Úprava přednášky byla podpořena projektem CZ.2.17/3.1.00/33274, který je financován Evropským sociálním fondem a rozpočtem hlavního města Prahy.

Evropský sociální fond Praha & EU: Investujeme do vaší budoucnosti

Plánování a rozvrhování

11

Roman Barták, KTIML

roman.bartak@mff.cuni.cz
http://ktiml.mff.cuni.cz/~bartak

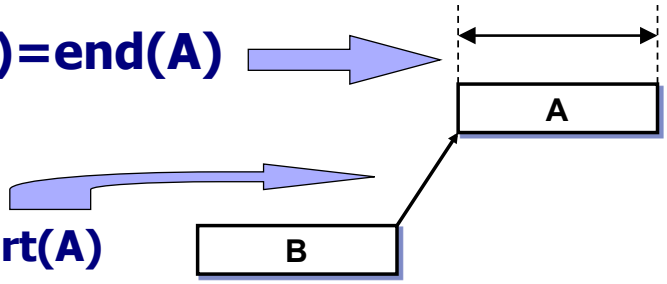
Rozvrhování jako CSP

- **Rozvrhovací problém** je statický, takže může být přímo **zakódován jako CSP**.
- Splňování podmínek se používá pro **kompletní rozvrhování**.

CSP model:

- **proměnné**
  - pozice aktivity A v čase a prostoru
  - alokace času: **start(A), [p(A), end(A)]**
  - alokace zdrojů: **resource(A)**
- **domény**
  - **termín dostupnosti** a **uzávěrka** pro časové proměnné
  - **alternativní zdroje** pro zdrojové proměnné
- **podmínky**
  - uspořádání aktivit a omezené kapacity zdrojů

- **časové vztahy**
  - **start(A)+p(A)=end(A)**
  - uspořádání
    - B « A
    - ↳ **end(B) ≤ start(A)**



- **omezení kapacity zdrojů**
  - unární zdroj (aktivity se nemohou překrývat)
    - A « B ∨ B « A
    - ↳ **end(A) ≤ start(B) ∨ end(B) ≤ start(A)**

Podmínky pro zdroje

Zdroje

- **Zdroje jsou v plánování a rozvrhování používány v trochu jiných významech!**
- **rozvrhování**
  - zdroj
    = **stroj** (prostor) pro zpracování aktivity
- **plánování**
  - zdroj
    = **materiál** konzumovaný/produkovaný danou aktivitou
  - unární zdroj z pohledu rozvrhování je často modelován jako logická podmínka (např. ruka je volná)

Typy zdrojů

- **unární zdroje**
  - v daném čase může být zpracována maximálně jedna aktivita
- **kumulativní zdroje**
  - několik aktivit se může zpracovávat paralelně, ovšem za předpokladu, že není překročena kapacita zdroje
- **produkovatelné/spotřebovatelné zdroje**
  - aktivita konzumuje/produkuje nějakou kapacitu zdroje
  - na zdroji musí zbýt nějaká minimální volná kapacita (konzumace) a maximální kapacita zdroje nemůže být překročena (produkce)

Unární zdroje

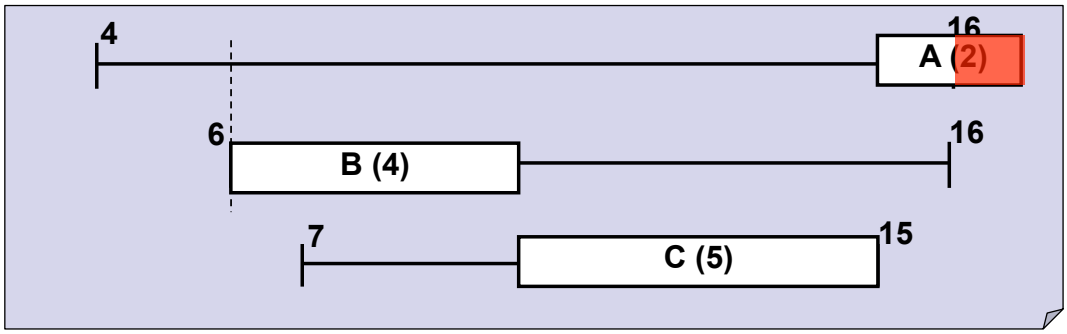
- **Aktivity se nemohou překrývat!**
  - v daném čase běží maximálně jedna aktivita, proto se těmto zdrojům říká **unární**
- Předpokládáme, že aktivity jsou **nepřerušitelné**.
  - nepřerušitelná aktivita zabírá zdroj od svého startu až do ukončení
- Jednoduchý model s disjunktivními podmínkami:
  - **A « B ∨ B « A**

    end(A) ≤ start(B) ∨ end(B) ≤ start(A)
  - těmto zdrojům se proto někdy říká **disjunktivní**

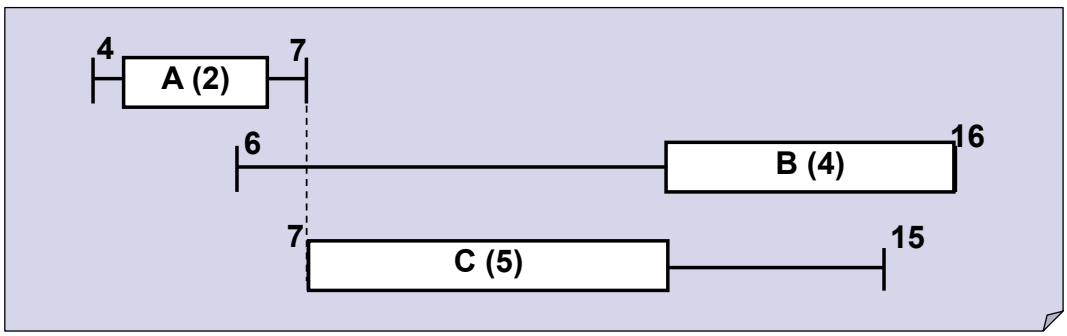
Baptiste & Le Pape (1996)

Edge finding

Co se stane, pokud aktivita A nebude zpracována jako první?



Pro A, B, C není dost času, a tedy aktivita A musí být první!



Edge finding
pravidla

Pravidla:

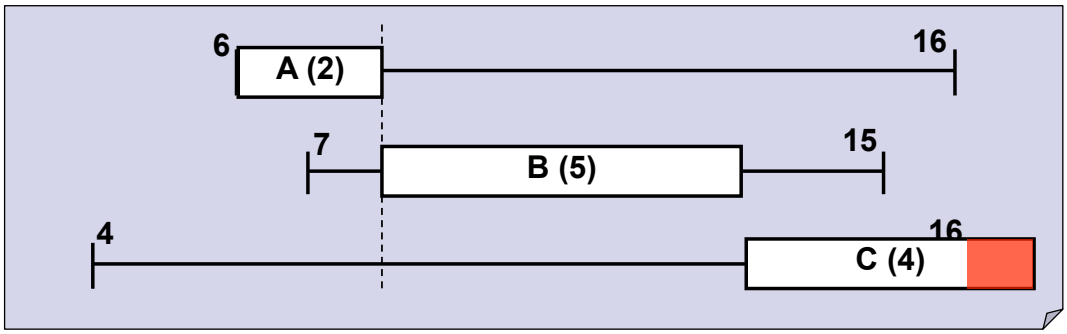
- **p(Ω ∪ {A}) > lct(Ω ∪ {A}) - est(Ω) ⇒ A«Ω**
- **p(Ω ∪ {A}) > lct(Ω) - est(Ω ∪ {A}) ⇒ Ω«A**
- **A«Ω ⇒ end(A) ≤ min{ lct(Ω') - p(Ω') | Ω'⊆Ω }**
- **Ω«A ⇒ start(A) ≥ max{ est(Ω') + p(Ω') | Ω'⊆Ω }**

V praxi:

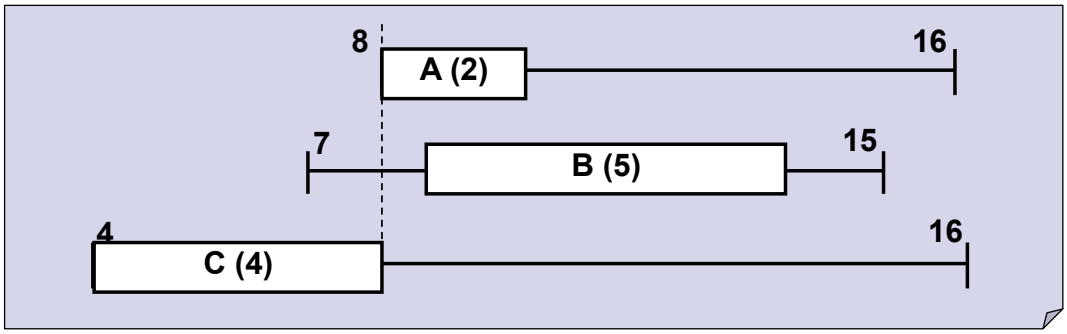
- je potřeba prozkoumat n.2^n dvojic (A,Ω) (to je moc!)
- místo Ω můžeme používat **intervaly úloh** [X,Y]
  {C | est(X) ≤ est(C) ∧ lct(C) ≤ lct(Y)}
  - ↳ časová složitost $O(n^3)$, často používaný inkrementální algoritmus
- existují také algoritmy se složitostí $O(n^2)$ a O(n.log n)

Not-first/not-last

Co se stane, pokud aktivita A bude zpracována jako první?



Pro B a C není dost času, a tedy aktivita A nemůže být první!



Not-first/not-last
pravidla

Pravidla not-first:

$p(\Omega \cup \{A\}) > lct(\Omega) - est(A) \Rightarrow \neg\ A \ll \Omega$

$\neg\ A \ll \Omega \Rightarrow start(A) \geq \min\{\ ect(B)\ |\ B \in \Omega\ \}$

(Symetrická) pravidla not-last:

$p(\Omega \cup \{A\}) > lct(A) - est(\Omega) \Rightarrow \neg\ \Omega \ll A$

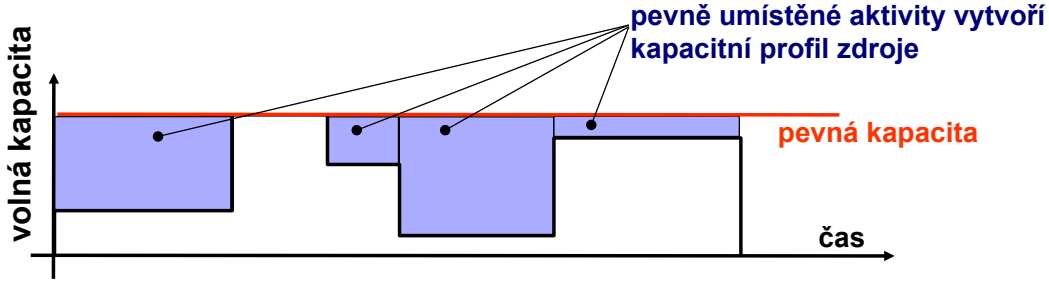
$\neg\ \Omega \ll A \Rightarrow end(A) \leq \max\{\ lst(B)\ |\ B \in \Omega\ \}$

V praxi:

- Může být implementováno s časovou složitostí $O(n^2)$ a paměťovou složitostí $O(n)$

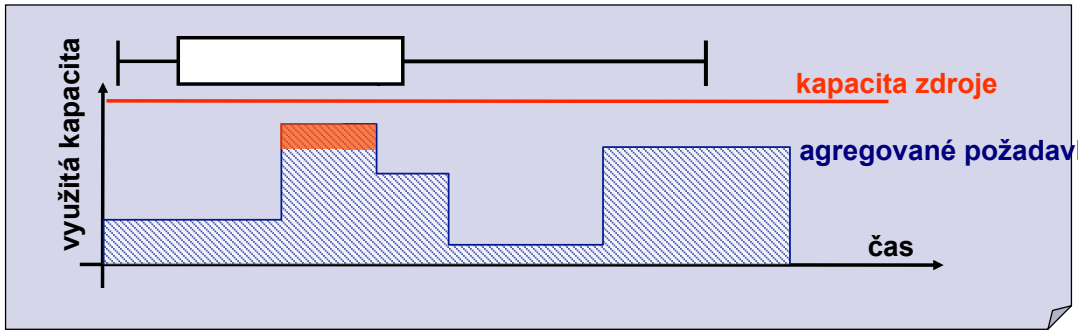
Kumulativní zdroje

- Každá **aktivita využívá jistou kapacitu** zdroje – **cap(A)**.
- Aktivity mohou být **zpracovávány paralelně**, pokud není překročena kapacita zdroje.
- Kapacita zdroje **může být v čase proměnná!**
  - takové zdroje lze modelovat pomocí v čase neměnné kapacity, od které se odečte kapacita pevně umístěných aktivit, čímž se v každém čase dosáhne požadované kapacity

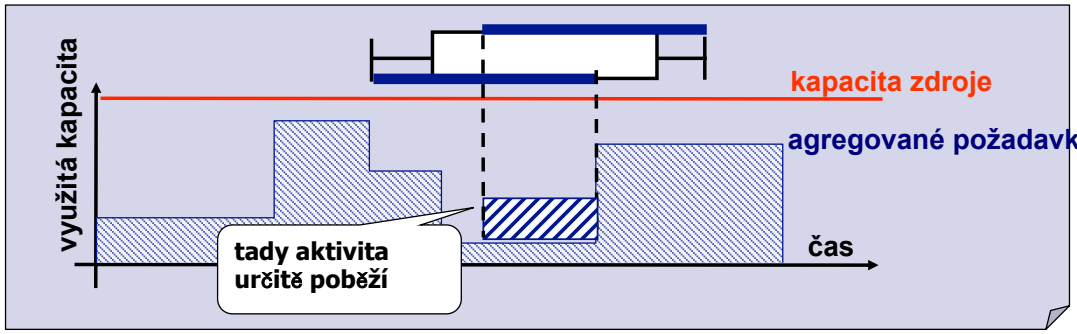


Agregované požadavky

Kde je dostatečná kapacita pro zpracování aktivity?



Jak se konstruuje graf agregovaných požadavků?



Podmínka tabulky

- Jak zajistit, že v žádném čase není překročena maximální kapacita?*

$$\forall t \sum_{start(A_i) \le t \le end(A_i)} cap(A_i) \le MaxCapacity$$

- **Tabulka** pro aktivitu A je množina Booleovských proměnných **X(A,t)** indikujících, zda A běží v čase t.

$$\forall t \sum_{A_i} X(A_i,t) \cdot cap(A_i) \le MaxCapacity$$

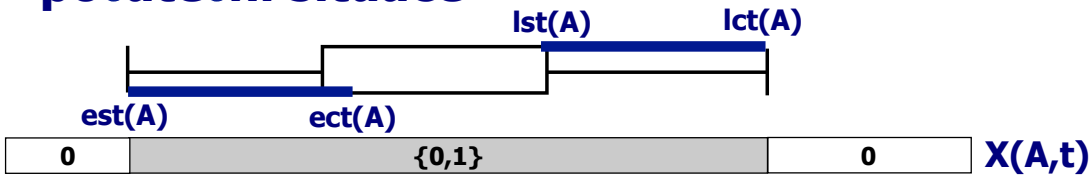
$$\forall t,i \;\; start(A_i) \le t \le end(A_i) \Leftrightarrow X(A_i,t)$$

*** předpokládáme diskrétní čas**

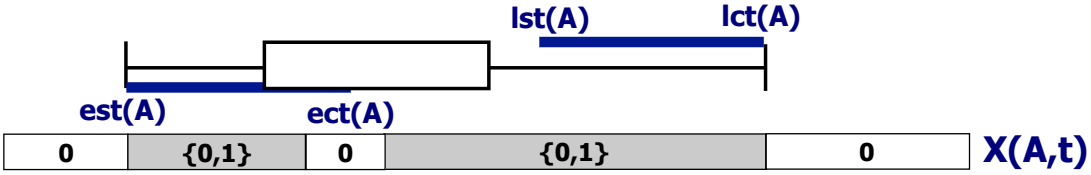
Podmínka tabulky

příklad filtrace

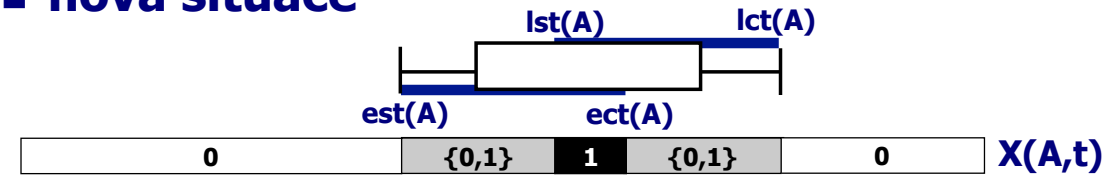
- počáteční situace



- některé pozice zakázané kvůli kapacitě



- nová situace



Baptiste et al. (2001)

Podmínka tabulky

pravidla

Jak realizovat filtraci přes podmínku

$$\forall t,i \;\; start(A_i) \le t < end(A_i) \Leftrightarrow X(A_i,t)\,?$$

Problém:

t slouží jako index i jako proměnná

start(A) ≥ min{t : ub(X(A,t))=1}

end(A) ≤ 1+max{t : ub(X(A,t))=1}

X(A,t)=0 ∧ t<ect(A) ⇒ start(A)>t

X(A,t)=0 ∧ lst(A)≤t ⇒ end(A)≤t

(lst(A)≤t ∧ t<ect(A) ⇒ X(A,t)=1)

Alternativní zdroje

- **Jak modelovat možnost použití alternativních zdrojů pro danou aktivitu?**
- Pro každý zdroj uděláme **duplikát aktivity**.
  - duplikát se účastní příslušných zdrojových podmínek, ale neomezuje další aktivity na daném zdroji
    - „neúspěch" u duplikátu znamená odstranění zdroje z domény proměnné res(A) příslušné aktivity
    - odstranění zdroje z domény proměnné res(A) znamená „smazání" odpovídajícího duplikátu
  - původní aktivita se účastní precedenčních podmínek (například v rámci úlohy)
  - omezení časů u duplikátu se propaguje do originálu aktivity a naopak

Alternativní zdroje
filtrace

- Nechť A_u je duplikát aktivity A alokovaný na zdroj $u \in res(A)$.

$$u \in res(A) \Rightarrow start(A) \leq start(A_u)$$
$$u \in res(A) \Rightarrow end(A_u) \leq end(A)$$
$$start(A) \geq \min\{start(A_u) : u \in res(A)\}$$
$$end(A) \leq \max\{end(A_u) : u \in res(A)\}$$
$$\text{prázdné okno pro } A_u \Rightarrow res(A) \setminus \{u\}$$

Ve skutečnosti se jedná o realizaci konstruktivní disjunkce mezi alternativními zdroji.

Relativní uspořádání

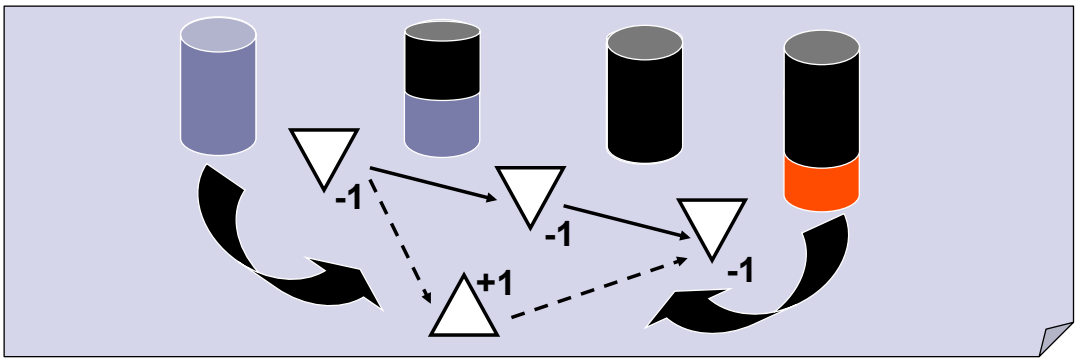
Pokud je **čas relativní** (uspořádání aktivit)

potom techniky edge-finding a agregovaných požadavků **nic neodvodí!**

Pořád ale můžeme používat informace o uspořádání aktivit a spotřebě/produkci daného zdroje!

Příklad:

Reservoár: aktivity konzumují a produkují zdroj



Cesta & Stella (1997)

Zdrojový profil

- Aktivita A „produkuje“ kvantitu **prod(A)**:
  - pozitivní číslo znamená **produkci**
  - negativní číslo znamená **spotřebu**
- **Optimistický zdrojový profil (orp)**
  - maximální možná úroveň zdroje v čase, kdy se A začne zpracovávat
  - aktivity, které **musí** být před A, se vezmou dohromady s produkčními aktivitami, které **mohou** být před A

$$orp(A) = InitLevel + prod(A) + \sum_{B \ll A} prod(B) + \sum_{B??A \ \&\ prod(B)>0} prod(B)$$

- **Pesimistický zdrojový profil (prp)**
  - minimální možná úroveň zdroje v čase, kdy se A začne zpracovávat
  - aktivity, které **musí** být před A, se vezmou dohromady s konzumačními aktivitami, které **mohou** být před A

$$prp(A) = InitLevel + prod(A) + \sum_{B \ll A} prod(B) + \sum_{B??A \ \&\ prod(B)<0} prod(B)$$

*B??A znamená, že pořadí A a B ještě není známé

Filtrace orp

- $orp(A) < MinLevel \Rightarrow fail$
  - "přestože veškerá produkce je plánována před A, pořád ještě není dosažena požadovaná minimální úroveň zdroje"

- $orp(\mathrm{A}) - prod(\mathrm{B}) - \sum_{\mathrm{B}\ll\mathrm{C}\ \&\ \mathrm{C}??\mathrm{A}\ \&\ prod(\mathrm{C})>0} prod(\mathrm{C}) < MinLevel \Rightarrow \mathrm{B} \ll \mathrm{A},$
  pro libovolné B takové, že B??A a *prod*(B)>0
  - "pokud je produkce v B plánována za A a minimální požadovaná úroveň zdroje není dosažena ani když všechny ostatní produkční aktivity jsou před A (které tam být mohou), potom B musí být před A"

Filtrace prp

- $prp(\mathrm{A}) > MaxLevel \Rightarrow \mathrm{fail}$
  - "přestože veškerá konzumace je plánována před A, je maximální úroveň zdroje (kapacita) překročena"

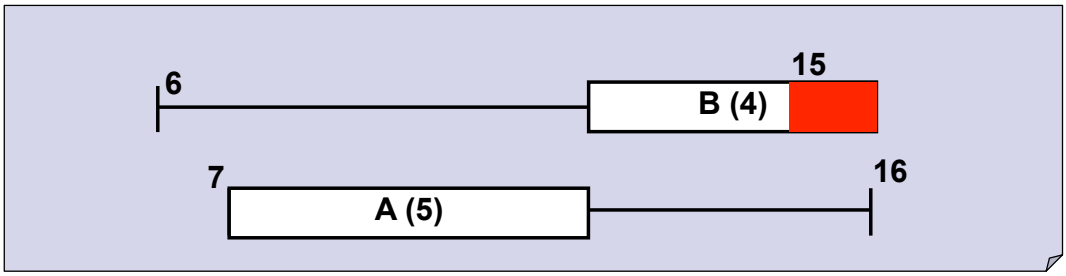
- $prp(\mathrm{A}) - prod(\mathrm{B}) - \sum_{\mathrm{B}\ll\mathrm{C}\ \&\ \mathrm{C}??\mathrm{A}\ \&\ prod(\mathrm{C})<0} prod(\mathrm{C}) > MaxLevel \Rightarrow \mathrm{B} \ll \mathrm{A},$
  pro libovolné B takové, že B??A a *prod*(B)<0
  - "pokud je konzumace v B plánována za A a maximální úroveň zdroje je překročena i když všechny ostatní konzumační aktivity jsou před A (které tam být mohou), potom B musí být před A"

Vilím (2002)

Detectable precedence

od časových oken k precedencím

Co se stane, pokud je aktivita A zpracována před B?



- Omezená časová okna mohou být použita pro detekci precedencí.
- est(A)+p(A)+p(B)>lct(B) ⇒ B«A

Laborie (2003)

Energy precedence

od precedencí k časovým oknům

Použití "energie" aktivit zpracovaných před A k odvození lepšího est(A)

start(A) ≥ max{ est(Ω') + ⌈e(Ω')/cap⌉ | Ω'⊆{C : C«A} }

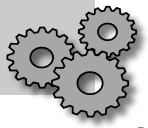
Pro unární zdroje

start(A) ≥ max{ est(Ω') + p(Ω') | Ω'⊆{C : C«A} }

stačí prozkoumat **Ω(X,A) = {Y | Y « A ∧ est(X) ≤ est(Y)}**

start(A) ≥ max{ est(Ω(X,A)) + p(Ω(X,A)) | X « A }

```
dur ← 0
end ← est(A)
for each Y∈{ X | X « A } in the non-increasing order of est(Y) do
    dur ← dur + p(Y)
    end ← max(end, est(Y) + dur)
end for
est(A) ← end
```



Optimalizace v CSP

- Účelová (kriteriální) funkce se v CSP typicky kóduje „speciální" podmínkou:

$$\mathbf{v = obj(Xs)}$$

Příklad: **makespan = max{end(A_i)}**

- je možné odvodit lepší meze pro v použitím současných domén pro proměnné Xs
  - **makespan_{min} = max{ect(A_i)}**
- Je možné omezit domény proměnnch Xs na základě aktuálních mezí proměnné v
  - **end(A_i) ≤ makespan_{max}**
- pro propagaci složitějších účelových funkcí se typicky používá relaxace problému

Rozvrhovací strategie

Schémata větvení

větvení = řešení disjunkcí

Tradiční rozvrhovací přístupy:

- **kritická rozhodnutí** se dělají **první**
  - vyřeš **úzká hrdla** …
  - definuje **tvar** prohledávacího stromu
  - vzpomeňme na princip prvního neúspěchu (**fail-first**)
- preferuj **alternativy s větší flexibilitou**
  - definuje **pořadí** větví pro prozkoumání
  - vzpomeňme princip prvního úspěchu (**succeed-first**)

Jak ale popsat co je kritické a co je flexibilní?

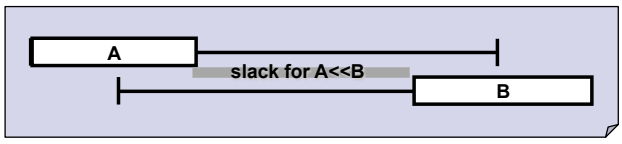
Smith and Cheng (1993)

Slack

Slack (tolerance) je formální popis flexibility.

- Slack pro **dané pořadí dvou aktivit**

  je „volný čas pro posun aktivit“



slack(A«B) = max(end(B))-min(start(A))-p({A,B})

- Slack pro **dvě aktivity** (bez určení pořadí)

  slack({A,B}) = max{slack(A«B),slack(B«A)}

- Slack pro **skupinu aktivit**

  slack(Ω) = max(end(Ω)) - min(start(Ω)) - p(Ω)

Smith and Cheng (1993)

Uspořádání větví

A«B v ¬A«B

- **Jaké aktivity mají být uspořádány první?**
  - nejkritičtější pár (fail-first)
  - pár s **minimálním slack({A,B})**
- **Jaké pořadí má být zvoleno?**
  - nejflexibilnější pořadí (succeed-first)
  - pořadí **maximalizující slack(A??B)**
- O(n^2) bodů volby

Baptiste et al. (1995)

Větvení první/poslední

(A«Ω v ¬A«Ω) nebo (Ω«A v ¬ Ω«A)

- **Máme hledat první nebo poslední aktivitu?**
  - rozhodni se podle **menší množiny** kandidátů na první resp. poslední aktivitu (first-fail)
- **Jaká aktivita má být vybrána?**
  - pokud se hledá první aktivita, potom preferuj aktivitu, která má **nejmenší min(start(A))**
  - pokud se hledá poslední aktivita, potom preferuj aktivitu, která má **největší max(end(A))**
- O(n) bodů volby

Zdrojový slack

- **Zdrojový slack** je definovaný jako slack množiny aktivit zpracovávaných daným zdrojem.
- **Jak používat zdrojový slack?**
  - volíme zdroj, na kterém budou **aktivity uspořádány jako první**
    - zdroj s minimálním slackem (**úzké hrdlo**)
  - volíme zdroj, na který **alokovat danou aktivitu**
    - zdroj s maximálním slackem (**flexibilita**)