# Plánování job-shopu (pokračování)

11. dubna 2023

1. Matematické programování a job shop
2. Metoda větví a mezí pro job shop
3. Posunování kritického místa *(shifting bottleneck)*

# Disjunktivní programování

- Formulace disjunktivního programování
  - nejčastěji používaná formulace matematického programování pro *job shop* bez recirkulace
    (bez recirkulace: úlohy prováděny na stroji nejvýše jednou)
  - vychází z disjunktivní grafové reprezentace

- Disjunktivní programování
  - jedná se o matematické programování
  - omezení rozděleny do množin konjunktivních omezení
    - všechna tato omezení musí být splněna
    - standardní matematické programování: všechna omezení konjuktivní

    a jedné nebo více množin disjunktivních omezení
    - z každé množiny disjunktivních omezení musí být alespoň jedno omezení splněno

# Disjunktivní programování a *job shop*

$y_{ij}$ značí startovní čas operace $(i, j)$ úlohy $j$ na stroji $i$

Minimalizace $C_{max}$

za předpokladu

$$
\begin{array}{rcll}
y_{kj} - y_{ij} & \geq & p_{ij} & \forall (i,j) \rightarrow (k,j) \in A \\
C_{max} - y_{ij} & \geq & p_{ij} & \forall (i,j) \in N \\
y_{ij} - y_{il} & \geq & p_{il} \text{ nebo } y_{il} - y_{ij} \geq p_{ij} & \forall (i,l),(i,j) : \\
 & & & \quad i = 1 \dots m \\
y_{ij} & \geq & 0 & \forall (i,j) \in N
\end{array}
$$

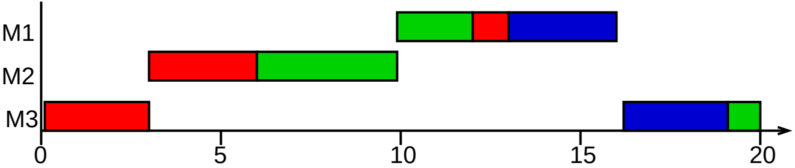
- Tato formulace ale nezajišťuje existenci standardní řešící procedury
- Problém velmi obtížný
- Ukážeme další přístupy: enumerační procedury (BB),
  heuristiky (posunování kritického místa)

# Typy rozvrhů

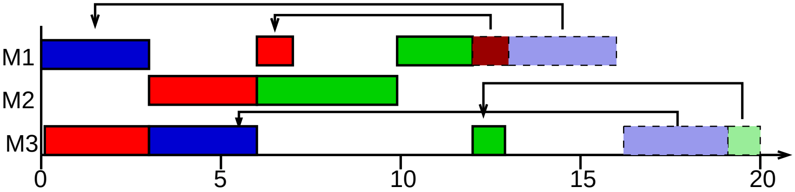
- Rozvrh je bez zdržení *(nondelay)*, pokud žádný stroj nečeká, když existuje dostupná operace
- Rozvrh je aktivní, pokud nemůže být žádná operace rozvrhována dříve změnou posloupnosti úloh na stroji bez pozdějšího naplánování jiné operace
  - redukce *makespan* aktivního rozvrhu je možná pouze zvýšením startovního času alespoň jedné operace
  - optimální rozvrh je aktivní rozvrh

# Příklad: aktivní vs. neaktivní rozvrh

Neaktivní rozvrh:




Aktivní rozvrh:



# Aplikace metody větví a mezí na *job shop*

- Víme: optimální rozvrh je aktivní rozvrh
- Metoda řešení
  - generování množiny aktivních rozvrhů
  - výběr nejlepšího rozvrhu
- Zlepšení
  - použití metody větví a mezí při generování
- Důsledek
  - potřebujeme algoritmus pro generování všech aktivních rozvrhů
- Značení
  - $\Omega$: množina všech operací, jejichž předchůdci už jsou narozvrhováni
  - $r_{ij}$: nejdřívější startovní čas operace $(i, j) \in \Omega$
    - může být spočítano pomocí výpočtu nejdelší cesty
  - $\Omega'$: podmnožina $\Omega$

# Generování množiny všech aktivních rozvrhů

1. Inicializace
   - $\Omega :=$ {první operace každé úlohy}
   - $r_{ij} := 0$ pro všechna $(i, j) \in \Omega$
2. Výběr stroje
   - spočítej pro současný částečný rozvrh

     $$t(\Omega) := \min_{(i,j)\in\Omega} \{r_{ij} + p_{ij}\}$$

     tj. kdy nejdříve může nějaká úloha z $\Omega$ skončit
   - $i^* :=$ stroj, na němž bylo dosaženo minima
3. Větvení
   - $\Omega' := \{(i^*, j) | r_{i^*j} < t(\Omega)\}$
   - pro všechna $(i^*, j) \in \Omega'$

     přidej do rozvrhu $(i^*, j)$ jako další operaci na stroji $i^*$

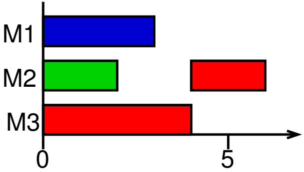
     smaž $(i^*, j)$ z $\Omega$

     přidej následníky úlohy $(i^*, j)$ do $\Omega$

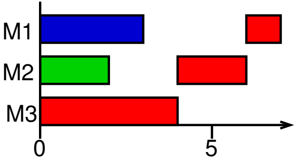
     běž na krok 2

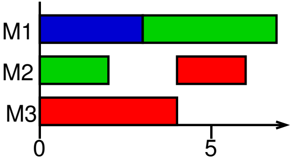
# Příklad: generování aktivních rozvrhů

- Úlohy: $J1 : (3,1) \rightarrow (2,1) \rightarrow (1,1)$
  $J2 : (1,2) \rightarrow (3,2)$
  $J3 : (2,3) \rightarrow (1,3) \rightarrow (3,3)$
- Částečný rozvrh:
  - neřešíme od začátku,
    začneme řešit už s tímto rozvrhem,
    aby byl postup demonstrativní




- $\Omega = \{(1,1),(3,2),(1,3)\}$
  - $p11 = 1, p32 = 3, p13 = 4 \quad r11 = 6, r32 = 4, r13 = 3$
- $t(\Omega) = \min(6+1, 4+3, 3+4) = 7$ např. $i^* = M1$
- $\Omega' = \{(1,1),(1,3)\}$
- Rozšířené částečné rozvrhy:



# Disjunkce vybrané při větvení

- Větvení algoritmu volí disjunkce
- Předpokládejme větvení $\Omega' = \{(i^*, j), (i^*, l)\}$

$\rightarrow$ výběr $(i^*, j)$ při větvení

přidání disjunkce $(i^*, j) \rightarrow (i^*, k)$ pro všechny dosud nenarozvrhované operace $(i^*, k)$

$\rightarrow$ výběr $(i^*, l)$ při větvení

přidání disjunkce $(i^*, l) \rightarrow (i^*, k)$ pro všechny dosud nenarozvrhované operace $(i^*, k)$

- Důsledek:
  - každý uzel stromu je charakterizován množinou $D'$ vybraných disjukcí

# Výpočet dolní hranice

Předpokládejme uzel $V$ s vybranými disjukcemi $D'$

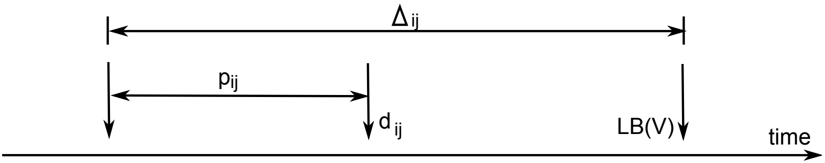
- Jednoduchá dolní hranice
  - spočítej kritickou cestu v $G(D')$
    dolní hranice $LB(V)$
- Vylepšená dolní hranice
  - uvažuj stroj $i$
  - povolíme překrývání operací na všech strojích kromě $i$
  - vyřeš problém na stroji $i$

# Podproblém: $1|r_j|L_{max}$

- Vypočítej nejdřívější startovní čas $r_{ij}$ všech operací $(i,j)$ na stroji $i$
  - nejdelší cesta ze zdroje v $G(D')$
- Vypočítej minimální množství času $\Delta_{ij}$ mezi:

  startem operace $(i,j)$ (tj. $r_{ij}$) a

  koncem rozvrhu (nejdelší cesta v $G(D')$ z uzlu do stoku)
- Získáme termíny dokončení $d_{ij} = LB(V) - \Delta_{ij} + p_{ij}$




- Vyřeš výsledný problém: $1|r_j|L_{max}$
  - viz dříve

# Vylepšená dolní hranice

- Vyřeš $1|r_j|L_{max}$ pro všechny stroje
- Výsledek: $L_1, \ldots, L_m$

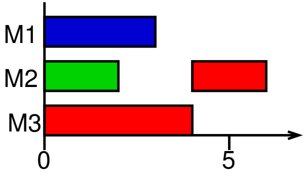
$$LB^{new}(V) = LB(V) + \max_{i=1...m} L_i$$

tj. výsledné řešení nemůže mít lepší kvalitu než nejlepší možné řešení pro každý stroj zvlášť, a proto zahrneme do dolní hranice nejhorší (největší) spočítané zpoždění

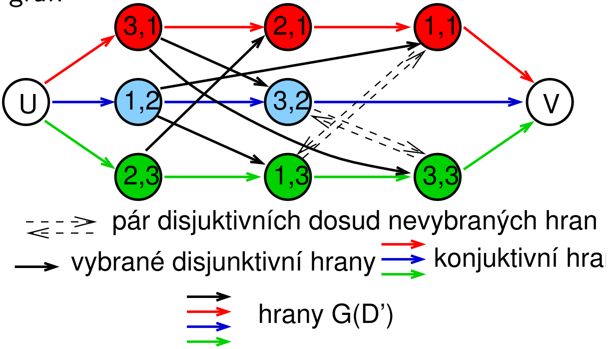
- Poznámky:
  - $1|r_j|L_{max}$ je NP-úplný problém
  - experimentální výsledky přesto ukazují, že se vyplatí řešit $m$ NP-úplných problémů pro každý uzel stromu
  - $20 \times 20$ instance jsou už obtížně řešitelné metodou větví a mezí

# Příklad: dolní hranice

Částečný rozvrh:

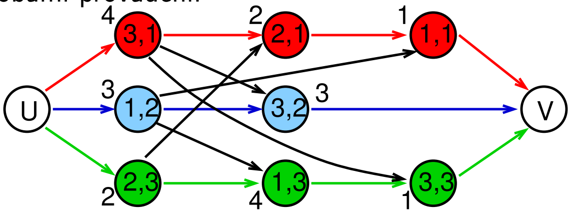



Odpovídající graf:



# Příklad: dolní hranice

Graf $G(D')$ s dobami provádění:



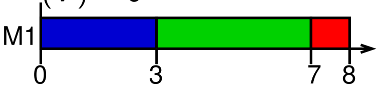

$LB(V) = l(U, (1,2), (1,3), (3,3), V) = 8$

Data pro úlohy na stroji $M1$:

| modrá | zelená | červená |
|---|---|---|
| $r_{12} = 0$ | $r_{13} = 3$ | $r_{11} = 6$ |
| $\Delta_{12} = 8$ | $\Delta_{13} = 5$ | $\Delta_{11} = 1$ |
| $d_{12} = 3$ | $d_{13} = 7$ | $d_{11} = 8$ |

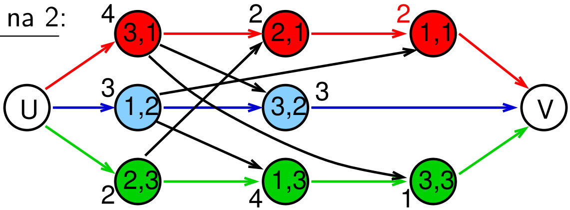
(víme: $d_{ij} = LB(V) - \Delta_{ij} + p_{ij}$)

Optimální řešení: $L_{max} = 0, LB^{new}(V) = 8$



# Příklad: dolní hranice

Změna $p11$ z 1 na 2:

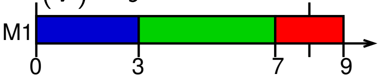



$$LB(V) = l(U,(1,2),(1,3),(3,3),V)$$
$$= l(U,(3,1),(2,1),(1,1),V) = 8$$

Data pro úlohy na stroji $M1$:

| modrá | zelená | červená |
|---|---|---|
| $r_{12} = 0$ | $r_{13} = 3$ | $r_{11} = 6$ |
| $\Delta_{12} = 8$ | $\Delta_{13} = 5$ | $\Delta_{11} = \underline{2}$ |
| $d_{12} = 3$ | $d_{13} = 7$ | $d_{11} = 8$ |

Optimální řešení: $L_{max} = 1, LB^{new}(V) = 9$



# Posunování kritického místa *(shifting bottleneck)*

- Úspěšná heuristika

  při řešení problému minimalizace *makespan* pro *job shop*

- Základní popis
  - iterativní heuristika
  - v každé iteraci je určen rozvrh pro jeden další stroj
  - používána re-optimalizace pro změnu už narozvrhovaných strojů

- Rozšiřitelnost: metoda lze rozšířit
  na obecnější *job shop* problémy
  - další objektivní funkce
  - *flexible flow shop*
    (paralelní stroj místo samostatných strojů)
  - nastavovací doba stroje

# Princip algoritmu

- Značení
  - $M$ je množina všech strojů
- Dáno
  - určen rozvrh pro podmnožinu $M_0 \subset M$ strojů
    - tj. je určen výběr disjunktivních hran
- Akce při jedné iteraci
  1. výběr stroje $k$, pro který ještě neexistuje rozvrh
     - tj. stroj z $M \backslash M_0$
  2. určení rozvrhu (výběru disjunktivních hran) pro stroj $k$ na základě daných (zafixovaných) rozvrhů pro stroje z $M_0$
  3. nové rozvrhování (= přeplánování) všech strojů z $M_0$ na základě ostatních daných (zafixovaných) rozvrhů, tj. přeplánujeme jeden stroj po druhém

# Princip výběru stroje a určení jeho rozvrhu

- Myšlenka:
  - výběr ještě nerozvrženého stroje, který působí nejvíce problémů, tzv. kritický (*bottleneck*) stroj
- Realizace:
  - spočítej pro každou operaci na nenarozvrhovaném stroji
    - nejdřívější možný startovní čas a
    - minimální zdržení mezi koncem operace a koncem úplného rozvrhu

    založeného na
    - zafixovaných rozvrzích strojů v $M_0$ a
    - pořadí úloh
  - spočítej pro každý nenarozvrhovaný stroj rozvrh respektující tyto nejdřívější termíny dostupnosti a zdržení
  - vyber stroj s maximálním koncovým časem a zafixuj rozvrh na tomto stroji

# Princip přeplánování strojů

- Myšlenka:
  - snaha redukovat *makespan* rozvrhu pro stroje v $M_0$
- Popis:
  - uvažuj stroje z $M_0$ jeden po druhém
  - smaž rozvrh vybraného stroje a spočítej nový rozvrh na základě
    - nejdřívějšího startovního času a
    - zdržení

    vyplývající z

    - ostatních strojů v $M_0$ a
    - pořadí úloh

# Plánování job-shopu: shrnutí

## Modelování a reprezentace

- disjunktivní grafová reprezentace
- matematické programování a job shop (+ řešení)

## Řešení

- metoda větví a mezí pro job shop
- heuristika: posunování kritického místa