Príklady na cvičenia ADS I.

Jan Hric, KTIML MFF UK
e-mail: Jan.Hric@mff.cuni.cz
http://ktiml.ms.mff.cuni.cz/~hric/vyuka/alg/cvads.pdf(,.ps)

30. ledna 2013

Zapocty 11/12: pre dialkove, kombinovane, ... studium a studentov z *mojich* cvičení: vypracujte: 6, 24, 43, 45, 65 (celkom 5 prikladov)

1 Asymptotická zložitosť

1 Porovnávanie algoritmov

a) Prečo sa algoritmy porovnávajú?
b) Ako sa porovnávajú? (vzhľadom k rôznym dátam) Prečo nás nezaujímajú konkrétne konštanty (multiplikatívna a aditívna)? Meranie veľkosti vstupu.
c) Čo sa porovnáva? (hľadiská)
d) -Čo sa optimalizuje?
e) Asymptotická notácia: $O, \Omega, \Theta, o, \omega$. *(Variantne: vybraná podpostupnosť)*

2 Príklady algoritmov

a) pre časovú zložitosť triedy $O(1)$, $O(\log n)$, $O(n)$, $O(n \log n)$, $O(n^2)$, $O(n^3)$, $O(2^n)$, $O(n!)$.
b) pre zložitosť $O(n^c)$*, kde c nie je celé číslo.*
c) Môže mať algoritmus časovú zložitosť $o(n)$*. Čo majú také algoritmy spoločné?*
d) Závislosť na dvoch (i viacerých) premenných.
e) funkcie s nie rastúcim chovaním: (niektoré) menšie dáta trvajú dlhšie. Ako odstrániť túto nepravidelnosť?

3 Výpočet konštánt

a) $\frac{1}{2}n^2 - 3n = \Theta(n^2)$. *Spočítajte konkrétne dosvedčujúce konštanty.*

4 Porovnanie dvojíc

Rozhodnite, ktoré vzťahy $f = X(g), X \in \{O, \Theta, \Omega, o, \omega\}$*, platia pre nasledujúce dvojice funkcií v premenných* n, m*, ostatné parametry označujú konštanty:*
a) $\log^k n \qquad n^\epsilon$*, (k a* ϵ *sú konštanty)*
b) $n^k \qquad c^n$,
b1) $n^k \qquad n^{k+\epsilon}$,
c) $\sqrt{n} \qquad n^{\sin n}$
d) $2^n \qquad 2^{\frac{n}{2}}$
e) $n^{\log m} \qquad m^{\log n}$
f) $\log(n!) \qquad \log(n^n)$
g) $(k_1)^n \qquad (k_2)^n$
h) $\log n \qquad \log_{10} n$
i) $2^n \qquad 2^{n+k}$
j) $2^{(n^{1/3})} \qquad 2^n$

5 Zoraďte

a) Zoraďte nasledujúce funkcie do postupnosti f_1, f_2, ... *tak, že* $f_1 = O(f_2)$, $f_2 = O(f_3)$, ...
b) Ďalej určite rozklad, v ktorom funkcie f a g ležia v jednej triede , práve ak $f(n) = \Theta(g(n))$.
$n! \qquad n^{\frac{1}{\log n}} \qquad e^n \qquad n^2 \qquad \ln n! \qquad 2^{2^{n+1}}$
$\ln\ln n \qquad n \qquad 4^{\ln n} \qquad n \log n \qquad \ln n^{\ln n}$
a ďalšie:
$1 \qquad n.2^n \qquad n^{\ln\ln n} \qquad \ln n \qquad (n+1)!$
2^k*, k pevné* $\qquad 2^{(n^{1/2})}$ ...

6 Vlastnosti $O()$. *ZAPOCET 11/12: b,! e,i3*

Dokážte alebo vyvráťte: Pre každú dvojicu funkcií $f, g: \mathcal{N} \to \mathcal{R}$ *platí:*
a) Ak $f(n) \in O(g(n))$*, potom* $g(n) \in O(f(n))$
b) Ak $f(n) = O(g(n))$*, potom* $2^{f(n)} = O(2^{g(n)})$
c) $f(n) = O(g(n))$*, akk* $g(n) = \Omega(f(n))$*, (symetria)*
d) $f(n) = O(f(n)^2)$
d2) $f(n) = \Theta(f(n/2))$*; pre ktoré f platí?*
e) $\forall h : f(n) = O(g(n))$ *a* $g(n) = O(h(n))$*, potom* $f(n) = O(h(n))$*; (tranzitivita)*
e2) $\forall h : f(n) = \Theta(g(n))$ *a* $g(n) = \Theta(h(n))$*, potom* $f(n) = \Theta(h(n))$*;*
f) f, g nezáporné: $\max(f(n), g(n)) = \Theta(f(n) + g(n))$
g) $f = O(h) \wedge g = O(h)$*, potom* $f(n) + g(n) = O(h)$*, (aritmetika)*
g2) $f = \Theta(h) \wedge g = \Theta(h)$*, potom* $f(n) + g(n) = \Theta(h)$
g3) $f = \Theta(h) \wedge g = \Theta(i)$*, potom* $f(n) + g(n) = \Theta(h(n) + i(n))$

h) $f = O(h) \wedge g = O(i)$, potom $f(n).g(n) = O(h(n).i(n))$, (aritmetika)
i) $f(n) + o(f(n)) = \Theta(f(n))$.
i2) $g = o(f)$, potom $f(n) + g(n) = \Theta(f(n))$.
i3) $f(n) = \Theta(h(n))$, $g = o(f)$, potom $f(n) + g(n) = \Theta(h(n))$.
j) …transformácia rastúcou funkciou h zachováva f=O(g) … hm, občas, za akých podmienok?
k) $f = o(h) \wedge g = o(h)$, potom $f(n) + g(n) = o(h)$, (aritmetika)
k2) $f = o(h) \wedge g = o(i)$, potom $f(n) + g(n) = o(h(n) + i(n))$ "malé" o
do prednášok: c) v jednom smere, tranzitivitu o():$\forall h : f(n) = o(g(n))$ a $g(n) = o(h(n))$, potom $f(n) = o(h(n))$; a pod.

7 Nájdite chybu *Dokážeme, že $f(n) = n^2$ je $O(n)$, indukciou. Pre $n = 1$ platí. Pre $n > 1$:*
$f(n) = O(n) + f(n-1) = O(n) + O(n)$ (z predpokladu) $= O(n)$.

8 Limity *Dokážte:*
a) $f(n), g(n)$ nezáporné:
$f(n) = O(g(n)) \rightarrow \exists c \in \mathcal{R}^+ \limsup_{n\to\infty}(f(n)/g(n)) = c$
b) $f(n), g(n)$ nezáporné:
$f(n) = o(g(n)) \rightarrow \limsup_{n\to\infty}(f(n)/g(n)) = 0$
c) Platia tvrdenia v opačnom smere?
d) Zformulujte a dokážte analogické tvrdenia pre Θ, Ω, ω

2 Stromy

9 Lineárna reprezentácia stromov *(programátorská úloha)*
Navrhnite sposob zápisu stromov (zatiaľ binárnych, nielen vyhľadávacích), aby ste z neho dokázali jednoznačne zrekonštruovať ľubovoľný daný strom.

Zobecnite aj pre B-stromy. (Neskor zobecníme aj na ľubovoľné grafové štruktúry)

Použitie: zadávanie príkladov v tomto (textovom) súbore, stromy na vstupe ako string, predávanie stromov medzi programami (sú aj iné sposoby predavania)

10 Porovnanie zložitosti operácií *Pre dynamické datové štruktúry popíšte:*
- aké majú operácie a ich výynam (find, insert, delete, min, max, succ, pred)
- ako je možné d.d.s. reprezentovat. (BVS, vyvážený BVS, Spojový zoznam, usporiadaný spojový zoznam, pole, usp. pole. Čím sa líší hašovacia tabulka - ktoré operácie neposkytuje?)
- prečo halda, zásobník a fronta nepatria medzi reprezentácie a sú pre špeciálne účely?
- aká je štruktúra položky v BVS, aké sú možné definície vyváženosti (pre naše účely: čokoľvek, čo zaručí maximálnu logaritmickú hĺbku
- aká je (najhoršia) zložitosť operácií pri jednotlivých reprezentáciách, ako sa to implementuje (s dorazom na BVS), prípadne či je nutné/možné použiť nejaký programátorský trik pre efektívnu implementáciu
– a v rámci toho vysvetlené pilné a líné datové štruktury (napr. pri vypúšťaní z usp. poľa)
– delete z poľa v O(1)

11 Najnevyváženejší strom.
a) Popíšte konštrukciu najnevyváženejšieho stromu (AVL, červeno-čierneho), tj. s najmenším počtom listov pre danu hlbku.
b) Dokážte (indukciou), že pre počet vrchlov tohoto stromu (pre AVL) platí $p_{n+2} = f_n - 1$ (pre poč. podmienky $f_0 = f_1 = 1$ a $p_0 = 1, p_1 = 2$). Tj. počet vrcholov rastie úmerne k Fib. číslam.
c) (rozširujúci) Dokážte indukciou, že pre Fib. čísla platí (správna varianta vzťahu): $f_n = ...(\frac{1+\sqrt{5}}{2})^n + (\frac{1-\sqrt{5}}{2})^n$

12 Operácie na AVL-stromu.
a) Do najnevyváženejšieho AVL stromu s hĺbkou 4, usporiadaného tak, že dlhšie vetvy smerujú doprostred, pridávajte nové mediány, kým sa nerotuje koreň. a2) Koľko prvkov sa pridá? b) Potom pridané prvky v rovnakom poradí vypustite (FIFO).
Rotácie: insert, jedn. ((1[inc]A[-1]2)B[-1]3) $\rightarrow$ (1A[0](2B[0]3)), insert, dvoj. ((1A[0](2[inc]B[0]3[inc]))C[-1]4) $\rightarrow$ ((1A2)B[0](3C4)), delete, jedn. ((1A[-1/0]2)B[-1]3[del]) $\rightarrow$ (1A[0/+1](2B[0/-1]3)), delete, dvoj. ((1A[+1](2B[x]3))C[-1]4[del]) $\rightarrow$ ((1A2)B(3C4))
*Pri vypúšťaní nastala situácia, keď sa znižuje podstrom vpravo a podtečie. Ak je podstrom vľavo vyvážený, *musíte* použiť jednoduchú rotáciu. Nájdite protipríklad, kedy dvojitá nefunguje.*

13 Implementácia rotácií
Rotácie sú symetrické, ale do jednoho kódu by sa neskladali ľahko jednoho.
Implementácia jednoduchej rotácie (koncepčne jednoduchá, ale neoptimálna): oddelím a samostatne ošetrím okrajové prípady (neexistuje rodič, neexistuje prostredný podstrom), zapamätám si (najviac 4 menené) objekty a

prepíšem im (správne) pointry. (Vlastne paralelné priradenie.)
Optimalizácie? Odlišnosti, ak nemám ukazatele na rodičov?

14 R-B stromy
a) Popíšte konštrukciu R-B stromu s najmenším (Rmin(n)) a najvačším (Rmax(n)) počtom vrcholov pre danú hĺbku n.
b) Pre čiernu hĺbku 4 skonštruujte strom ((Rmax A Rmin) B Rmin). Za listy považujeme, podľa konvencie, prazdne (externe uzly), tj. nil-pointry.
c) Pridávajte prvky pred vrchol, ktorý je v najnižšej reálnej vrstve tretí zľava, kým nebude zmenený koreň.
d) Pridané prvky vypustite, od posledného pridaného (LIFO)
Úpravy: insert: ((1Ar(2Br!3))Cb(4Dr5)) $\rightarrow$ ((1Ab(2Br3))Cr!(4Db5)) a symetria A/B, ((1Ar(2Br!3))Cb4b) $\rightarrow^2$ ((1Ar2)Bb(3Cr4) a symetria, delete: 1) pom. ((1Abb!2)Bb((3Cb4)Dr(5Eb6))) $\rightarrow$ (((1Abb!2)Br(3Cb4))Db(5Eb6)), 2) ((1Abb!2)Bx((3Cb4)Db(5Eb6))) $\rightarrow$ ((1Ab2)Bxb!((3Cb4)Dr(5Eb6))) ok/up, 3) ((1Abb!2)Bx((3Cr4)Db(5Eb6))) $\rightarrow$ ((1Abb!2)Bx(3Cb(4Dr(5Eb6)))) sub(4), 4) ((1Abb!2)Bx((3Cy4)Db(5Er6))) $\rightarrow$ (((1Ab2)Bb(3Cy4))Dx(5Eb6))) end

15 Varianty B-stromov
Popíšte operácie pre varianty B-stromov, a ich výhody a nevýhody:
a) Redundantné (hodnoty len v listoch, tj. externá repr.) vs. neredundantné (tj. interná repr.)
b) 2-3 stromy
c) Dychtivé vs. lenivé štiepenie
d) B-stromy, (prelievanie z/do susedov), skupinové štiepenie (2 na 3, príp. 3 na 4): naplnenie aspoň 2/3.*
e) B+ stromy, vodorovne previazané - v listoch (varianty)
f) vzťah B stromov, tj. 2-4 stromov a červeno-čiernych stromov

16 Implementácia rotácií *Napíšte pseudokód a) pre jednoduchú LR rotáciu b) pre dvojitú LR rotáciu, ak sú stromy i) jednoducho spojené, ii) obojsmerne spojené. Pre obojsmerne previazane stromy navrhnite updatovacie procedury tak, aby sa smerníky na rodiča nastavovali "automaticky" (a nezabúdali ste ich zmeniť).* V jazykoch s pattern matching sa dá napísať transformácia priamo (v príslušnej syntaxi): ((1A(2B3))C4) *to* ((1A2)B(3C4)).

3 Haldy

17 Davkové budovanie haldy *Pre binárnu (heapsortovú) a binomiálnu haldu ukážte, že dávkovo (pri znalosti n prvkov dopredu) dokážete haldu vybudovať v čase $O(n)$.*
pozn. nejde to zvrchu pre binárnu haldu: $\sum_{i=1}^{n} \log(i) = \Theta(n \log n)$

4 Hašovanie

18 Na zamyslenie
a) Je vhodné kombinovať hašovanie s vyhľadávacími stromami, ak sú "retiazky" príliš dlhé?
b) Porovnajte spotrebu pamäti pri hašovaní s retiazkami a pri otvorenej adresácii.
c) Čo je to primárne klastrovanie (pri jednej z metód otvorenej adresácie)? Prečo je to problém?
d) Ako je možné realizovať delete pri otvorenej adresácii? Popíšte chovanie pseudodelete pri jednotlivých operáciách na haš. tabuľke. (member, insert, delete)
e) Je možné pri lineárnej adresácii realizovať skutočné delete. Ako? Popíšte správny algoritmus.
f) Ako sa líši zložitosť Delete v najhoršom prípade pri lineárnej adresácii pri prechádzaní prvkov spredu a zozadu?
g) Porovnajte z hľadiska klastrovania a lokality prístupu krok 1 a krok i pri lineárnej otvorenej adresácii.
h) ! Má hašovanie nejakú nevýhodu oproti vyhľadávacím stromom? Čo neumožňuje (efektívne)?
i) Ako sa odstránia pseudovypustené prvky?
j) Prečo je výhodnejšie fyzické delete než pseudodelete? Kde (všade) sa prejaví penalizácia?
k) Má zmysel tabuľka s viacerými prvkami v jednej položke? (:hry, bloky)
l) Prečo je nepríliš vhodné pre hašovanie použiť súčet postupnosti (bez násobenia konštantami)
m) Má zmysel hašovanie bez kolízií? Aplikácie? (bude vyššie: Ako vytvoriť bezkolíznu haš. funkciu? - nie nutne minimálnu.)

19 Oblasť pretečenia.
a) Navrhnite a popíšte riešenie kolízií pomocou oblasti pretečenia. Popíšte jednotlivé operácie. Aká je zložitosť operácií?
b) Motivácia: otvorená adresácia so zaručeným časom pridania.
c) Je možné realizovať delete (a ako)?

20 Hašovanie pre Join v databázach
a) Máte dve množiny prvkov D1, resp. D2, s (nejednoznačnými) kľúčami. Ako nájsť všetky dvojice dát

$d1 \in D1$ a $d2 \in D2$, t.ž. $key(d1) = key(d2)$.
b) Určite zložitosť jednotlivých prístupov. Okrem hašovania je použiteľné neusporiadané a usporiadané prehľadávanie. (usporiadané, ak máme lineárne usporiadanie).

21 Dynamizácia

a) Ak sa tabulka zaplní (do kritickeho faktoru α), potom zväčšíme tabuľku (x-krát, napr. 2x) a prehašujeme. Dokážte, že (aj) rastúce tabuľky majú amortizovanú zložitosť vkladania konštantnú (bez ohľadu na počet prehašovaní). Spočítajte amortizačnú konštantu pre rastúce tabuľky, ak vloženie 1 prvku do 1 tabuľky má cenu 1 a tabuľka sa zväčšuje na dvojnásobok.
b) počítanie dvomi sposobmi a vysvetlenie rozdielu: 1) spočítanie celkovej ceny pridávaní, 2) inkrementálne pridávanie
c) Navrhnite sposob, ako zmenšovať tabuľku, aby čas operácií ostal konštantný a tabuľka nebola "příliš" prázdna. Uvažujte aj striedanie operácií.
c2) Ako sa bude sa líšiť implementácia pri delete a pseudodelete?
d) Navrhnite sposob, ako prehašovanie vykonať v reálnom čase. (trade-off: aj za cenu zvýšenej pamäti (a prípadne vačšej multiplikatívnej konštanty pre čas))
pozn.: dynamizácia datových štruktúr sa dá použiť nielen u hašovacích tabuliek

22 Univerzálne hašovanie

a) Metoda: rozdelenie hodnoty na kúsky pevnej veľkosti (s hodnotami do m) a spočítanie lineárnej kombinácie s náhodnými koeficientami ($0..m-1$), ktoré určujú zvolenú funkciu. (m je velkosť tabuľky)
b) Rozbor, ze kľúče kolidujú pre $1/m$ funkcií, tj. lin. kombinácií. (Hm - nutná algebra)
c) Je možné použiť metodu pre kúsky kľúča veľkosti 1 bit? Zdovodnite. Popíšte.
d) Q: je možné (resp. vhodné) kúsky kľúča použiť opakovane?
e) Ako sa bude metoda chovať, ak máme len 2 kúsky kľúča? (s hodnotami v rozsahu $0..m-1$)

23 Poznámky

a) Ako previesť hodnotu (reťazec, n-ticu) na číslo? (pre hašovanie pomocou modulo)
b) Kolizie budú vždy, je vhodné vedieť, pri akom naplnení sa začnú nepriaznivo prejavovať (pre prislusne metody riesenia kolizii). Optimálnu hodnotu α pre prehašovanie je možné spočítať (alebo zmerať).
c) hašovanie v kryptografii - iné požiadavky, iné funkcie: CRC (Cyclic Redundancy Code), MD5 (Message Digest); iné techniky (salt - osolenie); systém si pamätá (dlhý) hash hesla
d) Zobristovo hašovanie - v hrách, pre (typicky boolovské, alebo obecné) rysy, pomocou XOR
e) Tvorba haš. funkcií: pomocou + alebo Xor (tj. binárneho plus bez pretečenia) (tj. nevyužíva väčšie vzory).
f) delete z hľadiska prvku poľa: ktoré kľúče mohli dojsť až sem (pri danom stave tabuľky).
g) Prehašovanie je dávková operácia (tj. periodická rekonštrukcia datovej štruktúry).

5 Rozdel a panuj

24 Substitucna metoda *ZAPOCET 11/12: dva podpriklady ! h), l)*

Asymptoticky (presne) odhadnite:
a) $T(n) = T(\sqrt{n}) + \log n$
(b) $T(n) = 2.T(\lfloor n/2 \rfloor) + 17$ *-nevhodne*
(c) $T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + 17$ *-nevhodne*
d) $T(n) = T(n/3) + T(2n/3) + \Theta(n)$
d2) $T(n) = 3T(n/2) + 4T(n/4) + \Theta(n^2)$, *! volte vhodný tvar $T(n)$;*
e) $T(n) = T(n/3) + T(n/2) + \Theta(n)$
e2) $T(n) = T(2n/3) + T(n/2) + \Theta(n^2)$
f) $T(n) = 3T(n/2) + 4T(n/4) + O(n^{1.5})$
f2) $T(n) = 3T(n/2) + 4T(n/4) + O(n \log n)$
g) $T(n) = 4T(n/3) + \Theta(n^1)$ *-(tesny) odhad zhora*
h) $T(n) = T(3n/5) + T(4n/5) + O(n)$, *dok.* $T(n) = O(n^2)$
i) $T(n) = T(n/4) + T(3n/4) + \log_4(n)$
j) $T(n) = 5T(n/3) + \log(n)$
k) $T(n) = 2T(n/2) + n \log(n)$
l) $T(n) = T(n/7) + T(5n/7) + O(n)$, *medián, na sedminy*
l2) $T(n) = T(n/3) + T(2n/3) + O(n)$, *medián, na třetiny*

Pri (tesných) odhadoch je (niekedy) vhodné mať dvojzložkový odhad, napr. $O(n^2)$ pomocou $cn^2 - d.fce\_rezie$. Prvý člen vyjde z rekurzie presne a druhý zachytí réžiu výpočtov.
Obecnejší než Master theorem je Akra-Bazzi veta/metóda.

25 Nasobenie dlhych cisel

a) Lepsie nez $O(n^2)$.
a2) varianta: odčítanie vnútri, sčítanie nakoniec. Može pretiecť (o carry), pri práci so záp. číslami.
b) Osetrenie zvlastnych pripadov, pomocou dodatocnych operacii.
c) Analogicka idea: nasobenie komplexnych cisel $a + bi$ a $c + di$ na 3 realne nasobenia.
d) pocitanie pomocou odcitania, pretecenie aj tak hrozi

26 Dimenzia fraktalov

a) Pri konstrukcii (pravidelnych) fraktalov sa pouziva rekurzia. Dimenzia sa da spocitat pomocou Master teoremu (alebo subst. metody)

27 Strasenov alg.
a) Je (ne)pouzitelny na boolovske matice (tj. obsahujuce len 0-1)? (Tazke/tvorive).
b) Upravte, aby bol. (S optimalizaciou:) Ako zabranite pocitaniu s velkymi cislami?

28 Mediány
Hledání k-tého z n prvků (a mediánu)
– klíčový fakt pro substituční metodu: $1/5 + 7/10 < 1$
– při dělení na trojice lineární algoritmus nevyjde: $1/3 + 4/6 = 1$
- vypouštíme prvky: počet vypuštěných (po hledání mediánu): $n.3/10$
- heuristika: pokud je k blízko začátku (nebo konce), lze zvolit pivota asymetricky, např. $k/3 + 1$ *(resp. horní celá část), abychom vyloučili co nejvíc (větších) prvků a přitom měli zaručeno, že pivot je větší než k.*
Zůstává $n.2/5 + 3.k/3$ *prvků*
b) Napište rekurzivní rovnici pro alg. hledání mediánu při dělení na trojice, (pětice), sedmice.

6 Triedenie

29 Rozhodovacie stromy
a) Čo obsahujú rozhodovacie stromy vo vnútorných vrcholoch, čo v listoch. Ako sú označené hrany?
b) Može byť rovnaká permutácia v strome viackrát?
c) Čomu v strome odpovedá jeden konkrétny výpočet triediaceho algoritmu pre nejaké vstupné datá?

30 Stavba rozhodovacich stromov
a) pre bublesort, $n = 4$*;*
b) kedy nastava situacia, ze vnutorny list ma len 1 *syna?*
c) navrhnite optimalizacie (bubblesortu), ktore dovolia programu orezavat strom
d) pre mergesort; popiste vhodnu strukturu pre reprezentaciu (neni vzdy vsetko dane)

31 Utriedte linearnym algoritmom
112, 311, 412, 232, 322, 144, 132, 231; (zapis do stlpca)
a) V akom poradi sa zapisuju cisla do prvkov vystupneho pola?
b) Je tato technika pouzitelna na bitove zapisy cisel. Popiste vyhody a nevyhody (z hladiska operacii, z hladiska HW).
c) Pri obvyklom vypocte zlozitosti klasickych a linearnych triediacich algoritmov sa nepouzivaju porovnatelne operacie. Vysvetlite problem a odstante ho.
d) Dokazete linearne triedit realne cisla? A (obvykle) pocitacove reprezentacie realnych cisel?

32 Stabilné triedenie *Ako zaistíte pre ľubovoľný triediací algoritmus (obecne nestabilný), že bude triediť stabilne (nie nutne na mieste). Idea: poradie ako druhotný kľúč. (Použitie napr. v skriptovacích jazykoch, Haskelli ...)*

33 Quicksort *, varianty výberu pivota*
Quicksort ako schema algoritmov podla strategie výberu pivota
Navrhnite niekolko variánt výberu pivota a popíšte ich vlastnosti.
Aplikujte výber na nerandomizovaný a randomizovaný quicksort.
Ako musím upravit QS, ak pripúšťam ako pivot aj prvok mimo vstup

34 CountingSort
a) (DC v prednáškach): navrhnut variantu algoritmu, ktorá bude stabilná, ak sa v 3. prechode prechádza vstupné pole spredu (stream z kompresie al. serializácie, mag. páska).
b) explicitne formulovat invarianty pre obsah pola countrov C po 2. prechode, v povodnom algoritme a podla a)

7 Grafy

35 Generické prechádzanie
a) Formulujte generický alg. prehľadávania grafu.
b) Čo je stratégia. Aké stratégie výberu vrcholu odpovedajú prehľ. do hĺbky, do šírky, best-first (odhad vzd. k cieľu).
c) príklad best-first: cesta robota, s prekážkami.
d) vyplňovanie plochy farbou (zoznam neprehľadaných, "optimalizácia posledného volania"). (Prehľadávanie skoro lineárneho grafu.)
e) Aké operácie potrebujeme po datovej štruktúre otvorených vrcholov. (Naše prehľadávanie sa líši od prehľadávania pomocou expanzie v UI.)
f) Ako použiť prioritnú frontu?
g) Ako rekonštruujeme (jednu) cestu do daného vrcholu? Ako rekonštruujeme cestu len z pomocných informácií, napr. čísla vrstvy alebo poradia otvorenia (alebo uzavretia) vrcholov? Bude cesta odpovedať ceste použitej pri prehľadávaní, resp. bude cesta najkratšia?

h) Charakterizujete biele, šedé, čierne vrcholy; otvorené a uzavreté.
i) Popíšte rozdiel medzi expanziou vrcholu naraz a prechádzaním jeho hrán postupne. Čo je hranica a čo je vnútro prehľadanej oblasti?
j) Popíšte abstraktné a konkrétne datové štruktúry pre reprezentáciu grafov. (aj matica incidencie VxE)

36 Prehľ. do šírky - optimalizácie.
a) Pre neorientovaný graf, koľko (najmenej) vrstiev potrebujeme pri prehľadávaní do šírky, ak chceme šetriť pamäť a bezpečne vypúšťať už spracované "čierne" vrcholy. Pritom sa nesmie stať, že by sme vrchol prehľadávali opakovane. Ako výstup stačí číslo vrstvy cieľového vrcholu, nepožadujeme rekoštrukciu cesty. (Aplikácia: určí sa dĺžka cesty ako predvýpočet pre generovanie vlastností/požiadaviek na cestu danej (pevnej) dĺžky.)
b) Ako môžeme chovanie ešte zlepšiť, ak nepracujeme s celými vrstvami naraz? (Pointer na otca v každom vrchole alebo dealokačná "inštrukcia" za celú (neprázdnu) skupinu.)
c) Aké (dve) datové štruktúry je možné použiť? Popíšte výhody a nevýhody.
d) Ako je možné skombinovať výhody? (malá réžia pamäti, dynamická veľkosť)
e) Aká je potrebná dĺžka fronty? Aké grafy majú majú frontu?

37 Prehľ. do hĺbky.
a) Ako prebieha prehľadávanie úplného grafu. Ako hlboká bude rekurzia?
b) Ako prebieha prehľadávanie úplného bipartitného grafu? Popíšte pre rovnaké a odlišné veľkosti dvoch častí.
c) Zopakujte druhy hrán pri prehľadávaní do hĺbky. Ktoré druhy možu vzniknúť pri prehľadávaní neorientovaného grafu? Charakterizujete druhy hrán.
(d) Je možné použiť postupné vypúšťanie čiernych vrcholov pri prehľadávaní do hĺbky (a obecne pri generickom prehľadávaní)? Pri variante s expanziou. - Idea: počítanie šedých susedov pre čierny vrchol a uvolnenie, až je počet 0)
(e) Ako použiť minulú techniku pre orientované grafy? Aké akcie budú priradené jednotlivým typom hrán? - výpočet riadený udalosťami)

38 Linearizácia pamäte.
a) Ako zlinearizovať (vypísať do súboru) stav operačnej pamäte s pointrami (na objekty) tak, aby ste boli schopní zrekonštruovať stav pamäte (obecne na iných adresách). Predpokladajte, že sú dané vstupné body, ktoré vedú zvonku (alebo jeden vstupný bod). Nedostupné objekty sa zahodia, t.j. je to možné použiť ako garbage collector. Aké datové štruktúry použijete?
b) Typické riešenie vyžaduje dva prechody štruktúry pri vypisovaní alebo načítaní. (Ak mám vzájomne prepojené objekty alebo cyklus.) Čo sa predáva medzi fázami? Navrhnite algoritmus tak, aby mu stačil jeden prechod pri vypisovaní a jeden pri čítaní. (vhodné pre sieťové aplikácie)

39 Topologicke usporiadanie *ZAPOCET 05/06: c2,c3,d)*
a) Topologicky usporiadajte graf 1-7,6,4, 2-1,5,7, 3-4,5, 4-., 5-4,6,7, 6-4, 7-6.
b) Daju sa vsetky mozne topologicke usporiadania ziskat pomocou DFS? Zdovodnite.
c) Klasifikacia hran z prikladu a) na stromove, spatne, dopredne a priecne; uvedte pocty.
c2) Formulujte podmienky na druhy hran v pojmoch biely, sedy, cierny vrchol (tj. nenavstiveny, navstiveny, hotovy)
c3) a pomocou cislovania udalosti podla e).
d) Platia pre vstupne a vystupne stupne hran jednotlivych druhov nejake omedzenia?
e) Vztah k udalostiam riadenemu programovaniu; cislovanie udalosti - vstupov a (poslednych) opusteni vrcholov.
f) Ako upravite algoritmus, aby ste mohli zabudat uz nepotrebne vrcholy? (ktore uz nebudu v buducnosti (opat) navstivene)

40 Aplikacie top.usp.
a) Optimalizacia podvyrazov a generovanie kodu.
b) Vypocet fib. cisel, graf fib(5).

41 Suvislosti prechadzania grafov
a) Postupne generovanie velkych grafov (implicitna reprezentacia, generovanie "on demand", "on-the-fly"),
b) Prevod prechadzania na udalostami riadeny algoritmus, jeden z modelov spracovania XML.
c) Hasovanie vrcholov, znovuprechadzanie casti.
d) Lokalne pamatanie smeru prichodu (pamat $O(1)$) - pre stromy (hier).

42 Polosuvisly graf *(tiez slabo suvisly g.) b) alebo c)*
Graf $G = (V, E)$ *je* polosouvisly, *ak* $\forall u, v \in V \exists$ cesta z u do v alebo cesta z v do u. *Pre nasledujuce algoritmy odhadnite maximalnu zlozitost.*
a) Zistite, ci je graf polosouvisly; pomocou matice dosazitelnosti.
b) Pomocou opakovaneho DFS. (pro a proti)

c) Pomocou DFS :-), v case $O(cas(DFS))$. Vznikne kondenzovany graf (SSK), akeho musi byt tvaru (ake su zakazane podgrafy (minory)?).

43 Priemer grafu *ZAPOCET 11/12 c)*
Priemer grafu je definovaný ako najdlhšia z najkratších ciest medzi ľub. dvomi vrcholmi.
a) Ako v grafe s jednotkovo ohodnotenými hranami pomocou prehľadávanie do šírky spočítate jeho priemer? Aká je zložitosť algoritmu?
b) Navrhnite algoritmus pre spočítanie priemeru grafu s nezáporne ohodnotenými hranami. Použijte Dijkstrov alg. Zložitosť?
c) Navrhnite algoritmus pre spočítanie priemeru grafu s nezáporne ohodnotenými hranami. Aká vyjde zložitosť, ak použijete ako podprocedúry (všetky) známe alg. pre výpočet min. cesty?
d) Možné optimalizácie? (pre jednotlivé alg., napr. kritickú cestu).

44 Súvislosť hustého grafu *Navrhnite heuristický algoritmus, ktorý pre hustý (neorientovaný) graf rýchlo overí, že je spojitý. Idea: nemusím expandovať všetky vrcholy, stačí ak sú dosažitelné. Vhodné (heuristické) usporiadanie prehľadávaných hrán. (Podľa akého kritéria možem hrany usporiadať staticky (predpočítaním) a dynamicky?)*

45 Reprezentanti SSK *pomocou DFS. ZAPOCET 11/12 a), a2)*
a) Navrhnite algoritmus, ktory pri jednom prechode DFS urci pocet SSK-komponent a zaroven urci reprezentantov tychto komponent. Reprezentanti budu prve navstivene vrcholy (s najmensim otvaracim cislom) v komponente. Navod: vylucte ostatne vrcholy.
a2) Urcite zlozitost algoritmu.
b) Optimalizujete spracovanie 1 vrcholu. Urcite zlozitost.
c) Pamatajte si vhodne data u vrcholu, aby ste pri opustani, tj. uzatvarani, vrcholu dokazali rozhodnut, ci je reprezentantom komponenty. Spracovanie vrcholov a hran je v konstantnom case a backup hodnot z navstivenych vrcholov umoznuje udrziadat vhodne data.

46 Kondenzace grafu *pomocou DFS. Navrhnite algoritmus, ktory pri jednom prechode DFS vyrobí kondenzaci daného grafu G. (tj. spojí vrcholy Silne Súvislých Komponent do jednoho vrcholu.)*

47 Tranzitívny uzáver grafu *pomocou DFS. Navrhnite algoritmus, ktorý pre daný orientovaný graf spočíta jeho tranzitívny uzáver, tj. doplní všetky hrany (x,y), ak z x vedie orientovaná cesta do y.*
a) pre acyklický graf
b) pre obecný graf (dodatočné doplňovanie vrcholov)
c) optimalizujte, aby ste vylúčili čo najviac vrcholov, ktoré nemožu prispieť, pretože ich následníci už sú zahrnutí. (Použite časy otvorenia a uzavretia.)
Varianta SSK a kondenzace grafu.

48 Vyfarbovanie plochy *Navrhnite heuristický algoritmus, ktorý v 2D obrázku prefarbí spojitú plochu (nie nutne konvexnú) z jednej farby na inú. Body plochy sú susedné, ak sú vedľa seba v riadku alebo stĺpci, tj. nie uhlopriečne. Heuristicky využite obvyklé vlastnosti obrázkov pre malú spotrebu pamäti.*
b) Navrhnite iný sposob spravovania vrcholov na zásobníku otvorených vrcholov, aby ste dosiahli malú spotrebu pamäti na typických obrázkoch s veľkými plochami. Idea: LCO – Last Call Optimization, v riadkovej štruktúre.
Expanzia vrcholu vs. postupné prehľadávanie hrán. Vyfarbovaná plocha (spolu so svojimi okrajmi) zadáva graf susednosti implicitne. Graf odvodený z obrázku nemusí odpovedať povodnej definícii susednosti, ale može byť odvodený.

49 Expanzia vrcholu
a) 2 sposoby, do prednasok, k ulohe farbenia plochy
b) oznacenie vrcholov: biele, šedé, čierne
ad) implicitne zadaný graf, datovou štruktúrou

50 Hrany kruznic, mosty, artikulacie *DC*
a) Navrhnite algoritmus, ktory urci v neorientovanom grafe vsetky hrany, ktore lezia na kruzniciach. Odhadnite jeho casovu zlozitost.
a2) Pomocou "leniveho vyhodnocovania" vylepsite algoritmus tak, aby sa prislusnost hran ku kruzniciam nepocitala opakovane. Priamociary alg. $O(n.m)$, mierne zlepseny $O(n.n)$, idealny (;-) $O(n+m)$ – viz (b).
b) Upravte algoritmus, aby to zvladnul v jednom prechode DFS (a na hranach a vrcholoch travil len konstantny cas (niekolkokrat)).
c) ?? Upravte algoritmus na identifikaciu tych orientovanych hran v orientovanych grafoch, ktore su castou orientovanych cyklov.
analogicky: mosty, artikulacie (ťažké) a dvojsúvislé komponenty; vztak k: "lenive vyhodnocovanie"

51 SSK pre konkrétny graf *ZAPOCET 07/08 1a-d*
a) Pre graf(y) nájdite klasickým algoritmom SSK. Pri možnosti výberu vrcholu alebo hrany postupujte podľa abecedy.
b) Napíšte nájdené poradie vrcholov po prvej fáze.

b2) Napíšte časy otvorenia a uzavretia vrcholov v prvej fáze.

c) Vyrobte kondenzáciu grafu

d) Spočítajte druhy hrán (stromové, ...)

1) (graf 2x4): A-B,E; B-C,D,F; C-B,D,G; D-C; E-F; F-E,G; G-H; H-nic. 13 hrán

1.1) stejný graf, začnite v E.

2) (graf 2x3): A-B,D; B-C,E; C-B,F; D-E; E-D,F; F-nic

3) písomka 2008

DFS pomocou expanzie vrcholu je pre SSK nevhodný; protipríklad: úplný DAG na 3 vrcholoch, A-B-C, A exp. B hore, C pod ním.

52 Serializacia a deserializacia grafu *alias I/O pointrovej struktury*
Beziaci program obsahuje v pamati "objekty" a pointry na ne (okrem nezaujimavych statickych datovych struktur). Program je pozastaveny a dynamicke data sa nemania.
a) Navrhnite metodu, reprezentaciu dat a algoritmy, ktore vam dovolia vypisat aktualny stav pamatovych datovych struktur a nasledne ich z tohoto zapisu spravne zrekonstruovat do izomorfnej podoby. Objekty su obecne roznych druhov, tj. rozne velke. Obecne mozete pri vystupe a/alebo vstupe pouzit viacnasobny prechod grafu odkazov.
b) Dokazete vystup a sucasne *vstup spravit v jednom prechode?*
Hakersky kutik: Pri vystupe sa nemusi robit konverzia adries (a vyhladavanie/mapovanie), mozeme pouzivat globalne adresy!

7.1 Min. kostra a cesty

53 Min. kostra
Pre graf na vrcholoch A,B,C,D,E,F,G,H s cenami hran AB 5, AC 11, AD 6, AF 4, BD 3, BE 3, BC 8, BG 8, CE 2, CH 1, DF 9, DG 7, EG 9, EH 3, FG 10, FH 13, GH 12 spocitajte min. kostru:
a) Hladovy algoritmom.
b) Rozsirovanim z vrcholu A
c) Rozsirovanim z vrcholu H
d) Vadia algoritmom zaporne hrany a zaporne cykly? Daju sa odstranit v preprocesingu?

54 Hierarchicke klastrovanie *ad minimalni kostra*
hm, nedokončene...

Hladovo spajam dva najblizsie vrcholy a nahradim ich centrom klustru

- moznosti pre novy vrchol: 1) stred, 2) vazeny stred, 3) vzdialenostny stred

- impl: "halda" komponent, ak je novy vrchol ...

55 Dijkstrov alg.
a) Pre graf z minuleho prikladu spocitajte min. cestu z A do H;
b) z E do vsetkych ostatnych (ina koncova podmienka)
c) Ktore vrcholy su z C do vzdialenosti 15 ? Dijkstra je prehladavanie "best-first" s ocenenymi hranami (vylepšenie prehľ. "do sirky"); v umelej inteligencii sa D.alg. pouziva s heuristikou pre odhad vzdialenosti do ciela (cielovej mnoziny), ktora umozni zmensit pocet otvaranych vrcholov

56 Dijkstrov alg. - varianty *a) Možná idea, ako použiť Dijkstrov alg. na graf so záporne ohodnotenými hranami, je pričítať k cene každej hrany dostatočne veľké kladné číslo. Ukážte, že tento algoritmus nedáva správne výsledky, tj. minimálna cesta v upravenom grafe nie je obecne zhodná s minimálnou cestou v povodnom grafe (a nie je ju možné jednoducho zrekonštuovať).*
b) Ako je možne "opatchovať" minulý algoritmus, aby dával správne výsledky (Vo vrchole uložíme obecnejšie informácie a vo väčšej pamäti)
c) Iný sposob úpravy D.a. je dovoliť prepočítavanie ohodnotenia vrcholov, pri zmene. Odhadnite zložitost (zdola, najhorší prípad) a prípadne ukážte, že počet spracovaných vrcholov je až exponenciálny.

57 Alg. Belman-Ford, optimalizácie
a) Navrhnite heuristickú optimalizáciu, ktorá umožní neprechádzať v každom cykle všetky hrany pri zachovaní korektnosti. Idea: aktívne vrcholy. (Zlepší sa týmto sposobom zložitosť v najhoršom prípade?)
b) Je možné ukončiť prechádzanie hlavného cyklu skor než po n-1 prechodoch? Popíšte postačujúcu podmienku.
c) Je možné pomocou aktívnychn vrcholov zistiť pred ukončením algoritmu, že sa v grafe nachádza záporný cyklus. Popíšte podmienku.

58 Datamining *Hladanie supportu*
Hm. Vrcholy ohodnote formulami, hrany odpovedaju operatorom pre zvacsovanie formuli. Mame syntakticke porovnavanie formuli a prah p. Ak formula f vybera z pevnej "databazy" a relevantnych a b nerelevantnych objektov, potom support f je a a presnost f je $a/(a+b)$.
a) Hladanie najlepsieho ohodnotenia (supportu) pre formule daneho druhu.
b) Hladanie najdlhsich vzorov s nadprahovym supportom. (Dynamicke ukoncenie.)
c) Hladanie najvacsej presnosti pri nadprahovom supporte.

Idea pre túto ulohu pochadza z clanku (vzor citacie): Jan Blaťák, Petr Kuba: Hledání častých vzoru v datech složité struktury, in: Datakon 2003, ed. Luboš Popelínský, Masarykova univerzita, Brno, 2003, pp. 193–203

59 Frekventovane itemsety *ad Dijkstra.*
- Je znama mnozina poloziek. Dalej je dany zoznam podmnozin poloziek, zvanych transakcie. Mame najst vsetky podmnoziny poloziek, ktore sa vyskytuju v mnohych transakciach (Viac nez prah; zoznam n najcastejsich)
- optimisticky odhad
- Dijkstra na implicitnom grafe

60 Dosazitelnost v grafe
a) Vysvetlite sposob pocitania dosazitelnosti (resp. vzdialenosti) pomocou nasobenia boolovskych natic, resp. specialneho nasobenia.
b) Kolko vykonani nasobenia potrebujete na najdenie vsetkych ciest az do dlzky n?
c) Je mozne tento sposob pouzit, ak su v grafe zaporne hrany?
d1) Definujte pojem "najkratsej cesty" tak, aby bol dobre definovany aj v grafe so zapornymi cyklami.
d2) Preco nejde algoritmus z (a) pouzit na (d1) ? (Metaotazka: Preco si mozem dovolit tuto otazku (d2), ak neviem, aku definiciu ste si zvolili?)

61 Najdite min. cesty *ZAPOCET (05/06:a) 07/08: c)*
Pre dane grafy urcite vsetky minimalne cesty, Floyd-Warshallovym algoritmom, v prirodzenom poradi. Znak '-' znamena neexistenciu (orientovanej) hrany:
a) na 4 vrcholoch A,B,C,D: -,3,8,-; -,-,4,-; 2,-,-,6; 6,12,15,- . Pre ZAPOCET vypiste matice vzdialenosti po kazdom prechode hlavneho cyklu. (SebaKontrola: 3 zmeny v prvom cykle).
b) na 4 vrcholoch A,B,C,D: -,3,8,11; 12,-,2,-; -,13,-,4; 1,-,14,-. (na cvicenia!)
c) na 4 vrcholoch A,B,C,D: -,2,6,-; -,-,3,-; 4,7,-,8; 1,5,-,-

62 Špeciálne násobenie matíc *a varianty.*
a) Pre grafy z minulého príkladu spočítajte dĺžku všetkých najkratších ciest pomocou špeciálneho násobenia matíc.
b) Je možné ukončiť násobenie skor, než po logaritmickom počte hlavných cyklov, ak sa hodnoty v matici v poslednom cykle nezmenili?
c) Je možné si pamätať pre rekonštrukciu cesty medziľahlý vrchol, pre ktorý sa našlo minimum? Ako sa v tomto prípade zrekonštruuje cesta? Pozn. Klasické určenie minulého vrcholu na ceste odpovedá postupu, akoby sme vyhodnotili posledný medziľahlý vrchol staticky pri naplňovaní matice a pri čítaní ho zistili v O(1).
d) Zistenie všetkých najkratších ciest (hrán, resp. vrcholov) - viz Kritická cesta.

63 Varianta F.-W.
a) Pre hľadanie (všetkých) min. ciest v grafe s obecne (aj záporne) ohodnotenými hranami, ale bez záporných cyklov je navrhnutá táto varianta F.-W. alg. Nájdeme najmenšiu hranu s cenou -c a jej hodnotu c pričítame ku všetkým hranám. Tým sa stanú hrany nezáporné a možeme použiť klasický F.-W. alg. pre hľadanie min. ciest. Dokážte správnosť alebo nesprávnosť popísaného algoritmu.
b) Je možné hľadať niektorými algoritmami maximálne cesty v grafe? Ktorými, prípadne za akých podmienok.
c) Je možné ukončiť počítanie skor, než po určenom počte hlavných cyklov, ak sa hodnoty v matici v poslednom cykle nezmenili?
d) Pri ručnom počítaní, ak jedna z ciest není definovaná, riadok, resp. stĺpec sa nezmení. Zdovodnite.
e) Ak je na niektorom mieste najmenšie číslo v riadku (alebo stĺpci), musíte toto miesto prepočítať? Tj. može sa toto číslo zmeniť?

64 Kritická cesta v DAG *(PERT) ZAPOCET 09/10 a), b)*
Metodou kritické cesty pro acyklické grafy spočítejte délku maximální (kritické) cesty, určete kritické činnosti a rezervy nekritických činností. Vrcholy grafu jsou $A - H$, ceny hran jsou: $AB10, AC8, AD1, BG9, CG7, CH3, DB4, DC12, EA5, ED6, FD2$.
b) Pri tejto úlohe typicky potrebujeme všetky kritické cesty, nielen jednu. Tj. obvyklý záznam jednoho predchodcu nie je dostatočný. Popíšte inú metódu, ako zistiť ktoré hrany a vrcholy ležia na (nejakej) kritickej ceste.
b2) Upravte túto metódu, aby ste zistili, ktorý vrchol (hrana) je posledný na nejakej kritickej ceste.
c) Popíšte transformáciu (orientovaného) grafu s ohodnotenými vrcholmi na graf s ohodnotenými hranami, pre ktoré sú určené preberané algoritmy.

65 Viterbiho alg. *ZAPOCET 11/12 a), a2), a3)*
Je daný acyklický orientovaný graf s počátečním vrcholem Z a koncovým vrcholem S. Hrany jsou ohodnoceny pravděpodobností přechodu mezi vrcholy. V každém vrcholu kromě Z a S jsou dány pravděpodobnosti vypsání 1 znaku z nějaké pevné abecedy Σ. Pravděpodobnost cesty je součin pravděpodobností na hranách cesty. Pravděpodobnost vypsání určité posloupnosti na dané cestě je určena součinem pravděpodobností vypsání jednotlivých znaků ve vrcholech. (Tj. hrany a vrcholy jsou nezávislé.)

a) (Zjednodušení: Vrstvy) Graf je navíc vrstvený, tj. vrcholy jsou rozděleny do vrstev $V_i, i = 0..k$ a hrany vedou pouze mezi vrstvami V_j a V_{j+1}. Nultá vrstva obsahuje pouze Z a poslední vrstva obsahuje pouze S. V jednom vrcholu se vypisuje 1 znak a pak se pokračuje hranou dál. Propojení mezi vrstvami není v obecnosti úplné.

Navrhněte algoritmus, který k dané posloupnosti P délky $k-1$ najde (jeden) nejpravděpodobnější způsob, jak byla P vygenerována a vydá příslušnou cestu ze Z do S.

a2) Odhadněte složitost algoritmu.

a3) Protože v obecnosti propojení mezi vrstvami není úplné, cesta nemusí existovat. Navrhněte (konzervativní) úpravu, aby nějaká cesta byla vždy vydána.

b) Graf není vrstvený, v jednom vrcholu se emituje 1 znak. Posloupnost musí být vypsána přesně mezi Z a S.

c) Jeden vrchol vypisuje posloupnosti znaků délky aspoň 1.

d) Vrchol může vypsat (s určitou pravděpodobností) prázdnou posloupnost.

d2) Jak lze simulovat vkládání, vypouštění (a záměnu znaků) v posloupnosti? Chceme aby podobné cesty měly (řádově) podobnou pravděpodobnost.

- V obecnosti: průchod DAG je dynamické programování. Aplikace. Určení pravděpodobností z trénovacích dat je jiná (netriviální) úloha.

V: (implicitne generovany graf), hladanie cesty v bludisku s (deterministickymi) nepriatelmi, najkratsej

V: povodi: hranovo oceneny strom s s sutokmi a n pristavmi (na sutoku alebo rieke), so stepenim $0-x$. a) Najst najdlhsiu cestu b) najkratsiu cestu c) cenu medzi urcenymi vrcholmi / vsetky ceny ?? d) zistit, ci existuje cesta danej ceny

metaotazka: Aky je vhodny, resp. najlepsi, tvar vstupnych dat pre dany algoritmus. Tj. zlozitost preprocesingu pocitame samostatne.

66 LU rozklad

*prevod (A=LU/I)-*Lna -1 (U/L -1), (L -1/I)- *L -(I/L)*

… stránka V. Majerecha