CHAPTER III

MODELY S PARALELNÝMI STROJMI

Majme danú množinu $\mathcal{M} = \{M_1, M_2, \dots, M_m\}$ m paralelných strojov, na ktorých sa má spracovať n úloh z množiny $\mathcal{J} = \{J_1, J_2, \dots, J_n\}$. Každá úloha J_i pozostáva z jedinej operácie $J_i = \{o_i\}$. Každá operácia o_i sa môže vykonať na ľubovoľnom stroji M_j s dobou spracovania p_{ij}. Ak na časy spracovania nie sú kladené ďalšie predpoklady, pôjde o najvšeobecnejší prípad $Rm|\beta|\gamma$ (paralelné rôzne stroje); ak sa dá písať $p_{ij} = p_i.q_j$ pôjde o systémy $Qm|\beta|\gamma$ (paralelné uniformné stroje), kde sa stroje líšia len rýchlosťou. Nakoniec ak doba spracovania operácie o_i je na všetkých strojoch rovnaká, t.j. ak $p_{ij} = p_i$ pôjde o systémy $Pm|\beta|\gamma$ (paralelné identidké stroje).

Paralelné identické stroje

Ďalej sa budeme zaoberať prípadom $Pm|\beta|f$, kde f je regulárne kritérium. Označme p_i čas spracovania úlohy J_i (je rovnaký na každom stroji). Úlohy môžu byť nezávislé v tom zmysle, že nezáleží na poradí ich vykonávania, alebo na množine úloh $\mathcal{J}$ môže byť definovaná precedenčná relácia $\prec$. Množina rozvrhov bez prerušenia tu nie je dominantnou množinou ani pre regulárne kritériá, preto sa povoľuje v niektorých modeloch prerušenie vykonávania úloh. Budeme predpokladať, že všetky úlohy prídu do systému naraz v čase 0.

Tvrdenie. *V systémoch $Pm||C_{\max}$ $Pm|pmtn|C_{\max}$ je dĺžka najkratšieho rozvrhu rovná*

$$L = \max\left\{\frac{1}{m}\sum_{i=1}^{n} p_i, \max_i p_i\right\}$$

Dôkaz.

Keďže jedna úloha sa nemôže naraz spracovávať na dvoch strojoch, dĺžka najkratšieho rozvrhu musí byť aspoň $\max_i p_i$. Pretože stroj môže v jednom čase spracovávať len jednu operáciu, musí byť L väčšie alebo rovné priemernej dobe obsadenia jedného stroja. Ukázali sme že

$$L \geq \max\left\{\frac{1}{m}\sum_{i=1}^{n} p_i, \max_i p_i\right\}$$

Skutočnosť, že v systéme $Pm|pmtn|C_{\max}$ už existuje prípustný rozvrh taký, že pre $C_{\max} = L$, ukáže nasledujúci algoritmus.

McNaughtonov algoritmus pre systém $Pm|pmtn|C_{\max}$.

Postupne priraďujeme úlohy tomu istému stroju, pokiaľ neprekročíme čas L z (X). Vykonávanie časti úlohy, ktorá by prekročila čas L prerušíme a zaradíme ďalšiemu stroju od času 0 atď. pokiaľ nepriradíme týmto spôsobom všetky úlohy.

Všeobecné paralelné stroje

Systém $Rm||C_{\max}$.

Nech p_{ij} je doba spracovania operácie o_i na stroji M_j. Označme x_{ij} pre $i = \{1, 2, \dots, n\}$, $j = \{1, 2, \dots, m\}$, binárnu premmennú, ktorá nadobúda hodnotu 1 práve vtedy, keď úloha J_i je priradená stroju M_j. Označme C trvanie rozvrhu (t.j. čas ukončenia poslednej úlohy za predpokladu vstupu úloh v čase 0.) Potom nájsť optimálny rozvrh v systéme $Rm||F_{\max}$ bez preempcie znamená riešiť nasledujúcu úlohu celočíselného programovania:

$$\text{Minimalizovať} \qquad C$$

za predpokladov

$$C - \sum_{i=1}^{n} p_{ij}.x_{ij} \geq 0 \qquad \text{pre} \quad j = \{1, 2, \dots, m\}$$

$$\sum_{j=1}^{m} x_{ij} = 1 \qquad \text{pre} \quad i = \{1, 2, \dots, n\}$$

$$x_{ij} = 0, 1$$

Takto definovanú úlohu je možné riešiť metódami celočíselného programovania. Je to však úloha NP–ťažká.

LPT heuristika pre $Pm||C_{\max}$

Pre systém $Pm||C_{\max}$ máme nasledujúci heuristický algoritmus:

Heuristický algoritmus pre model $Pm||C_{\max}$ (LPT algoritmus).

KROK 1: Vytvorme poradie úloh

$$p_{[1]} \geq p_{[2]} \geq \cdots \geq p_{[n]}. \tag{1.7}$$

KROK 2: V tomto poradí priraďujme úlohy strojom, ktoré sa najskôr uvoľnia.

Veta. *Majme systém $Pm||C_{\max}$. Označme dĺžku optimálneho rozvrhu a dĺžku rozvrhu získaného algoritmom LPT porade $C_{\max}(OPT)$, $C_{\max}(LPT)$. Potom platí:*

$$\frac{C_{\max}(LPT)}{C_{\max}(OPT)} \leq \frac{4}{3} - \frac{1}{3m}$$

Dôkaz.
Fixujme m a hľadajme najmenej početnú množinu úloh $\mathcal{J} = \{J_1, J_2, \dots, J_n\}$, pre ktorú neplatí tvrdenie vety, t.j.

$$\frac{C_{\max}(LPT)}{C_{\max}(OPT)} > \frac{4}{3} - \frac{1}{3m} \qquad (**)$$

Nech LPT rozvrh definuje permutácia $[i]$. Potom posledne zaradená úloha $J_{[n]}$ aj skončí ako posledná – t.j. skončí v čase $C_{\max}(LPT)$. Keby totiž skončila skôr, LPT rozvrh množiny $\mathcal{J}' = \mathcal{J} - \{J_{[n]}\}$ skončí v čase $C'_{\max}(LPT) = C_{\max}(LPT)$, ale pre dĺžku optimálneho rozvrhu $C'_{\max}(OPT)$ pre množinu $\mathcal{J}'$ bud platiť $C'_{\max}(OPT) \leq C_{\max}(OPT)$. V tom prípade by

$$\frac{C'_{\max}(LPT)}{C'_{\max}(OPT)} \geq \frac{C_{\max}(LPT)}{C_{\max}(OPT)} > \frac{4}{3} - \frac{1}{3m}. \qquad (***)$$

Dostali by sme tak menej početnú množinu úloh $\mathcal{J}'$, pre ktorú platí (**).

V "najmenšom" protipríklade teda úloha $J_{[n]}$ skončí v čase $C_{\max}(LPT)$ a teda začne v čase $C_{\max}(LPT) - p[n]$, pre ktorý platí:

$$C_{\max}(LPT) - p[n] \leq \frac{1}{m}\sum_{i=1}^{n-1} p[i],$$

z čoho dostaneme

$$C_{\max}(LPT) \leq p_{[n]}\left(1 - \frac{1}{m}\right) + \frac{1}{m}\sum_{i=1}^{n} p[i] = p_{[n]}\left(1 - \frac{1}{m}\right) + \frac{1}{m}\sum_{i=1}^{n} p_i.$$

Pretože pre dĺžku optimálneho rozvrhu platí:

$$C_{\max}(OPT) \geq \frac{1}{m}\sum_{i=1}^{n} p_i,$$

môžeme postupne písať

$$\frac{4}{3} - \frac{1}{3m} < \frac{C_{\max}(LPT)}{C_{\max}(OPT)} \leq \frac{p_{[n]}\left(1 - \frac{1}{m}\right) + \frac{1}{m}\sum_{i=1}^{n} p_i}{C_{\max}(OPT)} \leq$$
$$\leq \frac{p_{[n]}\left(1 - \frac{1}{m}\right)}{C_{\max}(OPT)} + \frac{\frac{1}{m}\sum_{i=1}^{n} p_i}{C_{\max}(OPT)} \leq \frac{p_{[n]}\left(1 - \frac{1}{m}\right)}{C_{\max}(OPT)} + 1$$

Dostali sme teda

$$\frac{4}{3} - \frac{1}{3m} < \frac{p_{[n]}\left(1 - \frac{1}{m}\right)}{C_{\max}(OPT)} + 1,$$

$$\frac{1}{3} - \frac{1}{3m} < \frac{p_{[n]}\left(1 - \frac{1}{m}\right)}{C_{\max}(OPT)}$$

$$\frac{1}{3}\left(1 - \frac{1}{m}\right).C_{\max}(OPT) < p_{[n]}\left(1 - \frac{1}{m}\right)$$

$$C_{\max}(OPT) < 3p_{[n]}$$

Pretože prevšetky i je $p_{[i]} \geq p_{[n]}$, z poslednej nerovnosti vyplýva, že pre každý stroj môžu byť v optimálnom rozvrhu naplánované najviac dve úlohy. Ak však existuje optimálny rozvrh s najviac dvoma úlohami pre každý stroj, potom je LPT-rozvrh optimálny. $\square$

Ďalšie výsledky pre systémy s paralelnými strojmi

ALGORITMUS PRE MODEL $Pm||\overline{F}$ BEZ PREEMPCIE.

KROK 1: Vytvorme poradie úloh

$$p_{[1]} \leq p_{[2]} \leq \cdots \leq p_{[n]}. \tag{1.7}$$

KROK 2: V tomto poradí priraďujme úlohy strojom, ktoré sa najskôr uvoľnia.

Algoritmus dáva optimálny rozvrh. Možno ho použiť aj pre modely s preempciou, pretože v tomto prípade tvoria rozvrhy bez prerušenia dominantnú množinu.

Model $Pm|prec|\mathbf{F_{max}}$s precedenčnou reláciou bez preempcie. .
Zavedením precedenčnej relácie na množine úloh $\mathcal{J}$ sa väčšina úloh stáva NP-ťažkou. Sú však niektoré jednoduché prípady, pre ktoré existuje polynomiálny algoritmus. Jedným z nich je problém $Pm|p_i = 1,\ intree|F_{\max}$, v ktorom má každá úloha jednotkový čas spracovania, t.j. $p_i = 1$ pre $i = 1, 2, \ldots, n$ a precedenčná relácia má tvar montážneho stromu – každý uzol má najviac jedného následníka. Tu dáva optimum nasledujúci algoritmus:

HUOV ALGORITMUS PRE SYSTÉM $Pm|p_i = 1,\ intree|F_{\max}$.

KROK 1: Terminálnym úlohám (bez následníkov) dáme značku a_i nulovú, ostatným priradíme značku podľa úrovne (dĺžky maximálnej dráhy od daného prvku ku niektorému z prvkov bez následníka). V priebehu výpočtu budeme označovať $\mathcal{I}$ množinu nezaradených úloh, $\mathcal{I}_0$ množinu tých úloh z $\mathcal{I}$, ktoré v $\mathcal{I}$ nemajú predchodcov. Ďalej označme $\langle t, t+1)$ časový interval, do ktorého budeme v kroku 2 zaraďovať úlohy.
Inicalizačne položme $t = 0$, $\mathcal{I} = \mathcal{J}$, $\mathcal{I}_0 \subseteq \mathcal{J}$ množina úloh bez predchodcov.

KROK 2: Ak je $|\mathcal{I}_0| \leq m$ (kde m je počet strojov) priradíme všetky úlohy z $\mathcal{I}_0$ strojom do časového intervalu $\langle t, t+1)$, ostatné stroje necháme pre tento časový interval voľné. Ak je $|\mathcal{I}_0| > m$, priradíme do časového intervalu $\langle t, t+1)$ m úloh s najväčšími značkami.

KROK 3: Zaradené úlohy vylúčime z množiny nezaradených úloh $\mathcal{I}$, a pokiaľ $\mathcal{I} \neq \varnothing$, aktualizujeme $\mathcal{I}_0$, položime $t := t + 1$ a ideme na krok 2.

Model $Pm|intree, pmtn|\mathbf{F_{max}}$ s precedenčnou reláciou v tvare montážneho stromu.
Ak je prerušenie operácií povolené a precedenčná relácia je v tvare montážneho stromu, môžeme na hľadanie optimálneho rozvrhu s kritériom $\mathbf{F_{max}}$ použiť nasledujúci algoritmus, ktorý nájde optimum aj v prípade nerovnakých dôb spracovania úloh.

MUNTZOV - COFFMANOV ALGORITMUS.

FÁZA ZNAČENIA:

KROK 1: Pre každú úlohu J_i priradíme značku a_i. Pre terminálne úlohy (úlohy bez následníkov) položme $a_i = p_i$.

KROK 2: Ak je priamy následník J_k úlohy J_i označený, položíme $a_i = p_i + a_k$. Ak ešte existuje neoznačená úloha, opakuj Krok 2.

FÁZA PRIRAĎOVANIA ČIASTOK ÚLOH

KROK X: Označme a inicializujme: čas zaradenia najurgentnejších úloh $t := 0$, okamžitý počet voľných strojov $h := m$, množina úloh, ktoré ešte neboli úplne zaradené $\mathcal{Z} = \mathcal{J}$, $\mathcal{I} \subseteq \mathcal{Z}$ množina nezaradených úloh bez predchodcov v $\mathcal{Z}$, $\mathcal{I}_0 \subseteq \mathcal{I}$ podmnožina s rovnakou maximálnou značkou a_i.

KROK 3: Úlohám z $\mathcal{I}_0$ s maximálnou hodnotou a_i, ktorých počet označme k, prideľujeme stroje. Ak je $k \leq h$, pridelíme každej takejto úlohe J_i celý stroj, čo označíme ako $c_i = 1$ a znížime počet voľných strojov $h := h - k$. Z $\mathcal{I}$ vylúčime úplne zaradené úlohy a upravíme $\mathcal{I}_0$, $k := |\mathcal{I}_0|$. Zostávajúcu kapacitu rozdelíme medzi úlohy z $\mathcal{I}_0$ s ďalšou najvyššou hodnotou a_i. Ak $k \leq h$, znovu postupujeme ako v predchádzajúcom odstavci. Ak $k > h$, potom pre každú takúto úlohu položíme $c_i = \frac{h}{k}$ (rozdelíme zostávajúcu kapacitu h pre k úloh) a položíme $h := 0$.
Takto postupujeme dovtedy, kým $h > 0$, alebo kým úplne nezaradíme všetky úlohy z $\mathcal{I}$.

KROK 4: Vypočítame najmenší časový okamžik $\tau > t$, kedy nastane jedna zo situácií:

a) Niektorá z úloh zaradených v kroku 3 je ukončená.
b) Vznikne situácia, že existujú dve úlohy J_i, J_j, pre ktoré platí $a_i - c_i.(\tau - t) \geq a_j - c_j.(\tau - t)$, ale $c_i < c_j$

Vynecháme dokončené úlohy z $\mathcal{Z}$, u ostatných redukujeme doby spracovania o realizované časti, t.j. $a_i = a_i - c_i.(\tau - t)$, $h := m$, aktualizujeme $\mathcal{Z}$, $\mathcal{I}$, $\mathcal{I}_0$, zmeníme začiatok ďalšieho rozvrhovacieho intervalu $t := \tau$ a vraciame sa na krok 3.
Výsledkom tejto fázy je rastúca postupnosť časových okamžikov $\tau_0 = 0, \tau_1, \tau_2, \dots, \tau_r$ určujúcich časové intervaly $\langle \tau_{i-1}, \tau_i)$. Predchádzajúci postup priraďuje každému takémuto časovému intervalu úlohy $J_{k_1}, J_{k_2}, \dots, J_{k_i}$, z ktorých žiadne dve nie sú v precedenčnej relácii a z ktorých sa majú v tomto časovom intervale vykonať čiastky $c_{k_1}, c_{k_2}, \dots, c_{k_i}$.

KONEČNÁ FÁZA ROZVRHOVANIA:

KROK 5: Časové okamžiky $\tau_1, \tau_2, \dots, \tau_r$, v ktorých dochádza v predchádzajúcom kroku k prerozdeleniu kapacity, delia trvanie rozvrhu na intervaly, kedy sa môžu prekrývať iba vzájomne nezávislé operácie. Použitím McNaughtonovho algoritmu v každom z týchto intervalov dostávame optimálny rozvrh.

Paralelné uniformné stroje

Systémy s paralelnými uniformnými strojmi sú charakteristické tým, že čas spracovania p_{ij} úlohy J_i na stroji M_j sa dá napísať ako $p_{ij} = p_i.q_j$, kde q_j sa nazýva faktor rýchlosti stroja M_j. V literatúre sa niekedy používa i zápis $p_{ij} = \dfrac{p_i}{v_j}$, kde $v_j = \dfrac{1}{q_j}$ sa nazýva rýchlosť stroja M_j.

Keďže systémy s preempciou dávali jednoduchšie výsledky pre paralelné identické stroje, budeme sa najprv zaoberať systémamami $Qm|pmnt|C_{\max}$. Skúmajme najprv veľmi špeciálny systém $Q2|pmnt, n = 2|C_{\max}$, t.j. systém s dvoma uniformnými strojmi M_1, M_2 s rýchlosťami v_1, v_2 a dvoma úlohami J_1, J_2 s parametrami p_1, p_2.

Nech x je časť úlohy J_1, ktorá sa vykoná na stroji M_2 (potom jej časť $p_1 - x$ sa vykoná na stroji M_1. Podobne nech $p_2 - y$ z úlohy J_2 sa vykoná na stroji M_2 a časť y na stroji M_1. Stroj M_1 bude pracovať $t_1 = \frac{(p_1-x)+y}{v_1}$, stroj M_2 pobeží $t_2 = \frac{(p_2-y)+x}{v_2}$. Ak sa má minimalizovať $C_{\max}$, mali by oba stroje bežať rovnaký čas, lebo inak by sa nevyužila ich kapacita naplno. Z $t_1 = t_2$ máme

$$\frac{(p_1 - x) + y}{v_1} = \frac{(p_2 - y) + x}{v_2},$$

odkiaľ

$$\frac{x-y}{v_1} - \frac{x-y}{v_2} = \frac{p_1}{v_1} - \frac{p_2}{v_2},$$

$$(x-y).\left(\frac{1}{v_1} - \frac{1}{v_2}\right) = \frac{p_1}{v_1} - \frac{p_2}{v_2},$$

$$(x-y) = \frac{\left(\frac{p_1}{v_1} - \frac{p_2}{v_2}\right)}{\left(\frac{1}{v_1} - \frac{1}{v_2}\right)} \tag{1}$$

Aby sa operácie neprekrývali, musí byť čas spracovania časti x úlohy J_1 na stroji M_2 rovný času spracovanie časti y úlohy J_2 na stroji M_1:

$$\frac{x}{v_2} = \frac{y}{v_1},$$

odkiaľ:

$$y = x.\frac{v_1}{v_2} \tag{2}$$

Dosadením za y z (1) do (2) dostávame:

$$(x-y) = x - x.\frac{v_1}{v_2} = x.\left(1 - \frac{v_1}{v_2}\right) = x.\left(\frac{v_2 - v_1}{v_2}\right) = \frac{\left(\frac{p_1}{v_1} - \frac{p_2}{v_2}\right)}{\left(\frac{1}{v_1} - \frac{1}{v_2}\right)} \tag{3}$$

odkiaľ:

$$x = \frac{\left(\frac{p_1}{v_1} - \frac{p_2}{v_2}\right)}{\left(\frac{1}{v_1} - \frac{1}{v_2}\right)}.\left(\frac{v_2}{v_2 - v_1}\right) = \frac{\left(\frac{p_1 v_2 - p_2 v_1}{v_1 v_2}\right)}{\left(\frac{v_2 + v_1}{v_1 v_2}\right)}.\left(\frac{v_2}{v_2 - v_1}\right) \tag{4}$$

$$x = \frac{(p_1 v_2 - p_2 v_1).v_2}{(v_2^2 - v_1^2)} \tag{5}$$

S využitím (2) a (5) možno písať:

$$y = \frac{(p_1 v_2 - p_2 v_1).v_1}{(v_2^2 - v_1^2)} \tag{6}$$

Keďže podľa predpokladov $p_1 \geq p_2$ a $v_1 \leq v_2$, je $(p_1v_2 - p_2v_1) \geq 0$, $(v_2^2 - v_1^2) \geq 0$, a preto je aj $x \geq 0$, $y \geq 0$.

Aby riešenie bolo prípustné, musí byť aj $(p_1 - x) \geq 0$, a tiež $(p_2 - y) \geq 0$.

$$p_1 - x = p_1 - \frac{(p_1v_2 - p_2v_1).v_2}{(v_2^2 - v_1^2)} = \frac{\left(p_1v_2^2 - p_1v_1^2 - p_1v_2^2 + p_2v_1v_2\right)}{(v_2^2 - v_1^2)} = $$
$$= \frac{\left(p_2v_1v_2 - p_1v_1^2\right)}{(v_2^2 - v_1^2)} = \frac{(p_2v_2 - p_1v_1).v_1}{(v_2^2 - v_1^2)}$$

$$p_2 - y = p_2 - \frac{(p_1v_2 - p_2v_1).v_1}{(v_2^2 - v_1^2)} = \frac{\left(p_2v_2^2 - p_2v_1^2 - p_1v_1v_2 + p_2v_1^2\right)}{(v_2^2 - v_1^2)} = $$
$$= \frac{\left(p_2v_2^2 - p_1v_1v_2\right)}{(v_2^2 - v_1^2)} = \frac{(p_2v_2 - p_1v_1).v_2}{(v_2^2 - v_1^2)}$$

Vidíme, že $(p_1 - x) \geq 0$ a tiež $(p_2 - y) \geq 0$ práve vtedy, keď

$$p_1v_1 \leq p_2v_2,$$

čo nastane práve vtedy, keď

$$\frac{p_1}{v_2} \leq \frac{p_2}{v_1}.$$

Ak čas spracovania väčšej úlohy na rýchlejšom stroji je väčší, než čas spracovania menšej úlohy na pomalšom stroji, nemožno úplne využiť oba stroje.

VETA. *Majme systém $Qm|pmtn|C_{\max}$. Nech platí:*

$$p_1 \geq p_2 \geq, \ldots, \geq p_n, \qquad v_1 \geq v_2 \geq, \ldots, \geq p_m.$$

Potom

$$C_{\max} \geq \left(\frac{p_1}{v_1}, \frac{p_1 + p_2}{v_1 + v_2}, \ldots, \frac{\sum_{i=1}^{m-1} p_i}{\sum_{j=1}^{m-1} v_j}, \frac{\sum_{i=1}^{n} p_i}{\sum_{j=1}^{m} v_j}\right).$$

Dôkaz.
Spracovanie všetkých úloh musí byť aspoň tak dlhé, ako spracovanie najväčšej úlohy na najrýchlejšom stroji, čo predstavuje prvý člen na maxima na pravej strane. Takisto $C_{\max}$ musí byť väčšie ako čas, potrebný na spracovanie dvoch najväčších úloh na dvoch najrýchlejších strojoch pracujúcich rovnaký čas, čo vyjadruje druhý člen maxima na prvej strane. Ďalšie členy maxima až na posledný sa určia tým istým spôsobom. Posledný člen maxima vyjadruje, že spracovanie celej dávky nemôže byť kratšie, ako spracovanie všetkých úloh na všetkých strojoch bežiacich rovnaký čas. □.

ALGORITMUS LRPT–FM PRE $Qm|pmtn|C_{\max}$.

V každom časovom okamžiku t určíme ρ_i – nepracovanú časť úlohy J_i. Zoraďme úlohy

$$\rho_{[1]} \geq \rho_{[2]} \geq \cdots \geq \rho_{[n]}. \qquad \text{(LRPT)}$$

V tomto poradí priraďme m úloh s najväčšími nespracovanými časťami strojom tak, že úlohy s najväčším nespracovaným zvyškom priradíme najrýchlejšiemu stroju, druhú úlohu v poradí LRPT priradíme druhému najrýchlejšiemu stroju atď. Na práve popísané pravidlo sa literatúra odvoláva ako na LRPT–FM (longest remaining processing time on fastest machine – nadlhší zostávajúci čas spracovania na najrýchejlšom stroji).

Algoritmus LRPT–FM má však nevýhodu – veľmi často vyžaduje rescheduling po veľmi krátkom intervale ϵ, čím dochádza k nekonečnému počtu prerušení. Možno ho však použiť vtedy, keď možné časy prerušení obmedzíme na konečnú diskrétnu množinu.

Systém $Qm|p_i = 1|C_{\max}$.

Označme:

$$x_{ijk} = \begin{cases} 1, & \text{ak sa úloha } J_i \text{ spracuje ako } k\text{-ta úloha na stroji } M_j, \\ 0 & \text{inak.} \end{cases}$$

Definujme

$$c_{ijk} = \frac{k}{v_j}.$$

Veličina c_{ijk} predstavuje čas unkončenia úlohy J_i, ak bola zaradená na k-tom mieste poradia pre stroj M_j. Hľadanie najkratšieho rozvrhu možno formulovať akko nasledujúcu úlohu bivalentného lineárneho programovania:

$$\text{Minimalizovať } \max_{i,j,k}\{c_{ijk}x_{ijk}\} \tag{A}$$

$$\text{za predpokladov: } \sum_{j=1}^{m}\sum_{k=1}^{n} x_{ijk} = 1 \quad \forall i = 1, 2, \ldots, n \tag{B}$$

$$\sum_{i=1}^{n} x_{ijk} = 1 \quad \forall j = 1, 2, \ldots, m, \ \forall k = 1, 2, \ldots, n \tag{C}$$

$$x_{ijk} \geq 1 \quad \forall i, k = 1, 2, \ldots, n, \quad \forall j = 1, 2, \ldots, m. \tag{D}$$

Podmienky (B) hovoria, že každá úloha bude do rozvrhu zaradená práve raz. Podmienky (C) hovoria, že ako k-ta úloha na stroji M_j bude zaradená práve jedna úloha.

Všimnime si, že ak platí (B), potom C_i čas ukončenia úlohy J_i možno vyjadriť ako

$$C_i = \sum_{j=1}^{m}\sum_{k=1}^{n} c_{ijk}x_{ijk} = \sum_{j=1}^{m}\sum_{k=1}^{n} \frac{k}{v_j} x_{ijk} \tag{E}$$

S využitím (E) možno vyjadriť viaceré kriteriálne funkcie ako napr. $\sum w_i C_i$ ako lineárnu funkciu a riešiť príslušnú minimalizačnú úlohu ako špeciálny prípad dopravnej resp. priraďovacej úlohy.

Existuje ešte iný prístup k riešenému problému.

Sevastjanovov algoritmus pre $Qm|p_i = 1|C_{\max}$.

.

Označme $C' = \dfrac{n}{\sum_{j=1}^{n} v_j}$. Potom C' je dolnou hranicou pre $C_{\max}$, ktorú možno dosiahnuť preemptívnym rozvrhovaním. Označme

$$\begin{aligned} k_1 &= [C'v_1] \\ k_2 &= [C'v_1] \\ \ldots&\ldots \\ k_m &= [C'v_1], \end{aligned}$$

kde $[x]$ označuje celú časť čísla x. Priraďme stroju M_i k_i úloh. Ostane nám $l = n - \sum_{j=1}^{m} k_j$ nepriradených úloh. Pretože $n = \sum_{j=1}^{m} C'.v_j$, platí:

$$l = n - \sum_{j=1}^{m} k_j = n - \sum_{j=1}^{m} [C'v_j] = \sum_{j=1}^{m} C'.v_j - \sum_{j=1}^{m} [C'v_j] =$$
$$= \sum_{j=1}^{m} (C'.v_j - [C'v_j]) \leq m - 1$$

Ostalo teda $m - 1$ nepriradených úloh. Tieto priradíme strojom nasledujúcou procedúrou:

Vyberieme stroj M_j, pre ktorý je podiel $\dfrac{k_j + 1}{v_j}$ najmenší a priradíme mu jednu z úloh. Položíme:

$$k_j := k_j + 1, \quad l = l - 1.$$

Ak $l = 0$, sú priradené všetky úlohy. Inak priraďovaciu procedúru zopakujeme.

Problém $Rm|pmtn|C_{\max}$

Nasledujúci postup je vhodný pre systémy $Rm|pmtn|C_{\max}$ $Qm|pmtn|C_{\max}$.

Označme $x_{ij} \in \langle 0, 1\rangle$ tú pomernú časť úlohy J_i, ktorá sa má spracovať na stroji M_j. Doba spracovania tejto časti úlohy bude $p_{ij}.x_{ij}$. Dolný odhad pre $C_{\max}$ dostaneme riešením nasledujúcej úlohy lineárneho programovania:

$$\text{Minimalizovať} \quad C \tag{A}$$

za predpokladov:

$$C - \sum_{i=1}^{n} p_{ij} x_{ij} \geq 0 \quad \forall j = 1, 2, \ldots, m \tag{B}$$

$$C - \sum_{j=1}^{m} p_{ij} x_{ij} \geq 0 \quad \forall i = 1, 2, \ldots, n \tag{C}$$

$$\sum_{j=1}^{m} p_{ij} x_{ij} = 1 \quad \forall i = 1, 2, \ldots, n \tag{D}$$

$$x_{ij} \geq 0 \tag{E}$$

Podmienky (B) hovoria, že dĺžka spracovania dávky je väčšia alebo rovná ako čas spracovania všetkých čiastok úloh pre každý stroj. Podmienky (C) vyjadrujú, že dĺžka rozvrhu je väčšia, než doba spracovania každej úlohy. Podmienky (D) hovoria, že súčet čiastok úloh pre všetky stroje dá celú úlohu.

Označme C^* optimálnu hodnotu kriteriálnej funkcie a x^*_{ij} optimálne riešenie, $t_{ij} = p_{ij}x^*_{ij}$ čas spracovnia časti úlohy J_i na stroji M_j pre $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, m$.

Výsledkom riešenia tejto úlohy lineárneho programovanie je nielen dolný odhad pre $C_{\max}$, ale aj určenie, aká čiastka úlohy J_i sa má vykonať na stroji M_j – označme ju $\hat{o}_{ij}$. Máme teda množinu strojov $\mathcal{M} = \{M_1, M_2, \ldots, M_m\}$, množinu úloh $\mathcal{J} = \{J_1, J_2, \ldots, J_m\}$. Každá z úloh sa skladá z čiastok $J_i = \{\hat{o}_{i1}, \hat{o}_{i2}, \ldots, \hat{o}_{im}\}$, pričom $\hat{o}_{i1}$ sa má spracovať na stroji M_1, $\hat{o}_{i2}$ na stroji M_2, atď. až $\hat{o}_{im}$ na stroji M_m s časmi spracovania $t_{i1}, t_{i2}, \ldots, t_{im}$, pričom nezáleží na poradí spracovania a je dovolené prerušenie čiastok $\hat{o}_{ij}$.

Táto úloha je vlastne problémom $Om|pmtn|C_{\max}$. Nasledujúci algoritmus je teda algoritmom na zaradenie čiastok úloh pre jednotlivé stroje v systéme $Rm|pmtn|C_{\max}$, ale aj plnohodnotným algoritmom pre hľadanie optimálneho rozvrhu pre $Om|pmtn|C_{\max}$.

Označme $\mathbb{T} = \{t_{ij}\}$ maticu typu $n \times m$. Riadky matice (je ich n) odpovedajú úlohám, m stĺpcov matice odopvedá strojom. Riadok matice $\mathbb{T}$ nazveme kritickým, ak $\sum_{j=1}^{m} t_{ij} = C^*$. V matici $\mathbb{T}$ môže existovať najviac m kritických riadkov, pretože

$$C^* m \geq \sum_{j=1}^{m} \sum_{i=1}^{n} t_{ij} = \sum_{i=1}^{n} \left(\sum_{j=1}^{m} t_{ij} \right). \tag{X}$$

Keby existovalo viac než m kritických riadkov, vonkajšia suma na pravej strane (X) by obsahovala viac než m sčítancov väčších než m, a teda vy bola väčšia než $C^* m$.

Stĺpec j matice $\mathbb{T}$ nazveme kritickým, ak $\sum_{i=1}^{n} t_{ij} = C^*$. Zostrojme štvorcovú diagonálnu maticu $\mathbb{Y} = \{y_{ij}\}$ typu $m \times m$, kde $y_{jj} = C^* - \sum_{i=1}^{n} t_{ij}$.

Definujme maticu $\mathbb{V}$ typu $(n + m) \times m$ predpisom

$$\mathbb{V} = \begin{pmatrix} \mathbb{T} \\ \mathbb{Y} \end{pmatrix}$$

Týmto sme k riadkom úloh $J_1, J_2 \ldots, J_n$ dodali riadky ďalším m fiktívnych úloh $J_{n+1}, J_{n+2} \ldots, J_{n+m}$, ktorými sa vyrovná zaťaženie všetkých strojov na C^* a ktorých zaradenie do rozvrhu v nejakom intervale $\langle b, c)$ bude znamenať, že v tomto intervale bude príslušný stroj bez práce.

Z prvkov matice $\mathbb{V}$ chceme vybrať m-prvkovú tzv. dekrementačnú množinu $\mathcal{U}$ kladných prvkov tak, aby:

- z každého stĺpca matice $\mathbb{V}$ bol v $\mathcal{U}$ práve jeden
- z každého riadku matice $\mathbb{V}$ bol v $\mathcal{U}$ najviac jeden
- z každého kritického riadku matice $\mathbb{V}$ bol v $\mathcal{U}$ práve jeden

Toto možno dosiahnuť nájdením prípustného riešenia nasledujúcich obmedzujúcich podmienok:

$$\begin{aligned}
\sum_{i=1}^{n+m} x_{ij} = 1 \qquad & \forall j = 1, 2, \ldots, m \\
\sum_{j=1}^{m} x_{ij} = 1 \qquad & \forall i = 1, 2, \ldots, n+m, \ i \text{ kritické} \\
\sum_{j=1}^{m} x_{ij} \le 1 \qquad & \forall i = 1, 2, \ldots, n+m, \ i \text{ nekritické} \\
x_{ij} \le \mathrm{signum}(v_{ij}) \qquad & \forall i = 1, 2, \ldots, n+m, \quad \forall j = 1, 2, \ldots, m. \\
x_{ij} \in \{0, 1\} \qquad & \forall i = 1, 2, \ldots, n+m, \quad \forall j = 1, 2, \ldots, m.
\end{aligned}$$

Bivalentná premenná $x_{ij} = 1$ práve vtedy, keď do množiny $\mathcal{U}$ vyberieme prvok v_{ij} a

$$\mathrm{signum}(v) = \begin{cases} \;\;\,1 & \text{ak } v > 0 \\ \;\;\,0 & \text{ak } v = 0 \\ -\;1 & \text{ak } v < 0 \end{cases}$$

Inou možnosťou nájsť množinu $\mathcal{U}$ je nasledovný postup. Zostrojíme bipartitný digraf $\mathbb{G} = (\{z, u\} \cup V_1 \cup V_2, H)$, kde $V_1 = \{M_1, M_2, \ldots, M_m\}$, $V_1 = \{J_1, J_2, \ldots, J_n + m\}$, z je fiktívny zdroj, u fiktívne ústie a H obsahuje všetky hrany typu (z, M_j) s dolnou i hornou kapacitou rovnou 1, (M_j, J_i) také, že $v_{ij} > 0$ s dolnou kapacitou 0 a hornou kapacitou 1 a všetky hrany typu (J_i, u) pre ktoré je horná kapacita rovná 1 a dolná 1, ak je riadok i kritický, 0 ak riadok i nie je kritický. Ak nájdeme v digrafe $\mathbb{G}$ prípustný tok, hrany typu M_j, J_i, ktorými tečie jednotkový tok, určujú hľadanú m-prvkovú dekrementačnú množinu $\mathcal{U}$.

Algoritmus.

KROK 1: $C := C^*$, $v_{ij} := t_{ij} = p_{ij}.x^*_{ij}$ pre $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, m$, $v_{n+j,j} := C^* - \sum_{i=1}^{n} v_{ij}$, $t := 0$

KROK 2: Nájdeme dekrementačnú množinu $\mathcal{U}$ a vypočítame

$$v_{\min} = \min_{v_{ij} \in \mathcal{U}} \{v_{ij}\} \qquad v_{\max} = \max_{i \in \{i' | v_{i'j} \notin \mathcal{U} \text{ pre } j=1,2,\ldots,m\}} \left\{ \sum_j v_{ij} \right\}$$

$$\delta = \min\{v_{\min}, C^* - v_{\max}\}$$

KROK 3: Do intervalu $\langle t, t + \delta$ zaradíme δ jednotiek z čiastok určených dekrementačnou množinou $\mathcal{U}$.

KROK 4: $\forall v_{ij} \in \mathcal{U}$ položíme

$$\begin{aligned} v_{ij} &:= v_{ij} - \delta, \\ t &:= t + \delta, \\ C &:= C - \delta, \end{aligned}$$

a aktualizujeme maticu $\mathbb{V}$. Ak $C > 0$ GOTO Krok 2, inak STOP.