AUTOMATYKA • 2005 • Tom 9 • Zeszyt 1–2

Zbigniew Buchalski*

Heurystyczny algorytm szeregowania zadań w systemie maszyn równoległych z równoczesnym rozdziałem zasobów

1. Wprowadzenie

Od wielu lat prowadzi się intensywnie badania problematyki czasowo-optymalnego szeregowania zadań i rozdziału zasobów [1–5]. Zadania optymalizacji zarówno dyskretnej, jak i ciągłej, należą do klasy problemów bardzo trudnych zarówno z teoretycznego, jak i obliczeniowego punktu widzenia i najczęściej należą do klasy problemów *NP*-trudnych.

Problem szeregowania zadań na maszynach równoległych z równoczesnym rozdziałem zasobów bardzo często spotykany jest w różnego rodzaju złożonych procesach produkcyjnych. Zasobami podzielnymi w sposób ciągły najczęściej są: gaz, energia elektryczna, paliwo. W wieloprocesorowych systemach komputerowych spotykamy się z szeregowaniem programów na procesorach oraz przydziałem zasobów w postaci stron pamięci operacyjnej do procesorów [2, 6]. Przy rozwiązywaniu tych problemów występują istotne trudności natury obliczeniowej, jak również pojawiają się bardzo trudne zagadnienia rozstrzygnięcia złożoności obliczeniowej, rozpatrywanych problemów.

Wyniki teorii złożoności obliczeniowej oraz rozmiar problemów praktycznych w sposób jednoznaczny eliminują z rozważań algorytmy dokładne, pozostawiając do zastosowania praktycznego jedynie algorytmy heurystyczne umożliwiające rozwiązanie postawionych problemów w krótkim czasie z zadowalającą dokładnością. Badania nad algorytmami heurystycznymi, dostarczającymi rozwiązań zagadnień, w których zastosowanie metod dokładnych jest nieefektywne lub wręcz niemożliwe, stanowią jedną z najszybciej rozwijających się gałęzi nauki [7–13].

W związku z rozwojem równoległych systemów przetwarzania informacji szczególnego znaczenia nabiera problem szeregowania zadań z systemem maszyn działających równolegle. Niniejsza praca dotyczy zagadnienia czasowo-optymalnego przydziału zasobów nieodnawialnych podzielnych w sposób ciągły do m różnych maszyn równoległych. W rozdziale drugim formułuje się problem oraz podany jest jego model matematyczny. W rozdziale trzecim przedstawiony został algorytm heurystyczny rozwiązujący postawiony problem, a w rozdziale czwartym podane zostały wyniki eksperymentów obliczeniowych przeprowadzonych na tym algorytmie.

* Instytut Cybernetyki Technicznej Politechniki Wrocławskiej

2. Sformułowanie zagadnienia

Rozpatrzmy dyskretny system produkcyjny zawierający maszyny połączone równolegle.

Na system maszyn równoległych nakładamy następujące założenia:

(i) posiada m różnych maszyn $M = \{1, 2, ..., k, ..., m\}$, na których należy wykonać n niezależnych zadań $Z = \{1, 2, ..., i, ..., n\}$;
(ii) zadanie może być wykonywane na dowolnej maszynie i w trakcie jego wykonywania nie może być przerywane;
(iii) liczba zadań do wykonania jest większa od liczby maszyn $n > m$;
(iv) realizacja każdego z zadań na maszynach musi następować niezwłocznie po zakończeniu wykonywania poprzedniego zadania lub nastąpić w chwili zerowej, gdy zadanie realizowane jest jako pierwsze na jednej z maszyn.

Niech N oznacza globalną ilość zasobów nieodnawialnych, a przez u_k oznaczmy tę część zasobów, które zostaną przydzielone k-tej maszynie w trakcie wykonywania zadań uszeregowanych na tej maszynie. Ograniczenie dotyczące zasobów jest następujące:

$$\sum_{k=1}^{m} u_k \le N, \quad u_k \ge 0, \quad 1 \le k \le m.$$

Czas wykonywania i-tego zadania na k-tej maszynie określony jest przez następującą funkcję $T_i(u_k, k)$:

$$T_i\left(u_k, k\right) = a_{ik} + \frac{b_{ik}}{u_k}, \quad u_k \in \left\{1, 2, ..., N\right\}, \quad 1 \le k \le m, \quad 1 \le i \le n \tag{1}$$

Parametry $a_{ik} > 0$ i $b_{ik} > 0$ charakteryzują i-te zadanie i k-tą maszynę.

Należy znaleźć takie uszeregowanie zadań na maszynach i taki przydział ograniczonych zasobów do maszyn równoległych, aby minimalizować czas zakończenia wykonania całego zbioru zadań T_{zak}.

Jeżeli oznaczymy przez $Z_k \subset Z$ zbiór zadań uszeregowanych na k-tej maszynie, to T_{zak} znajdziemy rozwiązując następujący problem minimalizacyjny

$$T_{zak} = \min_{\substack{Z_1, Z_2, ..., Z_m \\ u_1, u_2, ..., u_m}} \max_{1 \le k \le m} \left\{ \sum_{i \in Z_k} T_i\left(u_k, k\right) \right\} \tag{2}$$

Ograniczenia nałożone na rozwiązanie tego problemu są następujące:

(i) $Z_r \cap Z_s = \phi; \quad r, s = 1, 2, ..., m, \quad r \ne s, \quad \bigcup_{k=1}^{m} Z_k = Z;$

(ii) $\sum_{k=1}^{m} u_k \le N;$

(iii) $u_1, u_2, ..., u_m$ – całkowite dodatnie.

Dla uproszczenia problemu przyjmiemy najpierw, że zasoby nieodnawialne $u_1, u_2, ..., u_m$ są typu ciągłego. Przy tym założeniu wyznaczymy rozwiązanie optymalne, a następnie zaokrąglimy otrzymane wartości zasobów do najbliższych liczb naturalnych. Tak więc, czas T_{zak} znajdziemy rozwiązując następujący problem minimalizacji dyskretno-ciągłej

$$T_{zak} = \min_{\substack{Z_1,Z_2,...,Z_m \\ u_1,u_2,...,u_m}} \max_{1\le k\le m} \left\{ \sum_{i\in Z_k} T_i'(u_k, k) \right\} \tag{3}$$

przy następujących ograniczeniach:

(i) $Z_r \cap Z_s = \phi; \quad r, s = 1, 2, ..., m, \qquad r \neq s, \qquad \bigcup_{k=1}^{m} Z_k = Z;$

(ii) $\sum_{k=1}^{m} u_k \le N, \quad u_k \ge 0, \quad k = 1, 2, ..., m;$

gdzie $T_i' : [0, N] \times \{1, 2, .., m\} \to R^+$ jest rozszerzeniem następującej funkcji $T_i : \{1, 2, ..., N\} \times \{1, 2, ..., m\} \to R^+$ i określone jest przez funkcję:

$$T_i'(u_k, k) = a_{ik} + \frac{b_{ik}}{u_k}, \quad u_k \in [0, N], \quad 1 \le k \le m, \quad 1 \le i \le n \tag{4}$$

Do rozwiązania postawionego problemu pomocny będzie następujący lemat.

Lemat 1

Jeżeli $u_k^*, Z_k^*, k = 1, 2, ..., m$ *są rozwiązaniami zadania (3), to:*

(i) $\sum_{k=1}^{m} u_k^* = N; \quad u_k^* > 0, \quad k : Z_k^* \neq \phi, \quad k = 1, 2, ..., m;$

$u_k^* = 0, \quad k : Z_k^* = \phi, \quad k = 1, 2, ..., m;$

(ii) $\sum_{i\in Z_k^*} T_i'(u_k^*, k) = const, \quad k : Z_k^* \neq \phi, \quad k = 1, 2, ..., m.$

Warunek (i) w **lemacie 1** oznacza, że w przydziale czasowo-optymalnym zasobów i zadań do maszyn wykorzystuje się wszystkie jednostki zasobów, a warunek (ii), że czasy pracy tych maszyn, na których wykonywane są jakieś zadania, są identyczne.

Zdefiniujmy funkcję $F(Z_1, Z_2, ..., Z_m)$ określoną dla m zbiorów $Z_1, Z_2, ..., Z_m$, dla których zachodzi ograniczenie (i) dla wzoru (3).

Wartość tej funkcji jest rozwiązaniem następującego układu równań:

$$\begin{cases} \sum\limits_{i\in Z_k} a_{ik} + \dfrac{\sum\limits_{i\in Z_k} b_{ik}}{u_k} = F(Z_1, Z_2, ..., Z_m) \quad ; \qquad k : Z_k \neq \varnothing, \quad k = 1, 2, ..., m \\ \sum\limits_{k=1}^{m} u_k = N, \qquad u_k > 0, \qquad k : Z_k \neq \varnothing, \quad k = 1, 2, ..., m \end{cases} \tag{5}$$

Wykorzystując **lemat 1** oraz (5) zadanie minimalizacji (3) można przedstawić w następującej postaci

$$T_{zak} = \min_{Z_1, Z_2, ..., Z_m} F\left(Z_1, Z_2, ..., Z_m\right) \tag{6}$$

przy ograniczeniach:

(i) $Z_r \cap Z_s = \phi, \quad r, s = 1, 2, ..., m, \quad r \neq s;$

(ii) $\bigcup\limits_{k=1}^{m} Z_k = Z.$

Jeżeli $Z_1^*, Z_2^*, .., Z_m^*$ jest rozwiązaniem zadania (6), to $u_k^*, Z_k^*, k = 1, 2, ..., m$, gdzie

$$u_k^* = \begin{cases} \dfrac{\sum\limits_{i\in Z_k^*} b_{ik}}{F\left(Z_1^*, Z_2^*, ..., Z_m^*\right) - \sum\limits_{i\in Z_k^*} a_{ik}}; \quad k : Z_k^* \neq \phi, \quad 1 \leq k \leq m \\ 0 \qquad\qquad\qquad\qquad ; \; k : Z_k^* = \phi, \quad 1 \leq k \leq m \end{cases} \tag{7}$$

jest rozwiązaniem zadania (3).

3. Algorytm heurystyczny

Maszyny wchodzące w skład systemu maszyn równoległych różnią się pod względem szybkości wykonywanych zadań. Na szybkość tę wpływ ma ilość zasobów przydzielonych poszczególnym maszynom. Im więcej zasobów zostanie przydzielonych *k*-tej maszynie, tym będzie ona szybsza.

Zasoby przydzielone zostają do maszyn w następujący sposób:

1. miarą szybkości realizacji *i*-tego zadania przez *k*-tą maszynę jest tzw. współczynnik podziału zasobów β; $\beta > 1$;
2. zakładamy, że maszyną najszybszą jest maszyna pierwsza, a maszyną najwolniejszą jest maszyna *m*-ta;

3. maszynie *m*-tej przydzielamy u_m zasobów wg następującej zależności

$$u_m = \frac{N}{1 + \sum_{k=1}^{m-1} [(m-k)\cdot\beta]} \tag{8}$$

4. pozostałym maszynom przydzielamy zasoby wg następującej zależności:

$$u_k = (m-k)\cdot\beta\cdot u_m; \qquad k = 1, 2, ..., m-1 \tag{9}$$

Przedstawiony powyżej sposób przydziału zasobów do maszyn wykorzystany zostanie w zaproponowanym heurystycznym algorytmie szeregowania zadań na równoległych maszynach. Algorytm ten skonstruowany został w taki sposób, że najpierw szereguje on zadania na jednakowych maszynach, tj. takich, do których przydzielona została jednakowa liczba dostępnych zasobów, czyli $u_k = \frac{N}{m}$, k = 1, 2, …, m. Po tym uszeregowaniu następuje zróżnicowanie maszyn pod względem liczby przydzielanych im zasobów i sprawdzenie czy skrócony został czas zakończenia wykonywania wszystkich zadań T_{zak}.

Kolejne kroki algorytmu heurystycznego są następujące:

Krok 1.

Oblicz czasy wykonywania zadań na poszczególnych maszynach $T_i(u_k, k) = a_{ik} + \frac{b_{ik}}{u_k}$, i = 1, 2, …, n, k = 1, 2, ..., m dla zadanej wartości $u_k = \frac{N}{m}$ i losowo generowanych parametrów a_{ik}, b_{ik}.

Krok 2.

Uszereguj malejąco czasy wykonywania poszczególnych zadań i utwórz listę L tych zadań.

Krok 3.

Oblicz średni czas $T_{śr}$ wykonywania zadań przez każdą z maszyn według wzoru:

$$T_{śr} = \frac{\sum_{i=1}^{n} T_i(u_k, k)}{m}; \qquad i \in Z, k \in M, u_k = \frac{N}{m}.$$

Krok 4.

Przydzielaj kolejno najdłuższe i najkrótsze zadania z listy L do pierwszej wolnej maszyny aż do momentu, gdy suma czasów wykonywania zadań przydzielonych tej maszynie, nie przekroczy czasu $T_{śr}$. Przydzielone zadania usuń z listy L.

Krok 5.

Jeżeli jest maszyna, której nie przydzielono jeszcze żadnych zadań, to wróć do **kroku 4.** W przeciwnym wypadku przejdź do **kroku 6.**

Krok 6.

Jeżeli lista L nie została jeszcze wyczerpana, to kolejne zadania z tej listy przydziel do maszyn według algorytmu LPT(*Longest Processing Time*) aż do momentu wyczerpania się listy tych zadań.

Krok 7.

Oblicz czas zakończenia wykonywania wszystkich zadań T_{zak} dla uszeregowania zadań na maszynach utworzonego w **krokach 4–6** i dla $u_k = \frac{N}{m}$.

Krok 8.

Dla zadanego współczynnika β przydziel zasoby u_k, $k = 1, 2, ..., m$ poszczególnym maszynom wyliczone z zależności (8) i (9).

Krok 9.

Dla uszeregowania zadań na maszynach utworzonego w **krokach 4–6** i dla liczby zasobów u_k przydzielonych maszynom w **kroku 8** oblicz czas zakończenia wykonywania wszystkich zadań T_{zak}.

Krok 10.

Powtórz **krok 8** i **krok 9** dla następnych dziewięciu zwiększających się kolejno wartości współczynnika β. Po zakończeniu tych prób przejdź do **kroku 11**.

Krok 11.

Porównaj wartości czasów zakończenia wykonywania zadań T_{zak} z kolejnych prób i wybierz najkrótszy z tych czasów.

Krok 12.

Wyznacz dyskretne ilości zasobów $\hat{u}_k, k = 1, 2, \ldots, m$ według zależności:

$$\hat{u}_{\alpha(k)} = \begin{cases} \lfloor u_{\alpha(k)} \rfloor + 1; & k = 1, 2, \ldots, \Delta, \\ \lfloor u_{\alpha(k)} \rfloor & k = \Delta + 1, \Delta + 2, ..., m, \end{cases}$$

gdzie $\Delta = N - \sum_{j=1}^{m} \lfloor u_j \rfloor$ oraz α jest permutacją elementów zbioru $M = \{1, 2, ..., m\}$ taką, że $u_{\alpha(1)} - \lfloor u_{\alpha(1)} \rfloor \geq u_{\alpha(2)} - \lfloor u_{\alpha(2)} \rfloor \geq \ldots \geq u_{\alpha(m)} - \lfloor u_{\alpha(m)} \rfloor$.

Jeżeli istnieją takie maszyny, którym przydzielono zerowe ilości zasobów, to przydziel każdej z tych maszyn po jednej jednostce zasobu, pobierając je z kolejnych maszyn poczynając od maszyny, której przydzielono największą ilość zasobów.

4. Eksperymenty obliczeniowe

Przeprowadzono eksperymenty obliczeniowe na bazie przedstawionego algorytmu dla dziesięciu zwiększających się kolejno wartości współczynnika podziału zasobów β z prze-

działu [5, 15, ..., 95]. Parametry charakteryzujące *i*-te zadanie i *k*-tą maszynę a_{ik}, b_{ik} wylosowane zostały ze zbioru {5,0, 10,0, ..., 50,0} przez generator o jednostajnym rozkładzie prawdopodobieństwa. Dla każdej kombinacji *n* i *m* wygenerowano 50 instancji. Rezultaty analizy porównawczej algorytmu heurystycznego skonstruowanego dla potrzeb niniejszej pracy i znanego z literatury algorytm LPT przedstawione zostały w tabeli 1.

Tabela1
Wyniki analizy porównawczej algorytmu heurystycznego i algorytmu LPT

n/*m*	Liczba instancji, dla których:			Δ^H	S^H	S^{LPT}
	$T_{zak}^H < T_{zak}^{LPT}$	$T_{zak}^H = T_{zak}^{LPT}$	$T_{zak}^H > T_{zak}^{LPT}$	%	s	s
30/2	28	2	20	1,8	1,6	1,2
30/4	30	2	18	2,9	1,9	1,8
30/6	32	1	17	3,7	3,2	3,0
30/8	34	3	13	3,0	3,9	3,6
60/2	28	2	20	2,6	2,4	2,4
60/4	30	3	17	3,2	2,9	2,8
60/6	26	3	21	3,6	3,2	3,3
60/8	31	1	18	3,7	3,8	3,6
90/2	28	0	22	2,8	3,6	3,4
90/4	27	2	21	2,1	4,1	4,0
90/6	31	1	18	2,6	4,6	4,5
90/8	30	0	20	2,2	4,9	4,7
120/2	26	4	20	2,9	5,4	5,6
120/4	29	3	18	3,0	6,4	6,2
120/6	32	2	16	3,6	9,2	8,8
120/8	31	0	19	3,9	10,3	9,8

W tabeli 1 występują następujące wielkości:

n – liczba zadań;
m – liczba maszyn;
T_{zak}^H – czas zakończenia wykonywania wszystkich zadań ze zbioru *Z* przy wykorzystaniu algorytmu heurystycznego;
T_{zak}^{LPT} – czas zakończenia wykonywania wszystkich zadań ze zbioru *Z* przy wykorzystaniu algorytmu LPT;

Δ^H – średnia procentowa poprawa czasu T_{zak}^H w stosunku do T_{zak}^{LPT}

$$\Delta^H = \frac{T_{zak}^{LPT} - T_{zak}^H}{T_{zak}^H} \cdot 100\%;$$

S^H – średni czas obliczeń dla algorytmu heurystycznego;
S^{LPT} – średni czas obliczeń dla algorytmu LPT.

5. Uwagi końcowe

Przedstawione w poprzednim rozdziale eksperymenty obliczeniowe wykazały, że jakość szeregowania zadań na równoległych maszynach na bazie zaproponowanego w pracy algorytmu heurystycznego uległa poprawie w stosunku do szeregowania za pomocą znanego z literatury algorytmu LPT. Kilkuprocentowa poprawa czasu T_{zak}^H w stosunku do T_{zak}^{LPT} może być zachętą do dalszych prac nad efektywnymi algorytmami heurystycznymi.

Zastosowanie podanego w pracy algorytmu heurystycznego jest wskazane przede wszystkim dla systemów produkcyjnych o dużej liczbie zadań, gdyż wówczas średnia procentowa poprawa Δ^H jest największa. Zaproponowany algorytm może służyć zarówno do rozdziału operacji na stanowiska produkcyjne wyposażone w odpowiednie maszyny w dyskretnych systemach produkcyjnych, jak i do szeregowania programów w wieloprocesorowych systemach komputerowych.

Literatura

[1] Błażewicz J., Dell'olmo P., Drozdowski M., Speranza M.G.: *Scheduling multiprocessor tasks on three dedicated processors.* Information Processing Letters, 41, 1992, 275–280
[2] Błażewicz J., Drabowski M., Węglarz J.: *Scheduling multiprocessor tasks to minimize schedule length.* IEEE Transactions on Computers C–35, 1986, 389–393
[3] Boctor F. F.: *A new and efficient heuristic for scheduling projects will resources restrictions and multiple execution models.* European Journal of Operational Research, vol. 90, 1996, 349–361
[4] Ishii H., Martel C., Masuda T., Nishida T.: *A generalized uniform processor system*. Oper. Res., vol. 33, 1985, 346–362
[5] Janiak A.: *Single machine scheduling problem with a common deadline and resource dependent release dates.* European Journal of Operational Research, vol. 53, 1991, 317–325
[6] Buchalski Z.: *Optimization of programs scheduling and primary memory allocation in multiprocessing computer systems.* Information Systems Architecture and Technology ISAT'98, Wrocław, 1998, 246–253
[7] Bachman A., Janiak A.: *Jednomaszynowy problem szeregowania zadań czasowo i zasobowo zależnych przy kryterium minimalizacji czasu zakończenia wykonywania zadań.* Zeszyty Naukowe Politechniki Śląskiej Nr 1474, seria – Automatyka, Gliwice, z. 129, 2000, 23–32
[8] Bianco L., Błażewicz J., Dell'olmo P., Drozdowski M.: *Preemptive multiprocessors task scheduling with release times and times windows.* Annals of Operations Research, 70, 1997, 43–55
[9] Buchalski Z.: *Szeregowanie zadań w systemach wielomaszynowych z czasem realizacji zależnym od ilości zasobów*. Zeszyty Naukowe Politechniki Śląskiej, Nr 1474, seria – Automatyka, Gliwice, z. 129, 2000, 33–39

[10] Janiak A., Kovalyov M.: *Single machine scheduling subject to deadlines and resources dependent processing times.* European Journal of Operational Research, vol. 94, 1996, 284–291
[11] Józefowska J., Węglarz J.: *On a methodology for discrete-continuous scheduling.* European Journal of Operational Research, vol. 107, 1998, 338–353
[12] Józefowska J., Mika M., Różycki R., Waligóra G., Węglarz J.: *Rozwiązywanie dyskretno-ciągłych problemów rozdziału zasobów przez dyskretyzację zasobu ciągłego.* Zeszyty Naukowe Politechniki Śląskiej, Nr 1474, seria – Automatyka, Gliwice, z. 129, 2000, 221–229
[13] Nowicki E., Smutnicki C.: *The flow shop with parallel machines. A tabu search approach*. European Journal of Operational Research, 106, 1998, 226–253