

PB165 Grafy a sítě: Plánování s komunikací

Plánování s komunikací

- 1 Popis problému
- 2 Plánování seznamem
- 3 Heuristiky mapování
- 4 Shlukovací heuristiky

Plánování s komunikací a s precedencemi: přehled

Příklady aplikací

- **plánování komunikujících úloh s precedencemi**
 - acyklický graf precedenčních závislostí mezi úlohami
 - přenos dat po skončení úlohy následující úlohám
- **plánování acyklických workflows**

Algoritmy

- **optimální (polynomiální složitost)**
 - použitelné pouze pro velmi specializované problémy
 - obtížná rozšiřitelnost na obecné problémy
 - př. plánování na dvou procesorech, plánování stromů
- **heuristické**
 - **plánování seznamem (list scheduling)**
 - **heuristiky mapování (mapping heuristics)**
 - **shlukovací heuristiky (clustering heuristics)**

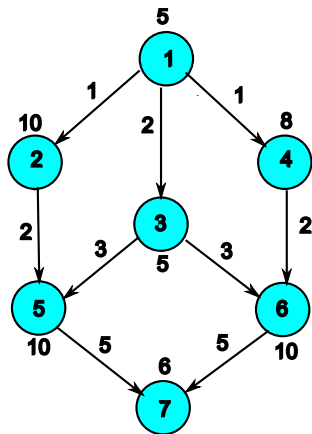
Úlohy a komunikace

Referenční dobu trvání p_i úlohy i

Precedenční omezení $i_1 \rightarrow i_2$

Komunikace: $size_{i_1, i_2}$

- $size_{i_1, i_2}$: velikost přenesených dat mezi úlohami i_1 a i_2
- pokud existuje $i_1 \rightarrow i_2$, pak $size_{i_1, i_2} \geq 0$, jinak $size_{i_1, i_2} = 0$
- pokud jsou i_1 a i_2 zpracovány na stejném stroji, tak lze čas na komunikaci (a tedy i velikost přenášených dat) zanedbat

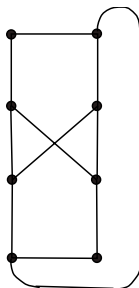
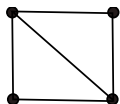


Hranově a vrcholově ohodnocený acyklický orientovaný graf

- vrcholy: úlohy
- orientované hrany: precedenční vztahy mezi úlohami
- ohodnocení hrany: velikost dat na komunikaci
- ohodnocení vrcholu: referenční doba trvání

Síť a komunikační zpoždění

- Topologie sítě: příklady



na uzlech dual-core procesor

- $transrate_{j_1, j_2}$: přenosová rychlost mezi sousedními stroji
- $setupmesg_j$: (inicializační) doba na poslání zprávy strojem j
- **Komunikační zpoždění** $c(i_1, i_2, j_1, j_2)$ pro poslání dat z úlohy i_1 úloze i_2 ze stroje j_1 na sousední stroj j_2

$$c(i_1, i_2, j_1, j_2) = \frac{size_{i_1, i_2}}{transrate_{j_1, j_2}} + setupmesg_{j_1}$$

Doba provádění. Optimalizace.

- $speed_j$: rychlost zpracování strojem j
- $setuptask_j$: (inicializační) doba na nastartování úlohy na stroji j
- **Doba provádění** p_{ij} úlohy i na stroji j :

$$p_{ij} = \frac{p_i}{speed_j} + setuptask_j$$

- **Objektivní funkce (performance measure)**
 - minimalizace makespan (čas dokončení poslední úlohy)
 - do makespan se započítává:
 - doba provádění + komunikační zpoždění

Plánování seznamem (list scheduling)

Jednoduché efektivní algoritmy

- založeny na seřazení úloh v **prioritní frontě**
- složitost algoritmu dána výpočtem priority

Algoritmus plánování seznamem

- 1 každému uzlu grafu přiřazena priorita
prioritní fronta inicializována úlohami bez předchůdců
úlohy ve frontě seřazeny v klesajícím pořadí dle priority
- 2 pokud existuje volný stroj a fronta je neprázdná, prováděj:
 - 1 odebrána první úloha z fronty
 - 2 volný procesor je vybrán pro spuštění úlohy
 - 3 jakmile provedeni všichni přímí předchůdci nějaké úlohy, tak je úloha přidána dle priority do fronty

Priority pro plánování seznamem

- Délka cesty přes uzly $i_1, i_2 \dots i_n$:

$$w1 \sum_{i=1}^n p_i + w2 \sum_{i=1}^{n-1} size_{i,i+1}$$

- $w1, w2$: koeficienty určující poměr mezi dobou trvání a velikostí přenesených dat
- Počáteční uzly: uzly bez předchůdců
- Koncové uzly: uzly bez následníků

Výpočet priority uzlu i (příklady)

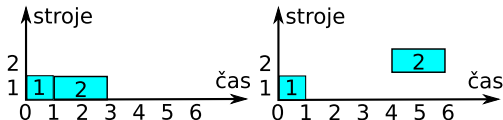
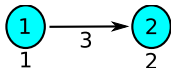
- **úroveň (level)**
délka nejdelší cesty z uzlu i do koncového uzlu
- **ko-úroveň (co-level)**
délka nejdelší cesty z počátečního uzlu do uzlu i

Plánování seznamem: nastavení

- **Zanedbání doby na komunikaci:**
 - pokud máme $i_1 \rightarrow i_2$ a úlohy i_1 a i_2 jsou prováděny na stejném stroji, tak dobu na komunikaci zanedbáme
- **Výběr stroje:**
 - pro provádění úlohy i je vybrán stroj, na kterém bude úloha dokončena nejdříve (v případě více možností je vybrán stroj s nejmenším indexem)
 - k provádění jsou vybírány pouze volné stroje
 - pozor, je třeba brát v úvahu, zda na některém stroji nedojde k zanedbání komunikace

Plánování seznamem: parametry

- Necht' $size_{i,j}$ přímo reprezentuje dobu na přenos dat
- Předpokládejme, že $w_1 = w_2 = 1$
- **Rychlost stroje**
 - pokud bychom předpokládali, že stroje mají stejnou rychlost, pak lze úlohy přímo plánovat dle zadané p_i a $size_{i,1,2}$
- **Koncový čas úlohy**
 - spočítán na základě délky cesty (z p_i a $size_{i,1,2}$)
 - lze uvažovat možné zanedbání komunikace (úloha je ve frontě až v okamžiku, když jsou všichni předchůdci dopočítáni, tj. známe jejich stroj a víme, zda dojde k zanedbání komunikace)

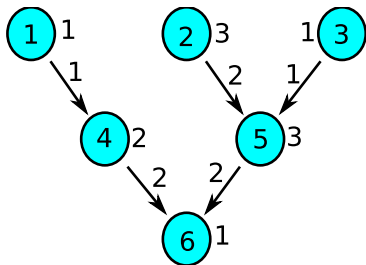


se zanedbáním komunikace
(na stejném stroji)

bez zanedbání komunikace
(na různých strojích)

Plánování seznamem: příklad

Plánování 6 úloh zadaných grafem na 2 stroje se stejnou rychlostí



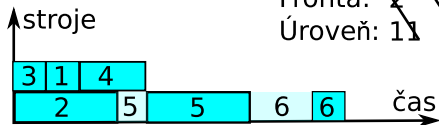
komunikace
zanedbána:

1->4

2->5

5->6

Fronta: ~~2~~ ~~3~~ ~~1~~ ~~4~~ ~~5~~ ~~6~~
Úroveň: 1 1 8 7 5 6 1



- tmavší obdélníky v rozvrhu znázorňují dobu, kdy úloha běží
- světlejší obdélníky odpovídají době na komunikaci pro danou úlohu
 - pro 5: komunikace 3 → 5, pro 6: komunikace 4 → 6

Plánování seznamem: cvičení

Vyřešte následující problém plánování s komunikací a s precedencemi pomocí plánování seznamem:

- jsou dány 2 stroje se stejnou rychlostí
- je dáno 6 úloh s dobou trvání
 $p_1 = 1, p_2 = 1, p_3 = 2, p_4 = 1, p_5 = 2, p_6 = 1$
- jsou dány precedence: $1 \rightarrow 2, 2 \rightarrow 4, 4 \rightarrow 6, 1 \rightarrow 3, 3 \rightarrow 5, 5 \rightarrow 6$
- $size_{i_1, i_2}$ reprezentuje dobu na přenos dat a je zadán takto:
 $size_{1,2} = 1, size_{2,4} = 3, size_{4,6} = 1,$
 $size_{1,3} = 2, size_{3,5} = 4, size_{5,6} = 2$

Plánování seznamem: diskuse

Výběr úlohy s nejvyšší úrovní = **výběr úlohy na kritické cestě**

Problémy při použití heuristiky

- změna kritické cesty
 - ⇐ komunikační zpoždění závisí na alokaci úlohy
 - ⇐ např. při alokaci na stejné stroje se zanedbatelným zpožděním nebo při nesteriomerné vzdálenosti strojů (a počtu hopů)

Řada heuristik založena na principu (několika) prioritních front

- rozsáhlé modifikace priorit
 - umožňují zachytit různé charakteristiky systému
- např. produkční plánovací systémy pro plánování úloh na počítačích
PBSPro, SGE

Heuristiky mapování (mapping heuristics)

- Modifikace plánování seznamem
- Více uvažovány reálné parametry:
 - topologie sítě, rychlost procesoru, přenosová rychlost, ...
- Doba provádění úlohy i na stroji j

$$p_{ij} = \frac{p_i}{speed_j} + setup_{task_j}$$

- Komunikační zpoždění

$$c(i_1, i_2, j_1, j_2) = \left(\frac{size_{i_1, i_2}}{transrate} + setup_{mesg} \right) \times hops_{j_1, j_2} + contdelay_{j_1, j_2}$$

- předpokládán konstantní $transrate$ a $setup_{mesg}$
- $hops_{j_1, j_2}$: počet hopů mezi stroji j_1 a j_2 , (délka nejkratší cesty mezi stroji při jednotkovém ohodnocení hran)
- $contdelay_{j_1, j_2}$: zpoždění dané zatížením linky mezi j_1 a j_2 (contention delay)

Heuristiky mapování (mapping heuristics)

- Konstruována a udržována **směrovací tabulka** obsahující
 - odhadované hodnoty p_{ij} a $c(i_1, i_2, j_1, j_2)$
 - pro každý procesor
 - počet hopů, preferovaná linka pro daný cíl, zpoždění dané zatížením
- **Úlohy plánovány podle nejvyšší úrovně** (podobně jako při plánování seznamem), v případě nejednoznačnosti vybrána úloha s větším množstvím přímých následníků
- Jakmile jsou dokončeni všichni předchůdci úlohy, tak je **úloha naplánována na stroj, kde nejdříve skončí**
- Koncový čas úlohy na stroji spočítán z
 - rychlost procesoru, přenosová rychlost, vybraná linka, počet hopů, zpoždění dané zatížením linky

Plánování se shlukovacími heuristikami (clustering heuristics)

- Plánování řeší
 - ① alokace úloh na stroje
 - ② seřazení úloh na stroji / umístění úloh v čase
- **Shlukovací heuristiky:** řeší alokaci úloh na stroje
- **Shluk:** množina úloh, která bude prováděna na stejném stroji
- **Princip plánování založeného na shlukování**
 - ① Shlukování úloh na nelimitovaný počet plně propojených procesorů
 - ② Namapování shluků a jejich úloh na daný počet procesorů (m) prováděním následujících operací
 - **spojování shluků:** pokud je počet shluků vyšší než m
 - **fyzické mapování:** přiřazení shluků na procesory tak, aby byla minimalizována komunikace (na této úrovni uvažována reálná konektivita mezi stroji)
 - **uspořádání úloh:** úlohy jsou na stroji prováděny tak, by byly splněny závislosti mezi úlohami

Shlukovací heuristiky

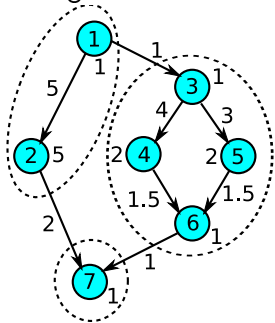
- Shlukovací heuristiky
 - bez backtrackingu (abychom se vyhnuly vysoké složitosti), tj. jakmile jsou jednou shluky spojeny nelze je zpětně rozpojit
- Iniciálně: jedna úloha = jeden shluk
- Typický krok heuristiky
 - spojení shluků + **vynulování hrany**, která je spojuje
 - vynulování hrany: úlohy prováděny na stejném procesoru, kde je čas na komunikaci zanedbatelný
- Kritérium pro výběr hrany na nulování redukce paralelního času rozvrhu

Naplánovaný graf úloh

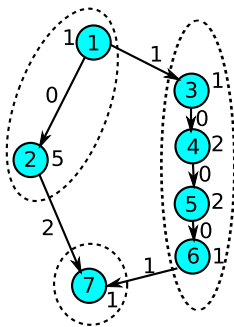
Naplánovaný graf úloh

- graf úloh **zahrnující** aktuální vynulování hran
- úlohy v jednom shluku jsou seřazeny (jako na procesoru) dle nejvyšší úrovně

původní graf úloh se 3 shluky



naplánovaný graf úloh



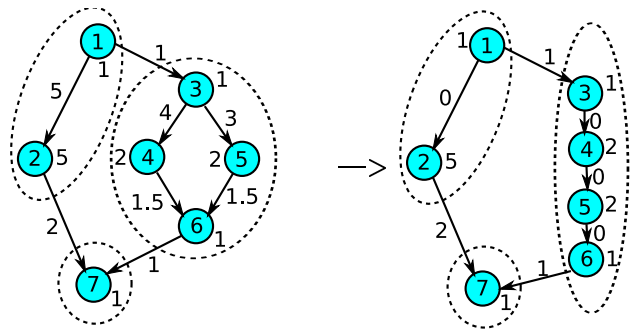
Paralelní čas a dominantní posloupnost

- Připomenutí: délka cesty přes uzly $i_1, i_2 \dots i_n$:

$$w1 \sum_{i=1}^n p_i + w2 \sum_{i=1}^{n-1} size_{i,i+1}$$

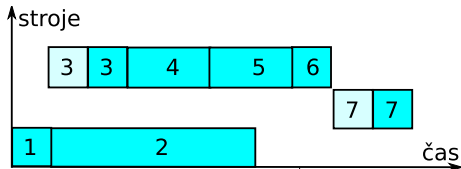
- **Dominantní posloupnost:**
nejdelší cesta v naplánovaném grafu úloh
- **Paralelní čas rozvrhu:**
doba nutná na provedení dominantní posloupnosti
- Pokud $w1 = w2 = 1$ a $size_{i,j}$ reprezentuje dobu na přenos
 \Rightarrow délka cesty odpovídá době na provedení úloh na cestě
 \Rightarrow paralelní čas rozvrhu = délka cesty dominantní posloupnosti

Dominantní posloupnost: příklad



Dominantní posloupnost 1,3,4,5,6,7 s délkou 10, tj. paralelní čas rozvrhu je 10

Odpovídající rozvrh na 3 strojích:



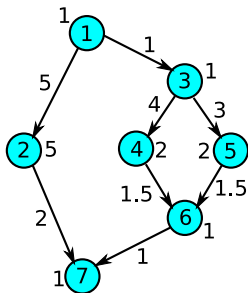
Algoritmus shlukovací heuristiky

- 1 Inicializace: všechny hrany označeny jako nevyzkoušené
každá úloha tvoří jeden shluk
- 2 Seřazení všech hran v grafu úloh v klesajícím pořadí
dle komunikační ceny
- 3 REPEAT
 - 1 vynulování největší nevyzkoušené hrany v seřazeném seznamu,
pokud nevzroste paralelní čas
 - 2 hrana je označena jako vyzkoušená
 - 3 když jsou spojeny dva shluky, úlohy jsou uspořádány dle nejvyšší úrovněUNTIL všechny hrany jsou vyzkoušené

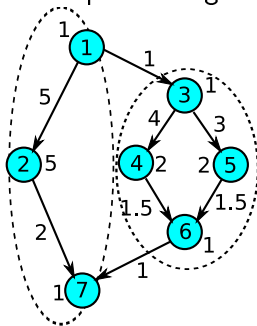
Komentář: podobně jako existují různé metody plánování seznamem, tak existují různé varianty shlukovacích heuristik

Shlukovací heuristiky: příklad

Problém:



Řešení pomocí algoritmu:



Postup:

seřazení hran, postupné zkoušení na vynulování a spojování shluků:

12 (vynulována, shluk 12), 34 (vynulována, shluk 34), 35 (vynulována, shluk 345), 27 (vynulována, shluk 127), 46 (vynulována, shluk 3456), 56 (vynulována), 13 (nevynulována, paralelní čas by z 10 pro dominantní cestu 1,3,4,5,6,7 vzrostl na 13 s dominantní cestou 1,2,3,4,5,6,7), 67 (nevynulována stejně jako 13)

Shlukovací heuristiky: cvičení

Vyřešte cvičení z kapitoly plánování seznamem pomocí shlukovací heuristiky.

Je v řešení nějaký rozdíl?

(náповěda: ano, u shlukovacích heuristik víme, na kterém stroji bude úloha zpracovávána předem, a proto je možné zahájit některé komunikace dříve, a tedy rozvrh skončí dříve)