Kombinatorická optimalizace
cvičení č.9
Rozvrhování a metoda větví a mezí

Roman Čapek, Přemysl Šůcha (capekrom@fel.cvut.cz, suchap@fel.cvut.cz)

13. dubna 2011

1 Rozvrhování

Rozvrhování se, obecně řečeno, zabývá přiřazením zdrojů úlohám v čase. Máme tedy množinu úloh, množinu zdrojů (procesorů), které tyto úlohy mohou využívat, a dále množinu omezení a kriteriální funkci, kterou chceme optimalizovat. Každá úloha, typicky ji značíme T_i, je specifikována svou dobou vykonávání (výpočetním časem) - *processing time* p_i a přiřazením na zdroj/více zdrojů. Dále může mít následující parametry:

- Doba, kdy nejdříve může začít její vykonávání - *release time* r_i,
- Doba, do kdy bychom ji chtěli mít dokončenou - *due date* d_i,
- Doba, do kdy musí být dokončena - *deadline* $\tilde{d}_i$,
- Priorita - *weight* w_i.

Běžně se užívá takzvaná $\alpha|\beta|\gamma$ notace, jinak také Graham-Blazewiczova [1, 2]:

- α obsahuje charakteristiku procesorů: počet (1-∞) a typ (P-paralelní, Q-uniformní...)
- β udává charakteristiku úloh a přídavných zdrojů: r_j - úlohy mají dané release time, $pmtn$ - je povolena preempce...
- γ definuje kritérium optimality: C_{max} - minimalizace celkové délky rozvrhu, L_{max} - minimalizace maximálního zpoždění, $\sum U_j$ - minimalizace počtu úloh, které překročí svůj duedate...

Například $P2||C_{max}$ je formální zápis rozvrhovacího problému, kde jsou k dispozici dva paralelní procesory a cílem je minimalizovat celkovou délku rozvrhu (i takto "jednoduchá" úloha ovšem spadá do kategorie NP-obtížných problémů). Výstupem rozvrhování je přiřazení úloh v čase na procesory a nejčastěji se zobrazuje ve formě Ganttova diagramu [1], kde na ose x je diskrétní čas a na ose y jednotlivé procesory (zdroje).

2 Metoda větví a mezí

Metoda větví a mezí (*branch and bound*) se používá pouze k řešení diskrétních optimalizačních úloh [1]. Tato metoda postupně konstruuje strom částečných řešení problému a postupně je rozvíjí (větvení-branch). Pokud narazí na nepřípustné částečné řešení nebo na částečné řešení takové, že už nemůže zlepšit nejlepší dosud nalezené, tento vrchol již dále nerozvíjí (meze-bound). Strom řešení si můžeme představit jako acyklický graf s jedním kořenovým vrcholem. Každý vrchol tohoto grafu představuje jedno částečné řešení problému, listy stromu představují řešení kompletní. K odříznutí podstromu dojde tehdy, pokud o něm víme:

a) Nenachází se v něm žádné přípustné řešení.

b) Nenachází se v něm optimální řešení.

Druhá možnost v podstatě pokrývá i možnost první, nicméně pro konstrukci algoritmů je vhodné je posuzovat odděleně.
Obecně rozlišujeme dva hlavní způsoby prohledávání stavového prostoru:

a) Prohledávání do šířky.

b) Prohledávání do hloubky.

V našem případě, tedy použití metody větví a mezí v rozvrhování, je výhodnější prohledávání stromu řešení do hloubky, protože každé získané přípustné řešení zvyšuje pravděpodobnost odřezávání dalších částí stromu bez jejich kompletního procházení.

3 Příklad - Bratleyův algoritmus

Příkladem na použití metody větví a mezí v problematice rozvrhování je problém formálně popsaný jako $1|r_j, \tilde{d}_j|C_{max}$, tedy rozvrhování na jednom procesoru, přičemž každá úloha má zadaný release time a deadline, tedy jakési časové okno, do něhož se musí vejít její vykonávání. Cílem je minimalizovat délku rozvrhu. Tento problém je rovněž ze třídy NP-obtížných problémů, takže pro něj neexistuje deterministický algoritmus běžící v polynomiálním čase. Jednou z variant jak tento problém řešit je použít takzvaný *Bratleyův algoritmus* [1] založený právě na metodě větví a mezí. Naším cílem je prohledat stavový prostor ve formě stromu řešení, kde každý koncový vrchol (list) reprezentuje jedno řešení problému. Ostatní vrcholy jsou částečná řešení složená ze seřazení některých úloh. K prořezávání stromu lze použít následující podmínky:

1) Překročení deadline některé úlohy.

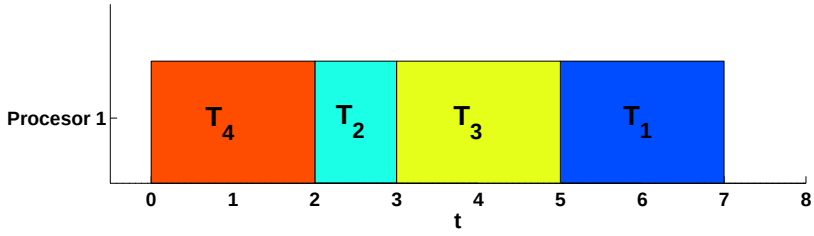
2) Odhad řešení nejlepšího možného řešení v podstromu je horší než nejlepší zatím dosažený výsledek.

3) Částečné řešení je optimální.

Ad 1) Pokud v některém vrcholu dojde k situaci, že nemůžeme přidat libovolnou úlohu bez překročení její deadline, nepovede tento částečný rozvrh nikdy k přípustnému řešení. Tento vrchol a celý strom pod ním tak můžeme odříznout. Vyplývá to z toho, že pokud přidáním některé úlohy za dosavadní částečný rozvrh dojde k překročení její deadline, určitě nepomůže její zařazení ještě dále.
Ad 2) Na počátku uděláme horní odhad maximální délky délky rozvrhu, například jako $UB = \max \tilde{d}_j$ pro $\forall j$. V každém vrcholu pak můžeme udělat odhad nejlepšího možného celkového řešení vzhledem k aktuálnímu částečnému řešení jako $LB = \max\left(\max\left(\min_{\forall j \in V} r_j, c\right) + \sum_{\forall j \in V} p_j,\ \max_{\forall j \in V}(r_j + p_j)\right)$, kde c je dosavadní délka rozvrhu (tedy čas dokončení zatím poslední přidané úlohy) a V je množina úloh, které ještě zbývá rozvrhnout. Pokud platí $LB > UB$, pak dalším větvením aktuálního vrcholu nemůžeme nalézt optimální řešení a je proto možné tuto část stromu odříznout.
Ad 3) Pokud je splněna takzvaná *release time property*, je částečný rozvrh optimální a stačí rozvíjet pouze aktuální vrchol. Vracením se ve stromu částečných řešení již nelze získat lepší rozvrh. Release time property říká, že pokud je délka částečného řešení menší nebo rovna nejmenší hodnotě release time dosud nerozvržených úloh, je toto částečné řešení optimální.

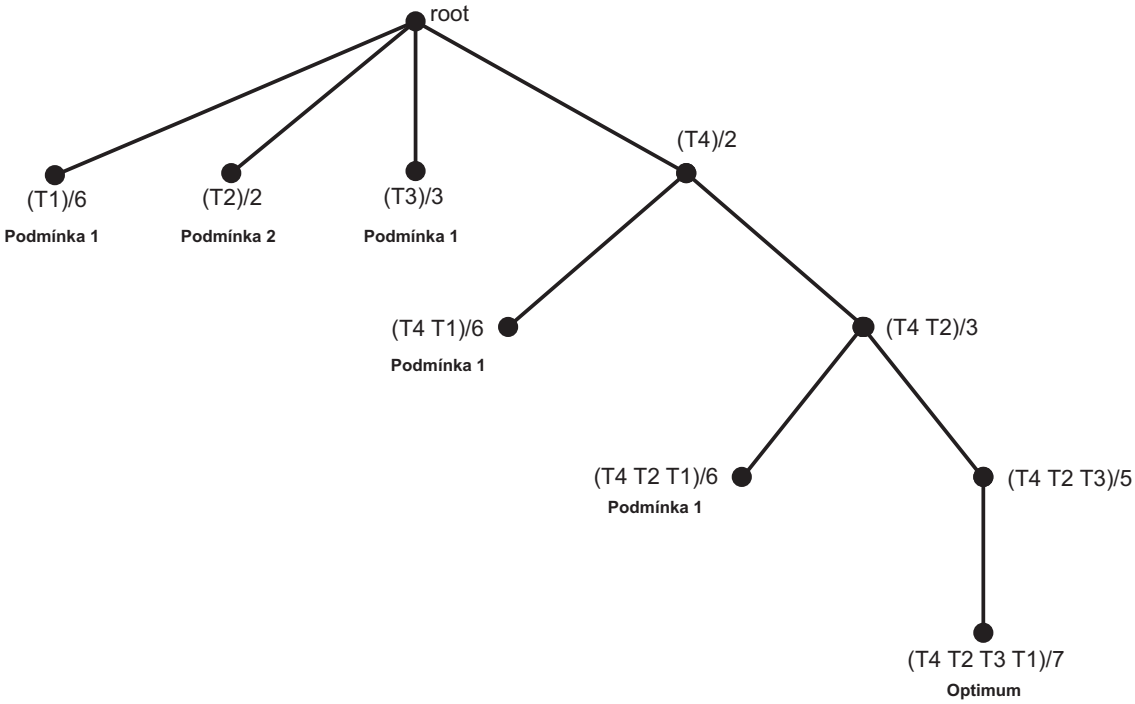
Příklad zadání instance problému $1|r_j, \tilde{d}_j|C_{max}$ je v levé části obrázku 1. Jedná se o problém se čtyřmi úlohami, každá má svůj processing time, release time a deadline. Úlohu vyřešíme pomocí Bratleyova algoritmu založeného na metodě větví a mezí. Výsledek v podobě Ganttova diagramu je v pravé části obrázku 1. Strom řešení bude mít $4! = 24$ listů. Pokud bychom měli zadání s 10 úlohami, bude počet kombinací seřazení úloh (a tedy listů stromu řešení) 3628800 a v případě 15 úloh půjde už o více $1.3 \cdot 10^{12}$ možných kombinací. Takže je jasně vidět, že už pro relativně malé instance je počet možných kombinací velmi vysoký a prohledávání celého stavového prostoru zcela neřešitelné.

Zadání:
p=(2,1,2,2)
r=(4,1,1,0)
d=(7,5,6,4)



Obrázek 1: Příklad zadání problému $1|r_j, \tilde{d}_j|C_{max}$ a jeho řešení

V tomto případě budeme konstruovat strom řešení tak, že z kořenového vrcholu vytvoříme čtyři větve, kde každý ze čtyř takto vyniklých vrcholů odpovídá umístění jedné úlohy do rozvrhu jako první. Tyto vrcholy poté opět rozvětvíme, tentokrát už každý jen na tři další (jedna pozice už je daná), poté vzniklé vrcholy větvíme dále na dva a v poslední fázi už nám z každého vrcholu vede jen jedna větev do listu stromu. Listů je celkem 24 (viz výše) a strom má i s kořenem 5 úrovní. Strom řešení prořezaný na základě výše zmíněných podmínek je na obrázku 2. Výsledek v podobě Ganttova diagramu je v pravé části obrázku 1.



Obrázek 2: Prořezaný strom řešení

Úkol: Naprogramujte Bratleyův algoritmus jako rekurzivní funkci v Matlabu. Vstupem budou tři vektory-processing time p, release time r a deadline $\tilde{d}$ a výstupem bude přiřazení úloh na procesor v čase a výsledná délka rozvrhu.

Poznámka: Kostru algoritmu naleznete zde Bratley.

Reference

[1] J. Blazevicz, *Scheduling Computer and Manufacturing Processes*. Springer, second ed., 2001.

[2] R. Graham, E. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan, “Optimization and approximation in deterministic sequencing and scheduling theory: a survey,” *Annals of Discrete Mathematics*, vol. 5, pp. 287–326, 1979.