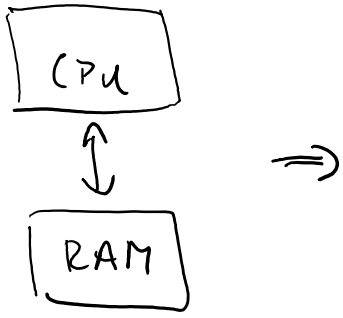
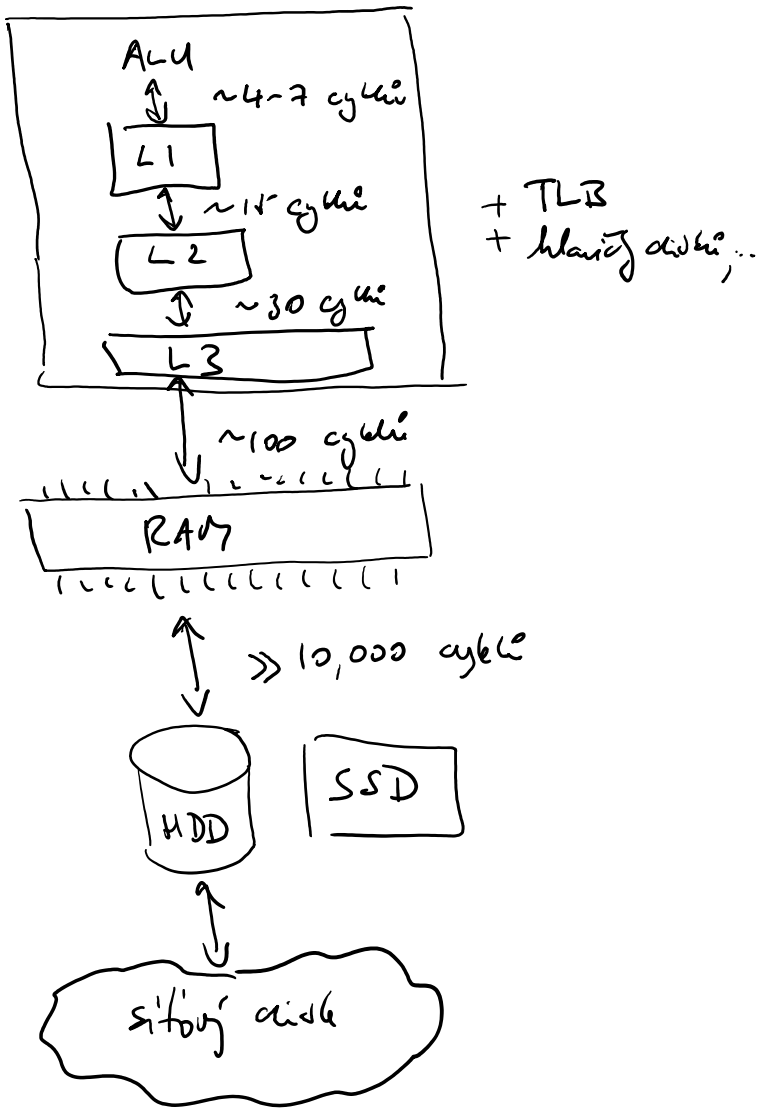
Paměťová hierarchie

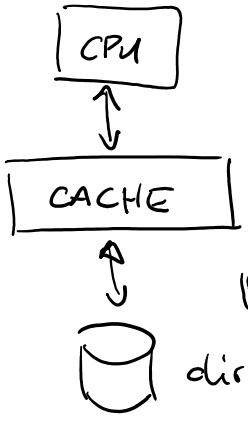
4 November, 2021 11:08





Model externí paměti

- měříme počet přenesených bloků mezi vyrovnávací pamětí (CACHE) a diskem
- soustředíme se na jednu úroveň vyrovnávací paměti



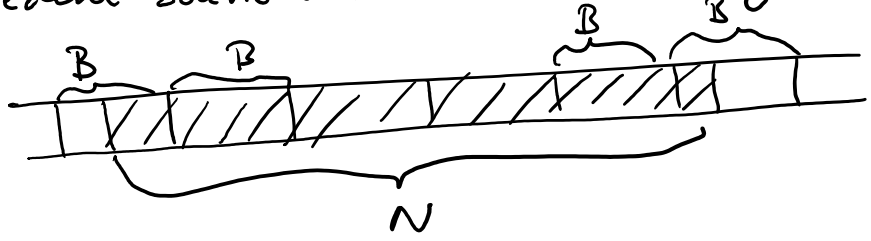
M... velikost

přenos po blocích velikosti B.

a zbytek ignorujeme

→ algoritmus optimalizovaný pro konkrétní úroveň

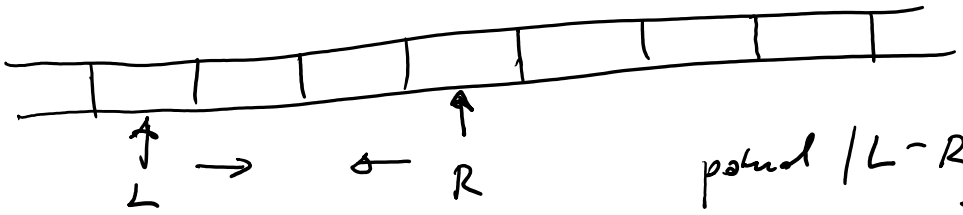
Př: • předení souvislého kusu dat délky N



Alg 1

$\lceil \frac{N}{B} \rceil + 1$ přenosů

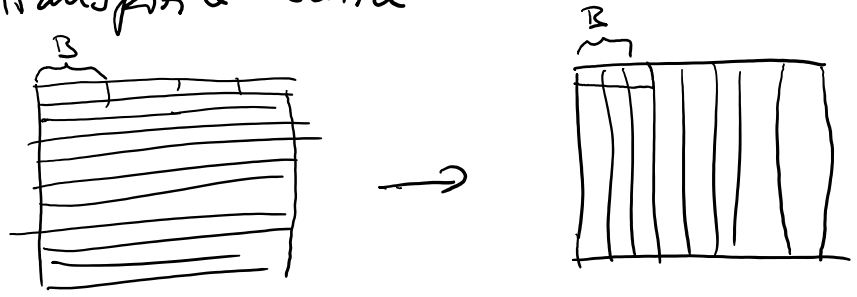
• binární vyhledávání v poli velikosti N



pokud $|L-R| \leq B$ nemusíme načítat další bloky

→ $\log N - \log B$ přenosů $= \log \frac{N}{B}$

• transpozice matice

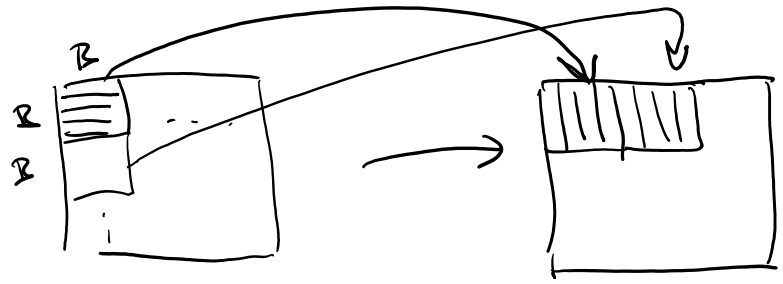


- matice v paměti uložena po řádcích, jednotlivé řádky jsou za sebou

→ naivní algoritmus - $\geq N^2$ přenosů (pokud $\frac{M}{B} < N$)

• pokud $M > B^2$ lze udělat algoritmus s $\frac{N^2}{B}$ přenosy (optimální)

- pokud $M > B^2$ lze udělat algoritmus s B přenosy (optimální)



transpozice po blocích velikosti $B \times B$ (Alg. 2)

(→ "cache aware algoritmy")

cache oblivious analýza

- algoritmus analyzujeme v neznámých M a B
  - chceme algoritmus, který se bude chovat optimálně pro každou volbu M a B, ale algoritmus nezávisí na M a B. (M a B se v algoritmu nikde nevyskytuje.)

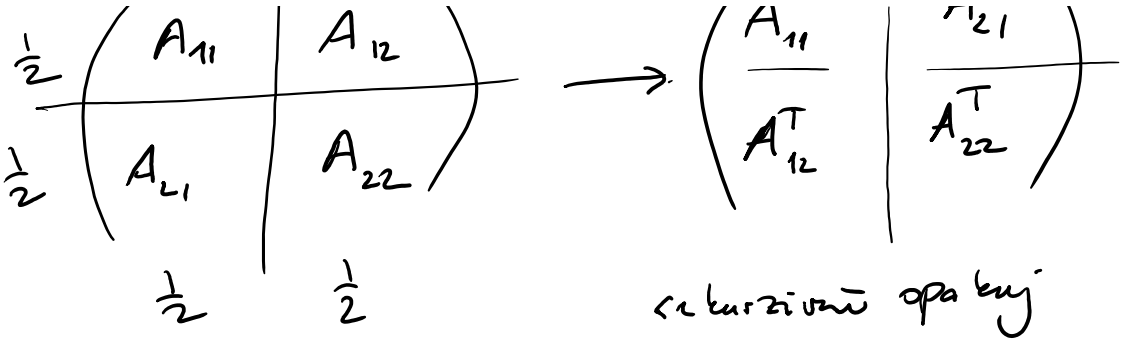
⇒ na každé úrovni optimální počet přenosů

⇒ celkově optimální počet přenosů.

Př.:
- přečtení souvislého kusu dat (Alg 1):
  - algoritmus nezávisí na M a B.
  - na každé úrovni $\lceil \frac{N}{B} \rceil + 1$ přenosů
  - líp to nelze
- transpozice matice (Alg2)
  - závisí na volbě B → není dobrý

transpozice matice - "cache oblivious" algoritmus

$$\frac{1}{2}\left(\begin{array}{c|c} A_{11} & A_{12} \end{array}\right) \longrightarrow \left(\begin{array}{c|c} A_{11}^T & A_{21}^T \end{array}\right)$$



rekurzivně opakuj

$T(A)$ ... transponuj A na místě

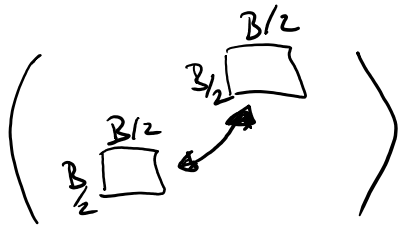
$TS(A,B)$ ... transponuj A & B a prohoď

$$T(A) := T(A_{11}), T(A_{22}), TS(A_{12}, A_{21}).$$

$$TS(A,B) := TS(A_{11}, B_{11}), TS(B_{22}, A_{22}), TS(A_{12}, B_{21}), TS(A_{21}, B_{12}).$$

- celkově n^2 operací
- počet přenosů $O(n^2/B)$ za předpokladu $M > B^2$ ("tall cache assumption")

Dk.:



v určitou chvíli algoritmus aplikuje T/TS na matice velikosti $\approx \frac{B}{2} \times \frac{B}{2}$. Ty zaschnují do max B bloků, čili všechny bloky mohou být načteny do cache zároveň

— B přenosů (podmnožic)

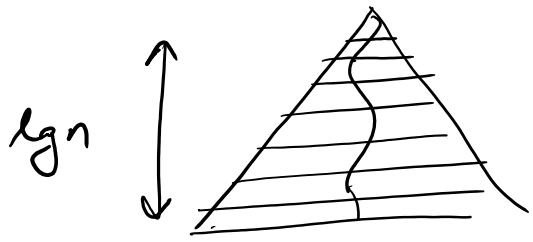
počet těchto volání je

$$\frac{N}{B/2} \times \frac{N}{B/2} = 4\frac{N^2}{B^2}$$

$B/2 \cdot B/2$ B^2

$\rightarrow 4\frac{N^2}{B^2} \cdot B = 4\frac{N^2}{B}$ přenosů $\boxed{B}$

Uložení stromů v paměti



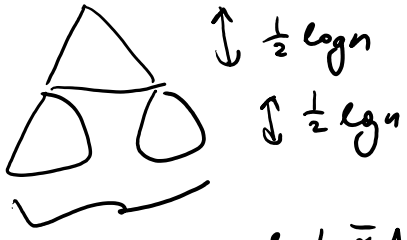
uložení po řádcích jako u regulární matice

$\log N - \log B$ přenosů při průchodu od kořene k listu

- B-stromy

Van Emde Boasovo rozložení

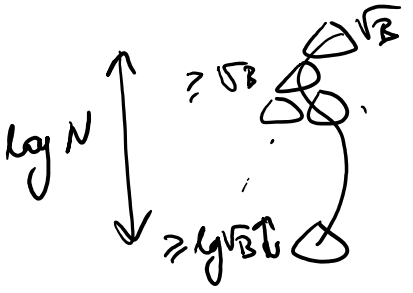
rekurzivně:



každý podstrom uložen souvisle v paměti

$\sqrt{n}$ stromů, každý velikosti $\sqrt{n}$

strom velikosti $\leq B$ je uložen v $O(1)$ souvislých blocích



cesta od kořene do listu projde $\leq \log N / \lg\sqrt{B}$ těmito podstromy.

$\rightarrow$ počet přenosů $O\left(\frac{\lg N}{\lg B}\right)$ $\boxed{B}$

třídění: mergesort $O\left(\frac{n}{B} \lg n\right)$ přenosů

k-cestný mergesort $O\left(\frac{n}{B} \lg_k n\right)$ přenosů

Funnelsort $O\left(\frac{n}{B} \lg_{\frac{M}{B}} n\right)$ přenosů pokud $M \geq B^2$

$\rightarrow O(n \lg n)$ operací

Násobení matic, FFT, ...

- Správa cache: Otázka: který blok vyhodit z cache, abychom mohli načíst požadovaný?

  problém: 1) neznáme budoucnost

  2) asociativní cache – daný blok může sedět pouze na některých místech v cache.

Sleator-Tarjan (1985):

1) není problém:

posloupnost přístupů $s_1, s_2, \ldots s_N$

LRU strategie: vyhodí blok nejdéle nepoužitý

OPT strategie: vyhodí blok, který bude použit za nejdelší dobu

↳ vyžaduje znalost budoucnosti

Věta: pokud dáme LRU strategii n_{LRU} bloků cache

a OPT strategii n_{OPT} bloků cache

pak $\#\text{výpadků LRU} \leq \frac{n_{LRU}}{n_{LRU} - n_{OPT}} \cdot \#\text{výpadků OPT} + n_{LRU}$

Př.: $n_{LRU} = 2 \cdot n_{OPT}$

$\rightarrow \#\text{výpadků LRU} \leq 2 \#\text{výpadků OPT}$

Pr.: $n_{LRU} = 2 \cdot n_{OPT}$

$\Rightarrow$ #výpadků LRU $\leq$ 2 #výpadků OPT $+ n_{LRU}$.

Dk.: rozdělíme $s_1, s_2, \ldots s_N$ na kusy zleva doprava, aby každý kus obsahoval n_{LRU} různých bloků (až na poslední)

k kusů

na každém kusu má LRU max. n_{LRU} výpadků

— // — OPT alespoň $n_{LRU} - n_{OPT}$ výpadků

#výpadků LRU $\leq n_{LRU} \cdot k$

#výpadků OPT $\geq (n_{LRU} - n_{OPT}) \cdot (k-1)$

$$\rightarrow \frac{\#\text{výpadků LRU}}{n_{LRU}} \leq \frac{\#\text{výpadků OPT}}{n_{LRU} - n_{OPT}} + 1$$