Algoritmuselmélet 17. előadás

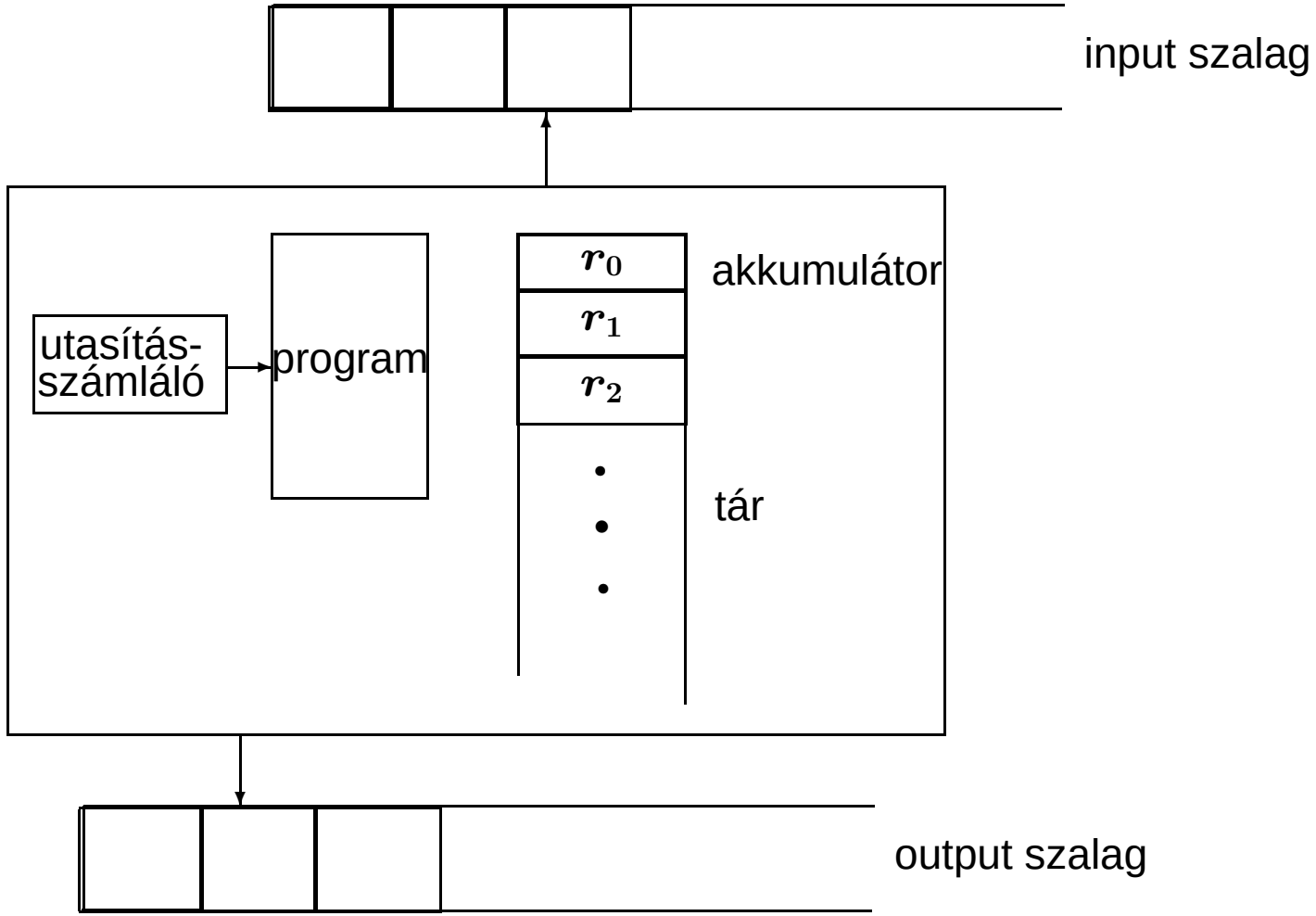
Katona Gyula Y.
Budapesti Műszaki és Gazdaságtudományi Egyetem
Számítástudományi Tsz.
I. B. 137/b
kiskat@cs.bme.hu

2002 Május 6.

A közvetlen elérésű gép (RAM)

Random Access Machine

$\Longrightarrow$ Memória minden cellája egy lépésben elérhető.



Utasítások $\Longrightarrow$

Aritmetika		*Adatmozgatás*		*Vezérlés*	
ADD	*op*	LOAD	*op*	JUMP	*címke*
SUB	*op*	STORE	*op*	JGTZ	*címke*
MULT	*op*	READ	*op*	JZERO	*címke*
DIV	*op*	WRITE	*op*	HALT	

op

$\Longrightarrow = i$: az i számot jelenti

$\Longrightarrow i$: i sorszámú cella tartalma

$\Longrightarrow *i$: az i sorszámú cellában levő sorszámú cella tartalma, pl. $r[0] = -4$ és $r[1] = 0$ $\Longrightarrow$ a $*1$ operandus jelentése -4

Az aritmetikai műveleteknél az első argumentum az akkumulátor, a második az op

$\Longrightarrow$ eredménye $\rightarrow$ akkumulátor

Például ha $r[0] = 17$ és $r[1] = 2$, akkor DIV 1 hatására a $8 = \lfloor 17/2 \rfloor$ érték kerül az akkumulátorba.

Aritmetika		*Adatmozgatás*		*Vezérlés*	
ADD	*op*	LOAD	*op*	JUMP	*címke*
SUB	*op*	STORE	*op*	JGTZ	*címke*
MULT	*op*	READ	*op*	JZERO	*címke*
DIV	*op*	WRITE	*op*	HALT	

A READ beolvassa az input cella tartalmát op-ba és lép
A WRITE kiírja az op-ot az output cellába és lép

A STORE op $\Longrightarrow$ akkumulátor tartalma $\rightarrow$ op
LOAD fordítva.

HALT $\Longrightarrow$ megáll.
JZERO: Ha $r[0] = 0$, akkor ugrik
JGTZ: Ha $r[0] \geq 0$, akkor ugrik.

Költségszámítás

Uniform költség: végrehajtott utasítások száma
Pl. az $f(n) = 3^{2^n}$ függvény kiszámítható $O(n)$ uniform költséggel:
$\Longrightarrow$ legyen $x := 3$, majd n-szer iterálva $x := x^2$.
De a k-adik iterációs lépésben két 2^k-jegyű számot szorzunk össze.
$\Longrightarrow$ Az eredménynek tehát 2^{n+1} jegye lesz. ↯

Logaritmikus költség: egy utasítás költsége a benne szereplő adatok összhossza. Egy program logaritmikus költsége a végrehajtott utasítások költségeinek az összege.

Példa: ADD $*1$ uniform költsége 1. A logaritmikus költsége viszont

$$\text{hossz}(r[0]) + \text{hossz}(r[1]) + \text{hossz}(r[r[1]]) + \text{hossz}(1),$$

ahol $r[i]$ a belső tár i-edik cellájának a tartalmát jelöli.

Ha nincs MULT és DIV akkor a logaritmikus költség valós költség
Legjobb ismert algoritmus $O(n \log n \log\log n)$

Ha tudjuk, hogy a RAM-program végig $\leq l$ hosszú szavakkal dolgozik $\Longrightarrow m$ uniform költség $\rightarrow O(lm)$ logaritmikus

Szimulációk

Tétel. [Turing-gép↔RAM szimulációk]
(1) Tetszőleges M Turing-gép szimulálható $O(T_M(n) \log T_M(n))$ logaritmikus költségű RAM programmal. A szimuláció uniform költsége $O(T_M(n))$.
(2) Egy $t(n)$ logaritmikus költségű, MULT és DIV utasításokat nem tartalmazó RAM-program szimulálható olyan N Turing-géppel, melynek az időigényére $T_N(n) = O(t^2(n))$ teljesül.

Bizonyítás: (1) M egy k-szalagos Turing-gép.
M bemenete $\Longrightarrow$ RAM input szalag
A RAM belső tárának első c celláját munkaterületként használjuk $\Longrightarrow$ M belső állapota, és itt kap helyet az a k cella is, amelyekben a M fejeinek helyzete van
$\Longrightarrow$ *összefésülve* az M szalagjainak tartalmát

A RAM programja $\Longrightarrow$ M átmeneteinek leírása $\Longrightarrow$ megvalósítható **if...then...** utasításokkal (indirekt címzés)

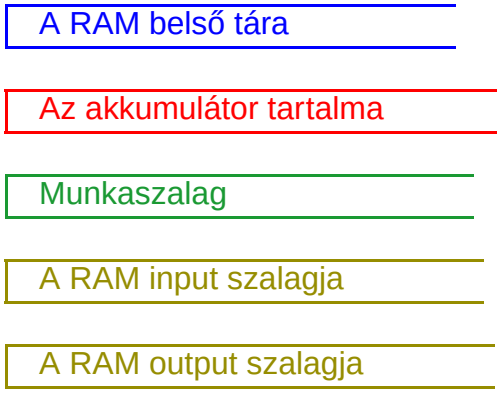
uniform költség: $O(T_M(n))$

logaritmikus költség: az M szalagjelei és belső állapotai (M-től függő) konstans hosszúságúak

a fejek helyét leíró mutatók pedig $O(\log T_M(n))$ bittel ábrázolhatók

$\Longrightarrow O(T_M(n) \log T_M(n))$

(2) A RAM-programot szimuláló N gép szalagjelei $\Longrightarrow \{0, 1, +, -, \#, ü\}$.



Első szalag:

Belső tár olvasása $\surd$ írása (végére) $\surd$
RAM alaputasítások $\Longrightarrow$ N állapotcsoportjai $\Longrightarrow$ ezek között átmenet

Lépésszám: az alaputasítások – a MULT és DIV kivételével – megvalósíthatók úgy, hogy N lépésszáma az utasításban szereplő adatok hosszával plusz az első szalag érdemi részének hosszával arányos legyen.
$\Longrightarrow$ egy lépés szimulációjának a költsége $O(t(n))$
összköltség: $t(n)O(t(n)) = O(t^2(n))$

Ha MULT és DIV is van benne: $O(t^3(n))$

Elágazás és korlátozás

Mit tegyünk ha kiderül, hogy a megoldandó probléma NP-teljes?

Legtöbbször van c^n-es algoritmus, de nem mindegy mekkora c.

Bontsunk esetekre, azt alesetekre, . . . $\Longrightarrow$ fa
Értékeljük az eseteket $\Longrightarrow$ bizonyos irányokban nem kell továbbmenni.
$\Longrightarrow$ (korlátozó heurisztika)

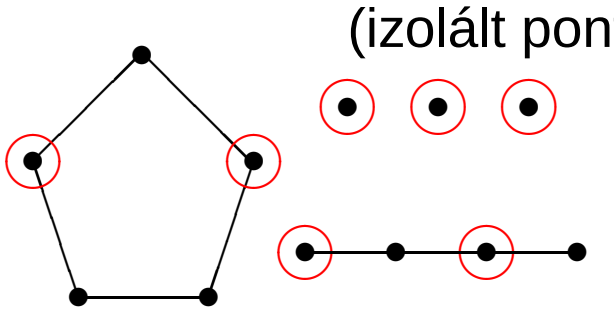
Pl. sakkállások

Feladat: Keressünk maximális méretű független ponthalmazt egy adott G gráfban.

NP-teljes
Minden részhalmazt végignézünk $\Longrightarrow$ $O(2^n)$ lépés

Jobb algoritmus

Észrevétel: *Ha G-ben minden pont foka legfeljebb kettő, akkor a feladat lineáris időben megoldható $\Longrightarrow$ G izolált pontok, utak és körök diszjunkt uniója. $\Longrightarrow$ komponensenként minden „második" pontot bevesszük a halmazba.*



MF(G)

1. Ha G-ben minden pont foka ≤ 2, akkor MF(G) az előbbi eljárás által adott maximális független halmaz, és a munkát befejeztük.
2. Legyen $x \in G$, $fok(x) \geq 3$.
   $S_1 :=$ MF($G \setminus \{x\}$)
   $S_2 := \{x\} \cup$ MF($G \setminus \{x$ és szomszédai$\}$).
3. Legyen S az S_1 és S_2 közül a nagyobb méretű, illetve akármelyik, ha $|S_1| = |S_2|$.
4. MF(G) $:= S$.

Legyen $T(n)$ az MF(G)-n ($|\ V(G)\ | \leq n$) belüli MF hívások maximális száma, beleértve MF(G)-t magát is.

Tétel. *Van olyan c állandó, hogy $T(n) \leq c\gamma^n$, tetszőleges n természetes számra, ahol γ a $\gamma^4 - \gamma^3 - 1 = 0$ egyenlet pozitív gyöke ($\gamma \approx 1,381$).*

Bizonyítás: Legyen $t(n) := T(n) + 1$.
$T(n) \leq T(n-1) + T(n-4) + 1$, ha $n > 4$. $\Longrightarrow$
$t(n) \leq t(n-1) + t(n-4)$, ha $n > 4$.
Indukcióval: $t(n) \leq c\gamma^n$
$n < 5$-re elég nagy c-vel ✓
$\Longrightarrow$ Ezután, ha $n \geq 5$, indukciós feltevésből:

$$\begin{aligned} t(n) &\leq t(n-1) + t(n-4) \leq c\gamma^{n-1} + c\gamma^{n-4} = \\ &= c\gamma^{n-4}(\gamma^3 + 1) = c\gamma^{n-4}\gamma^4 = c\gamma^n. \end{aligned}$$

✓

Összköltség: $O(n^d T(n)) = O(n^d \gamma^n) = O(1,381^n)$.

3-színezés keresése

Feladat: Adott G, keressünk egy 3-színezést.

NP-teljes
Minden lehetséges színezést végignézünk $\Longrightarrow$ $O(3^n)$ lépés

Ötlet: Bizonyos csúcsokat kiszínezünk pirosra, a többiről polinom időben el tudjuk dönteni, hogy kiszínezhetők-e kékkel és sárgával.

Összköltség: $O(2^n n^c)$.

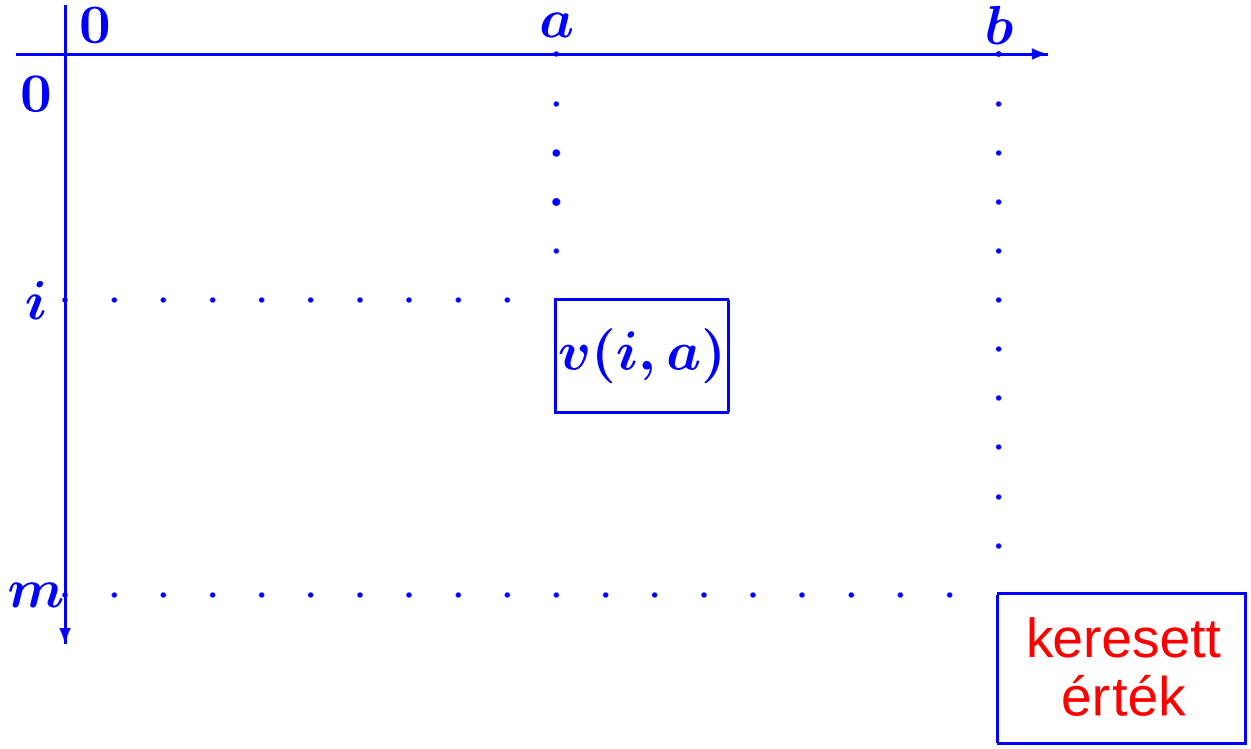
Dinamikus programozás

Táblázat kitöltése soronként, rekurziót használva.
Pascal háromszög, Bellman-Ford, Floyd

A Hátizsák probléma: Adottak az $s_1, \ldots, s_m$ súlyok, a b súlykorlát, a $v_1, \ldots, v_m$ értékek és a k értékkorlát. A kérdés, hogy van-e olyan $I \subseteq \{1, \ldots, m\}$ részhalmaz, melyre teljesül, hogy $\sum_{i \in I} s_i \leq b$ és $\sum_{i \in I} v_i \geq k$.

NP-teljes

Először kisebb problémára oldjuk meg: $v(i,a)$ a maximális elérhető érték az $s_1,\ldots,s_i$ súlyokkal, $v_1,\ldots,v_i$ értékekkel és a súlykorláttal megadott feladatra (mivel a maximális értéket keressük, nincs szükség értékkorlátokra).
Ekkor $v(0,a) = v(i,0) = 0\ \forall a,i$-re
cél $\longrightarrow v(m,b)$



$$v(i,a) = \max\{v(i-1,a); v_i + v(i-1,a-s_i)\}$$

$\Longrightarrow$ Soronként kitölthető $\Longleftarrow$ minden érték két felette levőből számolható.

Összköltség: $O(bL)$ nem polinomiális, mert b-től függ, nem $\log b$-től!

Definíció. *A b egész unáris ábrázolása: $0^b := 0 \cdots 0$ (összesen b darab 0 egymás után).*

Definíció. *Egy feladat egy egész input paramétere apró, ha unárisan számítjuk bele az input hosszába.*

Tétel. *A Hátizsák probléma apró súlykorlát esetén megoldható polinom időben.*