Rezervace

9. května 2022

Rezervační systém

- m strojů zapojených paralelně
- n úloh s
  - dobou provádění $p_1, \ldots, p_n$
  - termíny dostupnosti $r_1, \ldots, r_n$
  - termíny dokončení $d_1, \ldots, d_n$
  - váhy $w_1, \ldots, w_n$
- Úloha musí být prováděna v rámci daného časového intervalu
  - termín dostupnosti a dokončení nelze porušit
- Nemusí být možné realizovat všechny úlohy
- Cíl: vyber podmnožinu úloh,
  - pro kterou existuje konzistentní rozvrh
  - která maximalizuje danou objektivní funkci
    - maximalizace počtu prováděných úloh
    - maximalizace celkového množství provádění
    - maximalizace profitu realizovaných úloh (zadané váhy)

Základní modely

- Systém bez rezervy
  - úlohy trvají od termínu dostupnosti do termínu dokončení, tj. $p_j = d_j - r_j$
  - mluvíme o pevných intervalech nebo o intervalovém rozvrhování
- Systémy s rezervou
  - interval mezi termínem dostupnosti a dokončení může mít nějakou rezervu, tj. $p_j \leq d_j - r_j$

Aplikace rezervačních systémů

- Pronájem aut
  - čtyři typy aut: subcompact, střední velikost, plná velikost, sportovní typ
  - pevné množství každého typu
  - stroj = typ auta
  - úloha = zákazník požadující auto
  - zákazník může požadovat určitý(é) typ(y) auta
    - úloha může být prováděna na podmnožině strojů
  - v případě nedostatku některého typu auta může být nabídnut v některých případech silnější typ auta
- Rezervace pokojů v hotelu
- Rezervace strojů v továrně

Intervalové rozvrhování

- m strojů zapojených paralelně
- n úloh, pro úlohu j zadán
  - termín dostupnosti r_j
  - termín dokončení d_j
  - doba provádění $p_j = d_j - r_j$
  - množina M_j strojů, na kterých může být úloha j prováděna
  - w_{ij}: profit z provádění úlohy j na stroji i
- Cíl: maximalizovat profit z prováděných úloh
  - $w_{ij} = 1$: maximalizovat počet realizovaných úloh
  - $w_{ij} = w_j$: maximalizovat vážený počet realizovaných úloh

Formulace celočíselného programování

- Časové periody $1, \ldots, H$
- J_l: množina úloh, která potřebuje provádění v čase l (známe předem!)
- $x_{ij} = 1$ pokud je úloha j prováděna na stroji i

  $x_{ij} = 0$ jinak
- Maximalizace

$$\sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} x_{ij}$$

za předpokladu:

úloha běží nejvýše na jednom stroji

$$\sum_{i=1}^{m} x_{ij} \leq 1 \qquad j = 1, \ldots, n$$

v každém čase běží na každém stroji nejvýše jedna úloha

$$\sum_{j \in J_l} x_{ij} \leq 1 \qquad i = 1, \ldots, m \quad l = 1, \ldots, H$$

provádění úlohy j na stroji i: $\quad x_{ij} \in \{0, 1\}$

Jednotková doba trvání

- Každá úloha je dostupná přesně 1 časovou jednotku
- Co z toho plyne? Problém lze rozdělit na nezavislé problémy
  - víme přesně, které úlohy i jsou prováděny v každé časové jednotce
  - jeden podproblém pro každou časovou jednotku
- Výsledný problém pro časovou jednotku l:

  Maximalizace
  $$\sum_{i=1}^{m}\sum_{j=1}^{n} w_{ij}x_{ij}$$

  za předpokladu
  $$\sum_{i=1}^{m} x_{ij} \leq 1 \quad j = 1, \ldots, n$$
  $$\sum_{j \in J_l} x_{ij} \leq 1 \quad i = 1, \ldots, m$$
  $$x_{ij} \in \{0, 1\}$$

- Jedná se o problém přiřazení *(assignment problem)*
  - problém řešitelný v polynomiálním čase

Jednotková váha & identické stroje

- $w_{ij} = 1$; $M_j = \{1, \ldots, m\}$ pro všechna i, j; obecná p_j
  - tedy stroje jsou identické a
    cíl je maximalizovat počet realizovaných úloh
  - nejednotková p_j, nelze tedy řešit rozkladem v čase
- Algoritmus dávající optimální řešení

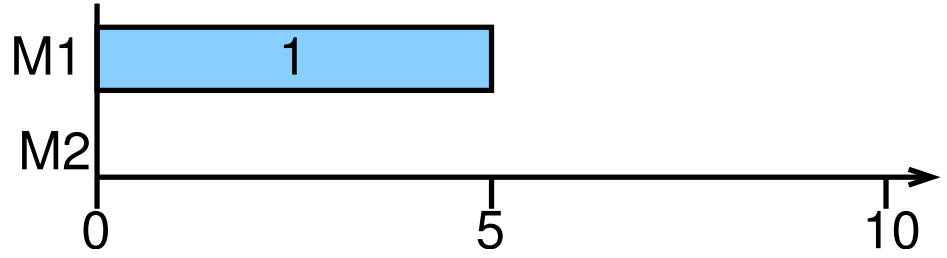
  Předpokládejme: $r_1 \leq \cdots \leq r_n$
  $J = \emptyset$ (J je množina už vybraných úloh pro provádění)
  for $j = 1$ to n do
    if existuje stroj dostupný v čase r_j then
      přiřaď j tomuto stroji
      $J := J \cup \{j\}$
    else urči úlohu j^* takovou, že $C_{j^*} = \max_{k \in J} C_k = \max_{k \in J}(r_k + p_k)$
      if $C_j = r_j + p_j > C_{j^*}$ then
        nezařazuj j do J
      else přiřaď j stroji j^*
        $J := J \cup \{j\} \setminus \{j^*\}$

Příklad: jednotková váha & identické stroje

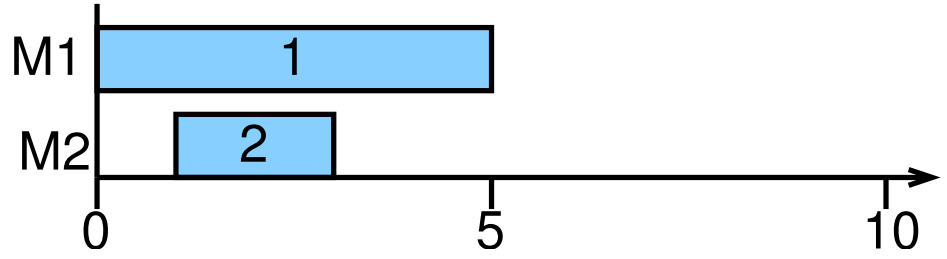
2 stroje a 8 úloh:

j	1	2	3	4	5	6	7	8
r_j	0	1	1	3	4	5	6	6
d_j	5	3	4	7	6	7	9	8

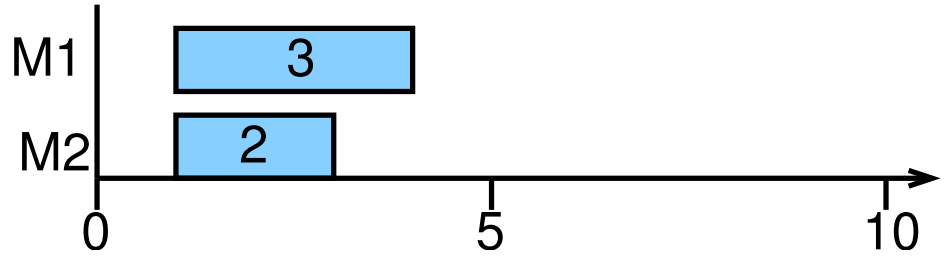
Iterace 1: $j = 1$



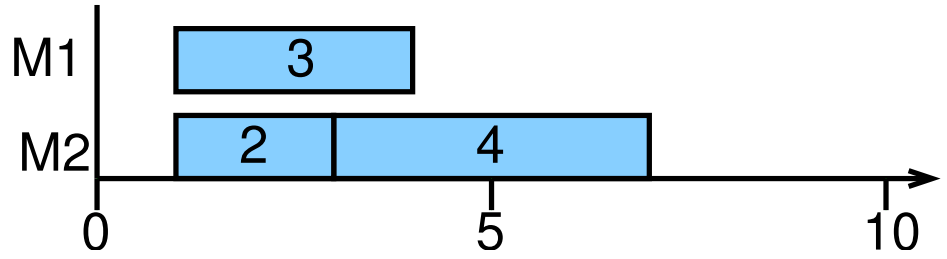
Iterace 2: $j = 2$



Iterace 3: $j = 3, j^* = 1$



Iterace 4: $j = 4$

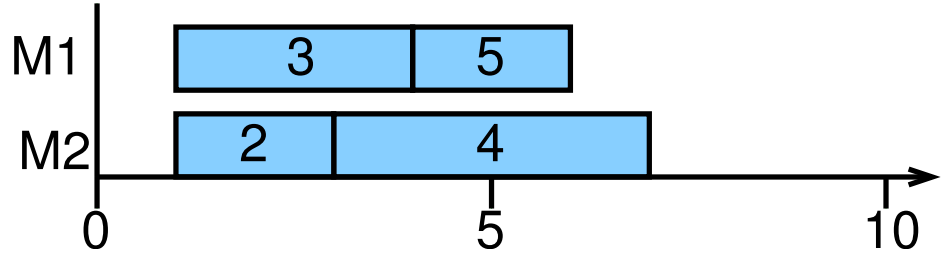


Příklad: jednotková váha & identické stroje

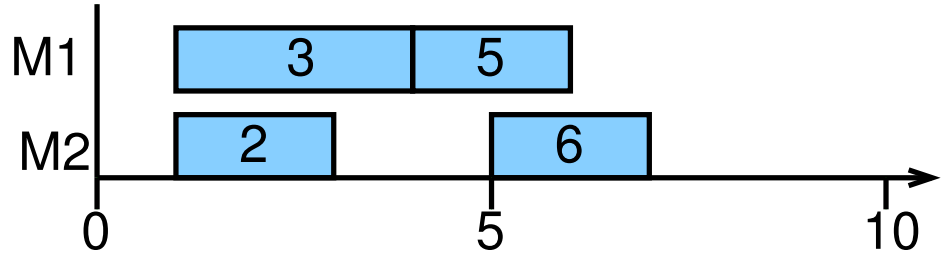
2 stroje a 8 úloh:

j	1	2	3	4	5	6	7	8
r_j	0	1	1	3	4	5	6	6
d_j	5	3	4	7	6	7	9	8

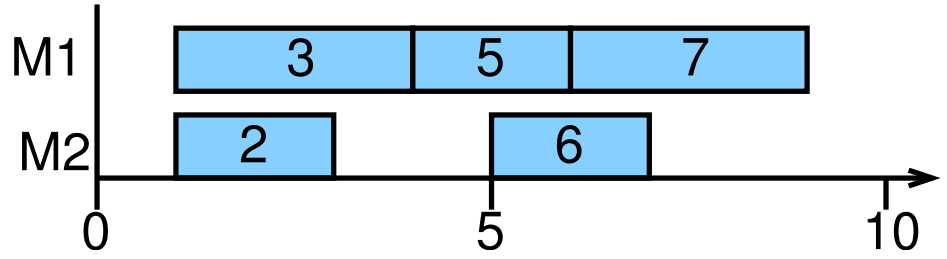
Iterace 5: $j = 5$



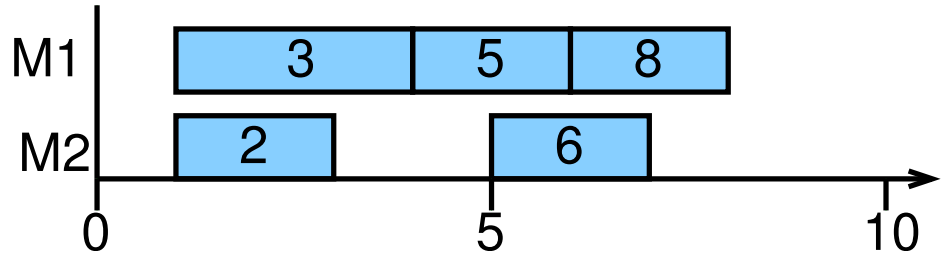
Iterace 6: $j = 6, j^* = 4$



Iterace 7: $j = 7$



Iterace 8: $j = 8, j^* = 7$



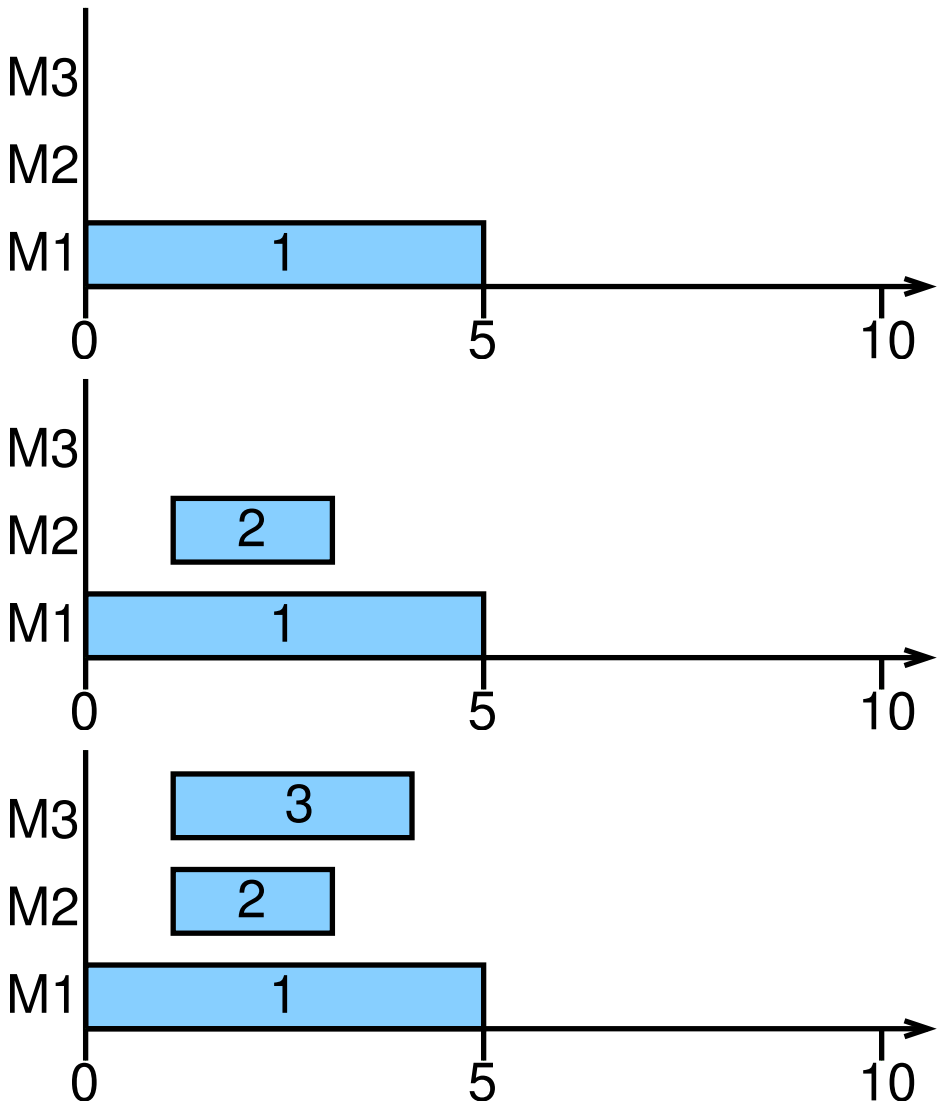
Jednotková váha a nelimitovaný počet identických strojů

- $w_{ij} = 1$ pro všechna i, j
- Všechny úlohy musí být realizovány
- Cíl: použít minimální počet strojů
- Algoritmus dávající optimální řešení

Předpokladejme: $r_1 \leq \cdots \leq r_n$
$M = \emptyset$ (M množina použitých strojů)
$i := 0$
for $j = 1$ to n do
  if stroj z M je volný v r_j
  then
    přiřaď j na volný stroj
  else
    $i := i + 1$
    $M := M \cup \{i\}$
    přiřaď úlohu j na stroj i

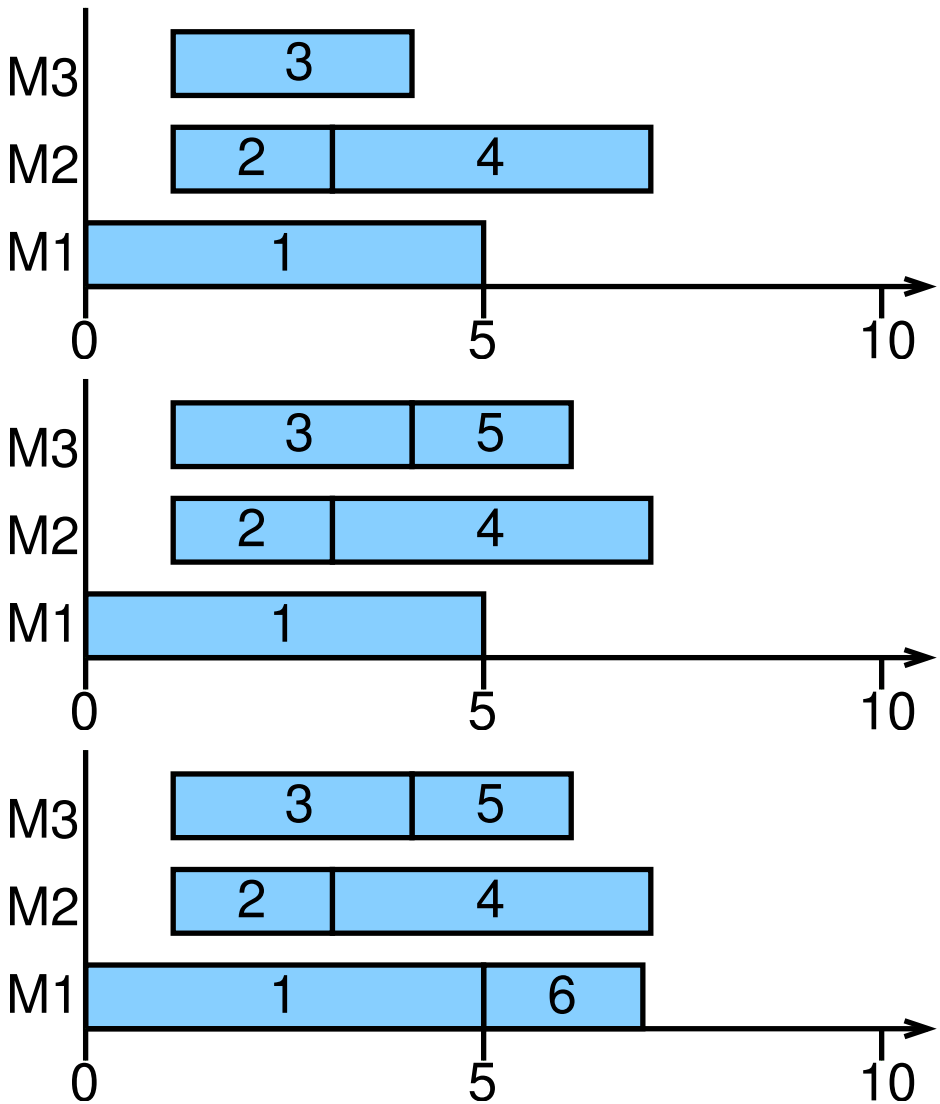
Příklad: jednotková váha a nelimitovaný počet identických strojů

j	1	2	3	4	5	6	7	8
r_j	0	1	1	3	4	5	6	6
d_j	5	3	4	7	6	7	9	8



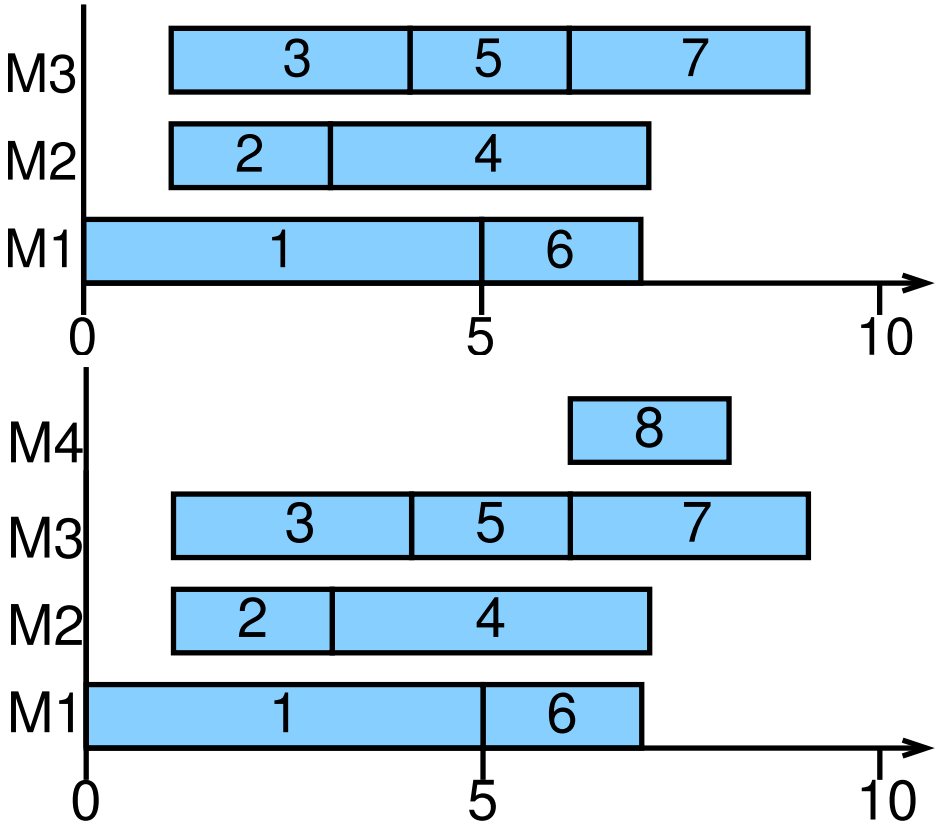
Příklad: jednotková váha a nelimitovaný počet identických strojů

j	1	2	3	4	5	6	7	8
r_j	0	1	1	3	4	5	6	6
d_j	5	3	4	7	6	7	9	8



Příklad: jednotková váha a nelimitovaný počet identických strojů

j	1	2	3	4	5	6	7	8
r_j	0	1	1	3	4	5	6	6
d_j	5	3	4	7	6	7	9	8



Barvení grafu

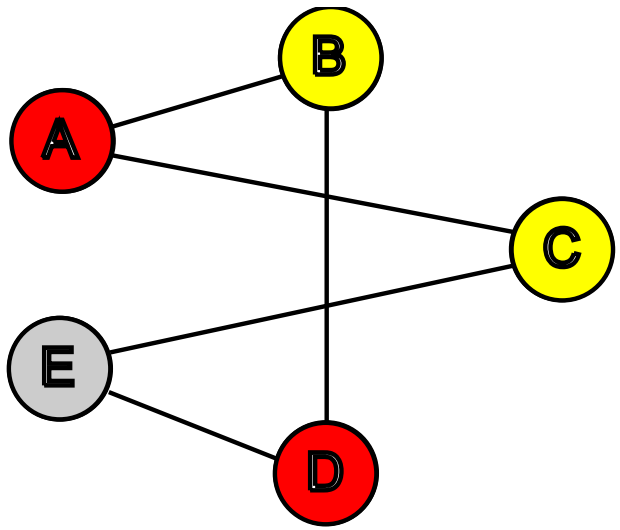
Problém barvení grafu

- Je možné obarvit vrcholy grafu s použitím n barev tak, aby žádné dva sousední vrcholy nebyly obarveny stejnou barvou?

Chromatické číslo grafu

- Minimální počet barev n nutný k obarvení grafu tak, by žádné dva sousední vrcholy nebyly obarveny stejnou barvou.

NP-úplný problém



Reformulace na problém barvení grafu

Problém s jednotkovou vahou a nelimitovaným počtem identických strojů lze reformulovat na problém barvení grafu

- vrchol = úloha
- hrana (j, k) znamená, že se úlohy j a k překrývají v čase
- každá barva odpovídá jednomu stroji
- přiřaď barvu (tj. stroj) každému vrcholu tak, že dva sousední vrcholy (překrývají se v čase) mají různou barvu (tj. stroje)

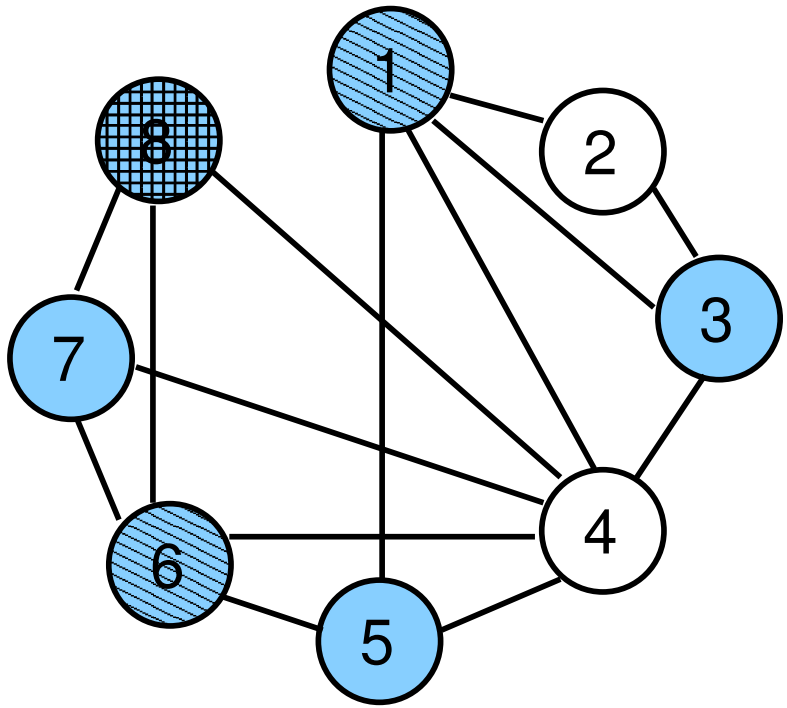
Poznámky:

- obecný problém existence obarvení grafu m barvami je NP-úplný
- uvažovaný rezervační problém je speciálním (jednodušším) případem problému barvení grafu
- heuristiky diskutovány v kapitole o *timetabling*

Příklad: reformulace

j	1	2	3	4	5	6	7	8
r_j	0	1	1	3	4	5	6	6
d_j	5	3	4	7	6	7	9	8

Odpovídající řešení problému barvení grafu:



Rezervační systém s rezervou

- Rezervační systém s rezervou: $p_j \leq d_j - r_j$
- Triviální případ
  - doby provádění = 1, identické váhy, identické stroje
  - řešení opět dekompozicí v čase
- Maximalizace váženého počtu aktivit
  - NP-těžký problém $\Rightarrow$ řešení heuristikami
  - kompozitní řídící pravidlo
    - předzpracování:
      určení flexibility úloh a strojů
    - algoritmus:
      nejméně flexibilní úloha na nejméně flexibilním stroji první

Předzpracování

- ψ_{it}: počet úloh, které mohou být přiřazeny
  na stroj i během intervalu $[t-1, t]$
  - možné využití stroje i v čase t
  - čím vyšší hodnota, tím je zdroj i v tomto čase flexibilnější

- $|M_j|$: počet strojů, které mohou být přiřazeny úloze j
  - čím vyšší hodnota, tím je úloha j flexibilnější

Prioritní indexy

- Prioritní index pro úlohu j: $I_j = f(w_j/p_j, |M_j|)$
  - uspořádání úloh podle: $I_1 \leq I_2 \leq \cdots \leq I_n$
  - čím menší je $|M_j|$, tím nižší je I_j
  - čím vyšší je w_j/p_j, tím nižší je I_j
    - snažíme se dát přednost kratším úlohám, protože maximalizujeme počet vážených provedených aktivit a delší úlohy jsou náročnější
  - např. $I_j = \frac{|M_j|}{w_j/p_j}$
- Prioritní index pro stroj
  - vybírá stroj pro úlohu
  - jestliže úloha potřebuje stroj v $[t, t + p_j]$
    pak výběr stroje i záleží na funkci s faktory $\psi_{i,t+1}, \ldots, \psi_{i,t+p_j}$, tj. na $g(\psi_{i,t+1}, \ldots, \psi_{i,t+p_j})$
  - např. $g(\psi_{i,t+1}, \ldots, \psi_{i,t+p_j}) = \left(\sum_{l=1}^{p_j} \psi_{i,t+l} \right) / p_j$

    nebo $g(\psi_{i,t+1}, \ldots, \psi_{i,t+p_j}) = \max(\psi_{i,t+1}, \ldots, \psi_{i,t+p_j})$

Algoritmus maximalizace váženého počtu aktivit

1. $j = 1$
2. Vyber úlohu j s nejnižším l_j a

   vyber mezi stroji a dostupnými časy zdroj a čas s nejnižší $g(\psi_{i,t+1}, \ldots, \psi_{i,t+p_j})$

   Zruš úlohu j jestliže nemůže být přiřazena ani jednomu stroji v žádném čase
3. Jestliže $j = n$ STOP

   jinak nastav $j = j + 1$ a běž na krok 2

Cvičení: maximalizace váženého počtu aktivit

- Nalezněte rozvrh pro daný problém

Úlohy	1	2	3	4	5	6	7
p_j	3	10	9	4	6	5	3
w_j	2	3	3	2	1	2	3
r_j	5	0	2	3	2	4	5
d_j	12	10	20	15	18	19	14
M_j	$\{1,3\}$	$\{1,2\}$	$\{1,2,3\}$	$\{2,3\}$	$\{1\}$	$\{1\}$	$\{1,2\}$

- pro $l_j = \frac{|M_j|}{w_j/p_j}$ a

  $g(\psi_{i,t+1}, \ldots, \psi_{i,t+p_j}) = \left(\sum_{l=1}^{p_j} \psi_{i,t+l}\right) / p_j$

- Řešení:

Úlohy	7	6	1	4	5	2
Stroje	2	1	3	3	1	2
Časy	11-14	14-19	5-8	11-15	2-8	0-10

úlohu 3 nelze umístit

Shrnutí

Rezervace

- Intervalové rozvrhování (rezervační systém bez rezervy)
  - celočíselné programování
  - jednotková doba trvání: formulace problému přiřazení (řešení pro každou čas. jednotku zvlášť)
  - jednotková váha & identické stroje (maximalizace počtu provedených aktivit)
  - jednotková váha & nelimitovaný počet identických strojů
- Rezervační systém s rezervou
  - kompozitní řídící pravidlo

Rozvrhování jako timetabling

9. května 2022

Rozvrhování = *timetabling*

- Doposud: rozvrhování = *scheduling*
  Nyní: rozvrhování = *timetabling*
  - důraz kladen na časové umístění objektů
  - vazby na časové uspořádání mezi objekty hrají malou roli
- Omezení operátorů/nástrojů *(operator/tool)*
  - mnoho operátorů
  - úloha potřebuje jeden nebo více odlišných operátorů
  - úlohy vyžadující stejné operátory nemohou běžet zároveň
  - možné cíle:
    rozvržení všech úloh v rámci časového horizontu H
    nebo minimalizace *makespan*
- Omezení pracovní síly
  - R identických pracovníků = jeden zdroj kapacity R
  - každá úloha potřebuje několik pracovníků
  - celkový počet pracovníků nesmí být nikdy překročen

Rozvrhování jako problém plánování projektu s omezenými zdroji

- Problém plánování projektu s omezenými zdroji
  resource-constrained project scheduling problem (RCPSP)
  - n úloh
  - N zdrojů
  - R_i kapacita zdroje i
  - p_j doba provádění úlohy j
  - R_{ij} požadavek úlohy j na zdroj i
  - $Prec_j$ (přímí) předchůdci úlohy j
- Rozvrhování s operátory
  - $R_i = 1$ pro všechna $i = 1 \ldots N$
- Rozvrhování s pracovní sílou
  - $N = 1$
- Oba problémy rozvrhování stále velmi obtížné
  - i při $p_j = 1$ NP-těžké
    operátoři $\equiv$ barvení grafu
    pracovní síla $\equiv$ *bin-packing*

Rozvrhování s operátory jako barvení grafu

- Omezíme se na problém s jednotkovou dobou trvání

- Úloha = uzel
- Úlohy potřebují stejného operátora = hrana mezi uzly
- Přiřazení barev (= času) uzlům
  - sousední uzly musí mít různé barvy
    tj. úlohy se stejným operátorem musí být prováděny v různých časech

- Problém existence řešení
  - tj. zadán časový horizont H a hledám rozvrh v rámci horizontu
  - může být graf obarven H barvami?
- Optimalizační problém
  - tj. minimalizace *makespan*
  - jaký nejmenší počet barev je třeba?
  - chromatické číslo grafu

Heuristiky pro barvení grafu

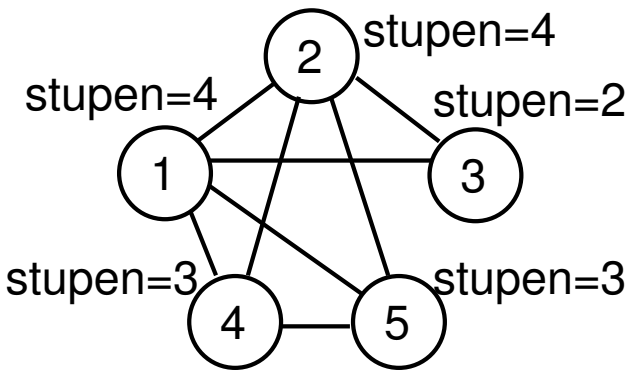
- Stupeň uzlu
  - počet hran spojených s uzlem
- Úroveň saturace
  - počet různých barev spojených s uzlem
- Intuice
  - obarvi uzly s vyšším stupněm dříve
  - obarvi uzly s vyšší úrovní saturace dříve
- Algoritmus
  1. uspořádej uzly v klesajícím pořadí podle jejich stupně
  2. použij barvu 1 pro první uzel
  3. vyber neobarvený uzel s maximální úrovní saturace
     v případě volby z nich vyber uzel
     s maximálním stupněm v neobarveném podgrafu
  4. obarvi vybraný uzel s nejmenší možnou barvou
  5. jestliže jsou všechny uzly obarveny STOP
     jinak běž na krok 3

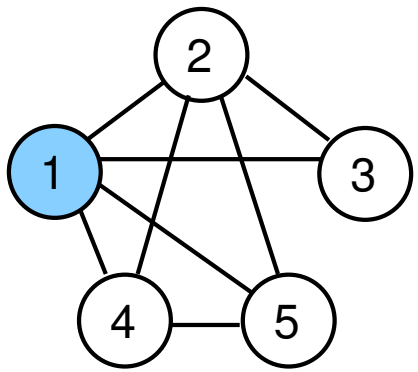
Příklad: rozvrhování schůzek

Vytvoř rozvrh pro 5 schůzek se 4 lidmi

- schůzka = úloha, člověk = operátor
- všechny schůzky trvají jednu hodinu

	1	2	3	4	5
Joe	1	1	0	1	1
Lisa	1	1	1	0	0
Jane	1	0	1	0	0
Larry	0	1	0	1	1

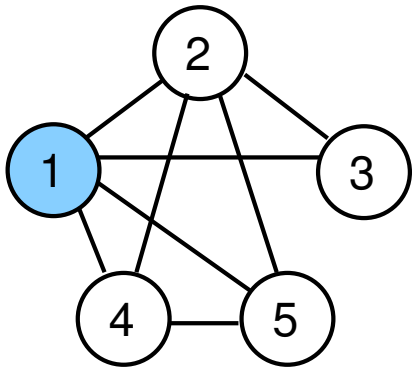




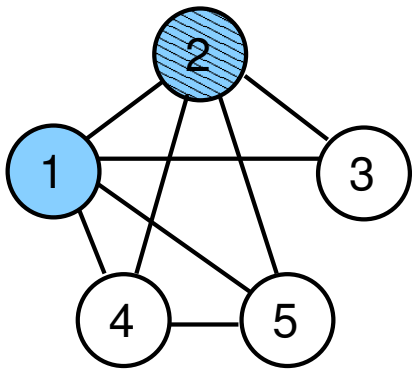
Můžeme vybrat buď úlohu 1 nebo úlohu 2

Např. vybereme 1 a obarvíme barvou 1

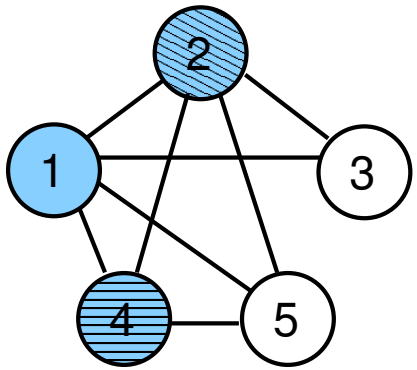
Příklad: rozvrhování schůzek (dokončení)



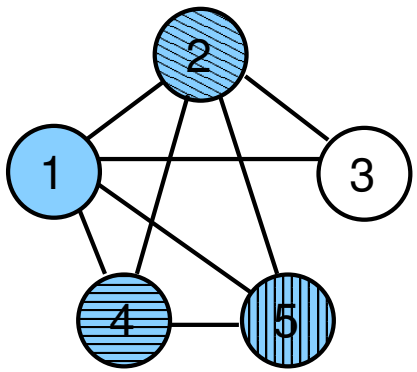
Úroveň saturace = 1 pro všechny úlohy
Vyber 2 vzhledem k nejvyššímu stupni



Úroveň saturace = 2 pro všechny uzly
Vyber 4 vzhledem k nejvyššímu stupni



Úroveň saturace = 2 pro uzel 3
Úroveň saturace = 3 pro uzel 5
Vyber 5 na obarvení



V posledním kroku obarvi 3
stejnou barvou jako 4
$\Rightarrow$celkem 4 barvy, tj. *makespan*=4

Příklad: rezervace vs. rozvrhování s operátory

Předpokládejme, že máme rezervační systém

Úloha	1	2	3
p_j	2	3	1
r_j	0	2	3
d_j	2	5	4

Lze přeformulovat na rozvrhování s operátory

Úloha	1	2	3
Operátor 1	1	0	0
Operátor 2	1	0	0
Operátor 3	0	1	0
Operátor 4	0	1	1
Operátor 5	0	1	0

úloha 1 běží v čase [0,2]

úloha 2 běží v čase [2,5]
úloha 3 běží v čase [3,4]

Vztah k rezervačním systémům

- Rozvrhování s operátory blízké rezervačnímu systému bez rezervy
- Oba problémy reformulovány na problém barvení grafu
  - rezervace: hrana = dvě úlohy se překrývají v čase
  - rozvrhování: hrana = dvě úlohy požadují stejného operátora
- Rozdíl ve složitosti
  - rezervace: překrývající se časové jednotky jdou po sobě
  - rozvrhování: úlohy často nevyžadují pouze sousední operátory

  $\Rightarrow$ rozvrhování s operátory je obtížnější problém

Rozvrhování s pracovní sílou: aplikace

- Řízení projektu ve stavebním průmyslu
  - dodavatel má realizovat n aktivit
  - doba trvání aktivity j je p_j
  - aktivita j požaduje pracovní skupinu velikosti W_j
  - precedenční omezení na aktivity
  - dodavatel má W pracovníků
  - cíl: dokončit všechny aktivity v minimálním čase
- Rozvrhování zkoušek
  - všechny zkoušky mají stejnou dobu trvání
  - všechny zkoušky jsou v místnosti s W sedadly
  - počet studentů předmětu j, kteří skládají zkoušku, je W_j
  - několik zkoušek může být narozvrhováno ve stejné místnosti, pokud je postačující počet sedadel
  - cíl: zkonstruovat rozvrh tak, že jsou všechny zkoušky narozvrhovány v minimálním čase

Reformulace pomocí problému plnění košů *(bin-packing)*

- Problém plnění košů
  - každý koš má kapacitu W
  - předměty velikosti W_j
  - cíl: naplnit minimální počet košů
- Uvažujme speciální problém rozvrhování s pracovní silou
  - předpokládejme jednotkovou dobu trvání
  - nelimitovaný počet strojů
  - minimalizace *makespan*
- Rozvrhování s pracovní silou jako problém plnění košů
  - předmět = úloha (s počtem pracovníků W_j)
  - kapacita koše = celkový počet pracovníků W
  - 1 koš = 1 časová jednotka
  - minimalizace počtu košů = minimalizace makespan
- Řešení problému plnění košů
  - NP-těžký problém
  - vyvinuta řada heuristik
  - heuristika prvního padnoucího *(first fit FF)* koše

  víme, že:

$$C_{max}(FF) \leq \frac{17}{10} C_{max}(OPT) + 2$$

Příklad: heuristika prvního padnoucího koše (FF)

- Předpokládejme 18 úloh a W=2100
  - úloha 1-6 požaduje 301 jednotek zdroje
  - úloha 7-12 požaduje 701 jednotek zdroje
  - úloha 13-18 požaduje 1051 jednotek zdroje
- FF heuristika:
  - přiřadíme prvních 6 úloh na jeden zdroj (301×6=1806)
  - pak přiřadíme vždy dvě úlohy na další tři zdroje (701×2=1402)
  - pak přiřadíme právě jednu úlohu na každý zdroj
- Velmi nekvalitní řešení vzhledem k uspořádání úloh

Heuristika prvního padnoucí koše se zmenšováním úloh

- Zlepšení FF heuristiky
- Uspořádání úloh od nejdelší k nejkratší
- První padnoucí koš se zmenšováním úloh
  (first fit decreasing FFD)
- Řešení příkladu:
  - seřadíme úlohy dle požadovaných jednotek zdroje, tj.
    13,14,15,16,17,18,7,8,9,10,11,12,1,2,3,4,5,6
  - úlohy bereme postupně a dáváme je na první zdroj, kam se vejdou
    13 dáme na zdroj 1, 14 dáme na zdroj 2, ..., 18 dáme na zdroj 6
    7 dáme na zdroj 1, 8 dáme na zdroj 2, ..., 12 dáme na zdroj 6
    1 dáme na zdroj 1, 2 dáme na zdroj 2, ..., 6 dáme na zdroj 6
- Víme, že

$$C_{max}(FFD) \leq \frac{11}{9} C_{max}(OPT) + 4$$

- FF i FFD mohou být rozšířeny
  - různé termíny dostupnosti

Diskuse

- Uvažovali jsme reprezentanty různých problémů
- V praxi
  - obecnější problémy (kombinace všech uvažovaných rysů problémů zároveň)
  - dynamické rezervační problémy
  - uvažování ceny (management zisku)
  - přerušení aktivit
  - ...

Shrnutí

Rezervace

- Intervalové rozvrhování (rezervační systém bez rezervy)
  - celočíselné programování
  - jednotková doba trvání: formulace problému přiřazení (řešení pro každou čas. jednotku zvlášť)
  - jednotková váha & identické stroje (maximalizace počtu provedených aktivit)
  - jednotková váha & nelimitovaný počet identických strojů
- Rezervační systém s rezervou
  - kompozitní řídící pravidlo

Timetabling

- plánování projektu s omezenými zdroji (RCPSP)
- rozvrhování s operátory a barvení grafu
  - heuristika pro barvení grafu
- rozvrhování s pracovní silou a problém plnění košů
  - heuristika prvního padnoucího koše
  - heuristika prvního padnoucího koše se zmenšováním úloh