Omezující podmínky (pokračování)

14. března 2022

1. Problém splňování podmínek
2. Rozvrhování jako problém splňování podmínek
3. Podmínky pro zdroje

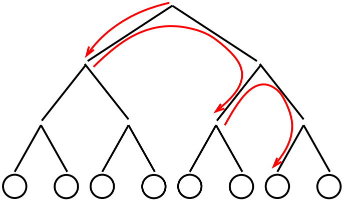
Prohledávání/přiřazování

Konzistenční techniky jsou (obvykle) neúplné
$\Rightarrow$ potřeba prohledávací algoritmus, který vyřeší "zbytek"

- př. X in 1..2, Y in 1..2, Z in 1..2, X$\neq$Y, Y$\neq$Z, X$\neq$Z
  AC neodstraní žádnou hodnotu ale problem je nekonzistentní

Přiřazovaní *(labeling)*

- prohledávání do hloubky (DFS/BT)
  - přiřaď hodnotu do proměnné
  - propaguj = udělěj
    problém lokálně konzistentní
  - vrať se v případě neúspěchu



- X in 1..5 $\equiv$ X=1 $\vee$ X=2 $\vee$ X=3 $\vee$ X=4 $\vee$ X=5

Obecně: prohledávací algoritmus řeší zbylé disjunkce

- X=1 $\vee$ X$\neq$1 standardní přiřazování
- X<3 $\vee$ X$\geq$3 dělení domén
- X<Y $\vee$ X$\geq$Y uspořádání proměnných

Kostra ohodnocování

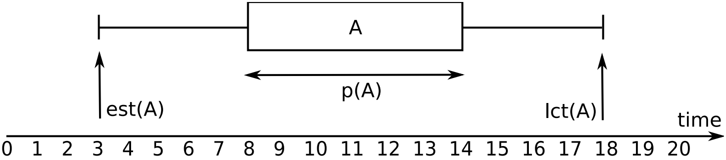
- Prohledávání do hloubky je kombinováno s AC, které omezuje prohledávaný prostor
- Technika pohledu dopředu (MAC)
- procedure labeling(V,D,C)
  if (všechny proměnné z V přiřazeny) then return V
  vyber dosud nepřiřazenou proměnnou x z V
  for (každou hodnotu v z D_x) do
      (TestOk,D') := consistent(V,D,C $\cup$ {x=v})
      if TestOk=true then R := labeling(V,D',C)
      if R$\neq$fail then return R
  return fail
  end labeling
- Před labeling je spuštěn iniciální běh konzistenčních algoritmů, aby byla zajištěna iniciální konzistence

Rozvrhování jako CSP: čas

Aktivita A: entita vyžadující prostor (zdroje) a čas

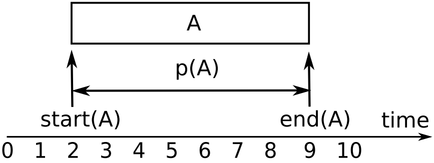
Proměnné a jejich domény pro časové přiřazení aktivity

- start(A): startovní čas aktivity
  - aktivita nemůže začít před svým termínem dostupnosti
  - est(A) = min(start(A)), earliest start time/nejdřívější startovní čas
- end(A): čas skončení aktivity
  - aktivita musí skončit před svým deadline
  - lct(A) = max(end(A)), latest completion time/nejpozdější koncový čas
- p(A): doba provádění aktivity
  - start(A) = {est(A), . . . , (lct(A)-p(A))}
  - end(A) = {(est(A)+p(A)), . . . , lct(A)}

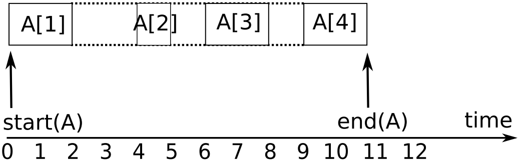


Rozvrhování jako CSP: základní omezení I.

- Nepřerušitelná aktivita: žádné přerušení během jejího zpracování
  - start(A) + p(A) = end(A)



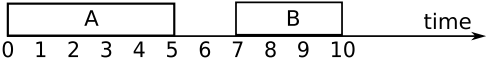
- Přerušitelná aktivita: může být přerušena během svého zpracování
  - start(A) + p(A) $\leq$ end(A)



p(A) = p(A[1]) + p(A[2]) + p(A[3]) + p(A[4])

Rozvrhování jako CSP: základní omezení II.

- Seřazení A<<B aktivity A,B
  (také: precedenční omezení mezi aktivitami A,B)
  - end(A) $\leq$ start(B)



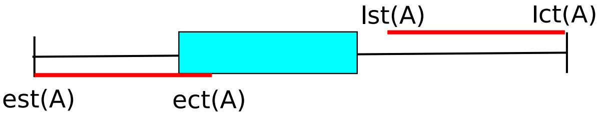
- Disjunktivní omezení: nepřekrývání aktivit A a B
  - nepřerušitelné aktivity
  - A<<B or B<<A
  - end(A) $\leq$ start(B) or end(B) $\leq$ start(A)
  - viz dále unární zdroje

Rozvrhování jako CSP: zdroje

Doménové proměnné pro zdroje:

- cap(A): požadovaná kapacita zdroje aktivitou A
  - unární/disjunktivní 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 (snižuje) nebo produkuje (navyšuje) aktuální množství zdroje
    - na zdroji musí zbýt nějaká minimální volná kapacita a maximální kapacita zdroje nemůže být překročena
    - příklad: reservoár
- resource(A): alternativní zdroje pro A
  - pro zpracování A vybereme jeden z alternativních zdrojů
  - jednotlivé hodnoty z domény resource(A) odkazují na konkrétní zdroje

Značení

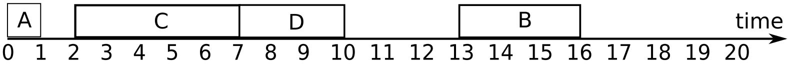


- est(A) nejdřívější startovní čas aktivity A (earliest start time)
- ect(A) nejdřívější koncový čas aktivity A (earliest completion time)
- lst(A) nejpozdější startovní čas aktivity A (latest start time)
- lct(A) nejpozdější koncový čas aktivity A (latest completion time)

- Ω je množina aktivit
- $p(\Omega) = \sum_{A \in \Omega} p(A)$
- $est(\Omega) = \min\{est(A) \mid A \in \Omega\}$
- $lct(\Omega) = \max\{lct(A) \mid A \in \Omega\}$

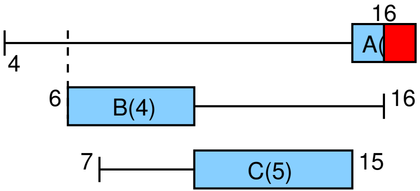
Unární zdroje

- Aktivivity se nemohou překrývat
  - v daném čase běží maximálně jedna aktivita, proto se těmto zdrojům říká unární
  - Grahamova klasifikace: jeden stroj
- Předpokládáme, že aktivity jsou nepřerušitelné
  - nepřerušitelná aktivita zabírá zdroj od svého startu až do ukončení
- Jednoduchý model s disjunktními omezeními
  - A<<B ∨ B<<A
    end(A)≤start(B) ∨ end(B)≤start(A)
  - těmto zdrojům se proto někdy říká disjunktní

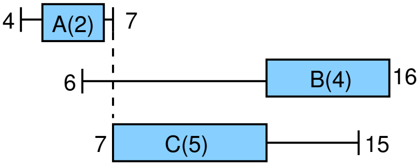


Hledání hran *(edge finding)*

- Baptiste & Le Pape (1996)
- Co se stane, pokud nebude aktivita A zpracována jako první?

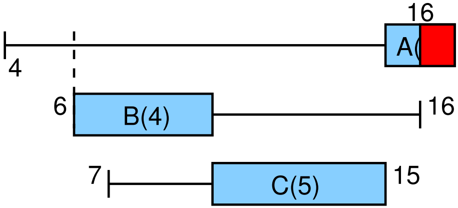


- Pro A,B,C není dost času, a tedy aktivita A musí být první



Hledání hran: příklad s odvozovacími pravidly

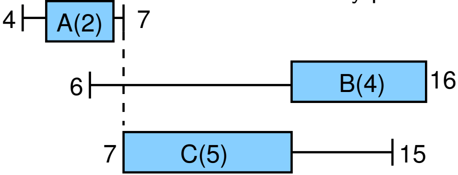
- $lct(\Omega \cup \{A\}) - est(\Omega) < p(\Omega \cup \{A\}) \Rightarrow A << \Omega$



$\Omega = \{B, C\}$

- $A << \Omega \Rightarrow end(A) \leq \min\{lct(\Omega') - p(\Omega') \mid \Omega' \subseteq \Omega\}$

všechny podmnožiny Ω musí stihnout své zpracování

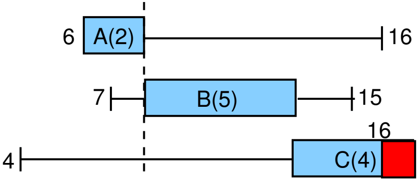


Hledání hran: všechna odvozovací pravidla

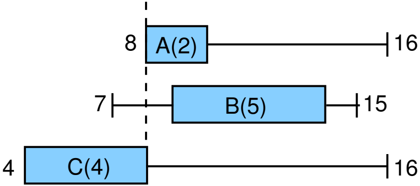
- Zopakujme tedy odvozovací pravidla: pro $end(A)$
  (co se stane, pokud nebude aktivita A zpracována jako první?)
  - $lct(\Omega \cup \{A\}) - est(\Omega) < p(\Omega \cup \{A\})$
    $\Rightarrow A << \Omega$
  - $A << \Omega \Rightarrow$
    $end(A) \leq \min\{lct(\Omega') - p(\Omega') \,|\, \Omega' \subseteq \Omega\}$
- Symetrická odvozovací pravidla: pro $start(A)$
  (co se stane, pokud nebude aktivita A zpracována jako poslední?)
  - $lct(\Omega) - est(\Omega \cup \{A\}) < p(\Omega \cup \{A\})$
    $\Rightarrow \Omega << A$
  - $\Omega << A \Rightarrow$
    $start(A) \geq \max\{est(\Omega') + p(\Omega') \,|\, \Omega' \subseteq \Omega\}$
- V praxi:
  - celkem existuje $n \cdot 2^n$ párů A,Ω (příliš mnoho!)
  - Carlier & Pinson 1994, Vilím & Barták & Čepek 2004
    algoritmus s časovou složitostí $O(n \log n)$
    (je nutné kontrolovat pouze některé páry A,Ω)

Ne-první/ne-poslední *(not-first/not-last)*

- Torres & Lopez 2000
- Co se stane, pokud aktivita A bude zpracována jako první?

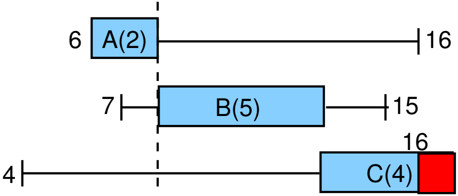


- Pro B a C není dost času, a tedy aktivita A nemůže být první



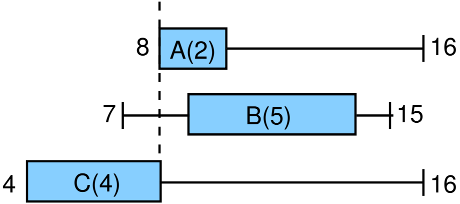
Ne-první/ne-poslední: př. s odvozovacími pravidly

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



$\Omega = \{B, C\}$

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



Odvozovací pravidla pro ne-první/ne-poslední

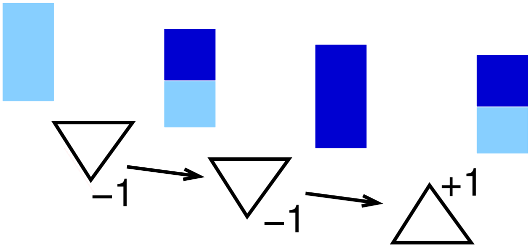
- Zopakujme Ne-první pravidla:
  (co se stane, pokud aktivita A bude zpracována jako první?)
  - $lct(\Omega) - est(A) < p(\Omega \cup \{A\})$
    $\Rightarrow \neg A << \Omega$
  - $\neg A << \Omega$
    $\Rightarrow start(A) \geq \min\{ect(B) | B \in \Omega\}$

- Ne-poslední (symetrická) pravidla:
  (co se stane, pokud aktivita A bude zpracována jako poslední?)
  - $lct(A) - est(\Omega) < p(\Omega \cup \{A\})$
    $\Rightarrow \neg \Omega << A$
  - $\neg \Omega << A \Rightarrow$
    $end(A) \leq \max\{lst(B) | B \in \Omega\}$

- V praxi:
  - Vilím 2004: algoritmus s časovou složitostí $O(n \log n)$

Produkovatelné/spotřebovatelné zdroje

- Zdroj = rezervoár
- Aktivita konzumuje nějaké množství zdroje cap(A)<0 nebo aktivita produkuje nějaké množství zdroje cap(A)>0
- Požadována minimální kapacita zdroje (při konzumaci) a maximální kapacita zdroje nemůže být překročena (produkcí)



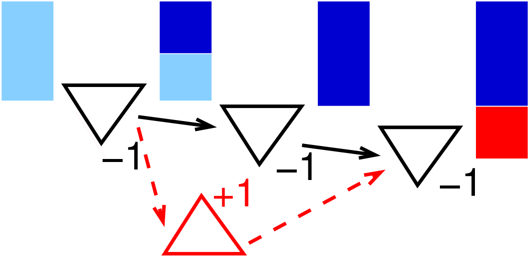
- Kumulativní zdroj může být chápan jako speciální případ rezervoáru
  - každá aktivita konzumuje cap(A) při startu a produkuje cap(A) na konci

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



lze odvodit nezbytnost přídavné aktivity produkující jednotku zdroje

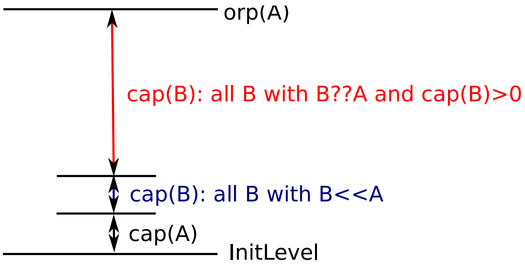
Optimistický zdrojový profil (*orp*) Cesta&Stella 1997

orp(*A*): 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

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

Příklad (opravdu pouze příklad jsou i jiné možnosti!):

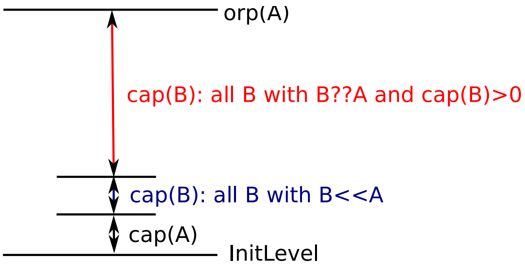


B??A znamená, že pořadí A a B ještě není známé

orp odvozovací pravidla I.

$orp(A) < MinLevel \Rightarrow fail$

- i když je veškerá produkce plánována před A
  není dosažena minimální požadovaná úroveň zdroje



$$orp(A) = InitLevel + cap(A) + \sum_{B<<A} cap(B) + \sum_{B??A\,\&\,cap(B)>0} cap(B)$$

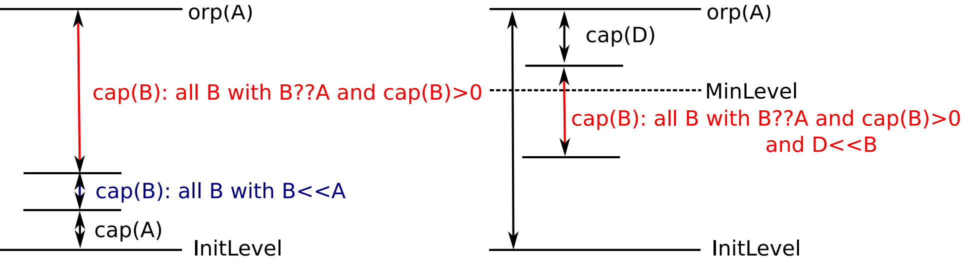
orp odvozovací pravidla II.

$orp(A) - cap(D) - \sum_{D<<B \,\&\, B??A \,\&\, cap(B)>0} cap(B) < MinLevel$
$\quad \Rightarrow D << A$

Uvažujme časový okamžik, kdy začne aktivita A

- pro libovolné D takové, že $D??A$ a $cap(D) > 0$:
  pokud je produkce v D plánována za A a minimální požadovaná úroveň zdroje není dosažena, pak musí být D před A

Příklad:



Optimistický zdrojový profil: cvičení

$orp(A) = InitLevel + cap(A) + \sum_{B<<A} cap(B) + \sum_{B??A \,\&\, cap(B)>0} cap(B)$

$orp(A) - cap(D) - \sum_{D<<B \,\&\, B??A \,\&\, cap(B)>0} cap(B) < MinLevel$
$\Rightarrow D << A$

- Máme dány aktivity A, B, C, X, Y, Z
  a jejich požadované kapacity jsou po řadě $1, 2, 3, 4, 5, -1$
- Jsou dány precedence: $A << B << C, X << Y$
- $InitLevel = MinLevel = 0$
- Jaká je hodnota $orp(A)$?
  $orp(A) = 0 + cap(A) + 0 + (cap(X) + cap(Y)) = 0 + 1 + 0 + (4 + 5) = 10$
- Může být X naplánováno po A? $10 - 4 - 5 = 1$, tj. ano
- Co se změní, pokud bychom přidali aktivitu D s následujícími vlastnostmi?
  - $cap(D) = -2$
  - $D << A$

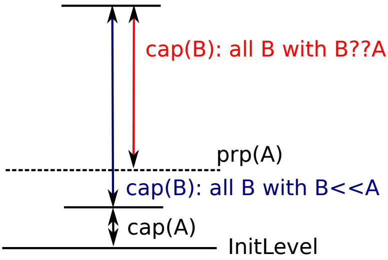
  $orp(A) = 0 + 1 + (\underline{-2}) + (4 + 5) = 8$ a dále $8 - 4 - 5 = -1$, tj. X nesmí být po A

$prp(A)$: 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

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

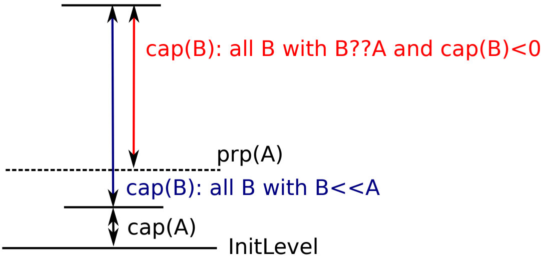
Příklad:



prp odvozovací pravidla I.

$prp(A) > MaxLevel \Rightarrow fail$

- i když je veškerá konzumace plánována před A, maximální povolená kapacita zdroje je překročena



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

prp odvozovací pravidla II.

$prp(A) - cap(D) - \sum_{D<<B \,\&\, B??A \,\&\, cap(B)<0} cap(B) > MaxLevel$
$\Rightarrow D << A$

Uvažujme časový okamžik, kdy začne aktivita A:

- pro libovolné D takové, že $D??A$ a $cap(D) < 0$:
  jestliže je konzumace v D plánována po A a je překročena maximální povolená úroveň zdroje, pak musí být D před A

Příklad: