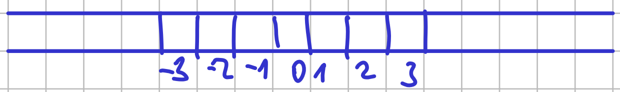
RAM



hodnoty – literál (konstanta) 42
– buňka [42]
– nepřímo adres. buňka [[42]]

konvence: [-1] = A … [-26] = Z

SEM:

goto SEM

Instrukce:
- $X \leftarrow Y$
- $X \leftarrow Y + Z$
- halt
- goto IHS
- if $X = Y$ then instrukce
  - $= \neq < > \leq \geq$

$X \leftarrow 5 + 11$

[-24] ↑ 5; 3, 4 → 5; 17 bitů

Výpočet:
- v paměti dostaneme vstup (zbytek paměti může obsahovat cokoli)
- provádíme instrukce … halt
- v paměti odevzdáme výstup

Cena instrukce:
1. jednotková — ! pozor na velká čísla !
   paradoxy
   Cv: součet $x_1 + \dots + x_n$ v konst. # instrukcí
2. jednotková, ale omezíme čísla šířkou slova W (bitů)
   Potřebujeme $W \geq \log n$ (kvůli čtení vstupu)
   Dovolíme $W \in O(\log n)$
   ↳ $|\text{čísla}| \leq 2^{c \cdot \log n} = n^c$ … polyvelká čísla
   ! omezili jsme prostor polynomem !
3. logaritmická – # bitů čísel, s nimiž pracujeme
   ! nepohodlné !
4. relativní logaritmická: $\left\lceil \dfrac{\#\text{ bitů čísel}}{\log n} \right\rceil$

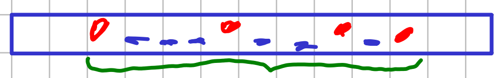
Df: Pro program a vstup x:

Čas běhu programu:
$t(x) :=$ součet cen provedených instrukcí

Prostor běhu:
$s(x) :=$ max. (sup) použitá adresa − min. (inf) použitá adresa + 1

může být i $+\infty$

nebo nekonstantní cenu buňky



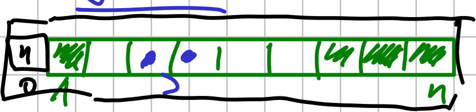
Df: Pro program:

Časová složitost: $T(n) := \max \{ t(x) \mid x \text{ je vstup velikosti } n \}$

prostorová složitost: $S(n)$ … $s(x)$ | …

velikost:
- # čísel
- # bitů
- # vrcholů a # hran
- # znaků řetězce

Algoritmus: Bublinkové třídění



1. Opakuj:
2. $p \leftarrow$ NE
3. Pro $i = 1, \dots, n-1$:
4. Pokud $x_i > x_{i+1}$:
5. $x_i \leftrightarrow x_{i+1}$
6. $p \leftarrow$ ANO
7. Dokud $p =$ ANO.

```
ZNOVU: P ← 0
       I ← 1
DALSI: J ← I+1
       if [I] ≤ [J] then goto OK
       T ← [I]
       [I] ← [J]
       [J] ← T
       P ← 1
OK:    I ← I+1
       if I < [0] then goto DALSI
       if P=1 then goto ZNOVU
       halt
```

4n−1 až 8n−5 (vnější)

4–8 instr. na průchod ↓ 4n−4 až 8n−8 na smyčku

1 až n cyklů ↓ časově $4n$ až $8n^2 - 5n + 1$

nejhorší případ aspoň $4n^2 - n + 1$

Důsledek: $4n^2-n+1 \le T(n) \le 8n^2-5n+1$

$\Omega(n^2)$ → $\Theta(n^2)$ ← $O(n^2)$

$10000n$

$2n^2$

Proč asymptotika?

1. je to snazší
2. konstanty strojově závislé
3. pro velké vstupy rozhoduje asymptotika

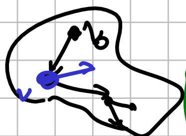
Typicky:
1. asymptoticky nejrychlejší alg.
2. ladíme konstanty

] pozor na malé vstupy

GRAFOVÉ ALGORITMY

- BFS (prohledávání grafu do šířky)
- DFS (… do hloubky)

DFS2(v):



1. Stav(v) ← Otevřený, $T \leftarrow T+1$, $in(v) \leftarrow T$
2. Pro hrany vw:
3. Pokud Stav(w) = nenalezen:
4. DFS2(w)
5. Stav(v) ← Zavřený, $T \leftarrow T+1$, $out(v) \leftarrow T$

O(1) na hranu

Stav(v): nenalezen, otevřený, zavřený

in(v)
out(v)

DFS(v_0): ∀v Stav(v) ← Nenalezen
in(v), out(v) ← ∅] $O(n)$
$T \leftarrow 0$
DFS2(v_0)

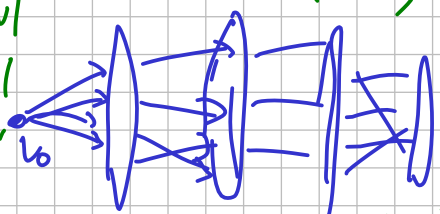
Notace: $G=(V,E)$ → Chceme čas $\Theta(n+m)$

$n := |V|$

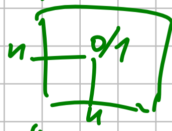
$m := |E|$

typ. orientovaný

multigrafy?



Reprezentace grafu:



matice sousednosti

seznamy sousedů

Lemma: DFS se zastaví v čase $O(n+m)$.

Stavy v: N → O → Z
⇒ vrchol otevřen max. 1
⇒ DFS2 se zavolá max. 1 na vrchol

Dk #1: $O\Big(n + \underbrace{\sum_{v\in V} \deg^{out}(v)}_{m}\Big) = O(n+m)$

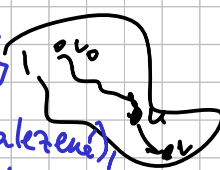
Dk #2: na ∀ hranu sáhnu nejvýš jednou a stojí to $O(1)$
→ celkem $O(n+m)$

Lemma: Po zastavení DFS je ∀v stav(v) = { nenalezený / zavřený ⇔ v je dosažitelný z v_0

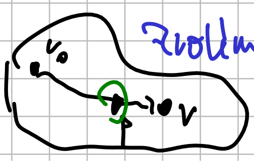
Dk: ⇒ indukcí podle běhu algoritmu: pokud vrchol zavřeme, je dosažitelný

předtím jsme otevřeli
⇑
DFS(v)
⇑
∃ hrana uv z otevřeného u
IP ⇒ u dosažitelné z v_0

⇐ sporem: kdyby ∃ špatné vrcholy (dosažitelné, ale nenalezené)



zvolíme v špatný, nejbližší z v_0



P := předposl. vrchol nejkr. cesty z v_0 do v
↳ p je dobrý ⇒ hrana pv musela být objev. ↯