CHAPTER II

SYSTÉMY S JEDNÝM STROJOM

Predpokladajme, že máme rozvrhovací systém, v ktorom každá úloha $J_i \in \mathcal{J}$ pozostáva s jedinej operácie $o_{i1} = o_i$ a pre každú operáciu je určený jediný stroj, na ktorom sa táto operácia môže vykonávať. Ďalej predpokladajme, že medzi úlohami z $\mathcal{J}$ neexistuje žiadna precedenčná relácia a pokiaľ dve úlohy J_i, J_j nevyžadujú ten istý stroj, spracovanie jednej nijako neovplyvňuje spracovanie druhej. Vtedy sa množina úloh $\mathcal{J}$ rozpadne na disjunktné podmnožiny $\mathcal{J}_1, \dots, \mathcal{J}_m$ také, že všetky úlohy ležiace v podmnožine $\mathcal{J}_j$ vyžadujú na svoje spracovanie (len) stroj M_j. V takomto prípade je spracovanie úloh z rôznych podmnožín na rôznych strojoch nezávislé, a daný rozvrhovací problém sa rozpadá na m nezávislých rozvrhovacích problémov – každý s jedným strojom.

Budeme teda predpokladať, že máme danú n–prvkovú množinu úloh $\mathcal{J} = \{J_1, J_2, \dots, J_n\}$, z ktorých každá pozostáva z jedinej operácie o_i – t.j. počet operácií i-tej úlohy $g_i = 1$ pre $1 \le i \le n$, počet strojov v systéme $m = 1$, pričom stroj je stále k dispozícii. Ďalej predpokladáme, že všetky úlohy z množiny $\mathcal{J}$ vstúpia do prázdneho systému naraz v čase 0, t.j. $r_i = 0$ pre $1 \le i \le n$ Ďalej predpokladáme, že pre každú úlohu J_i je dané $p_i = p_{i1}$ trvanie jej (jedinej) operácie a d_i plánovaný čas ukončenia.

Pre rozvrhovacie systémy s jedným strojom bez prerušenia operácií je rozvrh **S** určený postupnosťou disjunktných časových intervalov $(b_1, c_1), (b_2, c_2), \dots, (b_n, c_n)$, kde (b_i, c_i) je interval, v ktorom sa spracováva úloha J_i, pričom dĺžka tohoto intervalu $c_i - b_i = p_i$. Pre okamžik ukončenia úlohy C_i platí $C_i = c_i$.

Pre rozvrhovacie systémy s jedným strojom s prerušením operácií je rozvrh **S** určený postupnosťou disjunktných časových intervalov (b_{ik}, c_{ik}), $i = 1, 2, \dots, n$, $k = 1, 2, \dots, k_i$, kde k_i je počet čiastkových intervalov v ktorých sa vykonáva úloha J_i. Predpokladáme $b_{ik} < c_{ik} < b_{i(k+1)}$ pre každé i, k také,že $1 \le i \le n$ a $1 \le k < k_i$. Musí byť $\sum_{k=1}^{k_i} (c_{ik} - b_{ik}) = p_i$.

Označme W_i celkovú dobu čakania úlohy J_i v systéme, C_i okamžik jej ukončenia a F_i dobu jej pobytu v systéme. (W_i, C_i, F_i sú výstupné hodnoty riešenia rozvrhovacieho problému). Potom $C_i = F_i = p_i + W_i$, pretože $r_i = 0$

Napriek tomu, že rozvrhovacie problémy s jedným strojom sú relatívne jednoduché, majú svoj význam už tým, že sa tu dajú skúmať rôzne druhy rozvrhov a rôzne druhy rozvrhovacích kritérií a ich výsledky sú dostatočne názorné. Niekoľko výsledkov o týchto systémov možno použiť i v zložitejších prípadoch buď ako smerovanie bádania, alebo sú základom pre približné riešenie reálnych úloh. Existuje však množstvo reálnych situácií, ktoré možno dobre modelovať systémami s jedným strojom.

Typeset by $\mathcal{AMS}$-TEX

Definícia. Nech $\mathbf{S}$ je nejaký rozvrh pre systém s jedným strojom. Umelý prestoj v rozvrhu $\mathbf{S}$ je časový interval (t_1, t_2), v ktorom sa na stroji nevykonáva žiadna úloha, pričom v systéme je úloha čakajúca na spracovanie.

Definícia. Hovoríme, že $\mathcal{S}$ je dominantná množina rozvrhov pre kritérium f, ak v $\mathcal{S}$ existuje aspoň jeden optimálny rozvrh z hľadiska kritéria f.

Veta 1. *Majme systém $1||f$, resp. $1|pmtn|f$, kde f je regulárne optimalizačné kritérium. Potom existuje taký optimálny rozvrh $\mathbf{S}$ vzhľadom na kritérium f, v ktorom neexistuje umelý prestoj.*

Dôkaz pre systémy bez prerušenia (preempcie).

Nech rozvrh $\mathbf{S}$ definovaný postupnosťou intervalov $(b_1, c_1), (b_2, c_2), \ldots, (b_n, c_n)$ obsahuje umelý prestoj (t_1, t_2). Potom interval (t_1, t_2) je disjunktný s každým intervalom (b_i, c_i), t.j. pre každé i $1 \leq i \leq n$ platí buď $c_i \leq t_1$, alebo $t_2 \leq b_i$.

Zostrojme rozvrh $\mathbf{S}'$ definovaný postupnosťou intervalov $(b'_1, c'_1), (b'_2, c'_2), \ldots, (b'_n, c'_n)$ definovaný nasledovne:

$$b'_i = b_i \quad c'_i = c_i \qquad \text{ak } c_i \leq t_1 \tag{1.1}$$

$$b'_i = b_i - (t_2 - t_1) \quad c'_i = c_i - (t_2 - t_1) \qquad \text{ak } t_2 \leq b_i \tag{1.2}$$

Je vidieť, že pre okamžiky ukončenia úloh platí $C'_i = C_i$ alebo $C'_i = C_i - (t_2 = t_1)$ a teda $C'_i \leq C_i$. Pretože f je regulárne kritérium – t.j. je nerastúcou funkciou v každej svojej zložke, je $f(\mathbf{S}') \leq f(\mathbf{S})$. $\square$

Dôkaz pre systémy s prerušením operácií je analogický.

Veta. *Majme systém $1|pmtn|f$, kde f je regulárne optimalizačné kritérium. Potom existuje taký optimálny rozvrh $\mathbf{S}$ vzhľadom na kritérium f, v ktorom niet prerušení operácií.*

Nech $t_1, t_2, \ldots, t_r$ sú práve tie časové okamžiky, v ktorých sa začína alebo končí vykonávanie niektorej úlohy, resp. jej prerušenej časti. Keďže dominantnou množinou rozvrhov pre regulárne kritérium je množina rozvrhov bez umelých prestojov, v čase t_i pre $1 < i < r$ súčasne končí vykonávanie jedného úseku úlohy a súčasne začína spracovanie iného úseku inej úlohy. Ak v čase t_i končí spracovanie celej úlohy J_j, potom $C_j = t_i$.

Nech $\mathbf{S}$ je rozvrh s prerušením s časmi ukončenia $C_1, C_2, \ldots, C_n$, nech J_j je úloha, ktorá sa spracováva aspoň v dvoch úsekoch. Nech spracovanie predposledného úseku úlohy J_j je naplánované do časového intervalu $\langle t_p, t_{p+1})$ spracovanie posledného úseku úlohy J_j do časového intervalu $\langle t_q, t_{q+1})$, kde $p + 1 < q$. Ak spojíme spracovanie predposlednej a poslednej čiastky úlohy J_i do jedného časového úseku $\langle t_q + t_p - t_{p+1}, t_{q+1})$ a časy spracovania úsekov $\langle t_i, t_{i+1})$ posunieme do intervalov $\langle t_i + t_p - t_{p+1}, t_{i+1} + t_p - t_{p+1})$ dostaneme nový rozvrh $\mathbf{S}'$ s časmi ukončenia $C'_1, C'_2, \ldots, C'_n$, pre ktoré je $C'_i \leq C_i$ pre každé $i = 1, 2, \ldots, n$, z čoho pre regulárne kritérium f vyplýva $f(\mathbf{S}') \leq f(\mathbf{S})$. Takto postupne môžeme odstrániť všetky prerušenia v rozvrhu $\mathbf{S}$. $\square$

Vidíme teda, že optimálne riešenie vzhľadom na regulárne kritérium budeme hľadať medzi rozvrhmi bez prerušenia a bez umelých prestojov. Pre rozvrhovacie problémy bez preempcie pôjde vlastne len o to, v akom poradí naukladáme tesne za sebou intervaly spracovania pre úlohy z danej množiny $\mathcal{J}$. Rozvrh bude daný poradím – permutáciou úloh z $\mathcal{J}$. Preto tieto rozvrhy budeme označovať ako permutačné rozvrhy. Počet rôznych permutácií n-prvkovej množiny je $n!$, takže pokusy riešiť rozvrhovacie úlohy tohoto typu prezretím všetkých možností sú i pre malé čísla n odsúdené na neúspech. Pre mnohé optimalizačné kritériá však existujú veľmi jednoduché metódy príslušných rozvrhovacích problémov.

Nech $\mathcal{I}$ je množina prirodzených čísel typu $\mathcal{I} = \{1, 2, \dots, n\}$. Potom permutáciu π na množine $\mathcal{I}$ možno pokladať za vzájomne jednoznačné zobrazenie π konečnej množiny $\mathcal{I}$ do seba, čo možno symbolicky zapísať $\pi : \mathcal{I} \to \mathcal{I}$. Prvku $i \in \mathcal{I}$ priradí iný prvok $j \in \mathcal{I}$ podľa predpisu $j = \pi[i]$.

Majme dve permutácie $\pi,\ \tau$. Potom môžeme definovať permutáciu φ – zloženie permutácií $\pi,\ \tau$ predpisom $\varphi[i] = \pi\left[\tau[i]\right]$. Píšeme $\varphi = \pi.\tau$.

K najjednoduchším permutáciám patria permutácie π_{kl}, ktoré zamieňajú len poradie dvoch prvkov na miestach $k,\ l$, t.j.

$$\begin{aligned} \pi_{kl}[k] &= l \\ \pi_{kl}[l] &= k \\ \pi_{kl}[i] &= i \quad \text{pre } i \neq k,\ i \neq l \end{aligned}$$

Tvrdenie. *Každú permutáciu φ možno napísať ako zloženie konečného počtu permutácií typu $\pi_{k(k+1)}$, $1 \le k < n$.*

Dôkaz.

Nech $\varphi[1] = k$, potom $\pi_{12}.\pi_{23}.\dots.\pi_{(k-1)k}[1] = k$

Postupnými výmenami susedných prvkov sme dostali žiadaný prvok k na prvé miesto. Takisto postupnými výmenami typu $\pi_{23}.\pi_{34}.\dots.\pi_{(k'-1)k'}$ dostaneme prvok $\varphi[2]$ na druhé miesto pričom zachováme prvok na prvom mieste atď.

Cvičenia.

1. Vyjadrite permutáciu π_{kl} ako zloženie permutácií typu $\pi_{j(j+1)}$.
2. Dokážte, že všetky permutácie možno dostať ako zloženie cyklickej permutácie a permutácie π_{12}.

Pretože sa permutácia často týka indexov, z dôvodu zjednodušenia zápisu sa často permutácia zapisuje iba ako $j = [i]$ namiesto $j = \pi[i]$. V tomto zmysle budeme používať tento zápis i my.

Ak máme dané úlohy $\mathcal{J} = \{J_1, J_2, \dots, J_n\}$, zápisom [1] označíme číslo úlohy, ktorá sa vykoná prvá, [2] je číslo úlohy, ktorá sa vykoná na druhom mieste atď. Zápis $[3] = 7$ hovorí, že ako tretia sa vykoná úloha J_7. Podobne $C_{[i]}$ znamená dobu ukončenia úlohy, ktorá sa vykoná ako i–ta.

Minimalizácia času ukončenia poslednej úlohy

Majme ľubovoľný permutačný rozvrh v systéme $1||f$. Pre časy ukončenia úloh v ľubovoľnom permutačnom rozvrhu platí

$$C_{[1]} \le C_{[2]} \le \cdots \le C_{[n]}, \tag{1.3}$$

čo nie je nič iné ako samozrejmé tvrdenie, že úloha, ktorá sa spracuje ako prvá, sa ukončí skôr, než úloha spracovaná ako druhá atď. V tomto prípade $b_{[1]} = 0$ $C_{[1]} = c_{[1]} = p_{[1]}$, $C_{[i]} = c_{[i]} = \sum_{k=1}^{i} p_{[k]}$.

Všimnime si, že

$$\begin{aligned} C_{[1]} &= p_{[1]} \\ C_{[2]} &= C_{[1]} + p_{[2]} = p_{[1]} + p_{[2]} \\ &\dots\dots \\ C_{[n]} &= C_{[n-1]} + p_{[n]} = \sum_{i=1}^{n} p_{[i]} \end{aligned}$$

$$C_{\max} = C_{[n]} = \sum_{k=1}^{n} p_{[k]} = \sum_{i=1}^{n} p_i \tag{1.4}$$

Posledná rovnosť platí preto, lebo sčítanie ľubovoľného počtu sčítancov nezávisí od ich poradia. Vidíme, že $C_{\max}$ je pre všetky permutácie rovnaké.

Pretože všetky úlohy prídu do systému naraz v čase 0, sú všetky $r_i = 0$, je

$$F_{[i]} = C_{[i]} - r_{[i]} = C_{[i]} = \sum_{k=1}^{i} p_{[k]}. \tag{1.5}$$

Vidíme, že $C_{\max} = F_{\max}$ a táto hodnota nezávisí na permutácii $[i]$. Z hľadiska optimalizačného kritéria $C_{\max}$ resp. $F_{\max}$ sú teda všetky permutačné rozvrhy ekvivalentné.

Minimalizácia kritéria $(Cw)_{\max}$**,** $(Fw)_{\max}$**.**

Skúmajme kritérium $(Cw)_{\max} = \max_{1\le i\le n}\{C_i.w_i\}$ resp. $(Fw)_{\max} = \max_{1\le i\le n}\{F_i.w_i\}$. Vidíme, že $(Cw)_{\max} = (Fw)_{\max}$.

VETA. *Optimálny rozvrh* **S** *pre systém* $1||(Fw)_{\max}$*, resp.* $1||(Cw)_{\max}$ *je*

$$w_{[1]} \ge w_{[2]} \ge \dots \ge w_{[n]}\ . \tag{1}$$

Dôkaz.

Nech **S** je optimálny rozvrh s časmi ukončenia jednotlivých úloh $C_1, C_2, \dots, C_n$ a nech neplatí (1), t.j. existuje k, $1 \ge k \ge n-1$ také, že $w_{[k]} < w_{[k+1]}$. Definujme rozvrh $\mathbf{S}'$ určený permutáciou $[i]'$, pre ktorú platí: $[i]' = [i]$ pre $i \ne k$, $i \ne (k+1)$, $[k]' = [k+1]$, $[k+1]' = [k]$. Časy ukončenia jednotlivých úloh podľa rozvrhu $\mathbf{S}'$ označme $C'_1, C'_2, \dots, C'_n$. Podľa (1.4) môžeme písať $C'_{[i]'} = C_{[i]}$ pre všetky $i \ne k$, $i \ne k+1$, lebo

$$C'_{[i]'} = \sum_{j=1}^{i} p_{[j]'} = \begin{cases} \sum_{j=1}^{i} p_{[j]} = C_{[i]} \quad \text{ak } i < k, \\ \sum_{j=1}^{k} p_{[j]'} = C_{[k-1]} + p_{[k+1]} \quad \text{ak } i = k, \\ \sum_{j=1}^{k+1} p_{[j]'} = C_{[k-1]} + p_{[k+1]} + p_{[k]} = C_{[k+1]} \quad \text{ak } i = k, \\ \sum_{j=1}^{i} p_{[j]'} = C_{[k-1]} + p_{[k+1]} + p_{[k]} + \\ \qquad + \sum_{j=k+2}^{i} p_{[j]} = C_{[i]} \quad \text{ak } i > k+1. \end{cases}$$

Pretože $w_{[i]} = w_{[i]'}$ pre $i \ne k$, $i \ne k+1$ a $w_{[k]} = w_{[k+1]'}$, $w_{[k+1]} = w_{[k]'}$, je $C_{[i]}.w_{[i]} = C'_{[i]'}.w_{[i]'}$ pre všetky $i \ne k$, $i \ne k+1$. Platí:

$$C_{[k]}.w_{[k]} < C_{[k+1]}.w_{[k+1]},$$

lebo $C_{[k]} < C_{[k+1]}$ a $w_{[k]} < w_{[k+1]}$. Ďalej platí:

$$C'_{[k]'}.w_{[k]'} = C'_{[k]'}.w_{[k+1]} \leq C'_{[k+1]'}.w_{[k+1]} = C_{[k+1]}.w_{[k+1]},$$

a tiež

$$C'_{[k+1]'}.w_{[k+1]'} = C_{[k+1]}.w_{[k]} \leq C_{[k+1]}.w_{[k+1]},$$

z čoho už vidieť, že

$$\max_i\{C'_{[i]'}.w_{[i]'}\} \leq \max_i\{C_{[i]}.w_{[i]}\}.$$

Dokázali sme, ku každému, teda aj optimálnemu rozvrhu **S** dokážeme zostrojiť lepší alebo rovnako dobrý rozvrh **S**$'$, v ktorom platí (1).

Treba ešte ukázať, že pre optimalitu rozvrhu **S** stačí, aby platilo (1). Na to stačí ukázať, že ak platí $w[k] = w[k+1]$, permutácia $[*]'$ definuje permutačný rozvrh s rovnakou kriteriálnou funkciou ako pôvodná permutácia $[*]$. Rovnakým postupom dôkazu s predpokladom $w[k] = w[k+1]$ dôjdeme k záveru $C'_{[k+1]'}.w_{[k+1]'} = C_{[k+1]}.w_{[k+1]}$ a teda

$$\max_i\{C'_{[i]'}.w_{[i]'}\} \leq \max_i\{C_{[i]}.w_{[i]}\}.$$

□

Minimalizácia priemernej doby pobytu úloh v systéme

Priemernú dobu $\overline{F}$ pobytu úloh v systéme vypočítame ako

$$\begin{aligned}\overline{F} = \frac{1}{n}\sum_{i=1}^{n} F_{[i]} = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{i} p_{[k]} = \\ \underbrace{p_{[1]}}_{F_{[1]}} + \underbrace{p_{[1]} + p_{[2]}}_{F_{[2]}} + \underbrace{p_{[1]} + p_{[2]} + p_{[3]}}_{F_{[3]}} + \cdots + \underbrace{p_{[1]} + p_{[2]} + \cdots + p_{[n]}}_{F_{[n]}} = \\ \frac{1}{n}\sum_{i=1}^{n}(n-i+1)p_{[i]}\end{aligned} \quad (1.6)$$

Veta. *Majme systém* $1||\overline{F}$. *Priemerná doba pobytu úlohy v systéme* $\overline{F}$ *je minimálna práve vtedy, keď*

$$p_{[1]} \leq p_{[2]} \leq \cdots \leq p_{[n]}. \quad (1.7)$$

Dôkaz:

Dokážeme viac. Ukážeme, že hodnota $\overline{F_i^{\alpha}}$, kde $\alpha > 0$ je minimálna práve vtedy, keď platí (1.7). Tvrdenie vety je potom špeciálnym prípadom dokázaného tvrdenia pre $\alpha = 1$.

Majme rozvrh **S** určený permutáciou $[i]$ a nech existuje k, $1 \leq k < n$ také, že $p_{[k]} > p_{[k+1]}$. Definujme rozvrh **S**$'$ určený permutáciou $[i]'$, pre ktorú platí:

$[i]' = [i]$ pre $i \neq k,\ i \neq (k+1)$, $[k]' = [k+1]$, $[k+1]' = [k]$. Potom bude platiť $F_{[i]'} = F_{[i]}$ pre všetky $i \neq k,\ i \neq (k+1)$ a tak možno písať

$$\begin{aligned}f(\mathbf{S}) - f(\mathbf{S}') &= \frac{1}{n}\left(\sum_{i=1}^{n} F_{[i]}^{\alpha} - \sum_{i=1}^{n} {F'_{[i]'}}^{\alpha}\right) = \\ &= \frac{1}{n}\sum_{i=1}^{n}\left(F_{[i]}^{\alpha} - {F'_{[i]'}}^{\alpha}\right) = \frac{1}{n}\left(F_{[k]}^{\alpha} + F_{[k+1]}^{\alpha} - {F'_{[k]'}}^{\alpha} - {F'_{[k+1]'}}^{\alpha}\right).\end{aligned} \quad (1.8)$$

Platí:

$$F_{[k]} = \sum_{i=1}^{k} p_{[i]} = \sum_{i=1}^{k-1} p_{[i]} \ + p_{[k]} = T + p_{[k]} \tag{1.9}$$

$$F_{[k+1]} = \sum_{i=1}^{k+1} p_{[i]} = \sum_{i=1}^{k-1} p_{[i]} \ + p_{[k]} + p_{[k+1]} = T + p_{[k]} + p_{[k+1]} \tag{1.10}$$

$$F'_{[k]'} = \sum_{i=1}^{k} p_{[i]'} = \sum_{i=1}^{k-1} p_{[i]'} \ + p_{[k]'} = \sum_{i=1}^{k-1} p_{[i]} \ + p_{[k+1]} = T + p_{[k+1]} \tag{1.11}$$

$$\begin{aligned} F'_{[k+1]'} = \sum_{i=1}^{k+1} p_{[i]'} = \sum_{i=1}^{k-1} p_{[i]'} \ + p_{[k]'} + p_{[k+1]'} = \sum_{i=1}^{k-1} p_{[i]} \ + p_{[k+1]} + p_{[k]} = \\ = T + p_{[k]} + p_{[k+1]} \end{aligned} \tag{1.12}$$

Po dosadení z () až () do () máme

$$\begin{aligned} & f(\mathbf{S}) - f(\mathbf{S}') = \\ & = \frac{1}{n} \left((T + p_{[k]})^{\alpha} + (T + p_{[k]} + p_{[k+1]})^{\alpha} - (T + p_{[k+1]})^{\alpha} - (T + p_{[k]} + p_{[k+1]})^{\alpha} \right) = \\ & = \frac{1}{n} \left((T + p_{[k]})^{\alpha} - (T + p_{[k+1]})^{\alpha} \right) \end{aligned} \tag{1.13}$$

Pretože podľa predpokladu $p_{[k]} > p_{[k+1]}$, je i $(T + p_{[k]}) > (T + p_{[k+1]})$. Pretože x^{α} je rastúcou funkciou premennej x, je $(T + p_{[k]})^{\alpha} > (T + p_{[k+1]})^{\alpha}$, z čoho už vyplýva, že $f(\mathbf{S}) - f(\mathbf{S}') > 0$. Ak neplatí (1.7), rozvrh $\mathbf{S}$ nie je optimálny.

Na to, že z (1.7) vyplýva optimalita rozvrhu $\mathbf{S}$ stačí ukázať, že ak pre iný rozvrh $\mathbf{S}'$ s permutáciou $[i]'$ platí

$$p_{[1]'} \leq p_{[2]'} \leq \cdots \leq p_{[n]'} \tag{1.14,}$$

potom $f(\mathbf{S}) = f(\mathbf{S}')$. Permutáciu $\mathbf{S}$ však možno dostať z permutácie $\mathbf{S}$ aplikovaním konečného počtu susedných zámen na miestach k, $(k+1)$ takých, že $p_{[k]} = p_{[k+1]}$. Zmenu účelovej funkcie takejto zámeny popisuje vzťah (1.13), ktorého pravá strana bude v tomto prípade rovná 0. Keďže sa hodnota kriteriálnej funkcie zachová pri každej susednej výmene, zachová sa i pri výslednej permutácii $[i]'$ definujúcej rozvrh $\mathbf{S}'$.

Cvičenia.

1. Dokážte, že hodnota $\overline{F}$ pre rozvrh $\mathbf{S}$ daný permutáciou $[i]$ je maximálna práve vtedy, keď platí

$$p_{[1]} \geq p_{[2]} \geq \cdots \geq p_{[n]} \ . \tag{1.15}$$

Minimalizácia váženého súčtu dôb pobytu.

VETA 2. *Majme systém* $1||\overline{Fw}$. *Súčet vážených dôb pobytu v systéme* $\overline{Fw}$ *je minimálny práve vtedy, keď*

$$\frac{p_{[1]}}{w_{[1]}} \le \frac{p_{[2]}}{w_{[2]}} \le \cdots \le \frac{p_{[n]}}{w_{[n]}} \ . \tag{1.16}$$

Dôkaz.

Nech $1 \le k < n$. Definujme rozvrh $\mathbf{S}'$ s permutáciou $[i]'$, pre ktorú je $[k]' = [k{+}1]$, $[k+1]' = [k]$ a $[i]' = [i]$ pre $i \ne k,\ i \ne k+1$. Nech $f(\mathbf{S}) = \sum_{i=1}^{n} F_{[i]}.w[i]$. Potom dobu pobytu úlohy $[i]$ $F_{[i]}$ podľa rozvrhu $\mathbf{S}$ a dobu pobytu úlohy $[i]'$ $F'_{[i]'}$ podľa rozvrhu $\mathbf{S}'$ platí

$$F'_{[i]'} = F_{[i]} \qquad \text{pre } i \ne k,\ i \ne k+1 \ .$$

Pre $i = k,\ i = k+1$ platí (1.9) až (1.12).

$$\begin{aligned} f(\mathbf{S}) - f(\mathbf{S}') &= \sum_{i=1}^{n} F_{[i]}.w_{[i]} - \sum_{i=1}^{n} F_{[i]'}.w_{[i]'} = \sum_{i=1}^{n} \big(F_{[i]}.w_{[i]} - F_{[i]'}.w_{[i]'}\big) = \\ &= F_{[k]}.w_{[k]} + F_{[k+1]}.w_{[k+1]} - F'_{[k]'}.w_{[k]'} - F'_{[k+1]'}.w_{[k+1]'}. \end{aligned} \tag{1.17}$$

Po dosadení za $F_{[k]},\ F_{[k+1]},\ F'_{[k]'},\ F'_{[k+1]'}$ z (1.9) až (1.12) dostávame:

$$\begin{aligned} &f(\mathbf{S}) - f(\mathbf{S}') = \\ &= (T{+}p_{[k]}).w_{[k]}{+}(T{+}p_{[k]}{+}p_{[k+1]}).w_{[k+1]}{-}(T{+}p_{[k+1]}).w_{[k]'}{-}(T{+}p_{[k]}{+}p_{[k+1]}).w_{[k+1]'} = \\ &= (T{+}p_{[k]}).w_{[k]}{+}(T{+}p_{[k]}{+}p_{[k+1]}).w_{[k+1]}{-}(T{+}p_{[k+1]}).w_{[k+1]}{-}(T{+}p_{[k]}{+}p_{[k+1]}).w_{[k]} = \\ &\qquad = p_{[k]}.w_{[k+1]} - p_{[k+1]}.w_{[k]} \end{aligned}$$

Ak $\dfrac{p_{[k]}}{w_{[k]}} > \dfrac{p_{[k+1]}}{w_{[k+1]}}$ je $f(\mathbf{S}) - f(\mathbf{S}') > 0$. Ak $\dfrac{p_{[k]}}{w_{[k]}} = \dfrac{p_{[k+1]}}{w_{[k+1]}}$ je $f(\mathbf{S}) - f(\mathbf{S}') = 0$.

Podobne, ako pri dôkaze predchádzajúcej vety z posledného výsledku vyvodíme, že:

1. ak nie je splnená podmienka (1.16), príslušný rozvrh $\mathbf{S}$ nie je optimálny z hľadiska kritéria $\overline{Fw}$.
2. ak pre permutácie rozvrhov $\mathbf{S}$, $\mathbf{S}'$ príslušné permutácie splňujú podmienku (1.16), potom oba rozvrhy majú tú istú hodnotu kriteriálnej funkcie $\overline{Fw}$.

Z oboch posledných skutočností už vyplýva tvrdenie vety. □

Minimalizácia maxima omeškania resp. maxima časovej diferencie.

VETA 2. *Majme systém* $1||T_{\max}$ *resp.* $1||L_{\max}$, $\mathbf{S}$ *permutačný rozvrh. Na to, aby maximum omeškania* $T_{\max} = \max_{1\le i\le n}\{T_i\}$ *resp. maximum časových diferencií* $L_{\max} = \max_{1\le i\le n}\{L_i\}$ *jednotlivých úloh bolo minimálne stačí, aby*

$$d_{[1]} \le d_{[2]} \le \cdots \le d_{[n]} \ . \tag{1.18}$$

Dôkaz.

Nech $d_{[k]} > d_{[k+1]}$ pre nejaký permutačný rozvrh $\mathbf{S}$ s permutáciou $[i]$. Definujme rozvrh $\mathbf{S}'$ s permutáciou $[i]'$, pre ktorú je $[k]' = [k+1]$, $[k+1]' = [k]$ a $[i]' = [i]$ pre $i \neq k$, $i \neq k+1$. Chceme ukázať, že $f(\mathbf{S}') \leq f(\mathbf{S})$ pre kriteriálnu funkciu $f = L_{\max}$ resp. $f = T_{\max}$. Potom $C'_{[i]} = C_{[i]}$ pre $i \neq k$ a $L'_{[i']} = L_{[i]}$ pre $i \neq k$, $i \neq k+1$. Platí:

$$L_{[k]} = C_{[k]} - d_{[k]} \quad L_{[k+1]} = C_{[k+1]} - d_{[k+1]}$$

$$L'_{[k]'} = C'_{[k]} - d_{[k+1]} \quad L'_{[k+1]'} = C_{[k+1]} - d[k]$$

Pretože $C_{[k]} < C_{[k+1]}$ a $d_{[k]} > d_{[k+1]}$, je

$$L_{[k]} < L_{[k+1]}.$$

Na to, aby $f(\mathbf{S}') \leq f(\mathbf{S})$ stačí ukázať, že $L'_{[k]'} < L_{[i+1]}$ a $L'_{[k+1]'} < L_{[i+1]}$.
Pretože $C'_{[k]} < C'_{[k+1]'} = C_{[k+1]}$ je:

$$L'_{[k]'} = C'_{[k]'} - d_{[k+1]} < C_{[k+1]} - d_{[k+1]} = L_{[k+1]}$$

Pretože $d_{[k]} > d_{[k+1]}$, platí:

$$L'_{[k+1]'} = C_{[k+1]} - d_{[k]} < C_{[k+1]} - d_{[k+1]} = L_{[k+1]}$$

Pre kritérium $T_{\max}$ je dôkaz analogický. □

CVIČENIA.

1. Môže existovať viac optimálnych rozvrhov pre systém $1||f$, kde f je jedno z kritérií $(Fw)_{\max}$, $\overline{F}$, $\overline{Fw}$, $T_{\max}$, $L_{\max}$?
2. Formulujte postačujúcu podmienku pre to, aby rozvrh $\mathbf{S}$ maximalizoval
   - minimálnu časovú odchýlku $L_{\min} = \min_i\{L_i\}$
   - minimálne omeškanie $T_{\min} = \min_i\{T_i\}$ (Riešením je permutačný rozvrh, ktorý usporiada úlohy neklesajúco podľa časovej rezervy $d_i - p_i$.)

Minimalizácia váženého súčtu časových diferencií.
Skúmajme teraz kritériá $\overline{L} = \sum_{i=1}^n L_i$ a $\overline{Lw} = \sum_{i=1}^n L_i.w_i$
Platí

$$\overline{L} = \sum_{i=1}^n L_i = \sum_{i=1}^n (C_i - d_i) = \sum_{i=1}^n (F_i - d_i) = \sum_{i=1}^n F_i - \sum_{i=1}^n d_i \tag{1.19}$$

$$\overline{Lw} = \sum_{i=1}^n L_i.w_i = \sum_{i=1}^n (C_i - d_i).w_i = \sum_{i=1}^n (F_i - d_i).w_i = \sum_{i=1}^n F_i.w_i - \sum_{i=1}^n d_i.w_i \tag{1.20}$$

Keďže $\sum_{i=1}^n d_i$ vo vzťahu (1.19) resp. $\sum_{i=1}^n d_i.w_i$ vo vzťahu (1.20) sú konštanty, kritérium $\overline{L}$ je minimálne práve vtedy, keď je minimálne kritérium $\overline{F}$, kritérium $\overline{Lw}$ je mionimálne práve vtedy, keď je minimálne kritérium $\overline{Fw}$.

S použitím viet *, * možno tvrdiť:

VETA. *Majme systém* $1||\overline{L}$. *Priemerná časová diferencia úloh v systéme* $\overline{L}$ *je minimálna práve vtedy, keď*

$$p_{[1]} \le p_{[2]} \le \cdots \le p_{[n]} \ .$$

VETA. *Majme systém* $1||\overline{Lw}$. *Súčet* $\overline{Lw}$ *vážených časových diferencií úloh v systéme je minimálny práve vtedy, keď*

$$\frac{p_{[1]}}{w_{[1]}} \le \frac{p_{[2]}}{w_{[2]}} \le \cdots \le \frac{p_{[n]}}{w_{[n]}} \ .$$

Minimalizácia váženého maxima časovej odchýlky resp. omeškania.

Pre kritériá $(Lw)_{\max} = \max_i\{L_i.w_i\}$ a $(Tw)_{\max} = \max_i\{T_i.w_i\}$ možno nájsť optimálny rozvrh nasledujúcim algoritmom:

LAWLEROV ALGORITMUS.

Označme $\mathcal{I}$ množinu ešte nezaradených úloh, $p = \sum_{i\in\mathcal{I}} p_i$ celková doba spracovania nezaradených.

KROK 1. Položme $p := \sum_{i=1}^{n} p_i$, $\mathcal{I} := \{1, 2, \ldots, n\}$, $k := n$.
KROK 2. Položme $j :=$ to $i \in \mathcal{I}$, pre ktoré je $w_i(p - d_i)$ (pre kritérium $(Lw)_{\max}$) resp. $\max(0, w_i(p - d_i)$ (pre kritérium $(Tw)_{\max}$) minimálne.
KROK 3. Položme $[k] := j$, $\mathcal{I} := \mathcal{I} - \{j\}$, $k := k - 1$, $p := p - p_j$.
KROK 4. Ak $k = 0$ STOP, inak GOTO KROK 2.

Minimalizácia priemerného omeškania.

Pre rozvrhovaciu úlohu $1||\overline{T}$, kde $\overline{T} = \sum_{i=1}^{n} T_i$ $(T_i = \max(0, C_i - d_i)$ dlho nebol známy polynomiálny algoritmus, ani sa nedarilo nič dokázať o jej zložitosti. Tejto úlohe bolo venované enormné úsilie, kým sa v r. 1989 podarilo dokázať, že je NP-ťažká.

Vieme však ukázať niektoré jej vlastnosti.

VETA. *Pre rozvrhovaciu úlohu* $1||\overline{T}$ *platí*

a) Ak permutačný rozvrh $\mathbf{S}$ *s vlastnosťou* $d_{[1]} \le d_{[2]} \le \cdots \le d_{[n]}$ *obsahuje nanajvýš jednu omeškanú úlohu, potom je* $\mathbf{S}$ *optimálny rozvrh.*

b) Ak $d_1 = d_2 = \cdots = d_n$, *potom na to, aby bol rozvrh* $\mathbf{S}$ *optimálny stačí, aby pre príslušnú permutáciu platilo:*

$$p_{[1]} \le p_{[2]} \le \cdots \le p_{[n]} \ .$$

c) Ak pre permutačný rozvrh $\mathbf{S}$ *s vlastnosťou* $p_{[1]} \le p_{[2]} \le \cdots \le p_{[n]}$ *sú všetky úlohy omeškané, potom je* $\mathbf{S}$ *optimálny rozvrh.*

Rozvrhovacia úloha $1||\overline{Tw}$, kde $\overline{Tw} = \sum_{i=1}^{n} T_i.w_i$ patrí medzi NP–ťažké úlohy.

Minimalizácia počtu omeškaných úloh.

Ako poslednú úlohu permutačných rozvrhov uvedieme rozvrhovaciu úlohu, pri ktorej hľadáme rozvrh, ktorý minimalizuje počet omeškaných úloh.

MOOROV ALGORITMUS NA KONŠTRUKCIU ROZVRHU S MINIMOM OMEŠKANÝCH ÚLOH.

KROK 1. Vytvoríme permutačný rozvrh, pre ktorý je

$$d_{[1]} \leq d_{[2]} \leq \cdots \leq d_{[n]}$$

Vytvorme postupnosti $E := \{d_{[1]}, d_{[2]}, \ldots, d_{[n]}\}$, $L := \{\ \}$. Položme aktuálny počet členov postupnosti E $n_E := n$.

KROK 2. Ak žiadna dávka z E nie je omeškaná, postupnosť úloh z E zreťazená s ľubovoľným poradím úloh z L dáva optimálne riešenie. Počet prvkov postupnosti L určuje počet omeškaných úloh. STOP.
Inak pokračuj KROKOM 3.

KROK 3. V postupnosti E nájdeme prvú omeškanú úlohu $[k]$ (je to úloha na k–tom mieste vE). Medzi prvými k úlohami z E nájdeme úlohu $[j]$, pre ktorú je doba spracovania p_j maximálna. Presunieme prvok j z postupnosti E do množiny L. Upravíme patrične permutáciu $[i]$ ($x := [j]$, pre $i = j+1, j+2, \ldots, n_E$ polož $[i] := [i+1]$, $[n_E] := x$, $n_E := n_E - 1$. Potom $E = \{[1], [2], \ldots, [n_E]\}$, $L = \{[n_E+1], [n_E+2], \ldots, [n]\}$.
GOTO KROK 2.

Použitie dynamického programovamia

Ako sme už videli, niektoré regulárne kritériá vedú k NP-ťažkým úlohám. V tejto časti sa budeme zaoberať prípadom, keď regulárne kritérium $f(C_1, C_2, \ldots, C_n)$ možno napísať ako

$$f(C_1, C_2, \ldots, C_n) = \sum_{i=1}^{n} \mathbf{g}_i(C_i), \text{ kde } \mathbf{g}_i \text{ sú neklesajúce funkcie.} \tag{1.23}$$

Nech $\mathcal{J} = \{J_1, J_2, \ldots, J_n\}$, $\mathcal{Z} \subseteq J$, $\mathcal{Z}' = \mathcal{J} \setminus \mathcal{Z}$. Označme

$$q_{\mathcal{Z}} = \sum_{j \in \mathcal{Z}'} p_j = \sum_{j=1}^{n} p_j - \sum_{j \in \mathcal{Z}} p_j \tag{1.24}$$

Poznamenajme, že $q_{\mathcal{J}} = 0$, $q_{\varnothing} = \sum_{i=1}^{n} p_i$.

Ak sú všetky úlohy zo $\mathcal{Z}'$ naplánované pred úlohami zo $\mathcal{Z}$, potom prvá úloha zo $\mathcal{Z}$ začína v čase $q_{\mathcal{Z}}$ bez ohľadu na usporiadanie úloh v $\mathcal{Z}'$ a časy C_i ukončenia úloh zo $\mathcal{Z}$ závisia iba od usporiadania množiny $\mathcal{Z}$.

Ak je poradie úloh $\mathcal{Z}'$, $\mathcal{Z}$ optimálne, potom i usporiadanie úloh v $\mathcal{Z}$ musí byť optimálne bez ohľadu na usporiadanie úloh v $\mathcal{Z}'$.

Pre ľubovoľnú podmnožinu $\mathcal{Z} \subseteq J$ označme

$$G(\mathcal{Z}) = \min \left\{ \sum_{i \in \mathcal{Z}} \mathbf{g}_i(C_i) | \text{úlohy } J_i \in \mathcal{Z} \text{ sú zaradené najskôr v čase } q_{\mathcal{Z}} \right\} \tag{1.25}$$

Potom $G(\mathcal{J})$ je optimálna hodnota účelovej funkcie. Platí:

$$G(\mathcal{J}) = \min_{j \in \mathcal{J}} \{\mathbf{g}_j(p_j) + G(\mathcal{J} - \{j\})\} \tag{1.26}$$

a pre ľubovoľnú podmnožinu $\mathcal{Z} \subseteq J$:

$$G(\mathcal{Z}) = \min_{j \in \mathcal{Z}} \{\mathbf{g}_j(q_{\mathcal{Z}} + p_j) + G(\mathcal{Z} - \{j\})\} \tag{1.27}$$

Špeciálne je:

$$G(\{j\}) = \mathbf{g}_j(q_{\{j\}} + p_j) + G(\varnothing).$$

Pretože však:

$$q_{\{j\}} + p_j = \sum_{i=1}^{n} p_i - \sum_{i \in \{j\}} p_i - p_j = \sum_{i=1}^{n} p_i \quad \text{a } G(\varnothing) = 0,$$

platí:

$$G(\{j\}) = \mathbf{g}_j(\sum_{i=1}^{n} p_j). \tag{1.28}$$

Poznáme $G(\varnothing)$ a podľa (1.28) vieme vypočítať $G(\{j\})$ pre každú jednoprvkovú množinu. Ak poznáme $G(\mathcal{Z})$ pre všetky $(k-1)$-prvkové množiny tvaru $\mathcal{Z} - \{j\}$, potom použitím vzťahu (1.27) vypočítame $G(\mathcal{Z})$ aj pre ľubovoľnú k-prvkovú množinu $\mathcal{Z}$.

Heuristické postupy

Procedúry prehľadávajúce okolie.

Pri permutačných rozvrhoch si možno definovať okolie rozvrhu $\mathbf{S} = J_{[1]}, J_{[2]}, \ldots, J_{[n]}$ ako množinu všetkých rozvrhov, pri ktorých zameníme poradie úloh na dvoch susedných miestach. Podľa inej definície možno okolie rozvrhu $\mathbf{S}$ definovať ako množinu všetkých rozvrhov, ktoré z $\mathbf{S}$ dostaneme zámenou miest dvoch ľubovoľných úloh. Procedúry na prehľadávanie okolia prezrú všetky rozvrhy z okolia daného centrálneho rozvrhu $\mathbf{S}$ a z tých, ktoré majú menšiu hodnotu účelovej funkcie než $f(\mathbf{S})$, sa vyberie jeden s najmenšou hodnotou účelovej funkcie. Tento sa stáva novým centrálnym rozvrhom $\mathbf{S}$, v okolí ktorého sa znovu hľadá rozvrh s minimálnou hodnotou účelovej funkcie. Procedúra končí, keď sa v okolí rozvrhu $\mathbf{S}$ nenájde zlepšujúci rozvrh. Takto nájdené lokálne optimum je však zriedkakedy globálnym.

Na metóde prehľadávania okolí je založená i zložitejšia metóda zvaná *tabu search*. Táto umožňuje po dosiahnutí lokálneho optima aj takú voľbu nového centrálneho prvku, ktorý nezlepšuje kriteriálnu funkciu. Aby sa zabránilo návratu do toho istého lokálneho optima, metóda udržuje zoznam zakázaných prechodov (od toho tabu search). Takto sa umožní dostať sa po niekoľkých zhoršujúcich krokoch do okolia iného (možno lepšieho) lokálneho minima.

Náhodné vzorkovanie

Náhodné vzorkovanie – random sampling – je tiež jednou z metód na riešenie zložitých úloh. Rozvrh – poradie úloh – sa vytvorí náhodne použitím generátora náhodných čísel. Vygenerovanie takéhoto náhodného rozvrhu je rýchle, preto možno v krátkom čase vytvoriť vzorku N náhodných rozvrhov, ku každému vypočítať príslušnú hodnotu účelovej funkcie a rekordný rozvrh si zapamätať. Ak je

p pravdepodobnosť nájdenia optima pri jednom pokuse, pri N–prvkovej vzorke je pravdepodobnosť nájdenia optima $P_N = 1 - (1-p)^N$. Ak hľadáme permutačný rozvrh n úloh, potom je $p = \dfrac{1}{n!}$ a $P_N = 1 - \left(1 - \dfrac{1}{n!}\right)^N$.

Ukazuje sa, že náhodné pravdepodobnosť nájdenia optima je malá, preto je vhodné kombinovať metódu náhodného vzorkovania s metódou prehľadávania okolia, t.j. pre každý náhodne vygenerovaný rozvrh metódou prehľadania jeho okolia nájsť lokálne optimum.

Rozvrhy so započítaním prestavovacích časov

V prechádzajúcich rozvrhovacích úlohách sme ticho predpokladali, že stroj môže spracovávať ďalšiu úlohu bezprostredne po predchádzajúcej úlohe bez akéhokoľvek prestavovania. Vo všeobecnosti však zmena úlohy znamená i prestavenie stroja.

Veľmi často je čas na prestavenie stroja medzi úlohami nezávislý od typu ukončenej a začínajúcej úlohy. V takomto prípade sa tento prestavovací čas jednoducho zahrnie do času vykonávania každej úlohy p_i. S takto modifikovanými časmi vykonávania úloh už možno použiť všetky doteraz diskutované rozvrhovacie modely.

Sú však prípady, keď z charakteru vykonávania úloh vyplývajú prestavovacie doby, ktoré sú závislé na dvojici ukončovanej a začínajúcej úlohy.

Conway uvádza ako príklad takéhoto problému výrobňu farieb. V stroji na výrobu farieb sa vyrábajú farby Biela, Žltá, Modrá, Červená. Po ukončená výroby jednej farby sa musí stroj očistiť od zbytkov predchádzajúcej farby. Zbytky žltej farby neovplyvnia tak výrobu červenej, ako zbytky modrej výrobu bielej. Preto dôkladnosť čistenia stroja – a teda aj nastavovacia doba – závisí od predhádzajúcej a nasledujúcej farby. Nastavovacie doby možno zadať vo forme tabuľky

	B	Ž	M	Č
B	0	1	2	3
Ž	6	0	1	2
M	8	6	0	1
Č	9	8	6	0

Existuje $(4-1)! = 6$ rôznych výrobných cyklov, ktoré majú podľa tabuľky všeobecne rôzne sumárne nastavovacie doby:

1. B – Ž – Č – M – B 1+2+6+8=17
2. B – Ž – M – Č – B 1+1+1+9=12
3. B – Č – Ž – M – B 3+8+1+8=20
4. B – Č – M – Ž – B 3+6+6+6=21
5. B – M – Ž – Č – B 2+6+2+9=19

Tu sa už nastavovacie doby nedajú jednoducho zahrnúť do vykonávacích časov jednotlivých úloh. Existencia nastavovacích časov tu už mení charakter rozvrhovacieho procesu. Označme s_{ij} nastavovací čas pri prechode stroja od úlohy J_i k úlohe J_j pre $i, j = 1, 2, \ldots, n$. Naviac označme s_{0i} dobu nastavovenia stroja z počiatočného stavu pre vykonávanie úlohy J_i.

Potom

$$F_{[1]} = s_{0[1]} + p_{[1]}$$

$$F_{[1]} = F_{[2]} + s_{[1][2]} + p_{[2]}$$

$$F_{[n]} = F_{[n-1]} + s_{[n-1][n]} + p_{[n]}$$

Potom:

$$C_{\max} = F_{\max} = \sum_{i=1}^{n} s_{[i-1][i]} + \sum_{i=1}^{n} p_{[i]} \ .$$

Keďže posledná suma je nezávislá na permutácii i, optimálny permutačný rozvrh z hľadiska kritéria $C_{\max}$ resp. $F_{\max}$ bude ten, ktorý minimalizuje $\sum_{i=1}^{n} s_{[i-1][i]}$.

Úloha minimalizácie sumárnej doby nastavovania stroja je analogická s preslávenou úlohou obchodného cestujúceho. Pri tejto úlohe má obchodný cestujúci navštíviť n miest (vzdialenosti medzi ktorými sú dané maticou $\{s\}_{ij}$ a vrátiť sa do východzieho miesta, pričom treba minimalizovať celkovú prejdenú vzdialenosť. Úloha obchodného cestujúceho má niekoľko matematických modelov. V reči teórie grafov je formulovaná ako hľadanie minimálnej hamiltonovskej kružnice v hranovo ohodnotenom digrafe. Formulácia pomocou celičíselného lineárneho programovania je nasledujúca: Nech x_{ij} je $0-1$ premenná, pre ktorú $x_{ij} = 1$ práve vtedy, keď po úlohe meste i bezprostredne navštívime mesto j. Úloha obchodného cestujúceho je

$$\text{Minimalizovať} \sum_{i=1}^{n} \sum_{j=1}^{n} s_{ij} x_{ij}$$

za predpokladov

$$\sum_{i=1}^{n} x_{ik} \qquad \text{pre } k = 1, 2, \ldots, n \ ,$$

$$\sum_{j=1}^{n} x_{kj} \qquad \text{pre } k = 1, 2, \ldots, n$$

$$x_{ij} = 0 \text{ alebo } 1 \quad \text{pre } i = 1, 2, \ldots, n \text{ pre } j = 1, 2, \ldots, n$$

Ďalšou obmedzujúcou podmienkou je, aby matica x_{ij} definovala jedinú trasu obsahujúcu všetky mestá a končiacu v začiatočnom.

Úloha obchodného cestujúceho sa stala prototypom NP–zložitosti. Existujú totož úlohy, pre ktoré máme algoritmy, ktorých počet krokov možno zhora odhadnúť ako polynomiálnu funkciu ich rozmeru. Pre iné úlohy sa doteraz nepodarilo nájsť algoritmus, ktorý by tieto úlohy vyriešil v polynomiálnom čase. Jednou z takýchto úloh je úloha úloha obchodného cestujúceho. Pre celú triedu matematických problémov sa podarilo ukázať, že sú z hľadiska výpočtovej zložitosti ekvivalentné úlohe obchodného cestujúceho v tomto zmysle: Každú úlohu možno polynomiálnym počtom krokov previesť na druhú – t.j. pri existencii polynomiálneho algoritmu pre jednu by existoval polynomiálny algoritmus riešenia druhej. Týmto úlohám sa hovorí NP-ťažké. Keďže sa doteraz ani sústredeným úsilám nepodarilo nájsť polynomiálny algoritmus pre riešenie žiadnej NP-ťažkej úlohy, začína sa veriť, že takýto algoritmus neexistuje. Existujú však veľmi dobré suboptimálne postupy, ktoré nachádzajú riešenia zložitých kombinatorických úloh s veľmi dobrou hodnotou kriteriálnej funkcie. Pre ďalšie podrobnosti o úlohe obchodného cestujúceho odkazujeme čitateľa na literatúru [].

Nerovnaký príchod úloh do systému

Doteraz sme predpokladali, že všetky úlohy prišli do systému naraz v čase $r_1 = r_2 = \cdots = r_n = 0$. Teraz budeme predpokladať, že $r_i \geq 0$ pre $1 \leq i \leq n$. Ak pre systémy s príchodom všetkých úloh naraz bola dominantnou množinou rozvrhov množina všetkých rozvrhov bez prerušenia a bez umelých prestojov, pri tomto prípade tomu už tak nie je.

Cvičenie. Dokážte, alebo vyvráťte: Ak zakážeme prerušenie vykonávania úloh, potom je rozvrh $\mathbf{S}$ bez umelých prestojov práve vtedy, keď je reprezentovaný permutáciou $r_{[1]} \leq r_{[2]} \leq \cdots \leq r_{[n]}$.

Ak je rozvrh $\mathbf{S}$ bez prerušenia reprezentovaný permutáciou $[i]$, potom pre časy ukončenia $C_{[i]}$ platí

$$C_{[1]} = r_{[1]} + p_{[1]}$$

$$C_{[2]} = \max\left\{C_{[1]}, r_{[2]}\right\} + p_{[2]}$$

$$C_{[i]} = \max\left\{C_{[i-1]}, r_{[i]}\right\} + p_{[i]}$$

$$C_{[n]} = \max\left\{C_{[n-1]}, r_{[n]}\right\} + p_{[n]}$$

Príklad. Nech $r_1 = 0, p_1 = 4, d_1 = 15$, $r_2 = 2, p_2 = 10, d_2 = 12$. Uvažujme regulárne kritérium $\sum T$ resp. $T_{\max}$. Pre poradie vykonávania J_1, J_2 je $C_1 = 4$, $C_2 = 14$, $T_1 = 0$, $T_2 = 2$ pre poradie J_2, J_1 je $C_1 = 16$, $C_2 = 12$ $T_1 = 1$, $T_2 = 0$. Optimálny je teda rozvrh J_2, J_1 v ktorom je umelý prestoj v čase $\langle 0, r_2)$.

Ak dovolíme prerušenie, potom rozvrh, v ktorom sa najprv vykonajú dve jednotky z úlohy J_1, potom celá úloha J_2 a nakoniec zvyšok úlohy J_1 bude mať $C_1 = 14$, $C_2 = 12$, $T_1 = T_2 = 0$. Najlepším je teda rozvrh s prerušením.

Cvičenie. Zostrojte iné príklady s inými kritériami, kedy optimálny rozvrh (bez prerušenia) obsahuje umelý prestoj.

Veta. *V systéme* $1|r_i|C_{\max}$ *s nerovnakými príchodmi úloh pre je optimálny rozvrh* $\mathbf{S}$ *taký, že*

$$r_{[1]} \leq r_{[2]} \leq \cdots \leq r_{[n]}.$$

Dôkaz.
Nech pre niektoré k $1 \leq k < n$ je $r_{[k]} > r_{[k+1]}$. operácia $J_{[k]}$ môže začať najskôr v čase $\max\left\{C_{[k-1]}, r_{[k]}\right\}$, v ktorom je už ale v systéme úloha $J_{[k+1]}$, ktorá by mohla začať už skôr v čase $\max\left\{C_{[k-1]}, r_{[k+1]}\right\}$. Ak by sme len zamenili poradie vykonávania úloh $J_{[k]}, J_{[k+1]}$ na $J_{[k+1]}, J_{[k]}$ s tým, že spracovanie úlohy $J_{[k+1]}$ v novom rozvrhu začne v práve vtedy, kedy začalo spracovanie úlohy $J_{[k]}$ v pôvodnom rozvrhu, nezmenia sa hodnoty $C_{[k+2]}, C_{[k+3]}, \ldots, C_{[n]}$. Spracovanie úlohy $J_{[k+1]}$ však možno v novom rozvrhu posunúť do najskôr možného časového okamžiku $\max\left\{C_{[k-1]}, r_{[k+1]}\right\}$, čím sa môže vytvoriť možnosť posunúť spracovanie niektorých úloh $J_{[k+1]}, J_{[k+2]}, \ldots, J_{[n]}$ do skorších časových intervalov a tak prípadne znížiť hodnotu $C_{[n]}$.
Sériou susedných výmen tak dokážeme ku každému rozvrhu $\mathbf{S}$ vytvoriť rozvrh $\mathbf{S}'$ s permutáciou $r_{[1]} \leq r_{[2]} \leq \cdots \leq r_{[n]}$, pre ktorý platí $C_{\max}(\mathbf{S}') \leq C_{\max}(\mathbf{S})$. $\square$

Ak prerušenie úloh nie je dovolené, je hľadanie optimálneho rozvrhu pre systémy $1|r_i|f$, kde f je jedno z kritérií $L_{\max}$, $T_{\max}$ NP-ťažkým problémom.

Ďalej sa budeme zaoberať systémami $1|r_i, pmtn|f$ s nerovnakým príchodom úloh a s povoleným prerušením úloh.

Algoritmus pre hľadanie optimálneho rozvrhu v systémoch $1|r_i, pmtn|L_{\max}$, $1|r_i, pmtn|T_{\max}$.

KROK 0: Označme pre každú úlohu J_i τ_i čas potrebný na spracovanie jej zvyšku. Inicializačne položme $\tau_i = p_i$.

KROK 1: Označme $\mathcal{E} \subseteq \mathcal{J}$ množinu všetkých nedokončených úloh s rovnakým najskôr možným začiatkom a_1. Nech a_2 je ďalší najskôr možný začiatok spracovania nejakej ešte neukončenej úlohy. Ak všetky neukončené úlohy môžu začať v čase a_1, $a_2 = \infty$.

KROK 2: Z množiny $\mathcal{E}$ vyberieme úlohu i, pre ktorú je $d_i = \min_{j\in\mathcal{E}} d_j$. Určíme $L = min\{p_i, (a_2 - a_1)\}$ a do intervalu $\langle a_1, L)$ zaradíme spracovanie úlohy J_i.

KROK 3: Ak je spracovanie úlohy J_i ukončené, vyradíme ju zo zoznamu nedokončených úloh. Ak úloha J_i nie je ešte ukončená položme $\tau_i = \tau_i - L$.

KROK 4: Ak sú úplne zaradené všetky úlohy STOP. Inak GOTO KROK 1.

Poznámka. Predchádzajúci algoritmus možno použiť i pre konštrukciu optimálneho rozvrhu s kritériom $\overline{C}$, resp. $\overline{F}$, ak v KROKu 2 vyberáme za úlohu J_i tú, pre ktorú platí $p_i = \min_{j\in\mathcal{E}} p_j$.

Precedenčná relácia na množine úloh

Medzi úlohami pre jeden stroj môže existovať neprázdna precedenčná relácia $\prec$. Uvedieme zovšeobecnenie Lawlerovho algoritmu pre systém $1|prec|f$ pre systém s rovnakými príchodmi, neprázdnou precedenčnou reláciou $\prec$ a kritériom $(Tw)_{\max}$, $(Lw)_{\max}$, $(Fw)_{\max}$, resp. $(Cw)_{\max}$. Optimálny rozvrh budeme hľadať medzi permutačnými rozvrhmi bez prerušenia a bez umelých prestojov.

Zovšeobecnený Lawlerov algoritmus pre systémy $1|prec|f$, kde $f = (Tw)_{\max}$, $(Lw)_{\max}$, $(Fw)_{\max}$, resp. $(Cw)_{\max}$.

Označme $\mathcal{I}$ množinu ešte nezaradených úloh. Ďalej označme $\mathcal{I}_0 \subseteq \mathcal{I}$ podmnožinu takých nezaradených úloh, ktoré nemajú v $\mathcal{I}$ následníkov. $p = \sum_{i\in\mathcal{I}} p_i$ celková doba spracovania nezaradených úloh.

KROK 1. Položme $p := \sum_{i=1}^{n} p_i$, $\mathcal{I} := \{1, 2, \ldots, n\}$, nech $\mathcal{I}_0$ je množina úloh, ktoré nemajú následníkov. $k := n$.

KROK 2. Vezmime to $i \in \mathcal{I}_0$, pre ktoré je $f_i(p) = \min_{j\in\mathcal{I}_0} f_j(p)$.
(Pre kritérium $(Lw)_{\max}$) je $f_i(p) = w_i(p - d_i)$, pre kritérium $(Tw)_{\max}$) je $f_i(p) = \max(0, w_i(p - d_i)$, pre kritérium $(Fw)_{\max}$) je $f_i(p) = w_i p$) atď.)
Úlohu J_i zaraďme na k-te miesto tak, že končí v čase p.

KROK 3. Položme $[k] := i$, $\mathcal{I} := \mathcal{I} - \{i\}$, $\mathcal{I}_0$ množina tých úloh z $\mathcal{I}$, ktoré nemajú v $\mathcal{I}$ následníka, $k := k - 1$, $p := p - p_i$.

KROK 4. Ak $k = 0$ STOP, inak GOTO KROK 2.

VETA. *Pre systém* $1|prec|L_{\max}$ *resp.* $1|prec|T_{\max}$ *pre optimalitu rozvrhu* **S** *stačí, aby pre príslušnú permutáciu platilo:*

$$d'_{[1]} \leq d'_{[2]} \leq \cdots \leq d'_{[n]} \,, \tag{1.19}$$

kde

$$d'_i = \min\left(d_i, \min\{d_k \mid i \prec k\}\right)$$

Návod na dôkaz.
Dokazuje sa sporom. Nech **S** je rozvrh daný permutáciou $[*]$ a nech existuje k také, že $d'[k] > d'[k+1]$. Nech $\mathbf{S}'$ je rozvrh určený permutáciou $[*]'$ definovanou $[k]' = [k+1]$, $[k+1]' = [k]$ a $[i]' = [i]$ pre všetky $i \neq k$, $i \neq k+1$. Dôkaz treba ešte dokončiť.

Problém $1|r_i|L_{\max}$, $1|r_i, prec|L_{\max}$

VETA. *Problém* $1|r_i|L_{\max}$ *je NP-ťažký.*

Problém $1|r_i|L_{\max}$ je mimoriadne dôležitý preto, lebo sa často vyskytuje ako podproblém pri heuristických algoritmoch pre job shop a flow shop problém. Napriek tomu, že ide o NP-ťažký problém, existuje pomerne efektívny postup riešenia metódou vetiev a hraníc.

Pri tejto metóde tvoríme permutačný rozvrh postupne od začiatku. Pritom postup modelujeme stromom, ktorého koreň (vrchol úrovne 0) predstavuje prázdny parciálny rozvrh a vetví sa na n vrcholov úrovne 1, ktoré modelujú jednoúlohové praciálne rozvrhy. Každý z vrcholov úrovne 1 sa vetví na $n-1$ vrcholov úrovne 2, ktorých je $n.(n-1)$ a ktoré predstavujú dvojúlohové parciálne rozvrhy atď. Na úrovni $k-1$ je $n.(n-1).\dots.(n-k-1)$ vrcholov predstavujúcich $(k-1)$-členné parciálne rozvrhy a z ktorých sa každý vrchol vetví na $n-k$ vrcholov úrovne k. Ak vrchol na úrovni $k-1$ predstavuje parciálny rozvrh $J_1, J_2, \dots, J_k - 1$, treba uvažovať rozvetvenie do $J_1, J_2, \dots, J_k - 1, J_k$ len vtedy, keď

$$r_k < \min_{i \in \mathcal{I}} \{\max\{C_{k-1}, r_i\} + p_i\} \,, \tag{A}$$

kde $\mathcal{I}$ predstavuje množinu ešte nazaradených úloh, C_{k-1} je čas ukončenia úlohy J_{k-1}. Nech J_l je úloha, ktorá minimalizuje (A), t.j.

$$\max\{C_{k-1}, r_l\} + p_l = \min_{i \in \mathcal{I}} \{\max\{C_{k-1}, r_i\} + p_i\} \,, \tag{B}$$

Ak by sme na k-té miesto rozvrhu zaradili úlohu, ktorá nespĺňa (A), dostali by sme rozvrh, v ktorom sa pred úlohu na k-tom mieste dá vsunúť ešte úloha J_l bez zmeny časovej polohy ostatných úloh, čím dostaneme rozvrh, ktorý nie je horší než **S**. Táto vlastnosť vo väčšine prípadov značne zjednodušuje vetvenie.

Pre každý vrchol stromu – parciálny rozvrh $J_1, J_2, \dots, J_k$ je treba ešte určiť dolnú hranicu kritéria $L_{\max}$, ktorú určíme ako maximum hodnoty $L_{\max}$ pre parciálny rozvrh a hodnoty $L_{\max}$ pre ostávajúceh nezaradené úlohy pri povolení prerušenia vykonávania operácií – čo vedie k riešeniu systému $1|r_i, pmtn|L_{\max}$, pre ktorý je v tejto kapitole uvedený exaktný algoritmus riešenia.

Na prvý pohľad by sa mohlo zdať, že problém $1|r_i, prec|L_{\max}$ je ťažší než ten istý problém bez precedenčnej relácie. Na riešenie tejto úlohy môžeme použiť v podstate ten istý postup vetví a hraníc, ako bol práve popísaný s tým, že pri vetvení naviac kontrolujeme, či pridávaná úloha nenarušuje precedenčnú reláciu, čo je z enumeračného hľadiska výhodné, lebo ešte viac obmedzuje vetvenie.