

# PB165 Grafy a sítě: Úvod k plánování a rozvrhování

- 1 **Problém rozvrhování:** vlastnosti stroje, omezení, optimalizace
- 2 **Precedenční omezení a disjunktivní grafová reprezentace:** korespondence mezi rozvrhem a grafem
- 3 **Plánování projektu (precedenční omezení):** kritická cesta, kompromisní heuristika
- 4 **Barvení grafu:** algoritmus saturace, problémy přiřazení místností, rezervační problém, plánování operátorů
- 5 **Plánování na počítačové síti:**
  - Transport: doba na dopravu, doba na nastavení
  - Plánování datových přenosů
  - Paralelní úlohy s komunikací
  - Plánování s komunikací a s precedencemi: plánování seznamem, heuristiky mapování, shlukovací heuristiky

Rozvrhování na FI: PA167 Rozvrhování, PA163 Omezující podmínky

# Rozvrhování a plánování (scheduling)

## ● Zdroj/stroj

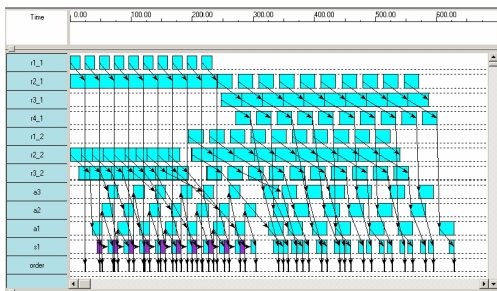
- kapacita
- dostupnost v čase
- rychlost

## ● Úloha/aktivita

- nejdřívejší startovní čas
- nejpozdější koncový čas
- doba trvání (na referenčním zdroji)
- počet zdrojů
- alternativní zdroje

## ● Rozvrhování

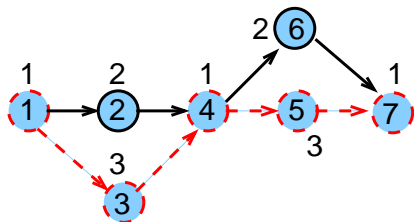
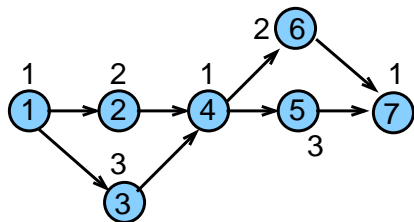
- optimální alokace/přiřazení zdrojů v čase množině úloh
  - omezené množství zdrojů
  - maximalizace zisku za daných omezení



Visopt ShopFloor System

# Příklad: rozvrhování s precedencemi

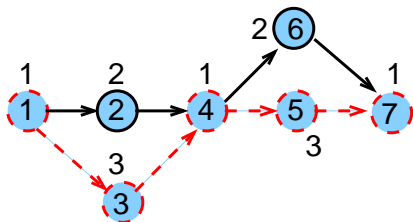
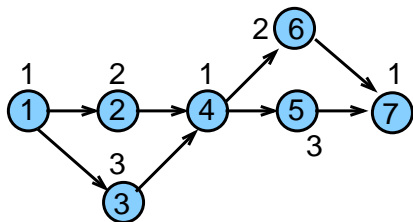
- Rozvrhování 7 úloh na 2 zdrojích
  - doba trvání úlohy + precedenční podmínky
  - nalezení rozvrhu tak, aby se minimalizovala doba nutná na realizaci všech úloh



kritická cesta

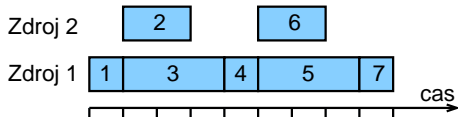
# Příklad: rozvrhování s precedencemi

- Rozvrhování 7 úloh na 2 zdrojích
  - doba trvání úlohy + precedenční podmínky
  - nalezení rozvrhu tak, aby se minimalizovala doba nutná na realizaci všech úloh



kritická cesta

- Možný rozvrh
  - na kritické (nejdelší) cestě nesmí vzniknout zdržení



- Stroje  $i = 1, \dots, m$
- Úlohy  $j = 1, \dots, n$
- $(i, j)$  operace nebo provádění úlohy  $j$  na stroji  $i$ 
  - úloha se může skládat z několika operací
  - příklad: úloha 4 má tři operace s nenulovou dobou trvání  $(2,4), (3,4), (6,4)$ , tj. je prováděna na strojích 2,3,6
- Statické parametry úlohy
  - doba trvání  $p_j$  ( $p_{ij}$ ): doba provádění úlohy  $j$  (na stroji  $i$ )
  - termín dostupnosti  $j$  (*release date*)  $r_j$ :  
nejdřívější čas, ve kterém může být úloha  $j$  prováděna
  - termín dokončení (*due date*)  $d_j$ :  
čas, do kdy musí být úloha  $j$  nejpozději dokončena
  - váha  $w_j$ :  
důležitost úlohy  $j$  relativně vzhledem k ostatním úlohám v systému

- **Stroje**  $i = 1, \dots, m$
- **Úlohy**  $j = 1, \dots, n$
- **$(i, j)$  operace** nebo provádění úlohy  $j$  na stroji  $i$ 
  - úloha se může skládat z několika operací
  - příklad: úloha 4 má tři operace s nenulovou dobou trvání  $(2,4),(3,4),(6,4)$ , tj. je prováděna na strojích 2,3,6
- **Statické parametry úlohy**
  - **doba trvání**  $p_j$  ( $p_{ij}$ ): doba provádění úlohy  $j$  (na stroji  $i$ )
  - **termín dostupnosti  $j$  (release date)**  $r_j$ :  
nejdřívejší čas, ve kterém může být úloha  $j$  prováděna
  - **termín dokončení (due date)**  $d_j$ :  
čas, do kdy musí být úloha  $j$  nejpozději dokončena
  - **váha**  $w_j$ :  
důležitost úlohy  $j$  relativně vzhledem k ostatním úlohám v systému
- **Dynamické parametry úlohy**
  - **čas startu úlohy (start time)**  $S_j$  ( $S_{ij}$ ):  
čas, kdy začne provádění úlohy  $j$  (na stroji  $i$ )
  - **čas konce úlohy (completion time)**  $C_j$  ( $C_{ij}$ ):  
čas, kdy je dokončeno provádění úlohy  $j$  (na stroji  $i$ )

## Grahamova klasifikace $\alpha|\beta|\gamma$

používá se pro popis rozvrhovacích problémů

- $\alpha$ : charakteristiky stroje
  - popisuje způsob alokace úloh na stroje
- $\beta$ : charakteristiky úloh
  - popisuje omezení aplikovaná na úlohy
- $\gamma$ : optimalizační kritéria



## Grahamova klasifikace $\alpha|\beta|\gamma$

používá se pro popis rozvrhovacích problémů

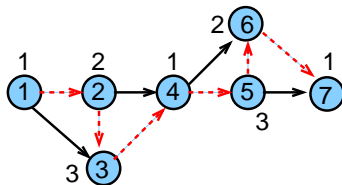
- $\alpha$ : charakteristiky stroje
  - popisuje způsob alokace úloh na stroje
- $\beta$ : charakteristiky úloh
  - popisuje omezení aplikovaná na úlohy
- $\gamma$ : optimalizační kritéria

<http://www.informatik.uni-osnabrueck.de/knust/class/>

- složitost a algoritmy pro jednotlivé rozvrhovací problémy

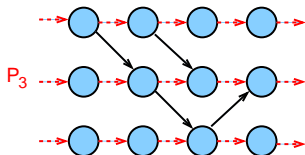
Jeden stroj 1:  $1 | \dots | \dots$

- nejjednodušší varianta
- speciální případ dalších složitějších prostředí stroje



Identické paralelní stroje  $P_m$

- $m$  identických strojů zapojených paralelně (se stejnou rychlostí)
- úloha je dána jedinou operací
- úloha může být prováděna na libovolném z  $m$  strojů



## Paralelní stroje s různou rychlostí $Q_m$

- doba trvání úlohy  $j$  na stroji  $i$  přímo závislá na jeho rychlosti  $v_i$
- $p_{ij} = p_j/v_i$
- příklad:  
několik počítačů s různou rychlostí procesoru

## Paralelní stroje s různou rychlostí $Q_m$

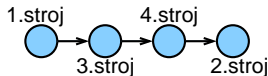
- doba trvání úlohy  $j$  na stroji  $i$  přímo závislá na jeho rychlosti  $v_i$
- $p_{ij} = p_j / v_i$
- příklad:  
několik počítačů s různou rychlostí procesoru

## Nezávislé paralelní stroje s různou rychlostí $R_m$

- stroje mají různou rychlost pro různé úlohy
- stroj  $i$  zpracovává úlohu  $j$  rychlostí  $v_{ij}$
- $p_{ij} = p_j / v_{ij}$
- příklad:  
vektorový počítač počítá vektorové úlohy rychleji než klasické PC

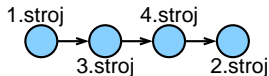
Multi-operační (*shop*) problémy

- jedna úloha je prováděna postupně na několika strojích
  - úloha  $j$  se skládá z několika operací  $(i, j)$
  - operace  $(i, j)$  úlohy  $j$  je prováděna na stroji  $i$  po dobu  $p_{ij}$
  - příklad: úloha  $j$  se 4 operacemi  $(1, j), (2, j), (3, j), (4, j)$



Multi-operační (*shop*) problémy

- jedna úloha je prováděna postupně na několika strojích
  - úloha  $j$  se skládá z několika operací  $(i, j)$
  - operace  $(i, j)$  úlohy  $j$  je prováděna na stroji  $i$  po dobu  $p_{ij}$
  - příklad: úloha  $j$  se 4 operacemi  $(1, j), (2, j), (3, j), (4, j)$

Open shop  $O_m$ 

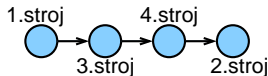
- multi-operační problém s  $m$  stroji (žádné nové vlastnosti)

Job shop  $J_m$ 

- multi-operační problém s  $m$  stroji
- pořadí provádění operací pro každou úlohu je předem určeno
- $(i, j) \rightarrow (k, j)$  určuje, že úloha  $j$  má být prováděna na stroji  $i$  dříve než na stroji  $k$   
příklad:  $(2, j) \rightarrow (1, j) \rightarrow (3, j) \rightarrow (4, j)$

Multi-operační (*shop*) problémy

- jedna úloha je prováděna postupně na několika strojích
  - úloha  $j$  se skládá z několika operací  $(i, j)$
  - operace  $(i, j)$  úlohy  $j$  je prováděna na stroji  $i$  po dobu  $p_{ij}$
  - příklad: úloha  $j$  se 4 operacemi  $(1, j), (2, j), (3, j), (4, j)$

Open shop  $O_m$ 

- multi-operační problém s  $m$  stroji (žádné nové vlastnosti)

Job shop  $J_m$ 

- multi-operační problém s  $m$  stroji
- pořadí provádění operací pro každou úlohu je předem určeno
- $(i, j) \rightarrow (k, j)$  určuje, že úloha  $j$  má být prováděna na stroji  $i$  dříve než na stroji  $k$   
příklad:  $(2, j) \rightarrow (1, j) \rightarrow (3, j) \rightarrow (4, j)$

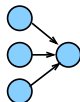
Multi-operační problémy = klasické detailně studované problémy  
operačního výzkumu

- reálné problémy mnohem komplikovanější
- metody řešení lze použít jako základ pro řešení složitějších problémů
- př. automobilová výrobní linka

## Precedenční podmínky

*prec*

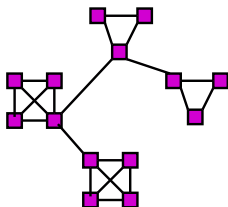
- úloha může být prováděna až po skončení další(ch) úloh
- pro úlohy  $a, b$  píšeme  $a \rightarrow b$ , což znamená  $S_a + p_a \leq S_b$



## Vhodnost stroje

*M<sub>j</sub>*

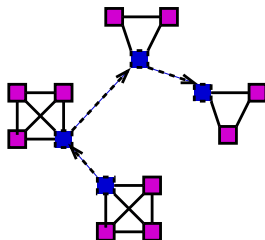
- podmnožina strojů  $M_j$ , na níž lze provádět úlohu  $j$
- př. úloha může být prováděna pouze na těch strojích v počítačové síti, kde jsou dostupná data





## Směrovací (*routing*) omezení

- udávají, na kterých strojích musí být úloha prováděna
- vazba na směrování v počítačových sítích
- 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

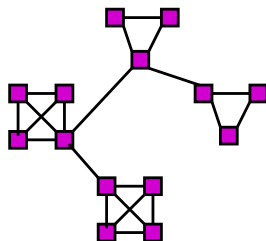


## Doprava a komunikace

 $t_{jkl}, t_{kl}, t_j$ 

- doba nutná na přepravu úlohy  $j$  mezi dvěma zařízeními  $k$  a  $l$ 
  - $t_{jkl}$  doba na přepravu ze stroje  $k$  na stroj  $l$  pro úlohu  $j$
  - $t_{kl}$  doba nezávislá na úloze
  - $t_j$  doba nezávislá na strojích
- omezení na přepravované/přenášené množství a možnou dobu přepravy
- omezení na propustnost (kapacitu) hrany/linky
- omezení na vzdálenost uzlů pro přepravu/přenos

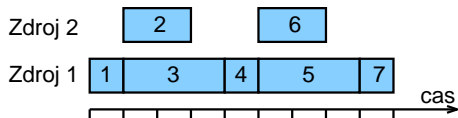
$t_{kl}$  dáno vzdáleností uzlů v síti/grafu:



- **Makespan**: maximální čas konce úloh

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

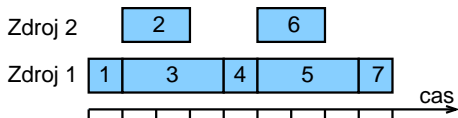
- Příklad:  $C_{\max} = \max\{1, 3, 4, 5, 8, 7, 9\} = 9$



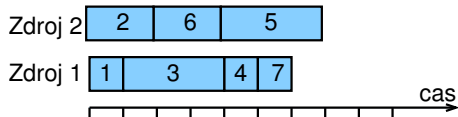
- **Makespan**: maximální čas konce úloh

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

- 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:  $C_{\max} = \max\{1, 2, 4, 5, 7, 4, 6\} = 7$

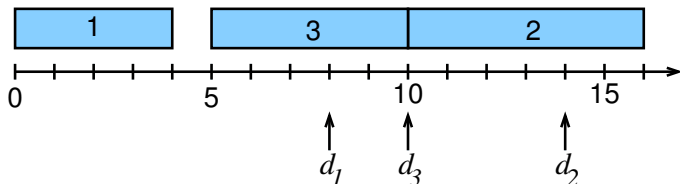


- Velmi často používané základní kritérium

- Zpoždění (*lateness*) úlohy  $j$ :  $L_j = C_j - d_j$
- Maximální zpoždění

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

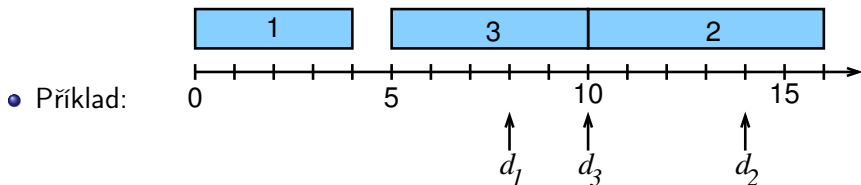
- Cíl: minimalizace maximálního zpoždění
- Příklad:



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

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

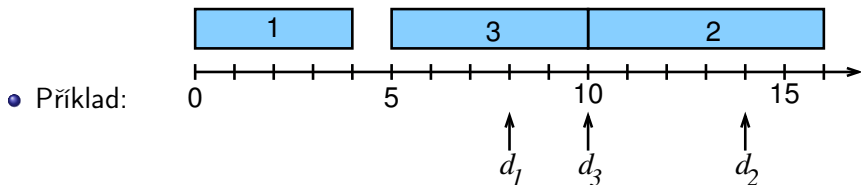
$$\sum_{j=1}^n T_j \quad \text{celkové zpoždění}$$



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

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

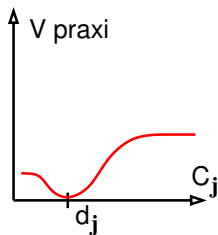
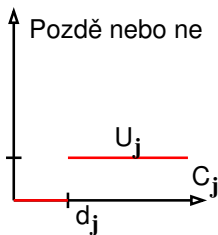
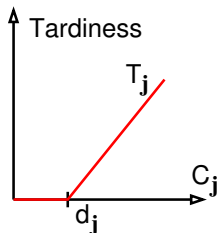
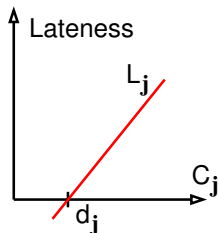
$$\sum_{j=1}^n T_j \quad \text{celkové zpoždění}$$



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

- Cíl: **minimalizace celkového váženého zpoždění**

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





Vypočítejte makespan, maximální zpoždění (*lateness*) a celkové zpoždění (*tardiness*) a celkové vážené zpoždění pro daný rozvrh.

Úloha	1	2	3	4	5	6
Čas startu úlohy	0	10	1	12	5	8
Váha úlohy	1	2	1	1	2	1
Doba trvání	1	2	4	1	3	2
Termín dokončení	4	13	5	14	7	8

Uveďte odpovídající značení i vztahy pro jednotlivá optimalizační kritéria.

# Příklady rozvrhovacích problému s Grahamovou klasifikací

- $1|prec|C_{max}$ 
  - plánování úloh provázaných precedencemi na jednom stroji s cílem minimalizovat makespan

# Příklady rozvrhovacích problému s Grahamovou klasifikací

- $1|prec|C_{\max}$ 
  - plánování úloh provázaných precedencemi na jednom stroji s cílem minimalizovat makespan
- $Pm|r_j, M_j|\sum w_j T_j$ 
  - systém s  $m$  stroji zapojenými paralelně, kde
    - úloha  $j$  přijde v čase  $r_j$  a má být naplánována do času  $d_j$
    - úloha  $j$  může být naplánována pouze na podmnožině strojů dané  $M_j$  a pokud není úloha  $j$  zpracována včas, tak je penalizována  $w_j T_j$

# Příklady rozvrhovacích problému s Grahamovou klasifikací

- $1|prec|C_{\max}$ 
  - plánování úloh provázaných precedencemi na jednom stroji s cílem minimalizovat makespan
- $Pm|r_j, M_j|\sum w_j T_j$ 
  - systém s  $m$  stroji zapojenými paralelně, kde
    - úloha  $j$  přijde v čase  $r_j$  a má být naplánována do času  $d_j$
    - úloha  $j$  může být naplánována pouze na podmnožině strojů dané  $M_j$  a pokud není úloha  $j$  zpracována včas, tak je penalizována  $w_j T_j$
- $Jm||C_{\max}$ 
  - job shop problém, kde je cílem minimalizovat makespan
  - velmi často studovaný a velmi známý typ job-shop problému

# Příklady rozvrhovacích problému s Grahamovou klasifikací

- $1|prec|C_{max}$ 
  - plánování úloh provázaných precedencemi na jednom stroji s cílem minimalizovat makespan
- $Pm|r_j, M_j|\sum w_j T_j$ 
  - systém s  $m$  stroji zapojenými paralelně, kde
    - úloha  $j$  přijde v čase  $r_j$  a má být naplánována do času  $d_j$
    - úloha  $j$  může být naplánována pouze na podmnožině strojů dané  $M_j$  a pokud není úloha  $j$  zpracována včas, tak je penalizována  $w_j T_j$
- $Jm||C_{max}$ 
  - job shop problém, kde je cílem minimalizovat makespan
  - velmi často studovaný a velmi známý typ job-shop problému
- $P_\infty|prec|C_{max}$ 
  - problém s neomezeným počtem strojů zapojených paralelně, kde jsou úlohy provázány precedenčními podmínkami a kde je cílem minimalizovat makespan
  - klasický problém plánování projektu

**Cvičení:** navrhňte různé rozvrhovací problémy pomocí Grahamovy klasifikace a popište je

# Úvod k plánování a rozvrhování

## 1 Terminologie a klasifikace

- Úvod
- Vlastnosti stroje
- Omezení
- Optimalizace
- Příklady

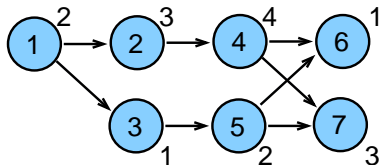
## 2 Grafová reprezentace pro:

- Precedenční omezení
- Disjunktivní grafová reprezentace

# Precedenční omezení

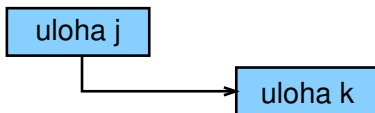
- Úloha může být prováděna až po skončení další(ch) úloh
  - úloha  $a$  před úlohou  $b$ :  $a \rightarrow b : S_a + p_a \leq S_b$
- **Orientovaný acyklický vrcholově ohodnocený graf**
  - uzly reprezentují úlohy
  - hrany reprezentují precedenční podmínky
  - ohodnocení vrcholu reprezentuje dobu trvání
  - graf bez cyklů (pro cyklický graf neexistuje žádné řešení)

Úloha	Doba trvání	Předchůdci
1	2	–
2	3	1
3	1	1
4	4	2
5	2	3
6	1	4,5
7	3	4,5

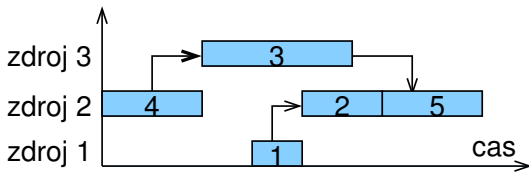


# Úloha jako obdélník

- Úloha jako uzel lze převést na **úloha jako obdelník**



- Horizontální strany obdelníku použity jako časové osy odpovídající době provádění úlohy





- Příklad: zprostředkování, instalace a testování rozsáhlého počítačového systému
  - projekt zahrnuje
    - evaluace a výběr hardware, vývoj software, nábor a školení lidí, testování a ladění systému, ...
  - precedenční vztahy
    - některé úlohy mohou být prováděny paralelně
    - úloha musí být realizována až po dokončení jiných úloh
  - cíl: minimalizovat cenu a čas na realizaci celého projektu
    - čas na realizaci projektu (makespan) lze promítnout do ceny projektu

- Příklad: zprostředkování, instalace a testování rozsáhlého počítačového systému
  - projekt zahrnuje
    - evaluace a výběr hardware, vývoj software, nábor a školení lidí, testování a ladění systému, ...
  - precedenční vztahy
    - některé úlohy mohou být prováděny paralelně
    - úloha musí být realizována až po dokončení jiných úloh
  - cíl: minimalizovat cenu a čas na realizaci celého projektu
    - čas na realizaci projektu (makespan) lze promítnout do ceny projektu
- Obecně: problémy plánování projektu
  - příští přednáška
- Rozšíření: plánování workflows
  - 1 orientovaný acyklický graf pro provádění úloh na počítačové síti
  - 2 obecné rozšíření: cyklické grafy + podmínky vyhodnocení cyklů

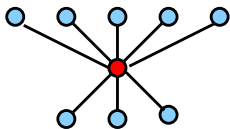
# Disjunktivní grafová reprezentace a multi-operační rozvrhování

- $n$  úloh
- $m$  strojů
- Jedna úloha je prováděna postupně na několika strojích
- Operace  $(i, j)$ : provádění úlohy  $j$  na stroji  $i$
- $p_{ij}$ : trvání operace  $(i, j)$
- Pořadí operací úlohy je předem stanoveno:
  - $(i, j) \rightarrow (k, j)$  specifikuje, že úloha  $j$  má být prováděna na stroji  $i$  dříve než na stroji  $k$
- Cíl: rozvrhovat úlohy na strojích
  - bez překrytí na strojích
  - bez překrytí v rámci úlohy
  - minimalizace *makespan*  $C_{max}$

tedy jedná se o job shop problém s minimalizací makespan  $Jm||C_{max}$

- Rozdílné typy aut na montážní lince
  - dvou-dveřové kupé, čtyř-dveřový sedan, ...
  - rozdílné barvy
  - rozdílné vybavení: automatická vs. manuální převodovka, posuvná střech, ...

- **Kritická místa** (*bottlenecks*)



- výkon stroje ovlivňuje tempo výroby
  - např. lakování (změna barvy vyžaduje časově náročné čištění)
- Cíl
  - maximalizace výkonnosti vhodným seřazením automobilů,
  - rovnoměrná pracovní zátěž na jednotlivých výrobních místech

# Příklad: job shop problém

Data:

- stroje:  $M1, M2, M3$
- úlohy:  $J1 : (3, 1) \rightarrow (2, 1) \rightarrow (1, 1)$   
 $J2 : (1, 2) \rightarrow (3, 2)$   
 $J3 : (2, 3) \rightarrow (1, 3) \rightarrow (3, 3)$



- doby trvání:  $p_{31} = 4, p_{21} = 2, p_{11} = 1$   
 $p_{12} = 3, p_{32} = 3$   
 $p_{23} = 2, p_{13} = 4, p_{33} = 1$

# Příklad: job shop problém

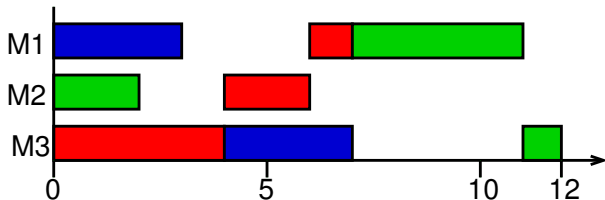
Data:

- stroje:  $M1, M2, M3$
- úlohy:  $J1 : (3, 1) \rightarrow (2, 1) \rightarrow (1, 1)$   
 $J2 : (1, 2) \rightarrow (3, 2)$   
 $J3 : (2, 3) \rightarrow (1, 3) \rightarrow (3, 3)$



- doby trvání:  $p_{31} = 4, p_{21} = 2, p_{11} = 1$   
 $p_{12} = 3, p_{32} = 3$   
 $p_{23} = 2, p_{13} = 4, p_{33} = 1$

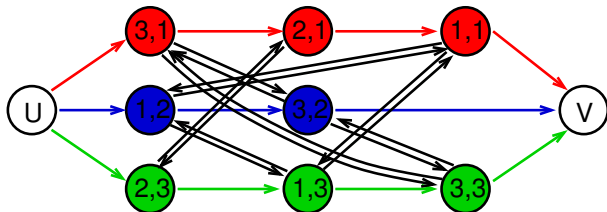
Řešení:



# Disjunktivní grafová reprezentace

Graf  $G = (N, A \cup B \cup P)$

- uzly odpovídají operacím  $N = \{(i, j) | (i, j) \text{ je operace}\} \cup \{U, V\}$
- dva pomocné uzly  $U$  a  $V$  reprezentující zdroj a stok
- **konjunktivní hrany**  $A$  reprezentují pořadí operací úlohy
  - $(i, j) \rightarrow (k, j) \in A \iff$  operace  $(i, j)$  předchází  $(k, j)$
- **disjunktivní hrany**  $B$  reprezentují konflikty na strojích
  - dvě operace  $(i, j)$  a  $(i, l)$  jsou spojeny dvěma opačně orientovanými hranami
- **pomocné hrany**  $P$ 
  - hrany z  $U$  ke všem prvním operacím úlohy
  - hrany ze všech posledních operací úlohy do  $V$



## Pojmy:

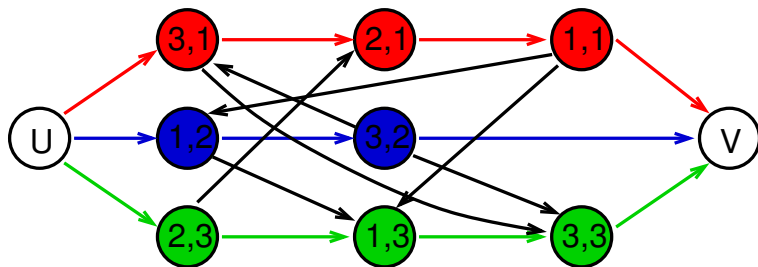
- Podmnožina  $D \subset B$  je nazývána **výběr**, jestliže obsahuje z každého páru disjunktivních hran právě jednu
- Výběr  $D$  je **splnitelný**, jestliže výsledný orientovaný graf  $G(D) = (N, A \cup D \cup P)$  je acyklický
  - jedná se o graf s konjunktivními hranami a vybranými diskjunktivními hranami

## Poznámky:

- splnitelný výběr určuje posloupnost, ve které jsou operace prováděny na strojích
- každý (konzistentní) rozvrh jednoznačně určuje splnitelný výběr
- každý splnitelný výběr jednoznačně určuje (konzistentní) rozvrh

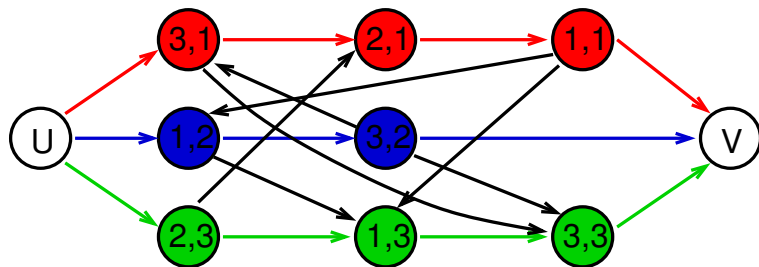


# Příklad: **nesplnitelný** výběr



V grafu existuje v důsledku nevhodného výběru hran cyklus:

## Příklad: *nesplnitelný* výběr



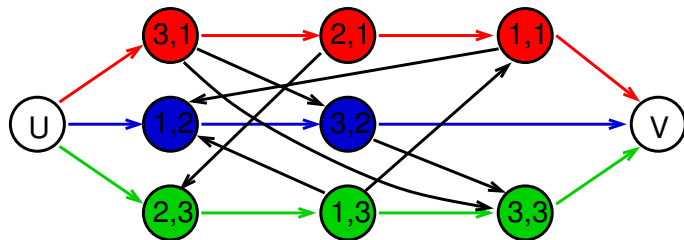
V grafu existuje v důsledku nevhodného výběru hran cyklus:

- $(1, 2) \rightarrow (3, 2)$
- $(3, 2) \rightarrow (3, 1) \rightarrow (2, 1) \rightarrow (1, 1) \rightarrow (1, 2)$

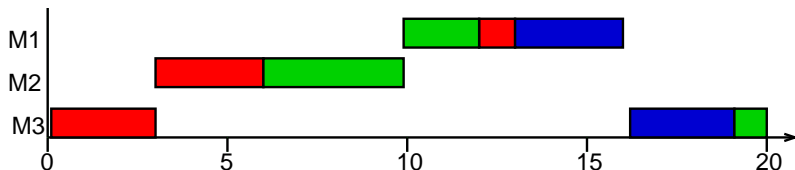
$\Rightarrow$  nelze splnit (k tomuto výběru neexistuje rozvrh)

# Příklad: splnitelný výběr

Jakým způsobem nalézt rozvrh pro daný splnitelný výběr?



Tedy: jakým způsobem lze nalézt tento odpovídající rozvrh:



# Výpočet rozvrhu pro výběr

Metoda: výpočet nejdelších cest z  $U$  do dalších uzlů v  $G(D)$

Technický popis:

- uzly  $(i, j)$  mají ohodnocení  $p_{ij}$ , uzel  $U$  má ohodnocení 0
- délka cesty  $i_1, i_2, \dots, i_r$ : součet ohodnocení uzlů  $i_1, i_2, \dots, i_{r-1}$
- spočítej délku  $l_{ij}$  nejdelší cesty z  $U$  do  $(i, j)$  a  $V$ ,  
např. použitím Dijkstrova algoritmu

# Výpočet rozvrhu pro výběr

Metoda: **výpočet nejdelších cest** z  $U$  do dalších uzlů v  $G(D)$

Technický popis:

- uzly  $(i, j)$  mají **ohodnocení**  $p_{ij}$ , uzel  $U$  má ohodnocení 0
  - **délka cesty**  $i_1, i_2, \dots, i_r$ : součet ohodnocení uzlů  $i_1, i_2, \dots, i_{r-1}$
  - spočítej délku  $l_{ij}$  nejdelší cesty z  $U$  do  $(i, j)$  a  $V$ ,  
např. použitím Dijkstrova algoritmu
- 1  $l_U = 0$  a pro každý uzel  $(i, j)$  s jediným předchůdcem  $U$ :  $l_{ij} = 0$
  - 2 vypočítej postupně pro všechny zbývající uzly  $(i, j)$  (a pro uzel  $V$ ):

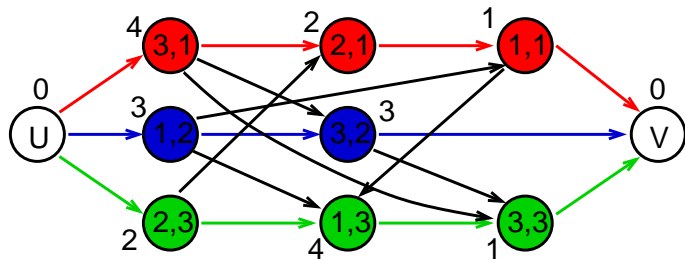
$$l_{ij} = \max_{\forall (k,l):(k,l) \rightarrow (i,j)} (l_{kl} + p_{kl})$$

tj. projdeme všechny předchůdce  $(k, l)$  uzlu  $(i, j)$

délka cesty do  $(i, j)$  přes  $(k, l)$  je  $l_{kl} + p_{kl}$

- zahaj operaci  $(i, j)$  v čase  $l_{ij}$
- délka nejdelší cesty z  $U$  do  $V$  je rovna *makespan*
  - tato cesta je kritická cesta

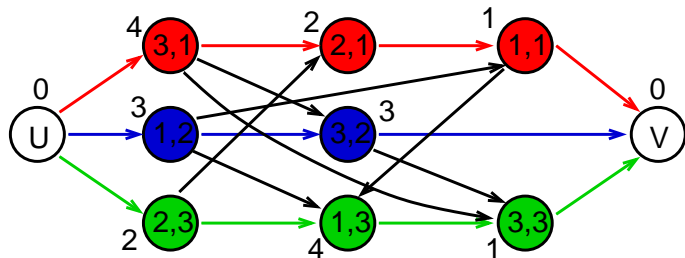
# Příklad: výpočet rozvrhu pro výběr



Výpočet  $l_{ij}$ :

uzel	(3,1)	(1,2)	(2,3)	(2,1)	(3,2)	(1,1)	(1,3)	(3,3)	V
délka									

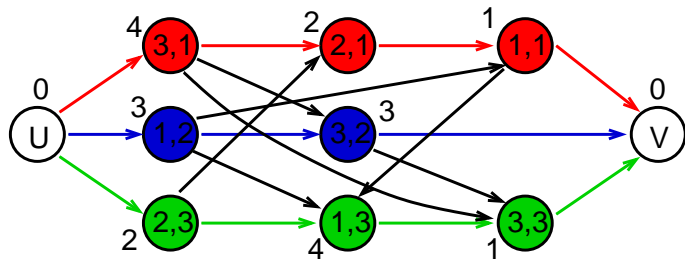
# Příklad: výpočet rozvrhu pro výběr



Výpočet  $l_{ij}$ :

uzel	(3,1)	(1,2)	(2,3)	(2,1)	(3,2)	(1,1)	(1,3)	(3,3)	V
délka	0	0	0						

# Příklad: výpočet rozvrhu pro výběr

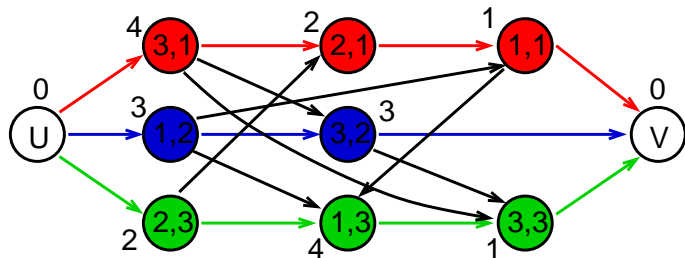


Výpočet  $l_{ij}$ :

uzel	(3,1)	(1,2)	(2,3)	(2,1)	(3,2)	(1,1)	(1,3)	(3,3)	V
délka	0	0	0	4	4				



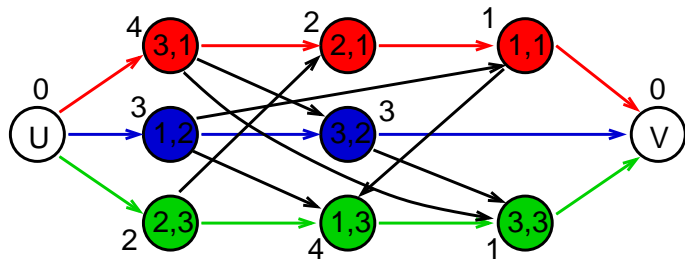
# Příklad: výpočet rozvrhu pro výběr



Výpočet  $l_{ij}$ :

uzel	(3,1)	(1,2)	(2,3)	(2,1)	(3,2)	(1,1)	(1,3)	(3,3)	V
délka	0	0	0	4	4	6			

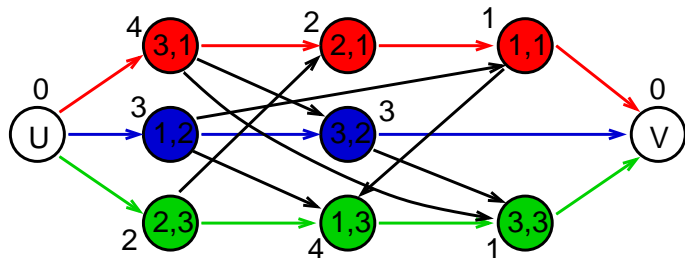
# Příklad: výpočet rozvrhu pro výběr



Výpočet  $l_{ij}$ :

uzel	(3,1)	(1,2)	(2,3)	(2,1)	(3,2)	(1,1)	(1,3)	(3,3)	V
délka	0	0	0	4	4	6	7		

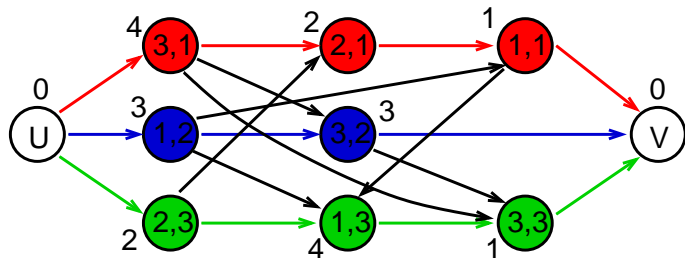
# Příklad: výpočet rozvrhu pro výběr



Výpočet  $l_{ij}$ :

uzel	(3,1)	(1,2)	(2,3)	(2,1)	(3,2)	(1,1)	(1,3)	(3,3)	V
délka	0	0	0	4	4	6	7	11	

# Příklad: výpočet rozvrhu pro výběr

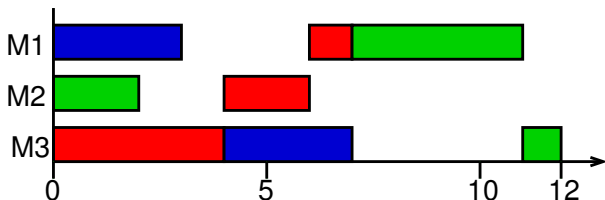


Výpočet  $l_{ij}$ :

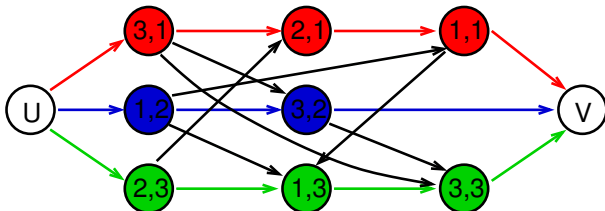
uzel	(3,1)	(1,2)	(2,3)	(2,1)	(3,2)	(1,1)	(1,3)	(3,3)	V
délka	0	0	0	4	4	6	7	11	12

# Konstrukce výběru pro daný rozvrh

Nalezněte výběr hran pro daný rozvrh:



Konstrukce odpovídajícího výběru: vybereme disjunktivní hrany, které odpovídají uspořádání operací úlohy v rozvrhu



Uveďte grafovou reprezentaci pro zadaný *job shop* problém a ukažte na něm příklad *splnitelného výběru* a *nesplnitelného výběru*. Pro Vámi vytvořený splnitelný výběr nalezněte odpovídající rozvrh.

Zadání problému je následující:

- máme 3 stroje
- máme 3 úlohy s následujícími precedencemi:

$$J1: (2, 1) \rightarrow (1, 1) \rightarrow (3, 1)$$

$$J2: (3, 2) \rightarrow (2, 2) \rightarrow (1, 2)$$

$$J3: (1, 3) \rightarrow (3, 3),$$

- doby trvání operací jsou:

$$p_{21} = 2, p_{11} = 4, p_{31} = 1,$$

$$p_{32} = 4, p_{22} = 2, p_{12} = 1,$$

$$p_{13} = 3, p_{33} = 3.$$