2003 - ... v.8 25.11.2002 - 28.11.2002 v.7 22.11. - 7.12.2000, v.6, 11.11'99 - 26.11. v.5, 11.11.1998 v.4, 7.,10.11.1997 v.3, 1.11.1996 v.2, 13.11.1995 v.1, 25.11.1991

Podmienky na zápočet: odovzdať okomentovaný program s (primerane veľkými) testovacími dátami a dokumentáciu. Dokumentácia obsahuje meno autora, zadanie, meno použitej implementácie, popis riešenia, programu a datových štruktúr, popis I/O (volania a výstupu) s príkladmi, podrobnejšie viz samostatný súbor na mojej WWW stranke (http://kti.ms.mff.cuni.cz/ hric/ − > výuka − > dokumentácia). Program má (obvykle) obsahovať hlavný predikát (popísaný v dokumentácii) pracujúci "dávkovo" a testovacie dáta. Používajte štandardný Prolog (napr. SWI, LPA alebo BinProlog).

Ak nie je v zadaní podrobnejšie určená (alebo odporúčaná) reprezentácia dát, je jej vymyslenie súčasťou úlohy (ale napr. grafy rozoberieme na cvičení). Vyberte si jednu z variánt a), b), ... jednotlivých príkladov, ale počítajte s tým, že pri veľkom záujme môžete dostať inú variantu. Niektoré príklady alebo ich varianty sú označené ako (príliš) ľahké (pomocou (1)) a nie sú vhodné ako zápočtové. Lahke/zjednodusene varianty sluzia (aj) ako navod na prvu verziu programu a ukazuju, ako sa dospelo k "spravne" tazkej variante.

Tento výčet nie je úplný. Vlastnej iniciatíve sa medze nekladú: ak máte inú rozumne ťažkú/ľahkú úlohu, rozmyslite si jej (presné) zadanie a navrhnite mi ju. (Taketo ulohy sem doplnam.) Primerane tazka uloha by mala obsahovat:

– netrivialne datove struktury (nielen zoznamy alebo matice) specificke pre problemovu domenu, bud vstupne (interprety, spracovanie jazykov) alebo vnutorne

– netrivialny algoritmus (na ktorom sa nieco naucite), ktory nie je prepisom vzorcov z ucebnic

– primeranu parametrizaciu z reality, nema to byt (len) matematicka abstrakcia problemu. Realizovane pomocou predavania konfiguracie alebo kontextu ako parametru.

– overitelne a zdokumentovatelne vysledky; nielen najde sa *nejaky* vystup alebo pouzije sa *nejaka* heuristika.

Vstupy predpokladajte v strukturovanej podobe (!; syntakticka analyza zadarmo), symbolickej (nie retazce). a obvykle davkovej. Vystupy vyrobte primerane informacne, obvykle znakove (ich graficka "krasa" nie je to podstatne). V niektorych kompaktnych alebo vypoctovo narocnych ulohach je vhodne naprogramovat ladiace tlace, ktore zdokumentuju priebezne chovanie vaseho algoritmu (napr. u rozvrhovacich problemov alebo hier a hlavolamov). Uloha by mala byt kompaktna, vhodne nie su kniznice mnohych (nepreviazanych) procedur (napr. podpora pre pracu s kombinaciami, permutaciami a variaciami). U jednoduchych uloh je mozne sa dohodnut na naprogramovani konkretneho netrivialneho algoritmu.

Čísla udávajú mnou odhadnutú zložitosť:

(0),(1) - príliš ľahké na zápočet (Alebo nechcem, aby ste úlohu riešili v danej podobe. Často je to varianta, ktorá je "málo" parametrická)

(2) ľahké - (4) ťažké

Ak nie je uvedené nič, je úloha vhodná ako zápočtová.

Možná realizácia testovacích dát a demo predikátu:

```
demo:-
data(X),my_main(X,V),write(in(X)-out(V)),fail.
```

Programy by mali umožniť trasovanie (na žiadosť), tj. vypisovanie nejakých medzivýsledkov, ktoré charakterizujú postup výpočtu.

ZADANIA PRÍKLADOV:

KAPITOLA 1: Datové štruktúry.

121. Analýza podgrúp.

a) Pre danú (konečnú) grupu nájsť všetky jej podgrupy. Pre každú podgrupu určiť mnoninu generátorov a ďalej určiť ich vzájomné inklúzie.

b) pre iné algebraické štruktúry

b1) napr. teleso

122. Grupový izomorfizmus.

Pre dve dané grupy zistiť, či sú izomorfné. Vydať

a) (1) jeden izomorfizmus

b) všetky izomorfizmy (v datovej štruktúre), backtrackingom, ale s inteligentným dopočítavaním

b1) nájsť odpovedajúce si generátory a vydať len ich korešpondencie

b2) nájsť rády prvkov a skúšať izomorfizmus len pre prvky rovnakého rádu

123. Generovanie konečných štruktúr. (2002)

Je daná báza (konečná, výčtom), nad ktorou máte generovať štruktúru. Sú dané niektoré "obvyklé" požiadavky: asociativita operácia, symetria, existencia (lavej, pravej) jednotky, dustributivita ...

a) Máte vygenerovať všetky štruktúry, ktoré splňujú požiadavky a zistiť ich počet.

b) vygenerovať a spočítať len neizomorfné varianty.

KAPITOLA 2: Grafy.

222. Hranová súvislosť.

Pre daný graf G určite jeho hranovú súvislosŤ.

230. (1) Dosažitelnosť v hypergrafe.

Orientovaný hypergraf je zadaný dvoma množinami: množinou vrcholov V a množinou hyperhrán H. Každá hyperhrana h je aspoň dvojprvková podmnožina V, v ktorej je určený jeden výstupný vrchol, ostatné vrcholy hyperhrany sú vstupné.

Dosažitelnosť vrcholu z danej počiatočnej množiny M vrcholov je definovaná takto: Každý vrchol z množiny M je dosažitelný z M. Ak v1 je výstupný vrchol hyperhrany h1 a všetky vstupné vrcholy h1 sú dosažitelné z M, potom je v1 dosažitelný z M; hovoríme, že hranu h1 sme použili. Minimálny počet použitých hrán, pomocou ktorých je v1 dosažitelný z M, je vzdialenosť vrcholu v1 od M.

Úloha: Vstup je orientovaný hypergraf G a počiatočná množina vrcholov M. Nájdite všetky vrcholy G, ktoré sú dosažitelné z M a vydajte nejakú množinu dosvedčujúcich hrán. (Pôvodne: Určite horný odhad vzdialenosti vrcholov od M.) Výsledok vráťte ako datovú štruktúru. (Hľadáme len horné odhady, pretože hrana sa môže započítať viackrát.) (Použite napr. variantu Dijkstrovho algoritmu)

231. Vzdialenosť v hypergrafe.

Ohodnotený orientovaný hypergraf je zadaný dvoma množinami: množinou vrcholov V a množinou hyperhrán H. Každá hyperhrana h je aspoň dvojprvková podmnožina V, v ktorej je určený jeden výstupný vrchol (ostatné vrcholy hyperhrany sú vstupné), a je ohodnotená cenou, tj. nezáporným reálnym číslom.

Cena dosiahnutia vrcholu z danej počiatočnej množiny M vrcholov (ďalej len cena vrcholu) je definovaná takto: Každý vrchol z množiny M má cenu 0 (vzhľadom k M). Ak v1 je výstupný vrchol hyperhrany h1 s cenou c_h1 a všetky vstupné vrcholy v_i hrany h1 majú postupne ceny c_i, potom cena v1 cez hranu h1 je súčet cien c_i a ceny c_h1. Cena vrcholu v1 je minimum ceny cez všetky použitelné hrany, v ktorých je v1 výstupný vrchol. Hrana je použitelná, ak všetky vstupné vrcholy majú stanovenú cenu.

Úloha: Vstup je ohodnotený orientovaný hypergraf G, počiatočná množina vrcholov M a cieľový vrchol V. Zistite, či je vrchol V dosažitelný a ak je, určite jeho cenu (resp. horný odhad) a vráťte zoznam použitých hrán (v nejakej datovej štruktúre) (použite napr. variantu Dijkstrovho algoritmu)

232. Presná vzdialenosť v ohodnotenom hypergrafe.

Zadanie hypergrafu ako v príklade 231, ale nechceme horný odhad, ale presnú hodnotu ceny vrcholu, tj. hrany použité viackrát (pre medzivrcholy) sa počítajú len raz. Návod: zobecnite počítanie z vzdialenosti jednoho vrcholu na vzdialenosť množiny vrcholov.

232b. presná vzdialenosť v hypergrafe.

To isté, ale hrany nemajú ohodnotenie a zaujíma nás len ich minimálny počet.

240. Izomorfizmus orientovaných hypergrafov.

Vydať všetky izomorfizmy hypergrafov. Hľadať rozkladom, tj. vrcholu môžu byť v odpovedajúcich si triedach rozkladu, ak majú rovnaký počet vstupných aj výstupných hyperhrán rovnakého stupňa. Rekurzívne zjeňovanie, keď stupeň (celkový počet vrcholov) je nahradený počtom prvkov v jednotlivých triedach.

240b. dtto, ale iný vhodný algoritmus (nie tupý backtracking)

241. Izomorfizmus neorientovaných hypergrafov.

To isté, ale pre neorientované hypergrafy.

242. Izomorfizmus rozkladom pre iné štruktúry

Berú sa do úvahy konkrétne vlastnosti štruktúry, napr. závisí na poradí vrcholov (u bin. op. v grupe), je viac druhov operácií (...)

a) grupy

b) stromy (1)

250. Toky v sitich.

Je dan orientovany acyklicky graf a jeho dva vrcholy zdroj a stok. Ulohou je najit naximalni tok mezi zdrojem a stokem. a) Libovolný alg. b) Jako zlepsujici cestu pouzit tu s nejvetsi propustnosti c) Jako zlepsujici cestu pouzit nejakou nejkratsi (vytvorit podsit pouze pro hrany z nejkratsich cest, postupne zvetsovat delku.)

260. Generovanie konečných struktur.

(aj inde?) Pre daných n prvkov (nosič) generujte všetky algebraické štruktúry daných vlastností, napr. grupoidy, grupy, (čiastočné) usporiadania, grafy ...

a) s vylúčením izomorfných

b) (1) bez vylučovania izomorfných

c) Navrhnúť jazyk na popísanie vlastností relácií/operácií, pomcou ktorého by sa zadávali požiadavky na štruktúry. Príklady vlastností: asociativita, symetria, reflexivita, (slabá) antisymetria, (jednostranná) distibutivita, existencia (jednostrannej) jedničky, idempotencia …

(d) preformulovať do omedzujúcich podmienok)

261. Testovanie konečných štruktúr.

(automatické dokazovanie) Pre konečné štruktúry vygenerované v príkl. 260c) testovať, či majú ďalšie požadované vlastnosti. Tie sú zadané analogickým jazykom.

262. Grupy izomorfizmov.

a) grafu

b) nejakých konečných štruktúr (viz 260)

Pre (každú) vygenerovanú štruktúru určiť a analyzovať jej grupu izomorfizmov. Analýza: generátory grupy, rozložiteľnosť …

KAPITOLA 3: Hříčky.

314. Oslík v záhradke.

Počiatočný stav:

xxxxxxxxxxxxxx xÚÄÄÄÄÄÄÄx xÄÄÄĹÄÄĹÄŮ x O Ä xÄÄÄĹÄÁÄĹÄ xŔÄÄÄÁÄÄÄÁÄŮx xxxxxxxxxxxxxx

Úlohou je nájsť postupnosť ťahov, ktoré dostanú veľkú kostku označenú O (oslíka) úplne doprava doprostred, aby sa dala vysunúť z plánu. Počas práce sa všetky časti musia pohybovať vo vyhradenom mieste 5 x 4.

315. Polohove hlavolamy (2004)

Zahrnuje Oslika (314), Lloydovu 15 (311), Rubikovu kocku (312), ?n-cube, tri kruhy v rovine. Hlavolamy maju pevny pocet miest a medzi nimi sa posuvaju "kosticky" rozneho tvaru a polohy. Zadava sa popis hlavolamu (tvar, povolene tahy a ich podmienky), pociatocne a koncove pozicie.

a) dosazitelnost (mnozin) pozicii

b) dosazitelnost pozicii na dany pocet tahov

c) heuristicke prehladavanie

d) prehladavanie s dopomocou: su dane vhodne priebezne medzipozicie (neuplne popisane)

x1) zobecnit aj na pentomino

x2) skladanie 3D povrchu 4D-kocky

316. Drotove hlavolamy. (2004)

So snurkou. Na zaklade 2D premietnutia drotoveho hlavolamu riesi vytiahnutie a navlecenie snurky.

330. Výroba krížovky.

a) Je zadaný slovník, tajnička (a prípadne jej poloha), tvar krížovky (prípadne neúplný), úlohou je zostaviť krížovku.

b) (Kompilácia krížovky) dtto, štruktúra krížovky je presne určená. Úlohou je „skompilovať" tvar krížovky na prologovský cieľ, ktorý obsahuje len volania slovníka a testy rovnosti písmen na zhodných miestach.

Druhy krížoviek:

A) normálna, s oddeľovacími čiarami

B) s oddeľovacími políčkami

C) slabiková

D) s diagonálne delenými políčkami

E) … a iné

332. Pentomino.

Daná plocha sa má presne pokryť danými obrazcami.

a) (1) Typicky je plocha delená na štvorce a pokrýva sa (dvanástimi) obrazcami z piatich spojených štvorcov. Obrazce sa môžu otáčat a prevracať.

b) dtto, ale tvary obrazcov sú súčasťou zadania

c) dtto, trojuholníková štruktúra plochy

d) dtto, šesťuholníková štruktúra plochy

e) priestorové pentomino

333. Tangram

a) (1) Ako pentomino, ale obrazce sú štvorce a ich diagonálne oddelené poloviny. Pre pevné tvary obrazcov.

b) Tvary obrazcov sú súčasťou zadania.

c) Obrazce a tvary sú aj obdĺžniky 2x1 a ich diagonálne oddelené poloviny.

334. Ethernity

a) Úloha ako pentomino. Plocha a obrazce sú dané trojuholníkmi a ich polovicami.

335. Maltézsky kříž.

a) Pre daných 6 častí zistí, či môžu priestorovo tvoriť maltézsky kríž.

b) dtto, ale jednoznačne, až na symetriu

c) (4) dtto (a) + zistí, či sa dá maltézsky kríž rozložiť na jednotlivé časti.

340. NIM.

Z daných n kôpok zápaliek odoberajú dvaja súperi. Úlohou je nájsť a odohrať optimálnu stratégiu pre nejakú variantu hry. Varianty hry sa líšia ukončením a povolenými ťahmi:

1) kto nemá ťah, prehral (obvyklá varianta)

2) kto tiahol posledný, prehral

Povolené ťahy (vždy musíme vziať aspoň jednu zápalku):

a) smie sa vziať ľubovoľný počet zápaliek z jednej kôpky

b) smie sa vziať najviac n zápaliek z jednej kôpky

c) smie sa vziať práve i zápaliek z jednej kôpky, možnosti pre i sú dané výčtom

d) a) + smie sa vziať zhodný počet záp. z dvoch kôpok

e) iné druhy ťahov, zadané externým programom "Stav − > Stav", resp. "Stav" − > "Množina_následníkov"

Nájsť optimálnu stratégiu znamená nájsť (všetky) prehrávajúce pozície: ak som na ťahu, tak vždy prehrám.

i) Pre program "všetky pozície" znamená všetky menšie než zadaný počiatočný stav.

ii) Program má nájsť výčtom n "najmenších" prehrávajúcich pozícií.

iii) Tabelácia pre pozície s najviac N (vstup) zápalkami na najviac K kôpkach (inteligentné hľadanie bez opakovania, výstup bez symetrií).

Napr. pre b1 s jednou kôpkou sú prehr. pozície k*(n+1) pre prirodzené k.

Varianty:

iv) povolené ťahy sú dané výčtom, ak niekto nemôže hrať, je to remíza

v) ťah môže odoberať zápalky z viac kôpok.

Upozornenie: Vybraná zápočtová úloha sa zostaví z poskytnutých možností (skoro) ľubovoľne, ale musí byť odstatočne parametrická.

341. Generické hľadanie optimálnej stratégie.

a) Po doplnení procedúr možný_ťah, vyhravajúca_pozícia, prehrávajúca_pozícia … s vhodnými parametrami hraje optimálnu hru za jednoho z hráčov.

b) dtto, ale tabelácia prehrávajúcich pozícií

342. Mastermind alias logik.

Logik je hra, keď počítač hádá rozloženie farebných kolíčkov v N daných zdierkach. Možných farieb je K a 1) môžu, 2) nemôžu sa opakovať. Odpoveď na každý tip sú dve čísla: počet (farebne) správne uhádnutých kolíčkov na správnych miestach a počet správne uhádnutých kolíčkov na nesprávnych miestach.

a) (1) Program háda. (Každá ďalšia voľba je možné riešenie vzhľadom k minulým výsledkom)

b) (1) Program hádá aj vyhodnocuje. Časť, ktorá vyhodnocuje, je inteligentný podvodník: vyberie si takú (správnu) odpoveď, ktorá poskytuje najväčší počet možných riešení. Vydávať nejakú skontrolovateľnú štatistiku o riešeniach.

b1) (4?) dtto (b), ale vyberie si odpoveď, ktorá zaručuje najdlhšie hádanie súpera. (Uvedomte si, že sa to nemusí zhodovať s odpoveďou podľa b)

351. Zakódované obrázky (2001).

a) Na štvorcovej sieti je čiernobiely obrázok. Pre každý riadok a stĺpec obrázku je daný popis, ktorý obsahuje počty súvisiacich čiernych pixelov v príslušnom riadku alebo spĺpci. Jednotlivé skupiny ciernych bodov sú oddelené aspoň jedným bielym. Úlohou je z daného popisu zrekonštruovať všetky vyhovujúce obrázky.

a1) Naviac sa skupiny nesmú dotýkať rohom. (podla hry Lode)

b) Pre šesťuholníkovú sieť

c) Pre trojuholníkovú sieť

d) Pre 3D sieť.

KAPITOLA 4: Hry.

Mierne varovanie: Šachové úlohy sú pomerne pracné.

401. Zohrávka šachovej koncovky K + V kontra K

402. Zohrávka šachovej koncovky K + D kontra K

403. Zohrávka šachovej koncovky K + 2S kontra K

404. Zohrávka šachovej koncovky K + V kontra K + P

405. Riešenie šachových úloh.

a) Dvojťažky

b) n-ťažky

c) pomocný mat, exoúlohy …

406. Generovanie šachových úloh.

Generovanie „zaujímavých" dvojťažiek, n-ťažiek, … Zaujímavosť, kvalita, možnosti sú dané materiálom, neexistenciou vedľajších riešení …

410. Radca bridžových výnosov.

Na základe pravidiel z literatúry, zadanej ruky vynášajúceho hráča a prípadne licitácie poradí najlepší výnos (prípadne niekoľko výnosov, prípadne podá zdôvodnenie)

411. Bridž - dopočítavanie kariet

a) Pre dané informácie o stole, vlastnej ruke a doterajšej zohrávke sa pokúsi dopočítať karty ostatných hráčov. Výsledok sú intervaly možného počtu kariet.

b) dtto, aj informácie z licitácie, prípadne už v predspracovanej podobe (u ľavého obrancu piky sú 5-list, kára sú dlhšie než srdcia, …)

412. Bridžový licitátor. (4)

Na základe kariet v jednej ruke a predchádzajúcej licitácie vydá dalšiu hlášku. (príklad interface)

Skontrolovateľné, tj. na základe pravidiel o "sile" ruky spocita vhodnú hlášku.

b) Systém, do ktorého by sa dali licitačné pravidlá dodávať samostatne.

415. Mariášový licitátor. (2002)

Ako 412 Bridžový licitátor, pre mariáš.

420. Pasiáns.

Hraje samostatne nejakú variantu pasiánsu, vypisuje povolené možnosti a zvolené ťahy.

440. (1) Dáma.

a) Vygenerovať z danej pozície dámy všetky platné ťahy.

b) Vygenerovať všetky možné skoky ľubovoľnej dĺžky.

c) (1) zjednodušené (a) - nie sú tam dámy.

d) (1) zjednodušené (b) - nie sú tam dámy.

450. Piškvorky.

Počítač hrá piškvorky. Analýzou pozície pomocou vzorov (a) alebo prehľadaním na niekoľko ťahov dopredu (b) počítač určí najlepší ťah. Nad touto častou je hracie prostredie, ktoré umožní hrať proti počítaču alebo dvom počítačom proti sebe. Typické použitie vzorov:

a) V "databáze" je niekoľko vzorov a k nim odpovedajúce (najlepšie) ťahy a ich hodnota. Vyberá sa ťah s najlepšou hodnotou medzi všetkými vzormi alebo s najlepšou sumou hodnôt.

b) Vzor má svoju cenu a jeho výskyt v pozícii je takto ocenený. Cena (koncovej) pozície je cenou všetkých obsiahnutých vzorov. Najlepší ťah sa zistí prehľadávaním stromu metodou minimax vzhľadom k zistenej cene koncových pozícií.

451. Miny (2004)

Hm. Windowsovska NT-hra.

a) Dabelsky miner. Podsunie minu, ak tam moze byt.

b) Dokonaly pyrotechnik. Urci bezpecne policka.

c) Pravdepodobnostny pyrotechnik. Na zaklade znalosti poctu zostavajucich min urci pravdepodobnosti vyskytu.

D) Sestuholnikova siet

E) Trojuholnikova siet.

452. Sokoban (2004)

Hra. Riesenie pozicii.

b) varianty: ?sestuholnikovy, trojuholnikovy

453. Mars explorer (2004)

Robot v prekazkach alebo bludisku hlada cestu. Robot sa nedokaze obracat na mieste, ale zatacky maju dany mimalny polomer.

a) vie, tj. moze cuvat.

b) nemoze cuvat. Ale ma zadany minimalny bezpecny odstup od prekazok. (alebo tvar)

454. Stavba zeleznice (2004)

V hornatom terene (z vyskovo zadanych polygonov) staviame trat o danom povolenom stupani a polomere zatacok. Co najlacnejsie.

460. Analýza pozície go.

a) Pre danú koncovú pozíciu go systém určí (približne) živosť skupína následne územie a spočíta výsledok.

b) Pre danú (skoro koncovú) pozíciu hry systém určí miesta, kde je ešte možné tvoriť územie (alebo ktoré sú otvorené pre boj).

c) Pre danú pozíciu ko systém spočíta druh ko: (zjednodušene) koľkokrát ho ktorý hrác musí vyhrať.

d) ...

470 ... úlohy z iných hier

Typicky koncovky sú primerane jednoduché.

KAPITOLA 5: Interprety a spracovanie "zdrojových" textov.

501. Analýza prologovských štruktúr.

a) Klauzule v danom vstupnom programe sa spracujú tak, že sa spočíta, koľkokrát bol ktorý atom (s četnosťou) použitý ako funkčný symbol v datovej štruktúre, ako predikátový symbol v hlave (volanie) a v tele (cieľ). Zoznam sa utriedi a vypíše.

b) dtto , ale jemnejšie rozdelenie - funkčné symboly sa rozlíšia podľa použitia v jednotlivých procedúrach. Inšpirácia z krížových referrencií.

Program sa číta zo vstupného súboru (alebo je predaný cez parameter ako zoznam klauzulí).

502. Graf závislosti.

a)(1) Ak klauzula procedúry P obsahuje volanie proc. Q, tak P závisí na Q. Zistite graf závislosti.

b) (a) + nepriame závislosti, cyklické závislosti

c) (a) + závislosti cez vstavané metapredikáty (argumenty setof, bagof, call, not ...)

503. Analýza výrazu s operátormi. (Syntaktická analýza)

Je daná postupnosť tokenov a zoznam infixných (binárnych) operátorov s precedenciami. Tokeny sú atomy, čísla a zátvorky. Úlohou je analyzovať postupnosť a previesť ju do štruktúry termu alebo prefixného zápisu. Možná metóda: nájde sa prvý operátor s najväčšou precedenciou a vstupná postupnosť sa rozdelí na spracovanie častí pre a za operátorom. Časti výrazov v zátvorkách sa preskakujú. Spracovanie môže skončiť chybou "neanalyzovateľný výraz".

a) (1) bez rozlišovania asociativity - vstup musí byť dostatočne uzátvorkovaný

b) (1) so zadanou asociativitou

c) s rôznymi druhmi zátvoriek

d) "zatvorky" majú vnútorné symboly ("kľúčové slová"), tj. ohraničujú viac podvýrazov než jeden. Každá zátvorka začína a končí symbolom (pre jednoduchosť), takže nie sú nutné priority a asociativita. Zatvorka je napr. if - then - else - endif, a pod. K zátvorke je daná výstupná reprezentácia, tj. meno funkčného symbolu. (tzv. distfixné operátory)

e) d+b, distfixné a binárne op., s prioritami a asociativitou

pr.?- analyzuj([op(plus,4), op(krat,2)], ['(',x, plus, x,')', krat, 4], X).

X = [krat, plus, x, x, 4]

X = [krat/2, plus/2, x, x, 4]

X = [krat, [plus, x, x], 4]

X = krat(plus(x,x),4)

... a varianty

Podkapitola: Hradlové siete.

Hradlová sieť je množina hradiel, kde každé hradlo má jediný výstup, svoje jedinečné meno, operáciu a zoznam vstupov (v danom poradí). Vstupy sú buď mená iných hradiel alebo mená vstupných bodov siete. Sieť je acyklická, tzn. žiadne hradlo nezávisí na svojom výstupe. Výsledok jednoho hradla sa posiela na všetky vstupy daného mena, môže ich byť niekoľko.

Príklad reprezentácie siete pre výpočet x*y+z+x: [h(h(1),plus,[h(2),z,x]), h(h(2),krat,[x,y])]

, kde h(1) a h(2) sú mená hradiel, x,y,z sú mená vstupných bodov, hradlá majú rôzny počet vstupov.

512. Alokácia registrov

Hradlová sieť je množina hradiel, kde každé hradlo má jediný výstup, svoje jedinečné meno, operáciu a zoznam vstupov (s daným poradím). Vstupy sú buď mená iných hradiel alebo mená vstupných bodov siete. Sieť je acyklická, tzn. žiadne hradlo nezávisí na svojom výstupe.

Vstup programu je hradlová sieť. Úloha je

a) (1) usporiadať hradlá tak, aby každé hradlo bolo až za hradlami, ktoré počítajú jeho vstupy.

b) priradiť vstupným bodom siete a výsledkom hradiel "registre CPU". Register sa môže použiť pre výsledok ďalšieho hradla, ak hodnotu z neho už nepotrebujete na vstupe žiadneho hradla. Výstupný register hradla je vždy odlišný od všetkých vstupných registrov. Počet registrov nie je omedzený, ale snažte sa ich použiť čo najmenší počet (tj. vyhnite sa triviálnemu riešeniu, kedy výsledku každého hradla je priradený iný register).

513. (1) Prevod hradlovej siete na polynóm.

a) (1) Je dana acyklicka hradlova siet, kde hradla maju najviac dva vstupy a jeden vystup, ktory sa môže vetviť. Hradla su troch druhov podla funkcie ktoru realizuju: Plus, Krat, Unarne minus. Napiste program, ktory pre konkretne hradlo v sieti vypocita polynom, ktory vyjadruje zavislosť vystupu zvoleneho hradla na vstupe siete.

b) Varianta za a) má nevýhodu, že spoločné podvýrazy (podsiete) sa vo výslednom výraze objavia viackrát. Výstup v tejto variante bude postupnosť výrazov a ich prirazení, tak aby sa aj viackrát použité hradlá spočítali len raz. Výsledok je najkratšia postupnosť priradení, ktorá spočíta výraz tak, že žiadne hradlo sa nepočíta viackrát.

514. (1) Prevod hradlovej siete na logický výraz.

a) (1) Je dana acyklicka hradlova siet, kde hradla maju najviac dva vstupy a jeden vystup, ktory sa môže vetviť. Hradla su troch druhov podla funkcie ktoru realizuju: And, Or, unarne Not. Napiste program, ktory pre konkretne hradlo v sieti vypocita formulu, ktora vyjadruje zavislosť vystupu zvoleneho hradla na vstupe siete.

b) analogicky k 513b pre logický výraz.

515. (1) Hľadanie ekvivalencií v hradlovej sieti.

Hradlová sieť je množina hradiel, kde každé hradlo má jediný výstup, svoje jedinečné meno, operáciu a zoznam vstupov (s daným poradím). Vstupy sú buď mená iných hradiel alebo mená vstupných bodov siete. Sieť je acyklická, tzn. žiadne hradlo nezávisí na svojom výstupe.

Hradlá v sieti sú ekvivalentné, ak majú zhodnú operáciu a ekvivalentné alebo zhodné vstupy. Je daná hradlová sieť. Nájdite v nej všetky ekvivalentné hradlá a vyrobte novú sieť, v ktorej všetky navzájom ekvivalentné hradlá nahradíte jedným z nich.

Návod: Hradlá najprv usporiadajte a potom prechádzajte sieť od najnižšej vrstvy.

516. (1) Prevod aritmetického výrazu na hradlovú sieť

Hradlová sieť je množina hradiel, kde každé hradlo má jediný výstup, svoje jedinečné meno, operáciu a zoznam vstupov (s daným poradím). Vstupy sú buď mená iných hradiel alebo mená vstupných bodov siete. Sieť je acyklická, tzn. žiadne hradlo nezávisí na svojom výstupe.

Vstup programu je daný výraz v tvare prologovského termu s číslami, identifikátormi a operáciami +, *, -, /, sin a cos. Vyrobte k danému výrazu hradlovú sieť, ktorá bude počítať zhodný výraz zhodným postupom (tj. neoptimalizujte).

Máte k dispozícii hradlá s danými šiestimi operáciami. Hradlá s operáciami + a * môžu mať ľubovoľný počet argumentov a ich vstupom nesmie byť výsledok rovnakej operácie.

516b. (1) Prevod postupnosti výrazov na hr. sieť.

Podobné minulej úlohe. Prevádzate postupnosť priradení, raz spočítaný výsledok sa môže použiť niekoľko krát.

Motivácia: Jeden výraz má nevýhodu, že každý jeho podvýraz sa dá použiť len jedenkrát (implicitne na mieste výskytu)

517. (1) Prevod logického výrazu na hradlovú sieť

Hradlová sieť je množina hradiel, kde každé hradlo má jediný výstup, svoje jedinečné meno, operáciu a zoznam vstupov (s daným poradím). Vstupy sú buď mená iných hradiel alebo mená vstupných bodov siete. Sieť je acyklická, tzn. žiadne hradlo nezávisí na svojom výstupe.

Vstup programu je daný logický výraz v tvare prologovského termu s identifikátormi a operáciami and, or, not a imp. Vyrobte k danému výrazu hradlovú sieť, ktorá bude počítať zhodný výraz zhodným postupom (tj. neoptimalizujte).

Máte k dispozícii hradlá s danými štyrmi operáciami. Hradlá s operáciami and a or môžu mať ľubovoľný počet argumentov a ich vstupom nesmie byť výsledok rovnakej operácie.

518. Prevod vyrazov do stvoric. (2004)

Kompilacia postupnosti priradzovacich prikazov (Pascal, C, C++, Java) do stvoric (tj. hradiel).

a) optimalizacia spolocnych podvyrazov

b) bez optimalizacie

x) varianty

520. Interpret dataflow počítača.

Model dataflow počítača je (cyklická) hradlová sieť. Po hranách siete sa posielajú "pakety" - jednotky informácií a spracovania. Jednotlivé hradlá pracujú asynchronne, aktivujú sa, ak majú pakety na vstupoch. Hradlá obvykle požadujú pakety na všetkých vstupoch, aby sa mohli aktivovať. Okrem „datových" operácií sú potrebné aj „riadiace" operácie (a riadiace hrany a pakety), napr. if - 2 vstupy a 2 výstupy - jeden vstup je výsledok podmienky a podľa neho sa druhý vstup pošle na jeden z výstupov.

Asynchrónnosť znamená, že na spracovanie na určitom vstupe určitého hradla môže čakať niekoľko paketov na spracovanie (v danom poradí).

Jeden vygenerovaný paket sa obvykle posiela na jeden (určený) vstup. Môžu existovať deliace hradlá (posielajú jeden svoj vstup na všetky svoje výstupy), alebo delenie môže byť implicitné (ako u hr. sietí)

Úloha: naprogramovať interpret dataflow počítača a odladiť jednoduchý program, napr. faktoriál, alebo súčet vstupných paketov.

521. Petriho siete (2001)

Návrh reprezentácie, simulácia. Analýza siete: konečnosť (?), dosažiteľnosť stavu, vykonateľnosť danej množiny/postupnosti prechodov, ...

522. (Konecne) automaty (2004)

(Vhodne pre studentov 3. rocnika, ktori uz vedia, o com sa hovori.)

a) Simulator automatov. Reprezentacia by mala umoznit aj skladanie automatov.

b) Prevod regularneho vyrazu na konecny automat

c) prevod konecneho automatu na regularny vyraz. (?)

530. Optimalizácia (Pascalského) príkazu.

(a la 515. Hľadanie ekv. v hradlovej sieti)

Pascalský priradzovací príkaz sa prevedie na postupnosť priradzovacích príkazov tak, aby sa spoločné podvýrazy vyhodnocovali iba jedenkrát a uschovali v pomocných premenných. Výraz sa zadá v tvare vhodnej prologovskej reprezentácie.

a) len pre aritmetické operácie

b) vstavané funkcie, ...

531. Prevod siete na (pascalský) výraz.

a) (1) elementárne operácie

b) selektory polí a záznamov, ...

c) (4?) funkcie ...

532. (1) Konštrukcia logického výrazu.

a) (1) Pre danú logickú funkciu o n premenných zostrojí ekvivalentný logický výraz (zo spojek and, or, not).

a1) z daných spojek (nand alebo nor stačí)

b) dtto, minimálny

c) dtto (b), ale funkcia nemusí byť daná úplne, niektoré hodnoty môžu byť nevýznamné a je možné si ich zvoliť, aby výsledok bol čo najmenší.

533. (1) Analýza logickej siete.

Pre danú logickú sieť s and a or hradlami a s jedným výstupom hľadám hodnoty vstupov, pre ktoré je výstup pravda. Program vydá výsledok v datovej štruktúre (nie backtrackingom)

a) všetky možnosti

b) minimálne možnosti

c) sieť obsahuje aj not, ... (alias prehľadávanie AND-OR grafu)

540. L-systémy

Generovanie a interpretácia L-systémov na základe popisu. [2]

541. Riešič podmienok.

a) (4) Sú zadané podmienky v tvare rovností a nerovností na premenné a čísla. Premenné majú konečný obor hodnôt. Riešič hľadá možné riešenia pre dané podmienky tak, že propaguje známe informácie o jednotlivých premenných cez podmienky na ostatné premenné. Až nedokáže propagovať ďalšiu informáciu, zvolí si premennú a podľa nej vykoná rozbor prípadov.

b) v podmienkach môžu byť aritmetické operácie

c) Premenné v podmienkach sú logické - majú 2 možné hodnoty.

Použitie: riešenie algebrogramov, 8 dám na šachovnici, zostavovanie krížovky

542. Sústavy omedzení v konečných doménach

Omedzenia môžu byť troch druhov a sú dané v tvare termov:

1) obor(¡premenná¿ , ¡obor_hodnôt¿) kde premenná je atom reprezentujúci meno premennej a obor_hodnôr je množina elementárnych hodnôt, ktoré môže daná premenná nadobudnúť. Každá premenná má práve jeden obor hodnôt, ktorý môže byť aj jednoprvkový.

2) zhodne(¡množina_premenných¿) kde všetky premenné musia nadobudnúť tú istú hodnotu. Premenná sa vyskytuje najviac v jednom omedzení zhodne/1.

3) rozne(¡množina_premenných¿) kde premenné musia mať navzájom rôzne hodnoty. Vstup je zoznam omedzení a zoznam hodnôt niekoľkých premenných. Ak je premenná so známou hodnotou v omedzení zhodne/1, môžete jej hodnotu propagovať na ostatné premenné v omedzení. Ak je premenná so známou hodnotou v omedzení rozne/1, môžete jej hodnotu vylúčiť z oboru hodnôt ostatných premenných. Ak premennej ostane jednoprvkový obor hodnôt, premenná nadobudne túto hodnotu.

Výstup je buď informácia, že pri daných omedzeniach a hodnotách neexistuje riešenie (napr. ak jednej premennej sú v dvoch závislostiach priradené rôzne hodnoty) alebo zoznam premenných so známymi hodnotami

543. Sústavy aritmetických závislostí.

Závislosti sú dané v tvare termov: zavisle(¡zoznam_premenných¿ , ¡popis_závislostí¿) kde popis závislostí je množina trojíc: určujúce_premenné, určené_premenné, meno_predikátu s týmto významom: Ak sú známe hodnoty všetkých určujúcich premenných v niektorej trojici, potom je možné zavolať daný predikát s dvoma argumentami. Prvý argument je vstupný a obsahuje hodnoty určujúcich premenných, druhý argument je výstupný a po návrate z volania bude obsahovať spočítané hodnoty všetkých určených premenných. Predpokladajte, že potrebné predikáty sú pripravené mimo váš program. Vstup je zoznam závislostí a zoznam hodnôt niekoľkých premenných.

Známe hodnoty premenných sa použijú pre výpočet podľa závislostí. Výstup je buď informácia, že pri daných omedzeniach a hodnotách neexistuje riešenie (napr. ak jednej premennej sú v dvoch závislostiach priradené rôzne hodnoty) alebo zoznam premenných so známymi hodnotami

544. Sústavy aritmetických omedzení.

Omedzenia môžu byť dvoch druhov a sú dané v tvare termov: 1) obor(¡premenná¿ , ¡obor_hodnot¿) kde premenná je atom reprezentujúci meno premennej a obor_hodnôr je množina elementárnych hodnôt, ktoré môže daná premenná nadobudnúť. Každá premenná má práve jeden obor hodnôt, ktorý môže byť aj jednoprvkový.

2) zavisle(¡zoznam-premenných¿ , ¡predikát-redukcia-oborov¿) Predikát redukcie oborov má dva argumenty a použije sa vždy, ak sa zmení obor hodnôt aspoň jednej premennej. Prvý argument je vstupný a pri volaní musí obsahovať obory hodnôt premenných v poradí podľa zoznamu (1), druhý je výstupný a po návrate obsahuje nové zredukované obory hodnôt. Zredukovaný obor hodnôt môže byť aj prázdny.

Predpokladajte, že potrebné predikáty redukcie oborov sú pripravené mimo váš program a vy ich len používate. Vstup je zoznam omedzení a zoznam hodnôt niekoľkých premenných. Známe a neskôr aj zmenené hodnoty premenných sa použijú pre redukciu oborov hodnôt. Ak premennej ostane jednoprvkový obor hodnôt, premenná nadobudne túto hodnotu. Výstup je buď informácia, že pri daných omedzeniach a hodnotách neexistuje riešenie (napr. ak premenná má prázdny obor hodnôt) alebo zoznam premenných so známymi hodnotami a nový zoznam omedzení so zredukovanými obormi.

545. Generovanie grafu z podmienok.

a) Je daná postupnosť konečných omedzení vo forme zoznamu podmienok na rovnosť dvoch premenných, nerovnosť dvoch premenných alebo definícia premennej a jej domény. Úlohou je vygenerovať graf, kde vrcholy odpovedajú premenným a sú ohodnotené množinou prípustných hodnôt a hrany sú ohodnotené množinou prípustných dvojích hodnôt (v nejakom tvare).

b) na vytvorený graf sa vykoná statická redukcia: hodnoty vrcholov, ktoré nemajú žiadne odpovedajúce ohodnotenie hrany a susedného vrcholu, sa vypustia. Podobne hodnoty hrán, ktoré nemajú odpovedajúcu hodnotu vo vrchole. Vykonáva sa, kým je čo vypúšťať.

2a) obory hodnot sú celé alebo prirodzené čísla a podmienky môžu byť na rovnosť alebo nerovnosť dvoch výrazov. Výrazy obsahujú základné celočíselné operácie.

2b) analogicky

550. Overenie typov v Prologu.

Pre daný program a dané typy (a la Haskel) overiť, že program je typovo správny.

560. Prepisovací systém.

Je daná množina prepisovacích pravidiel (s premennými) tvaru ľavá_strana −− > pravá_strana a term t (bez premenných). pr. pravidla: krat(plus(x1,1),x2) −− > plus(x2,krat(x1,x2))

a) Nájdite v t všetky podtermy, na ktoré sa dá použiť nejaké prepisovacie pravidlo. Vydajte pravidlo (jeho meno alebo číslo), pozíciu v terme (napr. "3.2" znamená v treťom podterme druhý podterm: xx v plus(a, b, krat(c, xx))) a substitúciu.

b) Aplikujte všetky pravidlá, ktoré je možné použiť na t a novo vzniknuté termy. Úlohu si (kvôli konečnosti) vhodne omedzte, napr. aplikuje sa len určitý, parametrom zadaný, počet krát.

561. Hľadanie kritických párov

Kritický pár je dvojica termov, ktoré môžu vzniknúť z jednoho termu prepísaním rôznymi pravidlami.

Kritický pár v prepisovacom systéme vznikne, ak nejaký podterm na ľavej strane pravidla, ktorý nie je premenná, je možné prepísať podľa iného pravidla. Kritický pár tvorí pôvodná pravá strana a prepísaná ľavá strana.

a) Nájdite všetky kritické páry.

b) (4) (Zúplňovanie systému prepisovacích pravidiel)

Podkapitola: Doménové špecifické jazyky.

570. Kreslenie 2D obrázkov. (2003)

Navrhnúť jazyk a interpret pre "kreslenie" 2D obrázkov. (a la interpret mninových výrazov z cvičení) Elementárne objekty sú kruhy, štvorce, trojuholníky, (čiary) … Elementárne operácie sú transformácie otočenia, roztiahnutia/zúženia, (perspektivy) … a dalej prienik, zjednotenie (s daným prekrytím farieb, s parametrickým zmiešavaním (u farieb)) … Vlastný výstup stačí ako matica.

a) čiernobiele b) šedé c) farebné

571. Testovanie vlastností programov.

A la Quickcheck.

572. Generovanie programu pre plotter (2004)

a) Prevod zloziteho obrazku na postupnost „jednoduchych“ prikazov. Napr. srafovanie danej oblasti; co najmenej preruseni (zdvihnuti pera)

b) Optimalne.

573. Generovanie programu pre CNC stroj (2004)

Vyroba daneho tvaru (s urcitou danou presnostou) pomocou nastroja jednoducheho tvaru (valec, gula). Nastroj sa otaca v 5 osach. (3D plotter)

KAPITOLA 6. ... a ďalšie.

601. (1) Editačná vzdialenosť.

Editačná vzdialenosť dvoch reťazcov S a T je minimálny počet operácií nižšie vymenovaných druhov, ktoré musíme vykonať, aby sa S zmenil na T.

Obvyklé operácie sú INSert písmena, DELete písmena, REPlace jednoho písmena iným, EXChange vedľa seba stojacích písmen. ďalšia možná operácia je SWAp, kedy sa namiesto (postupnosti) písmen ABC napíše CBA

a) (1) Pre dva reťazce určiť ich editačnú vzdialenosť, ak sú povolené op. INS, DEL, REP, EXC. Vydať dosvedčujúcu editačnú postupnosť.

b) (1) Určiť editačnú vzdialenosť, ak sú povolené operácie dané parametrom.

c/ K danému slovu a počtu povolených chýb vygeneroval̂T prechodový automat /vyhľadávací stroj/, ktorý /pri interpretácii/ pre nejaké vstupné slovo určí minimálny počet chýb /tzv. Hammingovu vzdialenosť/ a prípadne editačnú postupnosť.

(Na túto úlohu /a,b/ sa dá použiť dynamické programovanie.)

d) konfigurovatelna editacna vzdialenost. (2004) Kazda operacia ma svoju cenu (i nekonecno ;-), zacatie diery a vsuvky ma svoju cenu (motivacia z porovnavania DNA), konkretne dvojice pismen pre nahradenie maju svoju cenu (motivacia z genetiky a preklepov).

ulohy: d1) minimalna vzdialenost dvoch retazcov, d2) vycet v poradi rastucej vzdialenosti, d3) ...

602. Univerzálny riešič dynamického programovania.

a) Zadá sa vzťah a kritérium, program si určí poradie a spočíta to. Ide vlastne o tabeláciu výpočtov, poradie môže byť dané implicitne, tj. podvýrazy sa vyhodnocujú podľa potreby.

b) určí sa aj poradie.

603. Letovanie plošných spojov.

... hm. Presné spočítanie dĺžky dynamickým programovaním pri jednom pravoľavom prechode (varianty: viac prechodov, lokálne cyklenie).

a) jednou hlavou / pájkou

b) viac hlavami, (ktoré do seba nemajú nabúrať)

(604.) (Iné úlohy dynamického programovania)

Minimálne

605. Kompresie (2004)

Nejaky algoritmus kompresii. Vhodne su asi slovnikove algoritmy, LZ77 posuvne okno, varianty LZR, LZSS LZB, LZH; LZ78 rastuci slovnik, varianty LZW, LZC,LZT, LZMW, LZJ, LZFG)

610. Rozvrhovanie. (viz stránka: R. Barták)

... Je daná ... uniformné/neuniformné zdroje, optimálne podľa kritéria / nejaký rozvrh, štartovacie a koncové časy činností vs bez nich, tvrdé vs. penalizačné časy, ...

Konkrétne úlohy: a) Letisko, b) alokace pultov (s ostrovmi), c) rozvrh sestričiek, d) vyrobný podnik, e) reklamy, f) osadzovanie súčiastok, letovanie (tj. viac obchodných cestujúcich) ...

611. Rozvrhovanie s roznorodými zdrojmi.

a) Máme daný výčet strojov vo výrobnej hale. Stroje sú roznych druhov. ďalej je daný zoznam výrobkov. U každého výrobku je popis jeho výrobnej postupnosti, tj. v ktorom čase (relatívne od začiatku jeho výroby) a ako dlho potrebuje jednotlivé druhy strojov (a v akom počte). Výroba každého výrobku je neprerušiteľná operácia. Napránujte výrobu, aby trvala čo najkratší čas.

b) Popis výrobku je prerušiteľný na miestach špecifikovaných vo výrobnej postupnosti. Naplánujte najkratšiu výrobu.

c) Na výrobkoch je dané čiastočné usporiadanie, ktoré určuje poradie, v akom sa musí začať ich výroba.

d) a) + Výroba je cyklická, chcem vyrobiť daný počet výrobkov v jednom cykle. Koniec a začiatok cyklu sa môže prípadne prekrývať. Snahou je naplánovať čo najkatší cyklus.

620. Heuristické hľadanie pre obchodného cestujúceho.

Navrhnuť rôzne heuristiky. Umožniť priebežné sledovanie.

630. Genetický algoritmus pre obchodného cestujúceho.

Hm.

631. Prostredie pre geneticke algoritmy.

Navrhnut nejaku reprezentaciu genomu, naprogramovat na nej zakladne geneticke operacie (mutaciu, krizenie, ...) a vyvinut prostredie, ktore bude zabezpecovat celkovy chod algoritmu podla danych parametrov. Ukazat chovanie na jednoduchom (umelom) priklade. Kriterium (fitness) vstupuje ako proceduralny parameter.

a) Genom je zoznam. Kriterium (fitness) je napr. pocet určitých znakov, zhoda s danym retazcom, počet určitých dvojznakových (resp. krátkych) postupností ...

b) Genom je matica. Kriterium je zhoda s danou maticou ...

c) Podpora populacii. Algoritmus nepracuje s jednou populaciou, ale s niekolkymi (pripadne s roznymi parametrami). Prostredie obsahuje operacie pre "komunikaciu" populacii.

632. Simulacia "Broucci" Hm.

640. Chemicke nazvoslovie

Zo vzorca vyrobiť český názov.

a) pre nesubstituované anorganické kyseliny a soli, a vybrané skupiny (amonna, kyanová ...)

b) aj kyslicniky (, kyanovodik a jeho derivaty)

c) aj pre substituované kyseliny. Jeden alebo viac kyslíkov je nahradených inými skupinami /napr. sírou, amidovou skup., ...: amidosíran draselný KSO3NH2/. Tiež spracovať: ??CaMg(CO3)2 uhličitan vapenatohorečnatý (dolomit).

d) Pre časť organickej chémie.

641. Chemické vzorce.

K názvu chemickej zlúčeniny vyrobiť vzorec.

b) pre časť organickej chémie.

Podkapitola: Spracovanie prirodzeneho jazyka

650. Morfologia. (2003)

Navrhnut system, ktory umozni popis jazykovych vzorov pre sklonovanie a casovanie. System zahrna datove struktury pre popis vzorov a interpretacny program. Vzory maju byt schopne popisat nielen koncovky, ale aj zmeny v kmeni (kun ¡-¿ kone, bra't ¡-¿ beru). Z rovnakych dat roznou interpretaciou sa vykona analyza aj generovanie.

a) Analyza. K danemu zakladnemu tvaru so vzorom a danemu tvaru slova urcit, ci tvar je odvoditelny. V tejto ulohe v principe dovolujeme "pregenerovanie", pretoze sa predpoklada, ze budeme zadavat len spravne tvary slova.

b) Generovanie (synteza). K danemu zakladnemu tvaru a popisu vzoru vygenerovat vsetky (alebo jeden urcity) tvary slova. Presne, bez nespravnych variant.

Data mozu pre obe ulohy mozu obsahovat len informacie o tvaroch slov alebo aj morfologicke informacie (rod, cislo, pad, osobu...), ktore sa hodia pri generovani.

Cielom je navrh obecnych datovych struktur a interpretu, nie popis celeho jazyka. Navrhnute dat. struktury by mali byt schopne popisat podobne ulohy: chemicke nazvoslovie (si'rovy' – siric"ity', uhelnaty' – uhlic"ity'), zdrobneliny (socha – soska), anglicke vzory: zdvojovanie spoluhlasok, vypustane e.

Idea popisu: pattern matching - porovnavanie retazcov (slov) so vzorkami

651. Eliza, cesky (2002)

Hm. a) Cesky

b) Anglicky. Navrhnut rozsirenia.

652. Konverzacna hra. (2003)

a) Na principe "adventury", analogicky Elize. Navrhnut netrivialne rozsirenie, napr. rozne modely chovania pre rozne "miestnosti", pre roznych ludi.

b) Spolupraca skupiny.

653. Generator pohadek (2004, Bojar)

Program bude z vestavěného slovníku generovat gramaticky správné české věty. Musí být schopen generovat věty obsahující:

- shodu podmětu s přísudkem - předloŘka plus shodný přívlastek k podstatnému jménu (podle toho velkého červeného medvěda) - vedlejŮí větu vztaŘnou (medvěd, který usnul) - řečnickou otázku a odpověď (Byl ten medvěd zlý? Nebyl.) - zájmeno, pokud uŘ v předchozím textu vygeneroval něco, k čemu se to zájmeno můŘe vztahovat

Program se zavolá jednoduŮe: chci pohádku o délce deseti vět. Výstupem je seznam slov a interpunkce.

Doporučené vestavěné limity: (AÝ lze snadno nastavit jinak!) - maximální počet přívlastků: 3 - maximální počet vnořených vedlejŮích vět: 3 - maximální vzdálenost mezi podstatným jménem a zájmenem: přes 1 větu

653.2 Generator popisov (2004)

(Povodne: generator dokumentacie) Z danej semantickej siete vygenerovat ceske (anglicke) vety podla danych omedzeni. Sem. siet je (multi)graf s ohodnotenymi (pomenovanymi) hranami a vrcholmi. Hrany a vrcholy su niekolkych *druhov* a vrcholy obsahuju *atributy*.

Omedzenia pre generaciu urcuju:

– tvar vydavanych viet (pocet privlastkov, pocet vnorenych viet, celkovu velkost vety a/alebo pocet popisanych vrcholov, pouzitie zamen (viz 653))

– strategiu prechodu grafom (do hlbky, do sirky, tematicky ...)

– druhy vydavanych informacii: filter - projekcia, co nas zaujima a ma byt na vystupe

– pripadne zakazanu, pozadovanu a preferovanu povrchovu syntax

Hrany obsahuju informaciu o vystupnom syntaktickom tvare (povrchovej syntaxi); tvarov moze byt niekolko. Atributy mozu byt multihodnoty. Jazyk nemusi byt dokonaly. Je mozne / vhodne vyuzit "modifikatory", ktore vylepsia vystup: v(miestnost) -¿ [v,miestnosti], od(cas) -¿ [od,casu], plural(kruh)-¿ [kruhy]

Priklad, popis grafu v tvare RDF: ¡vrchol¿ hrana ¡vrchol¿

```
<PRG005> druh <prednaska>
<PRG005> je_v <S3>, syntax [vr(1),je, v, vr(2)],
    [vr(1),kona, sa, v, vr(2)]
<S3> druh <miestnost>
```

```
<PRG005> nazov <Neproceduralne programovanie>,
    syntax ...
<PRG005> od <9:00>
<9:00> druh <cas>
<PRG005> do <10:30>
<10:30> druh <cas>
<PRG005> pre multi([I/2/31,I/2/32,...])
<I/2/31> druh <kruh> ...
```

Trivialny vystup je "Prednaska PRG005 je v !miestnost! S3. Prednaska PRG005 ma nazov Neproceduralne programovanie. Prednaska je od !cas! 9:00 do !cas! 10:30. ...". Vykricnikmi su oznacene miesta, kde by sa hodili modifikatory.

Lepsi vystup (ale nie dokonaly) pre prilis nazorny filter [vypisujDruhy([prednaska]), pouziZamena([prednaska]), vynechajHrany([do]), syntax(nazovvedlajsouVetou)] bude: "Prednaska PRG005, ktora sa nazyva Neproceduralne programovanie, kona sa v S3. Tato prednaska je od 9:00 pre I/2/31, ... I/2/39 a I/2/40".

a) Cestina vs. b) anglictina.

654. Prevod semantickej siete na RDF (2004)

a) Tam a naspat. Semanticka siet viz 653.2. RDF (viz http://w3c.org) su trojice (objekt, atribut, hodnota), hodnoty su atomicke alebo ine objekty.

b) Vcetne jednoduchych operacii na na RDF (hladanie najkratsich ciest daneho tvaru (reg. vyrazom))

c) Prevod medzi XML (viz 655) a sem. sietou

655. Rozbor publikacnich udaju (2004).

viz. Vypis v XML. napr.

```
<InProceedings name="enti_mis:2003">
    <author no='1'>Bojar, Ondřej</author>
    <author no='2'>Brom, Cyril</author>
    ...
    <title>ENTI -- Simulátor přirozeného
      prostředí lidského světa</title>
    <BookTitle>MIS 2003</BookTitle>
    <Pages from='3' to='14'/>
    <Publisher>MATFYZPRESS</Publisher>
    <PubAddress>Praha</PubAddress>
    <Year>2003</Year>
    <Month>January 18--25</Month>
    <Note>MSM113200006, LN00A063</Note>
}
```

vice o XML: http://www.w3c.org/

(orig. Bojar): Program dostane na vstup řetězec obsahující publikační údaje (např. "Ondřej Bojar, Cyril Brom, Milan Hladík, MikuláŬ Vejlupek, Vojtěch Toman, and David Voňka. 2003. ENTI - Simulátor přirozeného prostředí lidského světa. In MIS 2003, pages 3-14. MATFYZPRESS, January 18-25, 2003. MSM113200006, LN00A063." ap.). Program bude postupně navrhovat *smysluplná* čtení těch údajů a vypisovat je ve formátu BibTeXu, např.:

```
@InProceedings{enti_mis:2003,
    author = {Bojar, Ondřej and Brom, Cyril
      and Hladík, Milan and Vejlupek, MikuláŬ
      and Toman, Vojtěch and Voňka, David},
    title = "{ENTI -- Simulátor přirozeného
      prostředí lidského světa}",
    BookTitle = {MIS 2003},
    Pages = {3--14},
    Publisher = {MATFYZPRESS},
    PubAddress = {Praha},
    Year = {2003},
    Month = {January 18--25},
    Note = {MSM113200006, LN00A063}
}
```

Více o BibTeXu na webu. Např. http://newton.ex.ac.uk /tex/pack/bibtex/btxdoc/btxdoc.html

656. Rozbor inzeratov. (2004)

viz 655. Rozbor inzeratov z novin. Vystup v strukturovanom tvare.

Domeny: a) byty; b) auta; c) praca d) a dalsie

660. Transformacie XML (2004)

Pre dokument v XML reprezentovany v prologovskej strukture (!) navrhnite popis obecnych transformacii dokumentu do XML.

b) Realizujte aj vystup XML do obvyklej formy.

A DALSIE:

-xxx Backpropagation Hm.

- (28.11.2002, RK) Eliza; zjednodušovanie výrazov, formálne derivovanie; práca s polynomami viac premenných (aj spoločný deliteľ); zebra; fraktály - generovanie výrazov pre plotter; prevod algebraických notací, prevod formule do prenexného tvaru; kombinatorické algoritmy; generování všech struktur nad konečnou množinou (jaká jsou neisomorfní částečná uspořádání nad množinou o N prvcích) - viz 260; Gaussova eliminace ; práce s permutacemi; "expertní systém" ; manipulace s konečnými automaty; ...

[1] L. Kučera: Kombinatorické algoritmy, SNTL, Praha [2] M Chytil: Automaty a gramatiky, SNTL, Praha, 1984

PISOMKOVe priklady 15. a 19.11.1999:

Za neuspesne napisanie pisomky mate odovdzat niekolko prikladov. Vyberte si z lahkych zapoctovych (napr. 201,203,205,210,320,321...). Odovzdajte na papieri, za dva tyzdne (6.12. resp. 10.12.).

Nevhodne ako zapoctove priklady, typicky prilis lahke.

Polynómy - úvod: (Kapitolka z počítačovej algebry)

Základné spôsoby reprezentácie polynómov (vektorov, matíc,...) viacerých premenných môžeme rozdeliť

A) podľa toho, či sú uvedené všetky koeficienty na:

A.1) riedké - sú uvedené len nenulové koeficienty s príslušnou mocninou u premennej/premenných

A.2) husté - sú uvedené všetky koeficienty v rastúcom alebo klasajúcom poradí. V prípade rastúceho poradia nemusíme ani uvádzať stupeň polymonu.

B) podľa spôsobu zhlukovania premenných:

B.1) rekurzívne - polynóm viacerých premenných je zapísaný ako polynóm jednej premennej, ktorý má ako koeficienty polynomy v ostatných premenných

B.2) distributívne - polynóm je zapísaný ako súčet termov, z ktorých je každý súčin premenných a koeficientu - obvykle čísla.

B.3) zmiešané

C) Hornerovo schéma

101. (1)Prevod polynomu do žiadanej formy.

Zadanie: Vstup je ľubovolne uzátvorkovaný výraz s číslami, premennými a operáciami '+', '-', '*' a `'^'` (tj. polynóm), a zoznam premenných, ktorý udáva dôležitosť premenných. V príklade nižšie bude hlavný polynom v premennej 'x', jeho koeficienty v 'y'. Pr. volania: `?-uprav( (x-2*y)^2*(3*z+x), [x,y], X).`

Úloha: Preveďte vstup na: a) riedky rekurzívny zápis (1) b) riedky distributívny zápis (1) c) hustý rekurzívny zápis (1) d) hustý distributívny zápis (1)

e) riedke Hornerovo schéma (2)

f) husté Hornerovo schéma (2)

g1) (2) Premenné sú zadané v zozname zoznamov. Každý prvok, tj. zoznam premenných, reprezentuje jednu uroveň rekurzívneho vnorenia polynomov, premenné v zozname budú roznásobené. Pr. Zoznam [[x,y],[u,v]] znamená, že výsledok bude roznásobený polynóm v x a y a jeho koeficienty budú roznásobené polynómy v u a v. Výsledok je hustý zápis.

g2) dtto. Výsledok je riedky zápis.

102. (1)-ľahké. Prevod polynómu do riedkeho distributívneho tvaru

Riedky distributívny tvar polynómu viacerých premenných je zadaný zoznamom premenných, ktoré sa v polynóme vyskytujú, a zoznamom sčítancov - faktorov. Každý faktor je v tvare zoznamu, jeho hlava je absolútny člen a ostatné prvky sú exponenty jednotlivých premenných podľa poradia daného zoznamom. Zhodné mocniny u všetkých premenných sa nevyskytujú dvakrát (v takom prípade môžeme sčítať absolútne členy).

Pr. polynóm `(2*x^3+4)*y` je pri danom poradí premenných [y,x] zapísaný ako [[4,1,0],[2,1,3]], tj. `4*y^1*x^0+2*y^1*x^3`

Úloha: Vstup programu je zoznam premenných a polynóm v týchto premenných, ktorý obsahuje operácie sčítanie, odčítanie, násobenie a umocňovanie na prirodzené číslo. Preveďte polynóm do riedkeho distributívneho tvaru. Uvedomte si, že umocňovať a násobiť môžete aj celé polynómy, napr. `((x+2)*(y*z))^5`.

110. (1) Datová štruktúra trie.

Slúži na reprezentáciu postupností nad úplne usporiadanou množinou (napr. číslami, znakmi ...). Každý vrchol reprezentuje prefix nejakej postupnosti. Trie je (binárny) vyhľadávací strom podľa jednej zložky postupnosti, ktorý obsahuje v každom vrchole „trie" pre postupnosti s dosiaľ nájdeným prefixom. (pomôcka: neprázdny trie má 4 zložky: ľavý a pravý pod-trie pre vyhľadávanie podľa prvej zložky, informáciu pre tu končiaci prefix a pod-trie pre dlhšie postupnosti s týmto prefixom)

a) Implementovať vyhľadávanie, pridávanie, (uberanie) b) (a) + Výroba vyváženého trie z daných postupností

111. (1) Implementácia algoritmu Aho-Corasicková.

dopredná funkcia, spätná funkcia, interpret zvolenej d.š.

120. Hľadanie podgrúp.

Úvod. Grupa je daná, podobne ako grafy, nejakým abstraktným rozhraním, napr. pomocou predikátov elem(G,X), elems(G,Xs), unit(G,X), inv(G,X,IX) a oper(G,X,Y,XY), ktoré určujú pre danú grupu G postupne (backtrackingom) prvok, všetky prvky, jednotku, inverziu a binárnu operáciu. Podstatné na rozhraní je, že grupová operácia (a ostatné súčasti) môžu byť dané tabuľkou alebo predpisom výpočtu (napr. grupa násobenia modulo 1001).

a) Je daná grupa a množina generátorov. Úlohou je nájsť najmenšiu generovanú podgrupu a vyjadriť každý jej prvok pomocou generátorov.

b) Pre iné algebraické štruktúry ...

b1) napr. teleso

201. Izomorfizmus grafov.

a) (1) slepým prehľadávaním

b) algoritmus rozkladu na triedy so zhodnými stupňami vrcholov [1], viz 240.

ba) orientované

bb) neorientované

202. (1) Rozklad orientovaného grafu na silne súvislé komponenty. - efektívne

203. (0) Redukcia orientovaného grafu podľa daných tried vrcholov.

204. (1) Testovanie k-súvislosti grafu. a) hranovej: odobratie k-1 hrán nezvačší počet komponent b) vrcho-

lovej: -"- k-1 vrcholov -"- c) rozklad na 2-súvislé komponenty

205. Ofarbenie grafu.

a) (0) Minimálne, prehľadávaním.

b) (2) Heuristické, [1]

c) (1) Pomocou nezávislých množín.

V grafe sa nájde nezávislá množina, jej vrcholy sa ofarbia jednou farbou a potom sa ofarbí (indukovaný) podgraf na zostavajúcich vrcholoch.

206. Maximálne párovanie.

a) (0) prehľadávaním

b) (2) lepší algoritmus [1]

c) (1) párovanie v bipartitnom grafe

210. (1) Rozklad na Eulerovské grafy.

Pre daný graf vydať minimálny počet sledov (čiar jedným ťahom), ktoré graf nakreslia.

211. (1) Maximálny sled.

Pre daný hranovo ohodnotený graf zistiť sled s maximálnym ohodnotením.

220. Minimálne cesty.

a) Pre hranovo ohodnotený (orientovaný) graf s n vrcholmi sa spočítajú minimálne cesty medzi všetkými vrcholmi takto: Pole i,j v matici n x n obsahuje v k-tom kroku dĺžku minimálnej cesty z i do j cez medziľahlé vrcholy 1 až k. Na začiatku k=0 a matica obsahuje len priame cesty - hrany. Na konci k=n a cesta môže byť ľubovoľná. Matica v k+1. kroku sa spočíta z matice v k. kroku. (pozn.: zložitosť $O(n^3)$)

Naviac rekonštrukcia minimálnej cesty z matice.

221. (1) Minimálna kostra

a) Pre daný hranovo ohodnotený graf naprogramujte hľadanie minimálnej kostry b) dtto, ale maximálnej

301. (1) Šifrovanie zrkadlovou mriežkou. Mriežka sa namiesto otáčania prevracia, najprv podľa zvislej osy, potom vodorovnej, a nakoniec zvislej. Ďalšie prevrátenie podľa vodorovnej osy by ju dostalo do pôvodnej polohy. Šifrovaná postupnosť je dlhšia ako počet polí v mriežke. Je daná (správna) mriežka a vstup, výstup je postupnosť matíc písmen, prípadne linearizovaná.

302. (1) Výroba zrkadlovej mriežky. Postupne pri backtrackingu vydávať jednotlivé mriežky, ktoré: a) (1) nemajú políčka spojené hranou alebo rohom. b) majú najviac daný počet takých spojení

310. (1) Prelievanie vody. (Aplikácia prehľadávania grafu)

a) Je daných n nádob (n je vstupný parameter!), ich objemy, počiatočné naplnenie a požadované koncové naplnenie. Úlohou je nájsť nejakú postupnosť preliatí, ktoré vedú z počiatočného do koncového stavu.

b) dtto, ale najkratšiu postupnosť

c) Okrem nádob je ešte nevyčerpatelný zdroj a nepreplnitelný kanál. (Nádoby sú na začiatku typicky prázdne)

d) dtto (c), ale najkratšie riešenie

311. (1) Lloydova 15.

V štvorci 4 x 4 je 15 kostičiek s číslami a jedno volné pole.

a) poskladať

b) (a) nadaný počet ťahov

c) dosažiteľnosť dvoch pozícií

d) (c) na daný počet ťahov

312. (1) Zmenšená Lloydova 9.

Ako úloha 311, ale na 3 x 3.

313. (1) Rubikova kocka.

a) poskladať

b) (a) na daný počet ťahov

c) dosažiteľnosť dvoch pozícií

d) (c) na daný počet ťahov

320. (1) Výroba magických štvorcov

a) (0) nediagonálnych

b) (1) diagonálnych

c) dokonalých - aj prerušené uhlopriečky majú zhodný súčet Úlohu neriešte backtrackingom. Chcete vydať všetky štvorce, resp. určený počet, v datovej štruktúre.

321.(1) Generovanie latinských štvorcov

Latinský štvorec rádu n obsahuje čísla 1 až n, každé n krát. Sú usporiadané tak, že v žiadnom stĺpci ani riadku sa žiadne číslo nevyskytuje dvakrát.

Naprogramujte generovanie latinského štvorca daného rádu. Snažte sa o rýchle vylúčenie nevyhovujúcich možností, tj. každé číslo, ktoré pridávate do štruktúry štvorca, skontrolujte na čo najväčší počet podmienok voči dosiaľ vygenerovanej časti štvorca.

a) (1) riešenia vydávajte backtrackingom

b) Riešenia vydajte v datovej štruktúre, všetky alebo určený počet.

321. (1) Generovanie ortogonálneho latinského štvorca

Latinský štvorec rádu n obsahuje čísla 1 až n, každé n krát. Sú usporiadané tak, že v žiadnom stĺpci ani riadku sa žiadne číslo nevyskytuje dvakrát.

Ortogonálny latinský štvorec obsahuje dvojice čísel (i,j), 1 =< i,j =< n, každú jedenkrát. V prvej aj druhej zložke je to latinský štvorec.

Naprogramujte generovanie ortogonálneho latinského štvorca daného rádu. Snažte sa o rýchle vylúčenie nevyhovujúcich možností, tj. každé číslo alebo dvojicu, ktorú pridávate do štruktúry štvorca, skontrolujte na čo najväčší počet podmienok voči dosiaľ vygenerovanej časti štvorca.

a) (1) Riešenia vydávajte backtrackingom

b) Riešenia vydajte v datovej štruktúre, všetky alebo určený počet.

343. (1) Riešenie sústav algebrogramov.

a) S operáciami +,-,*

b) dtto, + presné celočíselné delenie, (div, mod ...)

350. (1) Reťazce domina.

Sú dané dominové kostičky a prirodzené číslo - počet reťazcov. Úlohou je zistiť, maximálne koľko kostičiek sa dá poskladať do daného počtu reťazcov.

510. (1) Výpočet hradlovej siete.

Acyklickej. a) (1) logickej b) (1) aritmetickej c) s externe, genericky zadanými operáciami Menám operácií odpovedajú prologovské procedúry a pri výpočte sa predajú ako parametre.

511. (1) Usporiadanie hradlovej siete. (alias topologické triedenie)

Hradlová sieť je množina hradiel, kde každé hradlo má jediný výstup, svoje jedinečné meno, operáciu a zoznam vstupov (s daným poradím). Vstupy sú buď mená iných hradiel alebo mená vstupných bodov siete. Sieť je acyklická, tzn. žiadne hradlo nezávisí na svojom výstupe.

Vstup programu je hradlová sieť. Úlohou je usporiadať hradlá tak, aby každé hradlo bolo až za hradlami, ktoré počítajú jeho vstupy.