ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1988
Seria:AUTOMATYKA z.94 Nr kol.970

Zbigniew Buchalski
Politechnika Wrocławska

ALGORYTM DLA PROBLEMU SZEREGOWANIA ZADAŃ NA MASZYNACH DLA PEWNYCH FUNKCJI CZASU WYKONYWANIA ZADAŃ

Streszczenie. W pracy rozważany jest problem czasowo-optymalnego przydziału zasobów nieodnawialnych podzielnych w sposób ciągły i n zadań do dwóch identycznych maszyn równoległych dla pewnych funkcji czasu wykonywania zadań. Wykazano, że problem jest NP-trudny. Sformułowano model matematyczny zagadnienia oraz przedstawiono dwa algorytmy heurystyczne. Podano także wyniki badań eksperymentalnych algorytmów oraz omówiono efektywność tych algorytmów.

1. Wstęp

W większości badanych zagadnień szeregowania zadań na maszynach zakłada się, że czasy wykonywania zadań są stałe i zależą w ogólnym przypadku tylko od maszyny, na której są wykonywane zadania. W prezentowanej pracy zakłada się, że czas wykonywania zadania na maszynie zależy od ilości zasobu nieodnawialnego podzielnego w sposób ciągły, przydzielonego tej maszynie.

Zagadnienia, w których czasy wykonywania zadań nie są stałe, były już badane w literaturze. Należy w tym miejscu przede wszystkim zwrócić uwagę na prace [7,9,10,11]. Przykładowo w dwóch ostatnich przyjmuje się, że czas wykonywania zadania może przyjmować wartości z określonego skończonego zbioru (tzw. sposoby wykonywania zadań).

W niniejszej pracy rozważany jest problem czasowo-optymalnego przydziału zasobów nieodnawialnych podzielnych w sposób ciągły i n zadań do dwóch identycznych maszyn równoległych. W praktyce zagadnienia kolejnościowe są dość skomplikowane i najczęściej są zagadnieniami NP-zupełnymi. Dla tych zagadnień chcąc uzyskać "dobre" rozwiązania należy stosować metody przeglądu (metoda podziału i ograniczeń, programowanie dynamiczne), które są czasochłonne, szczególnie w sytuacji, gdy liczba zadań do wykonania jest stosunkowo duża. Dlatego też w rozdziale trzecim tej pracy zaprezentowano dwa algorytmy heurystyczne dla rozwiązania postawionego zadania. Podane w rozdziale czwartym wyniki badań eksperymentalnych tych algorytmów potwierdzają rekomendację tych algorytmów do zastosowań praktycznych, gdy liczba zadań do wykonania jest stosunkowo duża.

2. Sformułowanie problemu. Podstawowe własności

W pracy rozpatrujemy dyskretny system produkcyjny, o którym zakładamy, że:

(i) posiada dwie identyczne maszyny M_1, M_2, na których należy wykonać n niezależnych zadań $J = \{1,2,\dots,n\}$,

(ii) zadanie może być wykonywane na dowolnej maszynie i w trakcie jego wykonywania nie może być przerywane,

(iii) posiada N jednakowych jednostek zasobu nieodnawialnego,

(iv) maszyna M_k w trakcie wykonywania zadań jej przydzielonych $I_k \subset J$ wykorzystuje część jednostek zasobu nieodnawialnego u_k i w każdej chwili może wykonywać tylko jedno zadanie; $k = 1,2$, $u_1 + u_2 \leq N$,

(v) czas wykonania zadania i-tego na maszynie M_k, jeżeli przydzielono jej u_k jednostek zasobu nieodnawialnego, określony jest funkcją

$$T_i(u_k) = a_i + \frac{b_i}{u_k}, \quad u_k \in [0,N], \quad k = 1,2, \quad i \in J, \tag{1}$$

gdzie $a_i > 0$, $b_i > 0$ - parametry określające zadanie i-te i maszynę M_k.

Rozważany przez nas problem polega na znalezieniu czasowo-optymalnego przydziału zadań I_k i zasobu nieodnawialnego u_k do maszyn M_k, $k = 1,2$, przy spełnieniu powyższych założeń.

Sformułowany powyżej problem można przedstawić jako następujące zadanie minimalizacji dyskretno-ciągłej:

$$\min_{\substack{I_1,I_2 \\ u_1,u_2}} \max\left\{\sum_{i \in I_1} T_i(u_1), \sum_{i \in I_2} T_i(u_2)\right\} \tag{2}$$

przy ograniczeniach

$$\text{(i) } I_1 \cup I_2 = J, \; I_1 \cap I_2 = \emptyset, \qquad \text{(ii) } u_1 + u_2 \leq N, \; u_1 \geq 0, \; u_2 \geq 0. \tag{3}$$

Uwzględniając fakt, że funkcje T_i są malejące, zadanie (2), (3) można zapisać w równoważnej postaci:

$$\min_{I \subset J} \left[\min_{u \in [0,N]} \max\left\{\sum_{i \in I} T_i(u), \sum_{i \in \bar{I}} T_i(N-u)\right\}\right], \tag{4}$$

gdzie $\bar{I} = J \setminus I$.

Z postaci powyższego zadania i wcześniejszych uwag wynika, że w rozwiązaniu optymalnym czas pracy obydwu maszyn jest identyczny. Stąd

$$\sum_{i \in I^*} T_i(u^*) = \sum_{i \in \bar{I}^*} T_i(N-u^*) \stackrel{\text{def}}{=} Q(I^*) \tag{5}$$

oraz $I^* \neq \emptyset$ i $I^* \neq J$,

gdzie para I^*, u^* jest rozwiązaniem zadania (4).

Z (5) po prostych przekształceniach otrzymujemy

$$Q(I^*) = \frac{1}{2}\left[\sum_{i\in J}(a_i+c_i) + \sqrt{\left[\sum_{i\in J}(a_i+c_i)\right]^2 - 4\Big(\sum_{i\in I^*} a_i \sum_{i\in \bar{I}^*} a_i + \sum_{i\in I^*} c_i \sum_{i\in \bar{I}^*} a_i + \sum_{i\in I^*} a_i \sum_{i\in \bar{I}^*} c_i\Big)}\right] \tag{6}$$

zaś $$u^* = \frac{N \cdot \sum_{i\in I^*} c_i}{Q(I^*) - \sum_{i\in I^*} a_i}, \tag{7}$$

gdzie $c_i \overset{def}{=} \frac{b_i}{N}$, $i \in J$.

Kładąc

$$x(I) \overset{def}{=} \sum_{i\in I} a_i - \frac{a}{2}, \quad y(I) \overset{def}{=} \sum_{i\in I} c_i - \frac{c}{2}, \quad I \subset J \tag{8}$$

$$a \overset{def}{=} \sum_{i\in J} a_i, \quad c \overset{def}{=} \sum_{i\in J} c_i$$

oraz uwzględniając (5), (6), (8), zadanie (4) sprowadza się do następującej postaci:

$$\min_{\substack{I\subset J \\ I\neq\emptyset,\ \bar{I}\neq\emptyset}} F(I), \tag{9}$$

gdzie $F(I) \overset{def}{=} x^2(I) + 2\,x(I)\,y(I)$.

Pokażemy teraz, że $F(I) < \max\{F(\emptyset), F(J)\}$ dla $I \subset J$ oraz $I \neq \emptyset$, $I \neq J$. Rzeczywiście, jeżeli $I \subset J$, $I \neq \emptyset$, $\bar{I} \neq \emptyset$, to $x(I) \in (-\frac{a}{2}, \frac{a}{2})$ i $y(I) \in (-\frac{c}{2}, \frac{c}{2})$. Stąd $x^2(I) + 2x(I)y(I) < \frac{a^2}{4} + 2\left|x(I)y(I)\right| < \frac{a^2}{4} + \frac{a\cdot c}{2} = x^2(\emptyset) + 2x(\emptyset)y(\emptyset) = x^2(J) + 2x(J)\cdot y(J)$.

Uwzględniając powyższe, zadanie (9) przyjmie ostateczną postać

$$\min_{I\subset J} F(I). \tag{10}$$

Z przedstawionego toku rozumowania wynika, że zadanie (10) jest równoważne wyjściowemu zadaniu (2), (3). Po rozwiązaniu zadania (10), tzn. wyliczeniu $I^* \subset J$, zestaw u_1^*, u_2^*, I_1^*, I_2^* taki, że $u_1^* = u^*$ (wyliczone wg (7)), $u_2^* = N - u^*$, $I_1^* = I^*$, $I_2^* = J \setminus I^*$ jest rozwiązaniem zadania wyjściowego. Dlatego też dalej będziemy rozważać tylko zadanie (10).

Zadanie (10) należy do klasy zadań NP-trudnych. Fakt ten jest konsekwencją tego, iż pewien szczególny przypadek zadania (10) określony warun-

kami $a_i = c_i \stackrel{def}{=} \alpha_i$, $i \in J$ jest równoważny NP-trudnemu problemowi PODZIAŁ ZBIORU [2] w wersji optymalizacyjnej:

$$\min_{I \subset J} \max \left\{ \sum_{i \in I} \alpha_i, \sum_{i \in \bar{I}} \alpha_i \right\}.$$

Rozważany przez nas problem był już badany we wcześniejszych pracach autora, np. w [4,5]. Do jego rozwiązania proponowano algorytm wyznaczający rozwiązanie optymalne, oparty na metodzie podziału i ograniczeń. Szczegółowy opis tego algorytmu przedstawiono w [4], a w przypadku maszyn nieidentycznych w [5]. Analizując działanie tego algorytmu zauważono, że czas obliczeń gwałtownie wzrasta przy dużej liczbie zadań (już przy $n > 50$). Fakt ten jest typowy dla tej klasy problemów optymalizacji dyskretnej i w przypadku, kiedy zależy nam na krótkim czasie obliczeń, jedynym podejściem jest zastosowanie algorytmów heurystycznych. W pracy [6] zaproponowano pewien algorytm heurystyczny dla ogólnej sytuacji, gdy występuje m identycznych maszyn. Algorytm ten można oczywiście zastosować do omawianego tutaj problemu, jednakże wydaje się, że specyficzne własności problemu występujące tylko przy dwóch maszynach umożliwiają skonstruowanie algorytmu efektywniejszego. Dlatego też w następnym rozdziale, dla wygody Czytelnika, przedstawimy wspomniany już algorytm heurystyczny z pracy [6], a następnie zaproponujemy nowy algorytm heurystyczny dostarczający z reguły lepszego rozwiązania w sensie wartości funkcji celu niż ten pierwszy.

3. Algorytmy heurystyczne

Jak już wspominaliśmy w poprzednim rozdziale, przedstawimy teraz algorytm heurystyczny z pracy [6] zaimplementowany dla przypadku dwóch maszyn. Algorytm ten będziemy dalej nazywać Algorytmem 1.

Algorytm 1

krok 1. Wyznacz czasowo-optymalne uszeregowanie n zadań na dwóch identycznych maszynach, tzn. parę I_1, I_2 przyjmując, że zadania są niepodzielne oraz czas realizacji i-tego zadania jest równy $T_i(N)$, $i \in J$.

krok 2. Wyznacz zasoby (u_1, u_2) przydzielone do poszczególnych maszyn przy założeniu, że maszyny te wykonują odpowiednio zadania ze zbiorów I_1, I_2 w taki sposób, aby czasy pracy obu maszyn były identyczne oraz żeby $u_1 + u_2 = N$, $u_1 \geq 0$, $u_2 \geq 0$.

W celu realizacji kroku 2 powyższego algorytmu należy skorzystać z przedstawionych już zależności (6) i (7) (dla $I^* = I_1$). Z kolei w kroku 1 trzeba rozwiązać klasyczny problem szeregowania n niepodzielnych zadań na dwóch równoległych identycznych maszynach. Odpowiednie algorytmy szeregowania można znaleźć w literaturze [1,3]. Problem ten, jak wiadomo, jest problemem NP-trudnym. Dlatego też wszystkie istniejące algorytmy rozwiązujące

ten problem mają złożoność niewielomianową. Jednakże istnieje szereg "dobrych" algorytmów heurystycznych (np. w [8]), które można zastosować do rozwiązania tego zagadnienia. Tutaj proponujemy przyjąć algorytm LPT (longest processing time) polegający na przydzielaniu najdłuższego zadania do aktualnie wolnej maszyny (np. w [2]).

Wykorzystując fakt, że analizowany problem (2), (3) jest równoważny w omówionym wcześniej sensie problemowi (10), oraz biorąc pod uwagę postać funkcji F(I) rekomendujemy następujący algorytm heurystyczny zwany Algorytmem 2 rozwiązujący problem (2), (3).

Algorytm 2

krok 1. Wylicz $a = \sum_{i \in J} a_i$, $c = \sum_{i \in J} c_i$, $x = -\frac{a}{2}$, $y = -\frac{c}{2}$. Połóż $k = 0$, $I^H = \emptyset$, $F_{min} = \infty$.

krok 2. Sporządź listę zadań $L = (j_1, j_2, \ldots, j_n)$ wg nierosnących wartości $\frac{c_i}{a_i}$, $i \in J$.

krok 3. Jeżeli $x^2 + 2xy \geqslant F_{min}$, to przejdź do kroku 4. W przeciwnym wypadku połóż $I = I^H$, $F_{min} = x^2 + 2xy$ i przejdź do kroku 4.

krok 4. Podstaw $k := k+1$. Jeżeli $k > n$, to połóż $I^H = \{j_n\}$, $x = -\frac{a}{2} + a_{j_n}$, $y = -\frac{c}{2} + c_{j_n}$, $k = 1$ i przejdź do kroku 5. W przeciwnym wypadku podstaw $I^H := I^H \cup \{j_k\}$, $x := x + a_{j_k}$, $y := y + c_{j_k}$ i przejdź do kroku 3.

krok 5. Jeżeli $x^2 + 2xy \geqslant F_{min}$, to przejdź do kroku 6. W przeciwnym wypadku połóż $I = I^H$, $F_{min} = x^2 + 2xy$ i przejdź do kroku 6.

krok 6. Podstaw $k := k+1$. Jeżeli $k \geqslant n$, to przejdź do kroku 7. W przeciwnym wypadku podstaw $I^H := I^H \cup \{j_{n-k+1}\}$, $x := x + a_{j_{n-k+1}}$, $y := y + c_{j_{n-k+1}}$ i przejdź do kroku 5.

krok 7. Wylicz Q(I) ze wzoru (6) (dla $I^* = I$) oraz u ze wzoru (7) (dla $I^* = I$) i połóż $I_1 = I$, $I_2 = J \setminus I$, $u_1 = u$, $u_2 = N-u$.

W następnym rozdziale przedstawimy analizę numeryczną zaprezentowanych powyżej algorytmów ze szczególnym uwzględnieniem takich elementów, jak czas obliczeń i dokładność produkowanych przez algorytmy rozwiązań.

4. Przykłady testujące i wyniki obliczeń

Przedstawione Algorytm 1 i Algorytm 2 zostały zaimplementowane w języku Fortran na minikomputerze IBM PC. Do analizy wykorzystano wcześniej już zaimplementowany na tym samym minikomputerze algorytm dokładny oparty na metodzie podziału i ograniczeń z pracy [4]. Przykłady testujące otrzy-

mano przez losowe generowanie danych. Dla określonej liczby zadań n wygenerowano 15 zestawów a_i, c_i, $i \in J$. Dla n = 10, 25, 50, 75, 100, 125, 150, 175, 200 zestawy te zostały wylosowane ze zbioru $\{0,0.1,0.2,\ldots,9.9,10.0\}$ przez generator o jednostajnym rozkładzie prawdopodobieństwa. Dla każdego zestawu danych wyznaczano I_1, I_2, u_1, u_2 wg Algorytmu 1 i wg Algorytmu 2, a następnie zestaw optymalny I_1^*, I_2^*, u_1^*, u_2^* wg algorytmu dokładnego (optymalnego). Z kolei dla odpowiedniego n i odpowiedniego zestawu a_i, c_i, $i \in J$ wyznaczano błąd względny r_1 Algorytmu 1 oraz błąd względny r_2 Algorytmu 2 wg wzorów:

$$r_1 \stackrel{def}{=} \frac{Q_1 - Q^*}{Q^*} \cdot 100\%,$$

$$r_2 \stackrel{def}{=} \frac{Q_2 - Q^*}{Q^*} \cdot 100\%,$$

gdzie Q_1 - wartość kryterium liczona za pomocą Algorytmu 1,
Q_2 - wartość kryterium liczona za pomocą Algorytmu 2,
Q^* - wartość kryterium liczona za pomocą algorytmu dokładnego (optymalnego).

Wyniki obliczeń przedstawiono w Tablicy 1.

Tablica 1.

Wyniki badań numerycznych Algorytmu 1 i Algorytmu 2

n	CPU	CPU1	CPU2	R1	R2
	sek	sek	sek	%	%
10	1	0.4	0.4	3.6	0.7
25	3	1.1	1.5	4.7	0.9
50	11	2.6	3.4	3.9	0.6
75	53	5.1	6.2	4.1	0.5
100	113	9.6	11.7	4.4	0.7
125	227	16.8	20.3	5.8	0.8
150	434	27.2	34.1	4.9	0.8
175	862	42.3	53.8	6.2	0.7
200	1896	68.8	84.3	5.7	0.9

CPU - średni czas obliczeń w sek dla 15 zestawów danych przy stosowaniu algorytmu dokładnego (optymalnego),

CPU1 - średni czas obliczeń w sek dla 15 zestawów danych przy stosowaniu Algorytmu 1,

CPU2 - średni czas obliczeń w sek dla 15 zestawów danych przy stosowaniu Algorytmu 2,

R1 - średnia arytmetyczna błędów względnych r_1 dla 15 zestawów danych przy stosowaniu Algorytmu 1,

R2 – średnia arytmetyczna błędów względnych r_2 dla 15 zestawów danych przy stosowaniu Algorytmu 2.

Jak widać, wyniki badań numerycznych upoważniają do stwierdzenia, że Algorytm 2 jest lepszy od Algorytmu 1, gdyż czas obliczeń CPU2 jest nieznacznie większy od CPU1, natomiast wartość kryterium obliczona za pomocą tego algorytmu jest dużo bardziej zbliżona do rozwiązania optymalnego niż obliczona za pomocą Algorytmu 1. Zaproponowany w pracy Algorytm 2 można rozszerzyć na przypadek, gdy liczba maszyn jest większa od dwóch. Aktualnie przeprowadzana jest analiza porównawcza pomiędzy Algorytmem 2 i Algorytmem 1 dla liczby maszyn większej od dwóch.

Ważnym problemem wydaje się rozpatrzenie innych modeli zadań (innych postaci funkcj $T_i(u_k)$), np. liniowych $T_i(u_k) = a_i - b_i u_k$.

LITERATURA

[1] Błażewicz J., Cellary W., Słowiński R., Węglarz J.: Algorytmy sterowania rozdziałem zadań i zasobów w kompleksie operacji. Wydawnictwo Politechniki Poznańskiej, Poznań 1979.

[2] Błażewicz J., Cellary W., Słowiński R., Węglarz J.: Badania operacyjne dla informatyków. WNT, Warszawa 1983.

[3] Dempster M.A.H., Lenstra J.K., Rinnooy Kan A.H.G.: Deterministic and stochastic scheduling. Proceedings of an Advanced Study and Research Institute on Theoretical Approaches to Scheduling Problems, 1981.

[4] Buchalski Z.: Czasowo-optymalny przydział pamięci operacyjnej i programów do procesorów w systemach wieloprocesorowych ze wspólną pamięcią operacyjną. Praca doktorska. Wydawnictwo Politechniki Wrocławskiej, Wrocław 1983.

[5] Buchalski Z.: Pewne zagadnienia przydziału zadań i zasobu nieodnawialnego do dwóch różnych maszyn w dyskretnym systemie produkcyjnym. Zeszyty Naukowe Politechniki Śląskiej 1984, nr 810, Automatyka z 74 s. 49-57.

[6] Buchalski Z.: Some Problem of Time-Optimal Allocation of Memory and Tasks in Multiprocessor Computer Systems. Praca Międzynarodowej Konferencji "Cybernetics'85", Warszawa 1985, s. 41-49.

[7] Ishii H., Martel C., Masuda T., Nishida T.: A Generalized Uniform Processor System. Oper. Res. 1985, vol. 33, nr 2, s. 346-362.

[8] Sahni S.: Algorithms for scheduling independent tasks, Journal of Assoc. Comput. Mach. 23, 1976, s. 116-127.

[9] Słowiński R.: Multiobjective Network Scheduling with Efficient Use of Renewable and Non-Renewable Resources. European Journal of Operational Research 1981, nr 7, s. 265-273.

[10] Talbot B.F.: Resource - Constrained Project Scheduling with Time--Resource Tradeoff. The Nonpreemtive Case. Mgt. Sci 1982, t. 28 nr 10, s. 1197-1210.

[11] Węglarz J.: Project Scheduling with Continuously - Divisible Doubly Constrained Resources, Mgt. Sci 1981, vol. 27, nr 3.

Recenzent: Dr inż.E.Toczyłowski
Wpłynęło do Redakcji do 1988-04-30.

АЛГОРИТМ ДЛЯ ПРОБЛЕМЫ СОСТАВЛЕНИЯ РАСПИСАНИЯ ЗАДАЧ НА МАШИНАХ ДЛЯ НЕКОТОРЫХ ФУНКЦИЙ ВРЕМЕНИ ВЫПОЛНЕНИЯ ЗАДАЧ

Р е з ю м е

В работе представлена задача временно-оптимального расписания непрерывных невосстанавливаемых ресурсов и n заданий на двух параллельных идентичных машинах для некоторых функций времени выполнения задач. Показано, что проблема является NP - трудной. Сформулирована математическая модель задачи и представлены два эвристических алгоритма. Представлены также результаты экспериментальных исследований алгоритмов и их эффективность.

ALGORITHM FOR PROBLEM OF SCHEDULING TASKS ON MACHINES FOR SOME PROCESSING TIME FUNCTIONS

S u m m a r y

This paper deals with the problem of time optimal allocation nonrenewable, continuous resources and n tasks to two identical, parallel machines for some processing time functions. It is shown that the problem is of NP-type. The mathematical model of the problem is formulated and two heuristic algorithms is presented. Some computational results and effectiveness of these algorithms are shown.