ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1990
Seria: AUTOMATYKA z.100 Nr kol.1082

Eugeniusz Nowicki

Politechnika Wrocławska
Instytut Cybernetyki Technicznej

ALGORYTMY APROKSYMACYJNE DLA JEDNOMASZYNOWEGO PROBLEMU SZEREGOWANIA ZE ZMIENNYMI CZASAMI WYKONYWANIA ZADAŃ

Streszczenie. W pracy rozważa się problem szeregowania zadań w dyskretnym systemie produkcyjnym z jednym gniazdem krytycznym. Przyjmuje się, że dla każdego zadania określony jest termin gotowości i tzw. końcówka zadania. Ponadto zakłada się, że czas wykonywania zadania można zmieniać w sposób ciągły w pewnym zadanym przedziale. Stawia się problem wyboru kolejności wykonywania zadań oraz ich czasów, tak by minimalizować globalny koszt. W pracy bada się przydatność pewnej klasy algorytmów aproksymacyjnych do rozwiązania tego problemu. Przeprowadza się dla nich analizę najgorszego przypadku w oparciu o zaproponowane postacie dolnych ograniczeń.

1. Wstęp

Niniejsza praca jest kontynuacją szeregu prac poświęconych szeregowaniu zadań na maszynach [6],[10],[14],[18],[19], których wspólnym elementem jest przyjęcie modelu zadania takiego jak w problemach PERT/koszt. W modelu tym zakłada się, że czas wykonywania zadania (czynności) można zmieniać w sposób ciągły w pewnym z góry zadanym przedziale. Zmiana czasu wykonywania zadania powoduje różny koszt wykonywania tego zadania - koszt rośnie, gdy czas maleje. Wzrost kosztu przy zmniejszonym czasie wykonywania jest wynikiem zaangażowania dodatkowych zasobów. Pełną interpretację takiego modelu można znaleźć na przykład w [3]. Konsekwencją przyjętego modelu zadania jest pojawienie się, oprócz tradycyjnego wskaźnika jakości uszeregowania (np. C_{max}, L_{max}, T_{max}, $\Sigma w_i C_i$, $\Sigma w_i T_i$), dodatkowego wskaźnika będącego sumarycznym kosztem wykonywania zadań. Ten ostatni zależy tylko od czasów wykonywania poszczególnych zadań, w odróżnieniu od pierwszego, który zależy także od kolejności ich wykonywania.

Dla każdego klasycznego (ustalone czasy wykonywania zadań) zagadnienia szeregowania zadań, notowanego wg Grahama za pomocą zestawu $\alpha|\beta|\gamma$, można teraz wyróżnić trzy następujące problemy optymalizacyjne. Pierwszy, P1, polega na tym, że na bazie dwóch wskaźników (tradycyjnego γ i dodatkowego) konstruuje się kryterium globalne równe ich kombinacji liniowej i interpretuje jako łączny koszt uszeregowania. W drugim, P2, tylko jeden ze wskaźników traktuje się jako kryterium, a pozostały jako dodatkowe ograniczenie przez ustalenie nieprzekraczalnej jego wartości. Z kolei w trzecim

problemie, P3, najbardziej ogólnym, szuka się zbioru (lub pewnej jego aproksymacji) wszystkich punktów Pareto-optymalnych ze względu na obydwa wskaźniki. Badania nad tym problemem są jednak efektywne tylko w tych wypadkach, gdy istnieją algorytmy wielomianowe dla problemów P1, P2. Przykładowo, istotne rezultaty otrzymano dla zagadnień $1|r_i|C_{max}$, $1||L_{max}$ oraz $1||T_{max}$; kluczem do wyznaczenia całego zbioru punktów Pareto-optymalnych był tutaj fakt, że permutacja optymalna nie zależy od czasów wykonywania zadań. Ogólnie rzecz biorąc, wyniki dotyczące złożoności obliczeniowej dla problemów P1, P2 nie są zachęcające, ponieważ znacznie obniża się (w porównaniu z zagadnieniami tradycyjnymi) bariera, powyżej której pojawiają się problemy NP-trudne. I tak, problemy P1 i P2 dla zagadnienia $F2||C_{max}$ są NP-trudne [14], [11] nawet przy założeniu, że zmienne są czasy wykonywania zadań tylko na jednej maszynie oraz zależność koszt-czas jest liniowa i identyczna dla wszystkich zadań. Podobnie problemy P1, P2 dla zagadnienia $1|r_i|C_{max}$, w którym zmianie podlegają nie czasy wykonywania a momenty gotowości, są nawet silnie NP-trudne, [9],[15]. Na podkreślenie zasługuje tutaj fakt, że mimo iż z NP-trudności problemu P1 dla zagadnienia $\alpha|\beta|\gamma$ nie wynika bezpośrednio NP-trudność problemu P2 dla tego samego zagadnienia, to w obu powyższych wypadkach odpowiednie transformacje wielomianowe są identyczne (z oczywistą różnicą między parametrem określającym górną wartość łącznego kosztu w decyzyjnej wersji P1 a parametrami określającymi górne wartości wskaźnika γ i wskaźnika dodatkowego w decyzyjnej wersji P2). Z powyższych rozważań wynika, że badania nad problemami P1, P2 dla zagadnień $\alpha|\beta|\gamma$ (nawet gdy dla zagadnienia $\alpha|\beta|\gamma$ istnieje algorytm wielomianowy) należy raczej ograniczyć do poszukiwania dobrych algorytmów aproksymacyjnych. Przegląd otrzymanych dotąd rezultatów i dotyczących złożoności obliczeniowej algorytmów wielomianowych i algorytmów aproksymacyjnych można znaleźć w pracach [13], [15].

Przedstawiana praca dotyczy problemu P1 dla zagadnienia $1|r_i,q_i|C_{max}$ i rozszerza wstępne wyniki otrzymane w [12], [15]. Konstruuje się tutaj pewną klasę algorytmów aproksymacyjnych, której parametrem są algorytmy aproksymacyjne dla klasycznego zagadnienia $1|r_i,q_i|C_{max}$ (al. Schrage [17], al. Pottsa [16] oraz al. Halla i Shmoysa [8]). Następnie przeprowadza się do tej klasy algorytmów analizę najgorszego przypadku w oparciu o znane i nowe postacie dolnych ograniczeń.

Praca była finansowana przez RP.I.02 "Teoria sterowania i optymalizacji układów dynamicznych i procesów dyskretnych".

2. Sformułowanie problemu i dolne ograniczenia

Dany jest zbiór n niezależnych zadań ponumerowanych kolejno $1,2,\ldots,n$, które należy wykonać na jednej maszynie. Zakładamy, że: (i) maszyna w każdej chwili czasowej może wykonywać nie więcej niż jedno zadanie oraz zada-

nia wykonywane są bez przerwań, (ii) zadanie j jest gotowe do wykonywania w chwili $r_j \geq 0$, $j=1,2,\ldots,n$, (iii) czas wykonywania zadania j na maszynie jest równy $p_j = a_j - x_j$, $0 \leq x_j \leq u_j$, gdzie x_j określa skrócenie normalnego czasu wykonywania a_j, zaś u_j $(0 \leq u_j \leq a_j)$ maksymalne skrócenie tego czasu, $j=1,\ldots,n$, (iv) zadanie j jest uważane za wykonane po upływie czasu $q_j \geq 0$ od momentu zakończenia jego wykonywania na maszynie, $j=1,2,\ldots,n$.

Niech $\pi=(\pi(1),\ldots,\pi(n))$ określa permutacje elementów zbioru $\langle 1,\ldots,n\rangle$, Π - zbiór wszystkich takich permutacji, a $X = \{x=(x_1,\ldots,x_n): 0 \leq x_j \leq u_j,\ j=1,\ldots,n\}$ - zbiór wszystkich dopuszczalnych wektorów skróceń czasów wykonywania zadań na maszynie. Oznaczmy przez $C_{max}(\pi,x)$ minimalny czas zakończenia wykonywania zadań, przy spełnieniu (i)-(iv), dla kolejności wykonywania zadań określonej przez π oraz czasów wykonywania $p_j = a_j - x_j$, $j=1,\ldots,n$. Zachodzi

$$C_{max}(\pi,x) = \max_{1 \leq i_1 \leq i_2 \leq n} [r_{\pi(i_1)} + \sum_{i=i_1}^{i_2} (a_{\pi(i)} - x_{\pi(i)}) + q_{\pi(i_2)}]. \qquad (1)$$

Przyjmujemy, że koszt skrócenia czasu wykonywania zadania j na maszynie o x_j jednostek równa się $c_j x_j$, gdzie $c_j \geq 0$ oznacza jednostkowy koszt skrócenia. Przyjmujemy dalej, że globalny koszt $K(\pi,x)$, dla ustalonej permutacji $\pi \in \Pi$ oraz wektora skróceń $x \in X$, jest równy kosztowi związanemu z długością uszeregowania $(C_{max}(\pi,x))$ plus sumaryczny koszt wykonywania zadań na maszynie, tzn. ma postać:

$$K(\pi,x) = w * C_{max}(\pi,x) + \sum_{j=1}^{n} c_j x_j , \qquad (2)$$

gdzie $w \geq 0$ - jednostkowy koszt związany z długością uszeregowania, który po odpowiedniej zmianie jednostek będziemy uważać za równy jedności. Ostatecznie problem formułujemy następująco:

Znaleźć kolejność wykonywania zadań $\pi^* \in \Pi$ oraz wektor skróceń $x^* \in X$ spełniające

$$K(\pi^*,x^*) = \min \{K(\pi,x) = C_{max}(\pi,x) + \sum_{j=1}^{n} c_j x_j : \pi \in \Pi,\ x \in X\}. \qquad (3)$$

Sformułowany powyżej problem jest silnie NP-trudny. Wynika to z faktu, iż szczególny jego przypadek z $u_j = 0$, $j=1,\ldots,n$ jest równoważny silnie NP-trudnemu zagadnieniu $1|r_j,q_j|C_{max}$. Zauważmy ponadto, że podobnie jak dla innych problemów typu P1 (patrz np. [10]), nie zmiejszając ogólności rozważań można przyjąć $c_j < 1$, $j=1,\ldots,n$. Rzeczywiście, jeżeli dla pewnego j, $c_j \geq 1$, to istnieje rozwiązanie optymalne (π^*,x^*) z $x_j^* = 0$. Stąd dla tych j, dla których $c_j \geq 1$, można określić nową trójkę u'_j, a'_j, c'_j taką, że $a'_j = a_j$, $u'_j = 0$, a jako c'_j przyjąć dowolną liczbę z $[0,1)$.

Przedstawimy teraz różne postacie dolnych ograniczeń minimalnej wartości

globalnego kosztu $K^*=K(\pi^*,x^*)$. Można je otrzymać przez odpowiednie relaksacje warunków (i)-(iv). Zachodzi

$$C_{max}(\pi,x) \geq \max(R,\ Q,\ \Sigma_{j=1}^{n} a_j - x_j), \qquad \pi\in\Pi,\ x\in X, \tag{4}$$

gdzie

$$R = \max_{1\leq j\leq n} r_j, \qquad Q = \max_{1\leq j\leq n} q_j. \tag{5}$$

Stąd

$$K^* \geq \min_{x\in X} [\max(R,\ Q,\ \Sigma_{j=1}^{n} a_j - x_j) + \Sigma_{j=1}^{n} c_j x_j]. \tag{6}$$

Z (i) i z oczywistej nierówności "$\min_{x\in X} \max_{1\leq i_1\leq i_2\leq n} \geq \max_{1\leq i_1\leq i_2\leq n} \min_{x\in X}$" dostaniemy kolejną postać dolnego ograniczenia

$$K^* \geq \min_{\pi\in\Pi} C_{max}(\pi,x'), \tag{7}$$

gdzie

$$x'_j = (1-c_j)u_j, \quad j=1,\ldots,n. \tag{8}$$

Wykorzystując relaksację wartości q_j lub r_j otrzymamy

$$K^* \geq \min_{\pi\in\Pi,x\in X} K_r(\pi,x) + \min_{1\leq j\leq n} q_j \tag{9}$$

oraz

$$K^* \geq \min_{\pi\in\Pi,x\in X} K_q(\pi,x) + \min_{1\leq j\leq n} r_j. \tag{10}$$

Wielkość $K_a(\pi,x)$, $a\in(r,q)$ jest równa $K(\pi,x)$ odpowiednio przy założeniu, że $q_j=0$, $j=1,\ldots,n$ lub $r_j=0$, $j=1,\ldots,n$. Do rozwiązania zadania $\min_{\pi\in\Pi,x\in X} K_a(\pi,x)$, $a\in(r,q)$ w pracy [13] podano algorytm wielomianowy o złożoności $O(n^2)$. Dla $a=r$ $(a=q)$ permutacja będąca jego rozwiązaniem jest permutacją wg niemalejących r_j (wg nierosnących q_j), a optymalny wektor skróceń wylicza się wg pewnego algorytmu zachłannego. Idea tego ostatniego polega na tym, że wśród wszystkich zadań j, których skracanie czasu wykonywania zmniejsza C_{max}, skracamy zadanie j' o najmniejszej wartości jednostkowego kosztu $c_{j'}$. Skracanie czasu wykonywania zadania j' prowadzimy tak długo, aż zajdzie jedna z dwóch sytuacji: dalsze skracanie nie zmniejsza C_{max} lub dalsze skracanie nie jest już możliwe, tzn. $x_{j'}=u_{j'}$. Proces jest kontynuowany do momentu, w którym nie ma już zadania, którego czas wykonywania można skrócić i skrócenie to zmniejsza C_{max}. Z powyższego wynika, że w przypadkach szczególnych, gdy $q_j=const$, $j=1,\ldots,n$ lub $r_j=const$, $j=1,\ldots,n$, problem P1 i P2 dla zagadnienia $1|r_j,q_j|C_{max}$ ma złożoność wielomianowa. Ponadto, jak już wspominaliśmy we Wstępie, algorytm wyznaczający zbiór wszystkich punktów Pareto-optymalnych (problem P3) ma też złożoność wielomia-

nową (zbiór ten ma postać wypukłej łamanej, a algorytm wylicza wszystkie punkty załamań).

Na koniec zauważmy, że

$$C_{max}(\pi,x) \geq \max_{1\leq i\leq n}(r_i+q_i+a_i-x_i), \quad \pi\in\Pi, \ x\in X.$$

Stąd otrzymamy kolejną postać dolnego ograniczenia

$$K^* \geq \min_{x\in X}\left[\max_{1\leq i\leq n}(r_i+q_i+a_i-x_i) + \sum_{j=1}^{n} c_j x_j\right]. \qquad (11)$$

3. Algorytmy aproksymacyjne

Przed sformułowaniem pewnej klasy algorytmów aproksymacyjnych dla rozważanego problemu (3) przedstawimy znane z literatury algorytmy aproksymacyjne dla klasycznego zagadnienia $1|r_i,q_i|C_{max}$. Przez $C_{max}(\pi)$ oznaczamy termin wykonania wszystkich zadań dla permutacji π. Algorytmy omawiamy w kolejności malejących współczynników najgorszego przypadku [1].

Pierwszy algorytm zaproponowany przez Schrage [17], dla którego współczynnik najgorszego przypadku równa się dwa, ma następującą postać:

Algorytm Schrage

Krok 1. Połóż $t:=\min_{1\leq j\leq n} r_j$; $U:=\phi$; $U':=(1,\ldots,n)$.

Krok 2. Wyznacz zadanie $j'\in U'$ takie, że $q_{j'}=\max(q_j: r_j\leq t, j\in U')$.

Krok 3. Połóż $U:=U\cup(j')$; $U':=U'-(j')$; $S_{j'}:=t$; $t:=\max(t+p_{j'}, \min_{j\in U'} r_j)$. Jeżeli $U'=\phi$, to stop; w przeciwnym wypadku idź do Kroku 2.

W wyniku działania algorytmu otrzymujemy momenty rozpoczęcia wykonywania zadań S_j, $j=1,\ldots,n$ określające kolejność wykonywania zadań, którą będziemy dalej oznaczać przez π^S. Algorytm ma złożoność obliczeniową $O(n\log n)$. Zbiory U, U' w kroku 2 określają odpowiednio zadania uszeregowane i wszystkie pozostałe w bieżącej chwili t. Warto tutaj zauważyć, że dodatkowe zastosowanie algorytmu Schrage dla problemu inwersyjnego do $1|r_i,q_i|C_{max}$ (jest to problem, w którym wielkości r_i i q_i zostały zamienione miejscami) nie zmniejsza współczynnika najgorszego przypadku. Rzeczywiście, wystarczy rozważyć następujące zagadnienie konkretne: $n=3$, $r_1=q_1=0$, $p_1=P$; $r_2=p_2=1$, $q_2=P$; $r_3=P$, $p_3=q_3=1$, gdzie $P>>1$. Algorytm Schrage daje $\pi^S=(1,2,3)$ oraz $C_{max}(\pi^S)=2P+1$. Z kolei zastosowanie tego algorytmu do problemu inwersyjnego daje $\pi'=(2,3,1)$ oraz $C_{max}(\pi')=2P+1$. Permutacja optymalna $\pi^*=(2,1,3)$ i $C_{max}(\pi^*)=P+4$. Stąd $\min(C_{max}(\pi^S), C_{max}(\pi'))/C_{max}(\pi^*) = (2P+1)/(P+4)$ i dąży do dwa przy P dążącym do nieskończoności.

Kolejny algorytm, zaproponowany przez Pottsa [18], ma współczynnik najgorszego przypadku równy 1.5 i złożoność obliczeniową $O(n^2\log n)$. W celu jego zdefiniowania wprowadzimy pojęcie tzw. zadania "konfliktowego". Niech

para (u_1,u_2) dla permutacji π spełnia $1\le u_1\le u_2\le n$ oraz $C_{max}(\pi)=r_{\pi(u_1)}+\sum_{j=u_1}^{u_2}p_{\pi(j)}+q_{\pi(u_2)}$; w przypadku gdy jest więcej par spełniających powyższe warunki, to przez parę (u_1,u_2) oznaczamy tę, dla której obydwa elementy są najmniejsze. Zadanie $\pi(k)$ będziemy nazywać zadaniem konfliktowym dla permutacji π, jeżeli $u_1\le k<u_2$, $q_{\pi(k)}<q_{\pi(u_2)}$ oraz $q_{\pi(h)}\ge q_{\pi(u_2)}$, $h=k+1,\ldots,u_2$.

Można pokazać, że jeżeli π jest otrzymane za pomocą algorytmu Schrage i nie istnieje dla niej zadanie konfliktowe, to π jest permutacją optymalną.

Algorytm Pottsa

Krok 1. Połóż $h:=1$; $C_{max}:=\infty$.

Krok 2. Wylicz π wg algorytmu Schrage, $C_{max}^{(h)}(\pi)$, (u_1,u_2) oraz zadanie konfliktowe $\pi(k)$. Jeżeli $C_{max}^{(h)}(\pi)<C_{max}$, to połóż $C_{max}:=C_{max}^{(h)}(\pi)$.

Krok 3. Jeżeli $h=n$ lub nie istnieje zadanie konfliktowe, to stop; permutację związaną z aktualną wartością C_{max} traktujemy jako permutację π^P produkowaną przez algorytm. W przeciwnym wypadku połóż $r_{\pi(k)}:=r_{\pi(u_2)}$ i idź do kroku 2.

Przyjęcie w kroku 3 algorytmu nowej wartości momentu gotowości dla zadania $\pi(k)$ powoduje, że we wszystkich następnych permutacjach generowanych w kroku 2 zadanie to występuje za zadaniami $\pi(k+1),\ldots,\pi(u_2)$.

W pracy [7] zaproponowano pewną modyfikację algorytmu Pottsa. Modyfikacja polega na tym, że w przypadku gdy $\max_{1\le i\le n} r_i-\min_{1\le i\le n} r_i<\max_{1\le i\le n} q_i-\min_{1\le i\le n} q_i$, to stosujemy algorytm Pottsa nie do zagadnienia podstawowego, ale do inwersyjnego. Odpowiednie badania eksperymentalne potwierdzały celowość takiego postępowania. Kontynuując tę drogę rozumowania, Halls i Shmoys [8] zaproponowali algorytm, którego główna idea polega na tym, że wyznaczają permutacje π' wg al. Pottsa dla zagadnienia podstawowego, a następnie permutację π'', też wg al. Pottsa, ale dla zagadnienia inwersyjnego. Jako permutację heurezy przyjmują $\pi^{HS}\in(\pi',\pi'')$ taką, że $C_{max}(\pi^{HS})=\min(C_{max}(\pi'),C_{max}(\pi''))$. Współczynnik najgorszego przypadku dla tego algorytmu jest równy 4/3.

Przystąpimy teraz do sformułowania, wspomnianej już wcześniej, całej klasy algorytmów aproksymacyjnych rozwiązujących problem (3). Poszczególne elementy tej klasy różnią się między sobą zastosowaniem różnych algorytmów aproksymacyjnych dla zagadnienia $1|r_i,q_i|C_{max}$. Elementem wspólnym jest konstrukcja permutacji w oparciu o dolne ograniczenie postaci (7).

Algorytm H(A)

Krok 1. Wyznacz $\pi^{H(A)}\in\Pi$ minimalizujące $C_{max}(\pi,x')$, stosując pewien algorytm aproksymacyjny A rozwiązujący zagadnienie $1|r_i,q_i|C_{max}$ dla czasów wykonywania zadań $p_j=a_j-(1-c_j)u_j$, $j=1,\ldots,n$.

Krok 2. Wyznacz $x^{H(A)}\in X$ minimalizujące $K(\pi^{H(A)},x)=C_{max}(\pi^{H(A)},x)+\sum_{j=1}^{n}c_jx_j$ przy ograniczeniu $x\in X$.

Złożoność obliczeniowa kroku 1 jest równa złożoności obliczeniowej zastoso-

wanego algorytmu A. Oczywiście, możliwe jest także optymalne rozwiązanie zadania występującego w tym kroku. Trzeba wtedy zastosować jeden z algorytmów (niewielomianowych) opartych na metodzie podziału i ograniczeń, [2],[7]. Zadanie z kroku 2 jest zadaniem programowania liniowego i może być rozwiązane wykorzystując bardzo skuteczne algorytmy wyznaczające tzw. krzywą kosztu projektu w zagadnieniach PERT/koszt, po ich drobnej modyfikacji.

4. Oszacowanie współczynnika najgorszego przypadku

Przedstawimy teraz oszacowanie współczynnika najgorszego przypadku dla całej klasy algorytmów H(A). W oszacowaniu tym jako parametr wystąpi współczynnik ρ^A najgorszego przypadku zastosowanego algorytmu A. Przedtem jednak przedstawimy pewne górne oszacowania wartości $K^{H(A)}=K(\pi^{H(A)},x^{H(A)})$.

Zachodzi

$$C_{max}(\pi,x) \le R + Q + \Sigma_{j=1}^{n} a_j - x_j, \quad \pi\in\Pi, \ x\in X.$$

Stąd

$$K^{H(A)} = \min_{x\in X}\,[C_{max}(\pi^{H(A)},x)+\Sigma_{j=1}^{n} c_j x_j] \le R + Q + \Sigma_{j=1}^{n} a_j - (1-c_j)u_j \ . \quad (12)$$

Z kolei

$$K^{H(A)} \le C_{max}(\pi^{H(A)},x') + \Sigma_{j=1}^{n} c_j(1-c_j)u_j \ . \quad (13)$$

Twierdzenie

$$K^{H(A)}/K^{*} \le 0.25(1+\sqrt{3})\rho^A + 1, \quad (14)$$

gdzie ρ^A - współczynnik najgorszego przypadku algorytmu A.

Dowód:

Przed dowodem własności (14) pokażemy kilka własności pomocniczych. Niech $\alpha=0.25(1+\sqrt{3})$ i $\gamma=0.5(3-\sqrt{3})$. Zachodzi $0\le\gamma\le\alpha\le 1$, $1/(4\alpha)=1-\gamma$ oraz

$$\gamma = 2(1-\alpha). \quad (15)$$

Stąd i uwzględniając, że $0\le c_j\le 1$, otrzymamy

$$c_j(1-\alpha c_j) \le 1-\gamma \ , \ j=1,\ldots,n. \quad (16)$$

Korzystając z (15), dostaniemy

$$(1-\alpha)(R+Q) \le 2(1-\alpha)\max(R,Q) = \gamma\max(R,Q), \quad (17)$$

gdzie R, Q zdefiniowano w (5). Z definicji ρ^A oraz z (7) wynika, że

$$C_{max}(\pi^{H(A)},x') \le \rho^A \min_{\pi\in\Pi} C_{max}(\pi,x') \le \rho^A K^{*}. \quad (18)$$

Ostatnia własność pomocnicza ma postać

$$(1-\alpha)(a_j-u_j) + c_j u_j(1-\alpha c_j) \le \min((1-\gamma)a_j,\ (1-\gamma)(a_j-u_j)+c_j u_j). \qquad (19)$$

Z (18) i z faktu, że $\alpha \ge \gamma$ wynika $(1-\alpha)(a_j-u_j)+c_j u_j(1-\alpha c_j) \le (1-\alpha)(a_j-u_j) + (1-\gamma)u_j \le (1-\gamma)(a_j-u_j) + (1-\gamma)u_j = (1-\gamma)a_j$. Podobnie, z nierówności $1-\alpha c_j \le 1$ otrzymamy $(1-\alpha)(a_j-u_j)+c_j u_j(1-\alpha c_j) \le (1-\gamma)(a_j-u_j)+c_j u_j(1-\alpha c_j) \le (1-\gamma)(a_j-u_j)+c_j u_j$, co kończy dowód nierówności (19).

Przejdziemy teraz do zasadniczej części dowodu. Z (12) i (13) dostaniemy

$$K^{H(A)} \le \alpha C_{max}(\pi^{H(A)},x') + \alpha\Sigma_{j=1}^{n} c_j(1-c_j)u_j + (1-\alpha)(R+Q)+(1-\alpha)\Sigma_{j=1}^{n} a_j-(1-c_j)u_j$$

$$= \alpha C_{max}(\pi^{H(A)},x') + (1-\alpha)(R+Q) + \Sigma_{j=1}^{n}[(1-\alpha)(a_j-u_j)+c_j u_j(1-\alpha c_j)].$$

Stąd i z (17), (19) otrzymamy

$$K^{H(A)} \le \alpha C_{max}(\pi^{H(A)},x') + \gamma\max(R,Q) + \Sigma_{j=1}^{n}\min((1-\gamma)a_j,\ (1-\gamma)(a_j-u_j)+c_j u_j)$$

$$= \alpha C_{max}(\pi^{H(A)},x') + \gamma\max(R,Q) + \Sigma_{j=1}^{n} \min_{0\le x_j\le u_j}[(1-\gamma)(a_j-x_j)+c_j x_j]$$

$$= \alpha C_{max}(\pi^{H(A)},x') + \min_{x\in X}[\gamma\max(R,Q) + (1-\gamma)\Sigma_{j=1}^{n}(a_j-x_j) + \Sigma_{j=1}^{n} c_j x_j]$$

$$\le \alpha C_{max}(\pi^{H(A)},x') + \min_{x\in X}[\max(R,Q,\Sigma_{j=1}^{n}(a_j-x_j)) + \Sigma_{j=1}^{n} c_j x_j].$$

Z powyższego i z (18), (6) ostatecznie dostaniemy

$$K^{H(A)} \le \alpha\rho^{A}K^{*} + K^{*} = (\alpha\rho^{A} + 1)K^{*},$$

co kończy dowód twierdzenia. ■

W Tabeli 1 przedstawiono, wynikające z udowodnionego twierdzenia, górne oszacowanie UB(A) współczynnika najgorszego przypadku dla algorytmu H(A).

Algorytm A	optymalny	Hall i Shmoys	Potts	Schrage
ρ^{A}	1.00	1.33	1.50	2.00
UB(A)	1.68	1.91	2.02	2.37

Tab 1. Górne oszacowanie UB(A) współczynnika najgorszego przypadku dla algorytmu H(A); odpowiednie wielkości podano w przybliżeniu.

Analizując te oszacowania, warto zauważyć, że algorytm (H), który w kroku 1 wybiera permutacje arbitralnie, a następnie wykonuje krok 2 bez zmian, ma współczynnik najgorszego przypadku równy trzy. Przykład, dla którego $K^H/K^*=3$, ma postać: n=3, $r_1=1$, $a_1=u_1=q_1=0$; $r_2=0$, $a_2=1$, $u_2=0$, $q_2=0$; $r_3=a_3=u_3=0$, $q_3=1$ (ponieważ $u_j=0$, to wielkości c_j są nieistotne). Rzeczywiście, jeżeli przyjmiemy $\pi^H=(1,2,3)$, to $K^H=3$, zaś $\pi^*=(3,2,1)$ i $K^*=1$. Z kolei algorytm, opisany w [15], który w kroku 1 wybiera permutacje wg niemalejących wartości r_j, ma współczynnik najgorszego przypadku równy dwa.

Aktualnie prowadzi się badania eksperymentalne zaproponowanych algorytmów, a w szczególności algorytmu H(HS) oraz algorytmu H(C), gdzie C jest algorytmem Cariera znajdującym permutację optymalną w kroku 1. W badaniach tych wykorzystuje się dolne ograniczenia K^* postaci (6), (9), (10), (11) oraz inne bazujące na relaksacji warunku o niepodzielności zadań. Wstępne wyniki badań są bardzo zachęcające. Przykładowo górne oszacowanie błędu względnego $((K^{H(HS)}-K^*)/K^*)*100\%$ jest nie większe niż 0.5%, co w praktycznych sytuacjach, przy niedokładnych danych wejściowych, jest całkowicie do przyjęcia.

Wykorzystując idee algorytmu H(A), można też podać algorytm aproksymacyjny dla problemu P2. Konkretyzując ten problem będziemy zakładać, że minimalizujemy $C_{max}(\pi,x)$ przy ograniczeniach $\sum_{j=1}^{n} c_j x_j \le R$, gdzie R jest nieprzekraczalną wartością sumarycznego kosztu wykonywania zadań. Proponujemy do jego rozwiązania następujący algorytm G(A):

Algorytm G(A)

Krok 1. Dla k=1,2 wyznacz $\pi^{(k)} \in \Pi$ minimalizujące $C_{max}(\pi,x^{(k)})$, stosując pewien algorytm aproksymacyjny A rozwiązujący zagadnienie $1|r_i,q_i|C_{max}$ dla czasów wykonywania zadań $p_j=a_j-x_j^{(k)}$, gdzie $x_j^{(1)}=0$, $x_j^{(2)}=u_j$, $j=1,\ldots,n$.

Krok 2. Wyznacz $y^{(k)} \in X$ minimalizujące $C_{max}(\pi^{(k)},x)$ przy ograniczeniach $x \in X$, $\sum_{j=1}^{n} c_j x_j \le R$; k=1,2.

Krok 3. Wybierz parę $(\pi^{G(A)},x^{G(A)}) \in \{(\pi^{(k)},y^{(k)}): k=1,2\}$ spełniającą $K(\pi^{G(A)},x^{G(A)})=\min\{K(\pi^{(k)},y^{(k)}): k=1,2\}$.

Algorytm G(A) ma taką samą złożoność obliczeniową jak algorytm H(A). Wszelkie analizy dotyczące jakości produkowanego przez niego rozwiązania są aktualnie otwartym problemem.

LITERATURA

[1] Błażewicz J.: Złożoność obliczeniowa problemów kombinatorycznych. WNT, Warszawa 1988.

[2] Carlier J.: The one-machine sequencing problem. European J. Oper. Res., 1982, 42-47.

[3] Elmaghraby S.E.: Activity networks, J. Wiley and Sons, New York 1977.

[4] Graham R.L. et al.: Optimization and approximation in deterministic sequencing and scheduling: a survey, Ann. Discrete Math. 5, 1979, 287-326.

[5] Lawler E.L., Lenstra J.K., Rinnooy Kan A.H.G., Shmoys D.B.: Sequencing and scheduling: Algorithms and complexity, Department of Operation Research, Statistics, and System Theory, Report BS-R8909, 1989.

[6] Grabowski J., Janiak A.: Job-shop scheduling with resource-time models of operations, European J. Oper. Res. 28, 1986, 58-73.

[7] Grabowski J., Nowicki E., Zdrzałka S.: A block approach for single machine scheduling with release dates and due dates, European J. Oper. Res. 26, 1986, 278-285.

[8] Hall L.A., Shmoys D.B.: Jackson's rule for single-machine scheduling: making a good heuristic better, Department of Mathematics, Massachusetts Institute of Technology, Cambridge, 1986, praca niepublikowana.

[9] Nowicki E.: Minimalizacja kosztu w jednomaszynowym problemie szeregowania ze zmiennymi czasami gotowości zadań, Materiały konferencji "Problematyka budowy i eksploatacji maszyn i urządzeń w ujęciu systemowym", NOT, Kraków 1986, 76-82.

[10] Nowicki E.: Algorytmy aproksymacyjne dla dwumaszynowego problemu przepływowego z wypukłą funkcją kosztu, Archiwum Automatyki i Telemechaniki, Vol.33, z.3, 1988.

[11] Nowicki E.: Minimalizacja kosztu w dwumaszynowym problemie przepływowym ze zmiennymi czasami wykonywania zadań, Zeszyty Naukowe Politechniki Śląskiej, Seria: Automatyka, Nr. 84, 1986, 153-162.

[12] Nowicki E.: Algorytmy aproksymacyjne dla problemu $1|r_i,q_i|C_{max}$ ze zmiennymi czasami wykonywania zadań, I Krajowa Konferencja "Badania operacyjne i systemowe", Książ 1988.

[13] Nowicki E., Zdrzałka S.: Problemy szeregowania ze zmiennymi czasami wykonywania zadań. Zeszyty Naukowe Politechniki Śląskiej Seria Automatyka , Nr 84, 1986, 163-176.

[14] Nowicki E., Zdrzałka S.: Two-machine flow shop scheduling problem with controllable job processing times, European J. Oper. Res. 34, 1988, 208-220.

[15] Nowicki E., Zdrzałka S.: Sequencing problems with controllable job processing times, Applied Discrete Mathematics, 1989, (w druku).

[16] Potts C.N.: Analysis of a heuristic for one machine with release dates and delivery times, Operations Research 28, 1980, 1436-1441.

[17] Schrage L.: Obtaining optimal solutions to resource constrained network scheduling problems, 1971, praca niepublikowana.

[18] Vickson R.G.: Choosing the job sequence and processing times to minimize total processing plus flow cost on a single machine, Oper. Res. 28, 1980, 1155-1167.

[19] Van Wassenhove L.N., Baker K.R.: A bicriterion approach to time/cost trade-offs in sequencing, European J. Oper. Res. 11, 1982, 48-54.

Recenzent Doc.dr h.inż.J.Kałuski

Wpłynęło do Redakcji do 1990-04-30.

APPROXIMATION ALGORITHMS FOR ONE-MACHINE SCHEDULING PROBLEMS WITH VARIABLE JOB PROCESSING TIME

Summary

The paper deals with the discrete production sequencing problem with single bottle-neck and variable job processing times. It is assumed that the cost of performing a job is a linear function of its processing time, and schedule cost to be minimized is the total processing cost plus maximum completion time cost. For the problem some approximation algorithms are formulated and the worst-case analysis is carried out for each of them.

АППРОКСИМАЦИОННЫЕ АЛГОРИТМЫ ДЛЯ СИСТЕМЫ ОБСЛУЖИВАНИЯ С ОДНИМ ПРИБОРОМ И С УЧЁТОМ ИЗМЕНЕНИЯ ДЛИТЕЛЬНОСТЕЙ ОБСЛУЖИВАНИЯ

Резюме

В роботе представлена проблема упорядочения задач в производственном процессе с одним критическим гнездом и с учётом изменения длительностей обслуживания. Принимая, что стоимость выполнения задачи является линейной функцией ее времени выполнения, ставится проблема минимизации: суммарной стоимости выполнения всех задач плюс стоимость, которая зависит от времени окончания этих задач. Для этой проблемы формулируются аппроксимационные алгоритмы и представляются оценки погрешности.