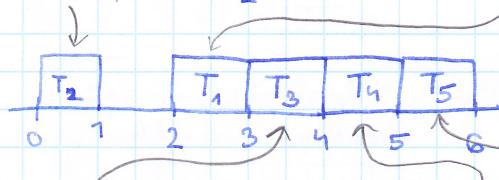


$$G = \{T_1, T_2, T_3, T_4, T_5\}$$

$$d = (3, 5, 4, 4, 6)$$

$$L=0, G=\{T_2\}$$



$$L=2, G=\{T_1, T_3\}$$

task	1	2	3	4	5
lateness	0	-4	0	-1	0

$$L=3, G=\{T_3, T_4, T_5\}$$

$$L=4, G=\{T_4, T_5\}$$

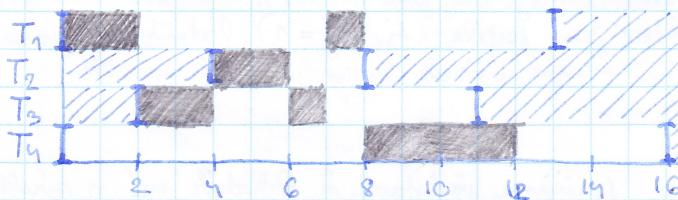
$$L=5, G=\{T_5\}$$

$$\begin{aligned} Př: & G = \{T_1, T_2, T_3, T_4\} \\ p = & \{3, 2, 3, 4\} \\ r = & \{0, 4, 2, 0\} \\ d = & \{13, 8, 11, 16\} \end{aligned}$$

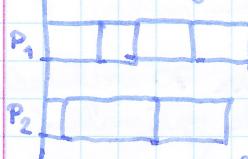
V čase  $t_0$  jsou ready úlohy  $T_1$  a  $T_4$ , ale  $T_1$  má blížší due date, takže zapne  $T_1$ . Potéž, dokud nebude ready nová úloha, bude do času  $t_2$ .

V čase  $t_2$  jsou ready úlohy  $T_1, T_3, T_4$ , a zúlohy  $T_1$  slyví už celat většinu jednoho dílu práce. Neblížší due date má  $T_3$ , spustím ji až do  $t_4$ .

V čase  $t_4$  jsou ready všechny úlohy ready, slyvající práce je  $p = \{1, 2, 1, 4\}$ . Spusťme  $T_2$ , pak  $T_3$ , pak  $T_1$  a nakonec  $T_4$ .



2 procesory, nedělitelné úlohy.



Minimalizujeme  $C_{max}$ . Je to jeho a ta samá úloha, jehož delení lze rozdat mezi oba počítače.

$C_{max}$   $\rightarrow$  Tolle je NP úplné

Ještě víc  $\downarrow$  preemption/migraci, ale nejdále jde vložit doho se stejnou fázi jednak 2 procesory, dělitelné úlohy: tolle je jednoduše řešit McNaughtonům algoritmem.

$$C_{max} = \max \{ \max(p_i), \frac{1}{2}p_i \cdot \sum p_i \}$$

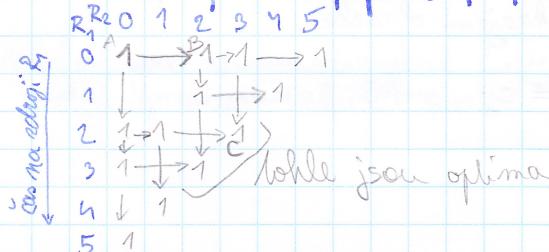
Postupně plníme první procesor úlohami. Jakmile úloha překáží  $C_{max}$ , lze ji vložit do druhého procesoru, pak plníme druhý procesor atd.

List scheduling - approximační algoritmus

uvažujeme si vhodně seřazený list úloh. Vybereme procesor, který skončí nejdřív, a k tomu nyníme úlohy, které je poté dostupný.

Rothkopf: Uvažujeme pole matic, pro každý sloupec má jednu osu. Celkově sloupek je nazýván UB (Upper Bound). Všechny hodnoty sloupu projdeme, a pokud tam je jednička, skončíme na ni navazat další sloupek.

$$Př: n=3, R=2, p=(2, 1, 2), UB=5$$



A: vložíme úlohu jedna, ta má  $p=2$ , a proto se posune o 2 po obou osách

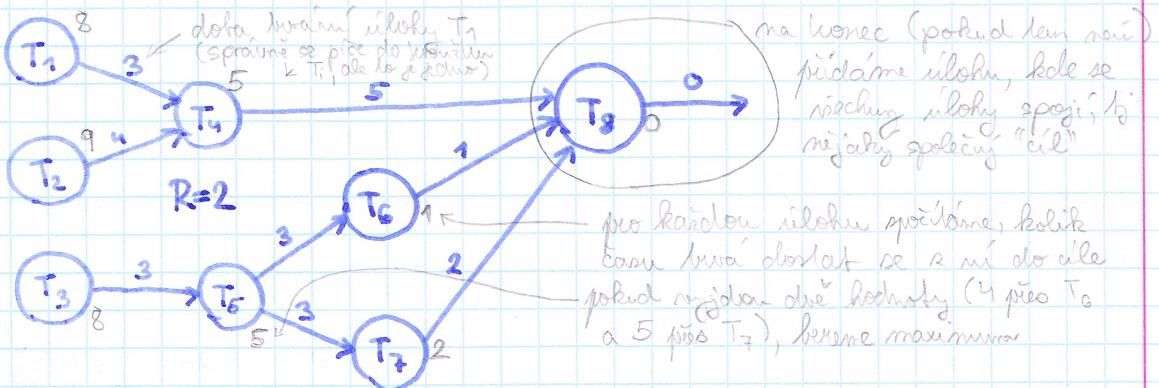
B: vložíme úlohu 2 coby 1, posune se o 1

C: existují dva novéby, které skončí v čase  $[2, 3] \Rightarrow 3$

A lehce maximální

Muntz & Coffman: Yestavístrom řešení a v něm se posune dál, když je vše potřebné hotovo. Uložení přesahuje virtuální zdroje, proto se ho nemůže počítat.

Př:



V čase  $A_0$  máme obětí úlohy  $T_1, T_2, T_3$ , s vzdáleností od cíle 8, 9, 8. Ze srovnání úloh vybereme  $T_2$  s meziúlohou vzdálenou 5 a do druhé srovnávky dáme systém:  $S_1 = \{T_2\}$ ,  $S_2 = \{T_1, T_3\}$ . V bázdě skupině spočítáme, jakou časť zdroje je s ní úloha mít:  $B_1 = 1/1S_1$ . Tady  $B_1 = 1$ ,  $B_2 = 0.5$ . Předpokládejme, že jak obětoho se ulevíme v  $S_1$  a  $S_2$  nezmění, kdežto  $\delta = 2$ .

V čase  $A_2$  máme úlohy  $T_1, T_2, T_3$ , s vzdálenimi 7, 7, 7. Rovnice jsou všechny na stejnou vzdálenou, tudíž v jedné srovnávce  $S$  a podílí se rovnoměrně o oba zdroje když  $B_1 = 2/3$ . Další změna je na vzdálení 5, a ta nastane za  $\delta = 3$ .

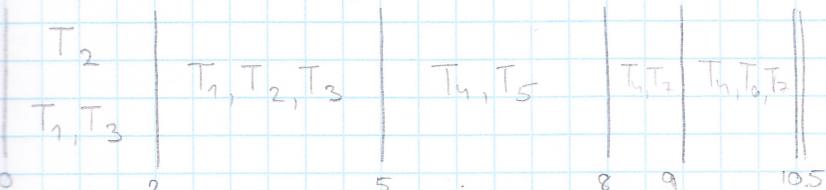
V čase  $A_5$  máme úlohy  $T_4, T_5$ , vzdálené 5, 5.  $S = \{T_4, T_5\}$ ,  $B = 1$ ,  $\delta = 3$ .

V čase  $A_7$  máme úlohy  $T_4, T_6, T_7$ , vzdálené 2, 1, 2.  $S = \{T_4, T_6, T_7\}$ ,  $B = 1$ ,  $\delta = 1$ .

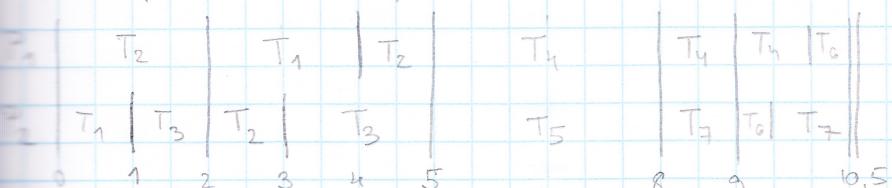
V čase  $A_9$  máme úlohy  $T_4, T_6, T_7$ , vzdálené 1, 1, 1.  $S = \{T_4, T_6, T_7\}$ ,  $B = \frac{2}{3}$ ,  $\delta = 1.5$ .

V čase  $A_{10.5}$  nám slíží jiná úloha  $T_8$  s nulačním časem vzdálení, kterou jsme užile přidali; algoritmus končí.

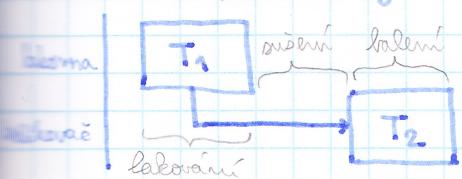
Rozvrh na virtuálních zdrojích:



Rozvrh po finálních úpravách



Temporal constraints: určuje čekací dobu mezi dvěma úlohami, během které většinou stroj volný, ale např. výrobek musí čekat. V diagramu se znázorňuje "tyčka" pod úlohou, která odsouhá další úlohy. V grafu se přímo zjistí cena kvadrátu - proto by měl být píšťadlo myši prázdné (análogie) nevhodné, když tyto constraints mohou mít rozdíly



čekací neboť tyčka ještě zdroje - lakovač může mít jenom jeden větší než všechno