ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1986
Seria: AUTOMATYKA z. 84 Nr kol. 894

Eugeniusz Nowicki

Politechnika Wrocławska
Instytut Cybernetyki Technicznej

MINIMALIZACJA KOSZTU W DWUMASZYNOWYM PROBLEMIE PRZEPŁYWOWYM ZE ZMIENNYMI CZASAMI WYKONYWANIA ZADAŃ*

Streszczenie. W pracy rozważa się dwumaszynowy problem przepływowy /flow-shop/, w którym kolejność wykonywania zadań oraz ich czasy wykonywania są zmiennymi decyzyjnymi. Zakładając, że koszt wykonywania zadania jest funkcją liniową jego czasu wykonania stawia się problem minimalizacji sumarycznego kosztu wykonywania poszczególnych zadań przy ograniczonym czasie zakończenia wykonania wszystkich zadań. Pokazuje się, że decyzyjna wersja tego problemu jest NP-trudna i następnie przy przyjęciu różnych założeń upraszczających konstruuje się algorytmy o złożoności wielomianowej. Dla przypadku ogólnego podaje się algorytm heurystyczny.

1. Wstęp

W ostatnich latach obserwuje się koncentrację badań nad tzw. mieszanymi problemami rozdziału zasobów. W szczególności przyjmuje się, że do wykonania operacji potrzeba jest wiele rodzajów zasobów odnawialnych i nieodnawialnych podzielnych w sposób ciągły lub dyskretny, co w znaczny sposób przybliża badany model do rzeczywistości.

W pracy rozważa się dwumaszynowy problem przepływowy zakładając model operacji taki, jak w problemach PERT/koszt. Przyjmuje się, że czas wykonania operacji na pierwszej maszynie można zmieniać w sposób ciągły w pewnym przedziale, co powoduje różne zużycie zasobu nieodnawialnego typu nakłady finansowe, czyli różny jest koszt wykonania operacji. Ponadto zakłada się, że koszt ten jest liniową funkcją czasu wykonania operacji. Przyjęcie tych założeń oznacza, że do wykonania operacji potrzebny jest jeden zasób odnawialny podzielny w sposób dyskretny /odpowiednia maszyna o pojemności jeden/ oraz jeden zasób nieodnawialny podzielny w sposób ciągły /nakłady finansowe/. Ponieważ dokładny przegląd dotychczasowych rezultatów dotyczących jednoczesnego uwzględnienia zasobów nieodnawialnych typu maszyn o pojemności 1 oraz modelu operacji typu PERT/koszt znajduje się w pracy [3] umieszczonej w tym samym zeszycie, ograniczymy się tutaj tylko do stwierdzenia, że najbardziej ogólny model mieszanego problemu rozdziału zasobów został podany przez Węglarza w [4].

Zagadnienie badane w pracy polega na znalezieniu takiej kolejności wykonywania zadań na maszynach oraz czasów wykonywania poszczególnych operacji wykonywanych na maszynie pierwszej i wchodzących w skład zadań,

* praca była częściowo finansowana przez RP.I.02 "Teoria sterowania i optymalizacja ciągłych układów dynamicznych i procesów dyskretnych"

żeby przy ograniczonym czasie zakończenia wykonania wszystkich zadań i spełnieniu ograniczeń charakteryzujących dwumaszynowy problem przepływowy minimalizować sumaryczny koszt wykonywania wszystkich operacji. Warto tutaj zauważyć, że przy odrzuceniu zasobu nieodnawialnego, tzn. maszyn, problem sformułowany powyżej sprowadza się do znanego z literatury problemu wyznaczania krzywej kosztu projektu przy specyficznych ograniczeniach kolejnościowych.

W pracy pokazano, że decyzyjna wersja badanego zagadnienia jest NP-zupełna i następnie przy przyjęciu różnych założeń upraszczających skonstruowano szereg algorytmów o złożoności wielomianowej wyznaczających rozwiązanie optymalne. Dla przypadku ogólnego podano algorytm heurystyczny.

2. Sformułowanie problemu

Dany jest zbiór n niezależnych zadań ponumerowanych 1,2,...,n, które należy wykonać na dwóch maszynach A, B. Każde zadanie j jest najpierw wykonywane na maszynie A w czasie $p_{1j} = a_j - x_j$, $0 \leq x_j \leq u_j$, a potem na maszynie B w czasie $p_{2j} = b_j$, gdzie x_j określa skrócenie czasu p_{1j}. Wielkość u_j /$a_j \geq u_j$/ oznacza możliwe maksymalne skrócenie czasu wykonywania zadania j na maszynie A. Zakładamy, że: /i/ każda maszyna wykonuje zadania bez przerwań, /ii/ wszystkie zadania są gotowe do wykonywania w chwili zerowej, /iii/ każda maszyna w dowolnej chwili czasowej może wykonywać nie więcej niż jedno zadanie, /iv/ kolejność wykonywania zadań na obu maszynach jest identyczna. Niech

$$X = \{x \in R^n : 0 \leq x_j \leq u_j,\ 1 \leq j \leq n\}$$

a π oznacza permutację elementów zbioru $\{1,2,\ldots,n\}$, zaś Π zbiór wszystkich takich permutacji. Oznaczmy dalej przez $C_{max}(\pi, x)$ czas zakończenia wykonywania wszystkich zadań przy spełnieniu /i/ - /iv/ dla permutacji $\pi \in \Pi$ i czasów wykonywania $p_{1j} = a_j - x_j$, $p_{2j} = b_j$, $j = 1,\ldots,n$, $x \in X$. Zachodzi $C_{max}(\pi, x) = \max_{1 \leq i \leq n} \left[\sum_{j=1}^{i} (a_{\pi(j)} - x_{\pi(j)}) + \sum_{j=i}^{n} b_{\pi(j)}\right]$. Przyjmujemy, że koszt wykonywania zadania j jest równy $c_j x_j$. Wielkość $c_j > 0$ określa jednostkowy koszt skrócenia czasu wykonywania zadania j na maszynie A.

Problem badany w tej pracy można sformułować następująco: Dla zadanego $T > 0$ znaleźć parę (π^o, x^o) będącą rozwiązaniem następującego zadania

$$\min_{x,\pi} \sum_{j=1}^{n} c_j x_j \qquad /1/$$

przy ograniczeniach:

$$C_{max}(\pi, x) \leqslant T, \quad \pi \in \Pi, \quad x \in X \qquad /2/$$

Ogólnie rzecz biorąc koszt wykonywania zadania przy $x = 0$ nie jest równy zero. Jednakże koszt ten nie zależy od permutacji π i dlatego nie zmniejszając ogólności rozważań mogliśmy go pominąć.

Przejdziemy teraz do pokazania pewnych własności rozwiązania optymalnego zadania /1/, /2/. Niech $K^o = \sum_{j=1}^{n} c_j x_j^o$, $T_1 = \min_{\pi \in \Pi} C_{max}(\pi, u)$, $T_2 = \min_{\pi \in \Pi} C_{max}(\pi, 0)$, gdzie $u = (u_1, u_2, \ldots, u_n)$. Zauważmy, że wielkości T_1, T_2 można wyznaczyć algorytmem Johnsona w czasie $O(n \log n)$. Zachodzą następujące własności:

Własność 1

Rozwiązanie optymalne zadania /1/, /2/ istnieje wtedy i tylko wtedy, gdy $T \geqslant T_1$.

Własność 2

Niech $T \geqslant T_1$. Jeżeli $K^o > 0$ to $T < T_2$. Jeżeli $T \leqslant T_2$ to $C_{max}(\pi^o, x^o) = T$.

Prawdziwość powyższych własności wynika z faktu, że dla każdej permutacji $\pi \in \Pi$, $C_{max}(\pi, x)$ jest funkcją nierosnącą każdej składowej x_j wektora $x \in X$. Własność 1 rozstrzyga problem istnienia rozwiązania zadania /1/, /2/, a własność 2 mówi, między innymi, że dla $T \geqslant T_2$ minimalny koszt wykonywania zadań jest równy zero. Dlatego też dalsze rozważania wystarczy ograniczyć do przypadku gdy $T_1 \leqslant T \leqslant T_2$. Kontynuując dalej analizę zadania /1/,/2/ zauważmy, że dla ustalonej permutacji π zadanie to sprowadza się do zadania programowania liniowego, dla którego istnieje algorytm o złożoności wielomianowej. Ponadto dla ustalonego $x \in X$ potrafimy w sposób efektywny sprawdzić czy istnieje permutacja π spełniająca $C_{max}(\pi, x) \leqslant T$. Tym niemniej złożoność problemu /1/,/2/ jest sprawą otwartą. Dlatego też w następnym punkcie przeprowadzimy analizę złożoności obliczeniowej tego problemu.

3. Złożoność obliczeniowa

W tym punkcie pokażemy, że decyzyjna wersja problemu /1/,/2/ jest NP-zupełna. Można ją sformułować następująco:

DWP: Dany jest 2-maszynowy przepływowy permutacyjny system wykonywania zadań /permutation flow-shop/, ze zmiennymi czasami wykonywania zadań $a_j - x_j$, b_j, $0 \leqslant x_j \leqslant u_j$, $j = 1, \ldots, n$, jednostkowe koszty skrócenia czasów wykonywania $c_j > 0$, $j = 1, \ldots, n$ i liczby K, T. Przyjmujemy, że n, a_j, b_j, u_j, $j = 1, \ldots, n$ są liczbami naturalnymi a K, T, c_j, x_j, $j = 1, \ldots, n$ nieujemnymi liczbami wymiernymi. Czy istnieje para (π, x) taka, że

$$\sum_{j=1}^{n} c_j x_j \leqslant K, \quad C_{max}(\pi, x) \leqslant T, \quad x \in X, \quad \pi \in \Pi \ ?$$

Twierdzenie 1

Decyzyjna wersja DWP problemu /1/,/2/ jest NP-zupełna.

Szkic dowodu: W dowodzie ograniczymy się do pokazania, że problem PODZIAŁ ZBIORU, o którym wiadomo, że jest NP-zupełny, jest wielomianowo transformowalny do DWP. Problem PODZIAŁ ZBIORU ma postać:

PODZIAŁ ZBIORU: Dane są liczby naturalne $t, \alpha_1, \ldots, \alpha_t$, B takie, że $\sum_{i=1}^{t} \alpha_i = 2B$. Czy istnieje $S \subset T = \{1,2,\ldots,t\}$ taki, że $\sum_{i \in S} \alpha_i = B$.

Będziemy rozważać następujący szczególny przypadek problemu DWP: $n=t+1$, $a_j = 2\alpha_j$, $u_j = 2\alpha_j$, $b_j = \alpha_j$, $j = 1,\ldots,t$; $a_n = B$, $u_n = 0$, $b_n = 2B$, $T = 4 \cdot B$, $K = 2 \cdot Bc$, $c_j = c$, $j = 1,\ldots,n$.

Załóżmy, że PODZIAŁ ZBIORU ma rozwiązanie, tzn. istnieje $S \subset T$ takie, że $\sum_{j \in S} \alpha_j = B$. Niech π będzie permutacją, których $t - |S|$ elementów należy do zbioru $T \setminus S$, następnym elementem jest n i dalej występują wszystkie elementy ze zbioru S. Połóżmy $x_j = 2\alpha_j$, $j \in T \setminus S$; $x_j = 0$, $j \in S$; $x_n = 0$. Zachodzi $C_{max}(\pi, x) = 4B = T$, $c \cdot \sum_{j=1}^{n} x_j = c \sum_{j \in T \setminus S} 2\alpha_j = 2Bc = K$.

Pokazaliśmy więc, że jeżeli PODZIAŁ ZBIORU ma rozwiązanie to szczególny przypadek problemu DWP określony powyżej ma też rozwiązanie.

W podobny sposób, choć wymagający dłuższego rozumowania, można pokazać, że jeżeli PODZIAŁ ZBIORU nie ma rozwiązania, to szczególny przypadek problemu DWP też nie ma rozwiązania. Ponieważ przynależność problemu DWP do klasy zadań NP jest oczywista, to tym samym dowód prawdziwości twierdzenia 1 został zakończony.

Z przedstawionego szkicu dowodu wynika, że DWP jest NP-zupełny nawet wtedy, gdy jednostkowe koszty skracania czasów wykonywania zadań na maszynie A są identyczne, tzn. $c_j = c$.

4. Algorytmy wielomianowe

W poprzednim punkcie pokazaliśmy, że problem DWP jest NP-zupełny. Wynika stąd, że problem /1/,/2/ jest NP-trudny, co powoduje, że nie istnieje algorytm o złożoności wielomianowej /jeżeli $P \neq NP$/ rozwiązujący ten problem. Nie znaczy to jednak, że przy przyjęciu dodatkowych ograniczeń nie można podać algorytmów efektywnych. Przeprowadzimy teraz analizę tych przypadków. W tym celu pokażemy kilka własności rozwiązania optymalnego zadania /1/,/2/ przy dodatkowym założeniu postaci $c_j = c$, $j = 1,\ldots,n$.

Prawdziwy jest następujący lemat:

Lemat 1

Niech $c_j = c$, $j = 1,\ldots,n$ oraz $A \triangleq \min \{c\,(C_{max}(\pi, 0) - T) : \bigvee_{\pi \in \Pi} C_{max}(\pi, u) \leq T\}$.

Zachodzi $K^o = c \sum_{j=1}^{n} x_j^o = A$ oraz każda permutacja $\pi \in \Pi$ taka, że

$C_{max}(\pi, u) \leq T$ oraz $c\left(C_{max}(\pi, 0) - T\right) = A$ jest permutacją optymalną dla zadania /1/, /2/.

Dowód lematu wynika bezpośrednio z tego, że dla każdej pary (π^{o}, x^{o}) będącej rozwiązaniem optymalnym zadania /1/,/2/ zachodzi $\sum_{j=1}^{n} x_j^{o} = C_{max}(\pi^{o}, 0) - T$.

Korzystając z powyższego lematu można łatwo wykazać prawdziwość poniższego twierdzenia, które będzie podstawą konstrukcji algorytmów, o złożoności wielomianowej, wyznaczających rozwiązanie optymalne zadania /1/,/2/.

Twierdzenie 2

Niech $c_j = c$, $j = 1,\ldots,n$ oraz przez $\Pi^{o} \subset \Pi$ oznaczmy zbiór wszystkich permutacji π takich, że $C_{max}(\pi, 0) = \min_{\pi' \in \Pi} C_{max}(\pi', 0)$. Podobnie niech $\Pi^{u} \subset \Pi$ będzie zbiorem wszystkich permutacji π takich, że $C_{max}(\pi, u) = \min_{\pi' \in \Pi} C_{max}(\pi', u)$. Jeżeli $\Pi^{o} \cap \Pi^{u} \neq \emptyset$, to każda permutacja $\pi \in \Pi^{o} \cap \Pi^{u}$ jest permutacją optymalną zadania /1/,/2/.

Jak już wspomniano dla ustalonej permutacji π zadanie /1/,/2/ redukuje się do zadania wyznaczenia jednego punktu na tzw. krzywej kosztu projektu. Do rozwiązania tego zadania można zastosować znany z literatury algorytm Kelley'a [2]. Jednakże ze względu na specyficzną postać ograniczeń kolejnościowych algorytm ten można odpowiednio zmodyfikować i właśnie taka zmodyfikowana wersja tego algorytmu o złożoności $O(n^2)$ jest przedstawiona poniżej. Dla uproszczenia notacji przyjęto, że $\pi = \langle 1,2,\ldots,n\rangle$.

Algorytm 1

1. Podstaw $x_j = 0$, $\Delta_j = \sum_{i=1}^{j} a_i + \sum_{i=j}^{n} b_i$, $1 \leq j \leq n$; $\bar{T} = \max_{1 \leq j \leq n} \Delta_j$; $\bar{K} = 0$; $k = \min\{j : \Delta_j = \bar{T}\}$.
2. Znajdź najmniejsze $1 \leq l \leq k$ także, że $c_l = \min\{c_j : 1 \leq j \leq k,\ x_j \neq u_j\}$. Jeżeli $l \neq 1$ znajdź $\Delta = \max_{1 \leq j < l} \Delta_j$ oraz $k' = \min\{1 \leq j < l : \Delta_j = \Delta\}$. Jeżeli $l = 1$ to połóż $\Delta = 0$.
3. Jeżeli $\min\{\bar{T} - \Delta, u_l\} \geq \bar{T} - T$ to $x_l = \bar{T} - T$; $\bar{T} := \bar{T} - x_l$; $\bar{K} := \bar{K} + c_l x_l$ i stop.
4. Jeżeli $u_l < \bar{T} - \Delta$ to $x_l = u_l$; $\bar{T} := \bar{T} - x_l$; $\Delta_j := \Delta_j - x_l$, $l \leq j \leq k$; $\bar{K} := \bar{K} + c_l x_l$ i idź do 2.
   Jeżeli $u_l \geq \bar{T} - \Delta$ to $x_l = \bar{T} - \Delta$, $\bar{T} : \bar{T} - x_l$; $\bar{K} := \bar{K} + c_l x_l$; $k = k'$ i idź do 2.

Zauważmy, że w trakcie działania algorytmu składowa x_j jest zmieniana, co najwyżej raz, $j = 1,\ldots,n$, co przy dowolnych ograniczeniach kolejnościowych w algorytmie Kelley'a nie jest prawdą.

Wspomniana wyżej złożoność obliczeniowa algorytmu 1 wynika z faktu, że krok 2 jest wykonywany co najwyżej n razy. Algorytm ten wyznacza rozwiązanie optymalne tylko wtedy, gdy $C_{max}(\pi, u) \leq T \leq T_2$. Jednakże gdy

$T < C_{max}(\pi, u)$, to nie istnieje rozwiązanie, a w przypadku gdy $T > T_2$ minimalna wartość kosztu jest równa 0. Po zakończeniu pracy algorytmu wektor x zawiera optymalne rozwiązanie zadania /1/,/2/ dla ustalonej permutacji π, $\bar{T} = T$ a $\bar{K}$ jest minimalną wartością kosztu.

Przejdziemy teraz do analizy trzech różnych przypadków, dla których podamy algorytmy o złożoności wielomianowej wyznaczające rozwiązanie optymalne zadania /1/,/2/. We wszystkich tych przypadkach będziemy zakładać, że $c_j = c$, $j = 1,\dots,n$. Przypomnijmy jeszcze, że rozważamy tylko sytuację, kiedy $T_1 \le T \le T_2$.

$a_j = a$, $b_j = b$, $j = 1,\dots,n$. Zauważmy, że w tym przypadku $\Pi^o = \Pi$. Stąd $\Pi^o \cap \Pi^u = \Pi^u$ i z twierdzenia 2 wynika, że każda para (π, x) taka, że $\pi \in \Pi^u$ a x jest wyznaczone wg algorytmu 1 dla permutacji π jest rozwiązaniem optymalnym /1/,/2/. Złożoność obliczeniowa wyznaczenia każdej takiej pary jest równa $O(n^2)$.

$b_j = b$, $u_j = u$, $j = 1,\dots,n$. Niech π będzie permutacją optymalną dla klasycznego zagadnienia $F_2 \| C_{max}$ przy $p_{1j} = a_j - u$, $p_{2j} = b$, wyznaczoną przez algorytm Johnsona. Wtedy $\pi \in \Pi^u$ oraz

$$\min\{a_{\pi(i)} - u, b\} \le \min\{a_{\pi(i+1)} - u, b\}, \quad i = 1,\dots,n-1. \qquad /3/$$

Pokażemy, że z powyższych nierówności wynika, że

$$\min\{a_{\pi(i)}, b\} \le \min\{a_{\pi(i+1)}, b\}, \quad i = 1,\dots,n-1. \qquad /4/$$

W dowodzie wystarczy się ograniczyć do przypadku, gdy $a_{\pi(i)} > a_{\pi(i+1)}$ dla pewnego $1 \le i \le n-1$. Wtedy jednak z /3/ dostaniemy: $\min\{a_{\pi(i)} - u, b\} = \min\{a_{\pi(i+1)} - u, b\}$. Stąd $b \le a_{\pi(i+1)} - u$, co kończy dowód. Kontynuując dalej rozważania zauważmy, że z nierówności /4/ wynika, że permutacja π jest rozwiązaniem klasycznego zagadnienia $F2 \| C_{max}$ przy $p_{1j} = a_j$, $p_{2j} = b$, czyli $\pi \in \Pi^o$. Tym samym pokazaliśmy, że $\Pi^o \cap \Pi^u \ne 0$ i na mocy twierdzenia 2 para (π, x), gdzie x jest wyznaczone wg algorytmu 1 dla permutacji π, jest rozwiązaniem optymalnym zadania /1/,/2/. Złożoność obliczeniowa wyznaczenia takiej pary jest równa $O(n^2)$.

$a_j = a$, $u_j = u$, $j = 1,\dots,n$. Niech π będzie permutacją optymalną dla klasycznego zagadnienia $F2 \| C_{max}$ przy $p_{1j} = a$, $p_{2j} = b_j$, wyznaczoną przez algorytm Johnsona. Wtedy $\pi \in \Pi^o$ oraz

$$\min\{a, b_{\pi(i+1)}\} \le \min\{a, b_{\pi(i)}\}, \quad i = 1,\dots,n-1. \qquad /5/$$

Rozumując analogicznie jak w poprzednim przypadku można pokazać, że z nierówności /5/ wynikają następujące nierówności

$$\min\{a - u, b_{\pi(i+1)}\} \le \min\{a - u, b_{\pi(i)}\}, \quad i = 1,\dots,n-1, \qquad /6/$$

czyli permutacja π jest rozwiązaniem klasycznego zagadnienia $F2 \| C_{max}$ przy $p_{1j} = a - u$, $p_{2j} = b_j$, a więc $\pi \in \Pi^u$. Stąd $\Pi^o \cap \Pi^u \ne 0$ i na mocy twierdzenia 2 para (π, x), gdzie x jest wyznaczone wg algorytmu 1 dla permutacji π , jest rozwiązaniem optymalnym zadania /1/,/2/. Złożoność obliczeniowa wyznaczenia takiej pary jest równa tak jak poprzednio $O(n^2)$.

Zauważmy, że w każdym z powyżej rozważanych przypadków permutacja optymalna π^o nie zależała od wartości T, czyli dla każdego $T_1 \le T \le T_2$ jest ona taka sama. Stąd i z lematu 1 mamy, że minimalna wartość kosztu dla każdego $T_1 \le T \le T_2$ jest równa $c(C_{max}(\pi^o,0) - T) = c(T_2 - T)$ oraz zero dla $T \ge T_2$. Dalej przez x^α oznaczmy wektor x wyznaczony algorytmem 1 dla $T = \alpha$ oraz niech ciąg $\langle i_1, i_2, \dots, i_s\rangle$, $s \le n$ określa poszczególne wartości l otrzymywane w kroku 2 algorytmu 1 dla $T = T_1$. Pokażemy teraz prostą konstrukcję wektora x^α dla $T_1 \le \alpha \le T_2$ na bazie ciągu $\langle i_1, \dots, i_s\rangle$ oraz x^{T_1}. Niech $0 \le k \le s$ będzie największą liczbą całkowitą taką, że $\sum_{j=1}^{k} x_{i_j}^{T_1} \le T_2 - \alpha$; wtedy $x_{i_j}^{\alpha} = x_{i_j}^{T_1}$, $j = 1,\dots,k$, $x_{i_{k+1}}^{\alpha} = T_2 - \alpha - \sum_{j=1}^{k} x_{i_j}^{T_1}$ /jeżeli ta wielkość nie jest zerowa/, a pozostałe składowe wektora x^α są równe zero. Poprawność powyższej konstrukcji wynika natychmiast z uwagi, która została zamieszczona bezpośrednio po sformułowaniu algorytmu 1. Ostatecznie para (π^o, x^α) jest rozwiązaniem zadania /1/,/2/ dla każdego $T_1 \le \alpha \le T_2$. Widzimy więc, że we wszystkich trzech rozważanych przypadkach potrafimy rozwiązać całą rodzinę zadań /1/,/2/ /parametrem tej rodziny jest T/ wykorzystując tylko raz algorytm 1. Przy dowolnych ograniczeniach kolejnościowych w klasycznym problemie PERT/koszt potrafimy to też zrobić ale pod warunkiem, że w trakcie działania algorytmu będziemy zapamiętywać kolejne wartości wektora x. Na koniec zauważmy, że złożoność przedstawionej procedury rozwiązywania całej rodziny zadań /1/, /2/ jest równa $O(n^2)$ /wg definicji złożoności obliczeniowej algorytmu wyznaczającego zbiór nieskończony podanej w [3]/.

5. Algorytm heurystyczny

Jeżeli permutacja π jest ustalona to, jak już wspominaliśmy, rozwiązanie zadania /1/,/2/ sprowadza się do rozwiązania zadania programowania liniowego. Zadanie to jest rozwiązywane przez algorytm 1 w czasie $O(n^2)$. Ponadto dla ustalonego $x \in X$ potrafimy w czasie $O(n \log n)$ wyznaczyć permutację $\pi \in \Pi$ minimalizującą $C_{max}(\pi, x)$; wystarczy zastosować algorytm Johnsona dla $P2||C_{max}$ gdy $p_{1j} = a_{1j} - x_j$, $p_{2j} = b_j$. Powyższe własności sugerują następujący algorytm heurystyczny dla rozwiązywania zadania /1/, /2/:

Algorytm heurystyczny

1. Połóż $x^{(0)} = u$; $k = 1$, $K^H = \sum_{j=1}^{n} c_j u_j$
2. Jeżeli $k > k_{max}$ to stop.

3. Wyznacz $\pi^{(k)}$ taką, że $C_{max}(\pi^{(k)}, x^{(k-1)}) = \min\limits_{\pi \in \Pi} C_{max}(\pi, x^{(k-1)})$.

4. Wyznacz $x^{(k)} \in X$ takie, że $C_{max}(\pi^{(k)}, x^{(k)}) \leqslant T$ i $\sum_{j=1}^{n} c_j x_j^{(k)} =$
$= \min\{\sum_{j=1}^{n} c_j x_j : x \in X,\ C_{max}(\pi^{(k)}, x) \leqslant T\}$. Jeżeli
$\sum_{j=1}^{n} c_j x_j^{(k)} < K^H$ to połóż $K^H = \sum_{j=1}^{n} c_j x_j^{k}$; $k := k + 1$ i idź do 2.
W przeciwnym wypadku stop.

Proponujemy przyjąć $2 \leqslant k_{max} \leqslant n$. Niech $k' \leqslant k_{max}$ oznacza etap, w którym algorytm zakończył działanie, tzn. największe k dla którego algorytm wykonuje jeszcze krok 4. Parę $(\pi^{(k')}, x^{(k')})$ będziemy traktować jako przybliżone rozwiązanie zadania /1/,/2/, a $K^H = \sum_{j=1}^{n} c_j x_j^{(k')}$, określa wartość funkcji celu dla tego rozwiązania. Złożoność algorytmu w świetle wcześniejszych uwag jest równa $O(k_{max} n^2)$. Zauważmy, że gdy algorytm skończy swoje działanie w kroku 4, to otrzymana para $(\pi^{(k')}, x^{(k')})$ spełnia warunki konieczne optymalności dla zadania /1/,/2/, tzn. $x^{(k')}$ minimalizuje $\sum_{i=1}^{n} c_i x_i$ przy ograniczeniu $C_{max}(\pi^{(k')}, x) = T$ oraz $\pi^{(k')}$ minimalizuje $C_{max}(\pi, x^{(k')})$.

Przedstawiony powyżej algorytm heurystyczny został przetestowany na 100 losowo wybranych przykładach konkretnych zadań /1/,/2/ dla n = 100. Każdy przykład jest scharakteryzowany przez zestaw a_j, u_j, b_j, c_j, $j = 1,\ldots,n$, T. Wielkości a_j, u_j, b_j, c_j zostały określone przez generator o rozkładzie równomiernym odpowiednio ze zbiorów $\{1,\ldots,a_{max}\}$, $\{1,\ldots,u_{max}\}$, $\{1,\ldots,b_{max}\}$, $[0,c_{max}]$. Przyjęto $a_{max} = b_{max} = 50$, $u_{max} = 25$, $c_{max} = 10$. Dla każdego zestawu wyliczono T_1 i T_2 i położono $T = (T_1 + T_2)/2$. Następnie wyznaczono $K^H = \sum_{j=1}^{n} c_j x_j^H$, gdzie x^H rozwiązanie otrzymane przez zastosowanie algorytmu heurystycznego,
$LB = \min\{\sum_{j=1}^{n} c_j x_j : x \in X,\ \sum_{j=1}^{n} x_j = T_2 - T\}$ oraz
$\rho = [(K^H - LB)/LB] \cdot 100\%$. Średnia arytmetyczna stu takich wyliczonych wartości ρ była równa 9%, a największa z nich 23%. Obliczenia zostały przeprowadzone na minikomputerze ZX-Spectrum.

Łatwo pokazać, że wartość LB szacuje od dołu minimalny koszt, tzn. $LB \leqslant K^o$. Stąd możemy stwierdzić, że średni błąd względny między kosztem dla rozwiązania heurystycznego a kosztem dla rozwiązania optymalnego, w analizowanej serii 100 przykładów, był nie większy niż 9%, a maksymalny błąd względny nie większy niż 23%. Należy więc przypuszczać, że badany algorytm heurystyczny znajduje parę (π^H, x^H), dla której koszt wykonania zadań K^H niewiele różni się od kosztu minimalnego K^o, tym bardziej, że wartość LB jest dość słabym oszacowaniem minimalnego kosztu.

Interesujące byłoby przeprowadzenie analizy najgorszego przypadku dla zaproponowanego algorytmu heurystycznego. Jest to jednak aktualnie problem otwarty.

6. Uwagi końcowe

Przedstawione w pracy rezultaty obowiązują także, jeśli nie będziemy zakładać, że kolejność wykonywania zadań na obu maszynach jest identyczna, tzn. zamiast permutacyjnego problemu przepływowego /permutation flow-shop/ rozważymy problem przepływowy /flow-shop/. Używając podobnych argumentów jak w [1] można pokazać, że w zbiorze rozwiązań optymalnych dla 2-maszynowego problemu przepływowego ze zmiennymi czasami wykonywania zadań istnieje rozwiązanie, w którym kolejność wykonywania zadań na obu maszynach jest identyczna.

Przedstawiony w pracy algorytm heurystyczny może być także zastosowany do rozwiązywania zadania minimalizacji kosztu wykonywania zadań w m-maszynowym permutacyjnym problemie przepływowym przy ograniczonym czasie zakończenia wykonywania wszystkich zadań i założeniu, że czas wykonywania każdej operacji można skracać. W tym przypadku w kroku 3 należy rozwiązać klasyczny problem $P||C_{max}$. Problem ten jest silnie NP-zupełny. Dlatego też do jego rozwiązania należy zastosować znany z literatury wielomianowy algorytm heurystyczny lub wykorzystać algorytmy oparte na metodzie podziału i ograniczeń, charakteryzujące się stosunkowo krótkim czasem obliczeń.

LITERATURA

[1] Conway R.W., Maxwell W.L., Miller L.W.; Theory of Scheduling. Addison-Wesley, Reading, Mass /1967/.

[2] Kelley J.E.Jr.; Critical-path planning and scheduling. Mathematical Basis, Oper. Res. Vol. 9 /1961/, 296-320.

[3] Nowicki E., Zdrzałka S.; Problemy szeregowania ze zmiennymi czasami wykonywania zadań. Automatyka z.84, Gliwice 1986.

[4] Węglarz J.; New models and procedures for resource allocation problems, Proc. 6th INTERNET Congress, Vol. 2 /VDI Verlag, Duesseldorf/, /1979/, 521-530.

Recenzent: Doc.dr h.inż.Jerzy Klamka

Wpłynęło do Redakcji do 1986.04.30

МИНИМИЗАЦИЯ СТОИМОСТИ В СИСТЕМЕ ПОТОЧНОГО ТИПА С ПОСЛЕДОВАТЕЛЬНЫМИ ДВУМЯ МАШИНАМИ С УЧЁТОМ ИЗМЕНЕНИЯ ДЛИТЕЛЬНОСТИ ОБСЛУЖИВАНИЯ

Р е з ю м е

В статье представляется система поточного типа с последовательными двумя машинами, в которой очерёдность выполнения задач и их время выпол-

нения являются переменными подлежащие выбору. Принимая, что стоимость выполнения задачи является линейной функцией её времени выполнения, ставится проблема минимизации суммарной стоимости выполнения всех задач при ограниченном времени окончания всех задач. Показывается, что проблема является P - полной. Затем для некоторого класса этих задач предлагаются полиномиальные алгоритмы. В общем случае формулируется эвристический алгоритм и представляются результаты вычислительных экспериментов проведённых для случайно избранных примеров.

MINIMIZING THE COST IN A TWO-MACHINE FLOW-SHOP WITH VARYING JOB PROCESSING TIMES

S u m m a r y

In the paper we consider a two-machineflow-shop in which both the sequence of jobs and the job processing times are controllable. The job cost is a linear function of the processing time. The problem is to find a sequence of jobs and job processing times minimizing the total cost under the given maximum job completion time. It is shown that the decision version of this problem is NP-complete even for identical costs per unit processing time. A number of polynomial algorithms for certain subproblems is presented and for the general case, a heuristic algorithm is given. The results of numerical simulations for heuristic algorithm are also given.