

Úvod do rozvrhování (dokončení)

21. února 2023

- 1 Klasifikace rozvrhovacích problémů (dokončení)
- 2 Složitost

- Precedenční podmínky *prec*
 - lineární posloupnost, stromová struktura
 - pro úlohy a, b píšeme $a \rightarrow b$, což znamená $S_a + p_a \leq S_b$
 - příklad: montáž kola
- Přerušování úlohy (*preemptions*) *pmtn*
 - při příchodu úlohy s vyšší prioritou je současná úloha přerušena
- Vhodnost stroje *M_j*
 - podmnožina strojů M_j , na níž lze provádět úlohu j
 - přiřazení místností: postačující velikost učebny
 - hry: počítač s HW grafickou knihovnou
- Omezení na pracovní sílu *W, W_l*
 - do problému zavedeme další typ zdroje
 - stroje mohou potřebovat operátory a úlohy lze provádět jen tehdy, pokud jsou dostupní *W* operátorů
 - mohou existovat různé skupiny operátorů se specifickou kvalifikací
 W_l je počet operátorů ve skupině l

- Směrovací (*routing*) omezení

- udávají, na kterých strojích musí být úloha prováděna
- pořadí provádění úlohy v multi-operačních problémech
 - job shop problém: pořadí operací předem stanoveno
 - open shop problém: pořadí operací úlohy (*route for the job*) stanoveno až při rozvrhování

- Nastavovací (*setup*) doba a cena

$s_{ijk}, c_{ijk}, s_{jk}, c_{jk}$

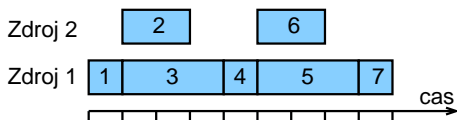
- závislé na posloupnosti provádění
- s_{ijk} čas nutný pro provádění úlohy k po úloze j na stroji i
- c_{ijk} cena nutná pro provádění úlohy k po úloze j na stroji i
- s_{jk}, c_{jk} čas/cena nezávislý na stroji
- příklady
 - problém obchodního cestujícího $1|s_{jk}|C_{\max}$

- Výroba na objednávku a na sklad
 - výroba zboží na sklad, pokud je u něj záruka spotřeby nutno uvážit cenu za skladování
 - výroba zboží na objednávku vynucuje úvahu termínů dokončení vyprodukované množství závislé na zákazníkovi
- Skladovací prostor a doba čekání při výrobě
 - omezené množství prostoru při výrobě
 - horní hranice počtu úloh čekajících ve frontě na stroj
 - **blokování**: úloha je zablokována na současném stroji, protože fronta na následujícím stroji je plná
- ...

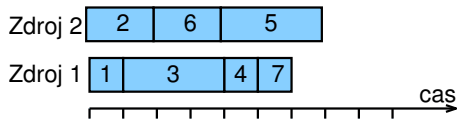
- **Makespan C_{\max}** : maximální čas konce úloh

$$C_{\max} = \max(C_1, \dots, C_n)$$

- Příklad: $C_{\max} = \max\{1, 3, 4, 5, 8, 7, 9\} = 9$



- Cíl: **minimalizace makespan** často
 - maximalizuje **výkon** (*throughput*)
 - zajišťuje **rovnoměrné zatížení strojů** (*load balancing*)
 - příklad: $C_{\max} = \max\{1, 2, 4, 5, 7, 4, 6\} = 7$

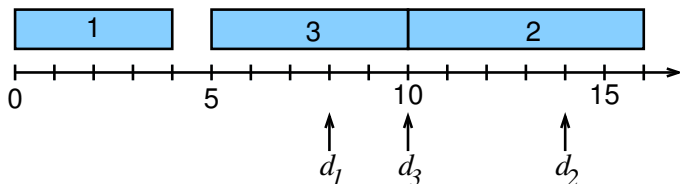


- Velmi často používané a základní kritérium

- Zpoždění (*lateness*) úlohy j : $L_j = C_j - d_j$
- Maximální zpoždění L_{\max}

$$L_{\max} = \max(L_1, \dots, L_n)$$

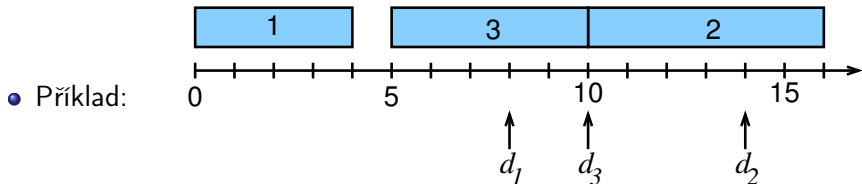
- Cíl: minimalizace maximálního zpoždění
- Příklad:



$$\begin{aligned}
 L_{\max} &= \max(L_1, L_2, L_3) = \\
 &= \max(C_1 - d_1, C_2 - d_2, C_3 - d_3) = \\
 &= \max(4 - 8, 16 - 14, 10 - 10) = \\
 &= \max(-4, 2, 0) = 2
 \end{aligned}$$

- Nezáporné zpoždění (*tardiness*) úlohy j : $T_j = \max(C_j - d_j, 0)$
- Cíl: minimalizace celkového zpoždění

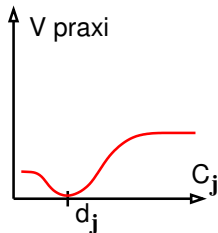
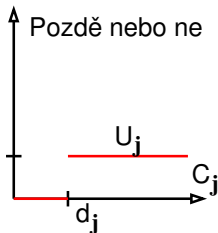
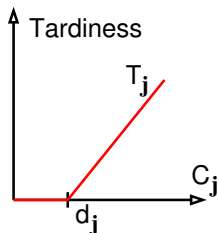
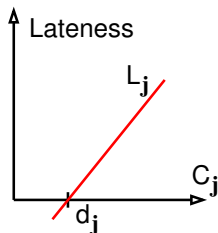
$$\sum_{j=1}^n T_j \quad \text{celkové zpoždění}$$



$$T_1 + T_2 + T_3 = \max(C_1 - d_1, 0) + \max(C_2 - d_2, 0) + \max(C_3 - d_3, 0) = \\ \max(4 - 8, 0) + \max(16 - 14, 0) + \max(10 - 10, 0) = 0 + 2 + 0 = 2$$

- Cíl: minimalizace celkového váženého zpoždění

$$\sum_{j=1}^n w_j T_j \quad \text{celkové vážené zpoždění}$$



- Cena za skladování vyrobeného zboží
- Cena za skladování při výrobě
(*Work-In-Process inventory cost*)
 - příliš velké množství právě vyráběného zboží může zaplnit linku
 - příliš dlouho odložené zboží může být znehodnoceno
- Délka skladování při výrobě svázána s časy konce úloh
⇒ minimalizace součtu časů konců úloh

$$\sum_{j=1}^n C_j$$

⇒ minimalizace váženého součtu časů konců úloh

$$\sum_{j=1}^n w_j C_j$$

celková hodnota daná skladováním při výrobě

- Robustnost
 - robustnější rozvrh vyžaduje méně změn při změně problému (porucha stroje, dopravní špička)
- Cena za nastavení (*setup*)
 - cena za připravení letadla na odlet (čištění, zásobování, doplnění pohonných hmot)
- Cena za pracovní sílu
 - cena za přiřazení zaměstnanců na konkrétní směnu

V problému často řada optimalizačních kritérií

- multi-kriteriální rozvrhování
 - *Pareto* optimalizace
- žádoucí vztah mezi nimi nemusí být jasně definovaný
 - co je důležitější?
- ani samotná kritéria nemusí být jasně definována
 - jak daný požadavek reprezentovat kritériem?

- Polynomiální problémy

- existuje algoritmus polynomiální složitosti pro řešení problému

- NP a NP-úplné problémy

- řešitelné nedeterministickým polynomiálním algoritmem
- potenciální řešení lze ověřit v polynomiálním čase
- v nejhorším případě exponenciální složitost (pokud neplatí $P=NP$)
- NP-úplný problém
 - libovolný problém v NP se na něj dá polynomiálně redukovat

- Příklady:

- Polynomiální

- $1||L_{\max} \quad P|pmtn|C_{\max} \quad 1||\sum w_j C_j$

- NP

- $1|r_j|L_{\max} \quad P2||C_{\max} \quad P2||\sum w_j C_j$

Řídící pravidla

21. února 2023

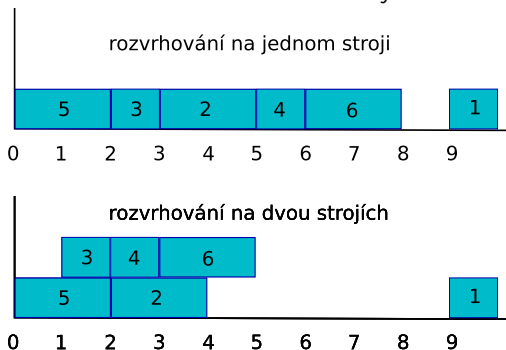
Řídící pravidla (*dispatching rules*)

Řídící pravidlo

- určuje pořadí (prioritu), ve kterém mají být úlohy prováděny
 - pokud má více úloh stejnou prioritu, úlohy jsou seřazeny náhodně (nebo např. dle čísla úlohy)
- jakmile se některý stroj uvolní, je vybrána nejprioritnější úloha

Příklad: použijte pro seřazení úloh **nejdřívější termín dostupnosti** r_j

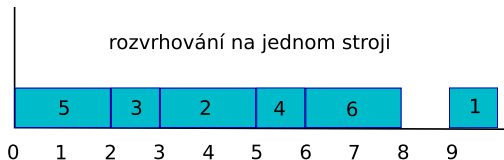
úlohy	1	2	3	4	5	6
r_j	9	2	1	2	0	3
p_j	1	2	1	1	2	2



Pravidla s termíny dostupnosti r_j a dokončení d_j

- Nejdřívější termín dostupnosti (*Earliest Release Date first ERD*)
 - ekvivalentní nejdříve-přijde-nejdříve-obsloužen (*First-Come-First-Serve*)
 - minimalizuje **odlišnosti v době čekání na stroji**
- Nejdřívější termín dokončení (*Earliest Due Date first EDD*)
 - směřuje k minimalizaci **maximálního zpoždění** mezi čekajícími úlohami
 - optimální pro $1||L_{max}$ (všechny úlohy dostupné na začátku)
 - pozor, i zde (zejména stejně jako **u všech pravidel**) **musíme brát v úvahu termín dostupnosti**, tj. úlohu lze plánovat teprve když je dostupná!!!
 - př. $r_2 = 3, d_2 = 5, r_3 = 0, d_3 = 6$ – dříve plánujeme úlohu 3

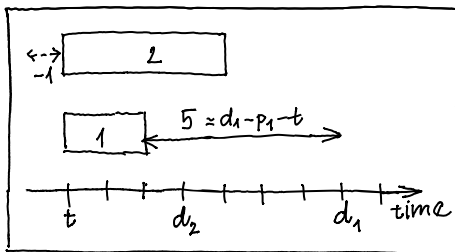
úlohy	1	2	3	4	5	6
r_j	9	3	0	2	0	6
p_j	1	2	1	1	2	2
d_j	10	5	6	9	2	8



Pravidla s termíny dostupnosti: minimální rezerva

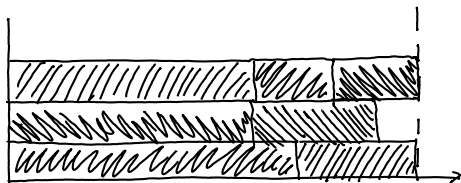
- Minimální rezerva (*Minimum Slack first MS*)

- $\max(d_j - p_j - t, 0)$
 - d_j termín dokončení
 - p_j doba provádění
 - t aktuální čas
- funkci \max používáme, abychom neměli záporné hodnoty pro úlohy, které už to určitě nestihnou
- minimalizace kritérií svázaných s termínem dokončení, tj. **maximální zpoždění**



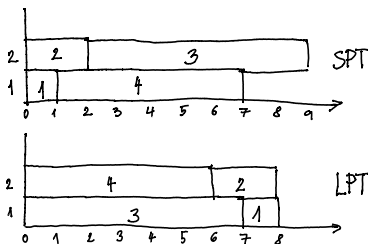
- **Statická pravidla** nejsou závislá na probíhajícím čase
 - pořadí se spočítá jako **funkce závislá na úloze a/nebo stroji**
 - pořadí nám definuje **prioritní frontu úloh**
 - př. nejdřívější termín dostupnosti
- **Dynamická pravidla** jsou závislá na čase
 - nutno **zahrnout do výpočtu funkce i aktuální čas**
 - uspořádání úloh závisí na čase \Rightarrow v každém čase je nutné určit znovu úlohu s nejvyšší prioritou a tu zpracováváme
 - př. minimální rezerva

- Nejdelší doba trvání (*Longest Processing Time first LPT*)
 - směřuje k rovnoměrnému zatížení paralelních strojů, tj. k minimalizaci **makespan**
 - myšlenka: kratší úlohy lze později využít pro vyrovnání zátěže na konci; jakmile jsou úlohy přiřazeny na stroje, tak je lze přeuspořádat bez změny zatížení



Pravidla s dobou trvání p_j

- Nejkratší doba trvání (*Shortest Processing Time first SPT*)
 - směřuje k **minimalizaci součtu časů konců úloh**, tj. **WIP** (Work In Process, cena za sklad při výrobě)



- Vážená nejkratší doba trvání (*Weighted Shortest Processing Time first WSPT*)
 - navíc w_j oproti SPT (řadím dle w_j/p_j)
 - **minimalizace váženýých součtu časů konců úloh**, tj. **WIP**
 - optimální pro jeden stroj, kde jsou všechny úlohy dostupné na začátku ($r_j = 0$ pro každou úlohu j)

- Kritická cesta (*Critical Path CP*)
 - vhodné pro precedenční omezení
 - vybírá úlohu, která je první v nejdelším řetězci dob provádění v grafu úloh daném precedencemi
 - vede k minimalizaci **makespan**
- Nejméně flexibilní úloha (*Least Flexible Job LFJ first*)
 - při zadání množiny vhodných strojů
 - vybírá se úloha, která může být prováděna na nejmenším počtu strojů (tj. nejméně alternativ)
 - vede k minimalizaci **makespan**
- Náhodné pořadí (*Service in Random Order SIRO*)
 - náhodný výběr úloh

- Jednoduchá na implementaci
- Optimální ve speciálních případech
- Zaměřeny na jedno optimalizační kritérium
- Kombinování několika řídicích pravidel: **kompozitní řídicí pravidla**

- **Použití v praxi**
 - pro řadu problémů příliš triviální
 - i tady lze např. použít jako generátor iniciálního řešení
 - nebo jako metodu pro řešení podproblémů
 - používá se pro složité problémy s vysokými nároky na propustnost
 - např. počet naplánovaných aktivit za vteřinunebo pro problémy s vysokým stupněm dynamiky
 - např. plánování úloh na počítače
(neznámá doba trvání, příchody nových úloh, výpadky strojů)