

HARMONOGRAMOWANIE ROZDZIAŁU ZADAŃ I ZASOBÓW W WIELOMASZYNOWYM SYSTEMIE PRODUKCYJNYM

Zbigniew BUCHALSKI

Streszczenie: W artykule przedstawiono zagadnienie czasowo-optimalnego przydziału n zadań niezależnych niepodzielnych i zasobów nieodnawialnych podzielnych w sposób ciągły do m maszyn pracujących równolegle. Czas wykonania i -tego zadania na k -tej maszynie określony jest pewną funkcją zależną od liczby jednostek zasobu przydzielonego k -tej maszynie oraz od parametrów charakteryzujących wykonywane zadanie. Dla zadanej funkcji czasu realizacji zadań zaproponowano pewien algorytm heurystyczny wyznaczający czasowo-optimalne harmonogramowanie zadań i przydział zasobów do maszyn równoległych. Przedstawiono wyniki badań numerycznych przeprowadzonych na tym algorytmie.

Słowa kluczowe: systemy wielomaszynowe, szeregowanie zadań, rozdział zasobów, algorytmy heurystyczne.

1. Wstęp

Intensywny rozwój równoległych systemów przetwarzania informacji pociągnął za sobą wzrost zainteresowania problematyką czasowo-optimalnego szeregowania zadań i rozdziału zasobów [1, 2, 3, 4, 5, 6, 7, 8]. Prezentowana praca bazuje na wynikach tych badań i jest kontynuacją wcześniejszych prac autora [9, 10, 11].

Problem szeregowania zadań na maszynach równoległych z równoczesnym rozdziałem zasobów bardzo często spotykany jest w różnego rodzaju złożonych procesach produkcyjnych. Czasy wykonywania zadań mogą być z góry zadane lub też mogą zmieniać się w zależności od współdzielonych zasobów. Zadania mogą być pewnymi procesami dynamicznymi opisywanymi za pomocą skomplikowanych modeli matematycznych.

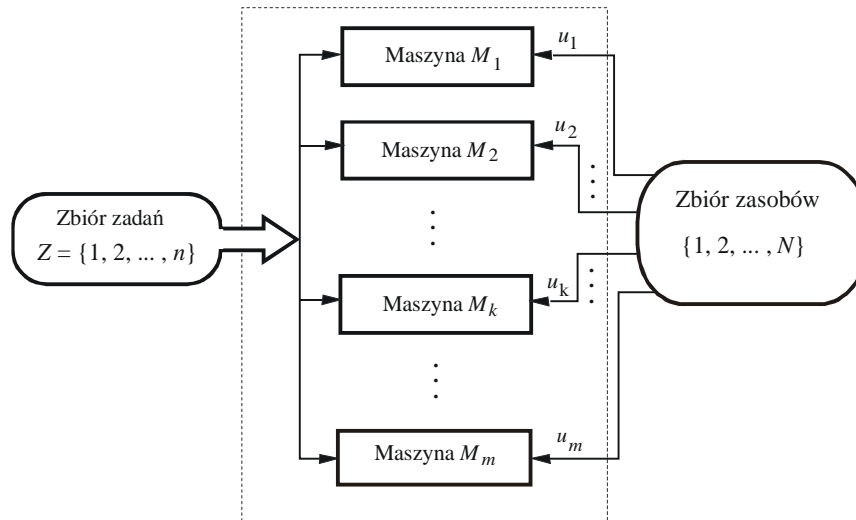
Wyniki teorii złożoności obliczeniowej oraz rozmiar problemów praktycznych w sposób jednoznaczny eliminują z rozważań algorytmy dokładne, pozostawiając do zastosowania praktycznego jedynie algorytmy heurystyczne umożliwiające rozwiązanie postawionych problemów w krótkim czasie z zadowalającą dokładnością. 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.

W niniejszym artykule przedstawiono pewien algorytm heurystyczny wyznaczający czasowo-optimalne harmonogramowanie rozdziału n zadań niezależnych niepodzielnych i jednostek zasobu nieodnawialnego podzielnego w sposób ciągły do m maszyn pracujących równolegle. Przedstawiono wyniki badań numerycznych przeprowadzonych na tym algorytmie dla losowo generowanych danych.

2. Sformułowanie problemu

Rozpatrzmy dyskretny system produkcyjny zawierający maszyny połączone równolegle

przedstawiony na poniższym rysunku:



Rys. 1. System maszyn równoległych

Na system maszyn równoległych nakładamy następujące założenia:

- (i) posiada m różnych maszyn $M = \{1, 2, \dots, k, \dots, m\}$, na których należy wykonać n niezależnych zadań $Z = \{1, 2, \dots, i, \dots, n\}$,
- (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:

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

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

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

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} .

3. Model formalny problemu

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

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

Ograniczenia nałożone na rozwiązanie tego problemu są następujące:

- (i) $Z_r \cap Z_s = \emptyset$; $r, s = 1, 2, \dots, m$, $r \neq s$, $\bigcup_{k=1}^m Z_k = Z$,
- (ii) $\sum_{k=1}^m u_k \leq N$,
- (iii) u_1, u_2, \dots, u_m - całkowite dodatnie.

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

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

przy następujących ograniczeniach:

- (i) $Z_r \cap Z_s = \emptyset$; $r, s = 1, 2, \dots, m$, $r \neq s$, $\bigcup_{k=1}^m Z_k = Z$,
- (ii) $\sum_{k=1}^m u_k \leq N$; $u_k \geq 0$, $k = 1, 2, \dots, m$,

gdzie: $T_i': [0, N] \times \{1, 2, \dots, m\} \rightarrow R^+$ jest rozszerzeniem następującej funkcji

$T_i: \{1, 2, \dots, N\} \times \{1, 2, \dots, m\} \rightarrow R^+$ i określone jest przez funkcję:

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

Do rozwiązania postawionego problemu pomocny będzie następujący lemat:

LEMAT 1

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

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

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

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

Warunek (i) w **LEMATIE 1** oznacza, że w przydziale czasowo-optimalnym 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 $F(Z_1, Z_2, \dots, Z_m)$ określoną dla m zbiorów Z_1, Z_2, \dots, Z_m , dla których zachodzi ograniczenie (i) dla wzoru (3). Wartość tej funkcji jest rozwiązaniem następującego układu równań:

$$\begin{cases} \sum_{i \in Z_k} a_{ik} + \frac{\sum_{i \in Z_k} b_{ik}}{u_k} = F(Z_1, Z_2, \dots, Z_m); & k : Z_k \neq \emptyset, \quad k = 1, 2, \dots, m \\ \sum_{k=1}^m u_k = N; \quad u_k > 0, \quad k : Z_k \neq \emptyset, \quad k = 1, 2, \dots, m. \end{cases} \quad (5)$$

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

$$T_{zak} = \min_{Z_1, Z_2, \dots, Z_m} F(Z_1, Z_2, \dots, Z_m), \quad (6)$$

przy następujących ograniczeniach:

$$(i) Z_r \cap Z_s = \emptyset, \quad r, s = 1, 2, \dots, m, \quad r \neq s,$$

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

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

$$u_k^* = \begin{cases} \frac{\sum_{i \in Z_k^*} b_{ik}}{F(Z_1^*, Z_2^*, \dots, Z_m^*) - \sum_{i \in Z_k^*} a_{ik}}; & k : Z_k^* \neq \emptyset, \quad 1 \leq k \leq m, \\ 0 & ; k : Z_k^* = \emptyset, \quad 1 \leq k \leq m \end{cases} \quad (7)$$

jest rozwiązaniem zadania (3).

4. Algorytm heurystyczny

Maszyny wchodzące w skład systemu maszyn równoległych różnią się pod względem szybkości wykonywanych zadań. Na szybkość tą 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:

- miarą szybkości realizacji i -tego zadania przez k -tą maszynę jest tzw. współczynnik podziału zasobów β ; $\beta > 1$,
- zakładamy, że maszyną najszybszą jest maszyna pierwsza, a maszyną najwolniejszą jest maszyna m -ta,
- maszynie m -tej przydzielamy u_m zasobów wg następującej zależności:

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

- pozostałym maszynom przydzielamy zasoby wg następującej zależności:

$$u_k = (m-k) \cdot \beta \cdot u_m; \quad k = 1, 2, \dots, m-1. \quad (9)$$

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 $u_k = \frac{N}{m}$, $k = 1, 2, \dots, m$. 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

$$T_i(u_k, k) = a_{ik} + \frac{b_{ik}}{u_k}, \quad i = 1, 2, \dots, n, k = 1, 2, \dots, m \text{ dla zadanej wartości}$$

$$u_k = \frac{N}{m} \text{ i losowo generowanych parametrów } a_{ik}, b_{ik}.$$

Krok 2. Uszereguj malejąco czasy wykonywania poszczególnych zadań i utwórz listę L tych zadań.

Krok 3. Oblicz średni czas T_{sr} wykonywania zadań przez każdą z maszyn wg wzoru:

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

Krok 4. Przydziel kolejno m pierwszych zadań z listy L do kolejnych maszyn od pierwszej poczynając a na m -tej kończąc, a następnie usuń te zadania z listy L .

Krok 5. Przydzielaj kolejno najkrótsze zadania z listy L do kolejnych maszyn od pierwszej poczynając aż do momentu, gdy suma czasów realizacji zadań przez kolejne maszyny nie przekroczy czasu T_{sr} . Przydzielone zadania usuń z listy L .

Krok 6. Jeżeli lista L nie została jeszcze wyczerpana to kolejne zadania z tej listy przydziel do maszyn wg algorytmu LPT .

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 danego współczynnika β przydziel zasoby $u_k, k = 1, 2, \dots, m$ 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 siedmiu 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 $\hat{u}_k, k = 1, 2, \dots, m$ według zależności:

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

gdzie $\Delta = N - \sum_{j=1}^m \lfloor u_j \rfloor$ oraz α jest permutacją elementów zbioru $M = \{1, 2, \dots, m\}$.

$$m) \text{ taką, że } u_{\alpha(1)} - \lfloor u_{\alpha(1)} \rfloor \geq u_{\alpha(2)} - \lfloor u_{\alpha(2)} \rfloor \geq \dots \geq u_{\alpha(m)} - \lfloor u_{\alpha(m)} \rfloor.$$

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.

5. Wyniki badań numerycznych

Przeprowadzono badania numeryczne na bazie przedstawionego algorytmu dla ośmiu zwiększających się kolejno wartości współczynnika podziału zasobów β z przedziału [3, 6, ..., 24]. Parametry charakteryzujące i -te zadanie i k -tą maszynę a_{ik} , b_{ik} wylosowane zostały ze zbioru {3.0, 6.0, ..., 60.0} przez generator o jednostajnym rozkładzie prawdopodobieństwa. Dla każdej kombinacji n i m wygenerowano 30 instancji. Rezultaty analizy porównawczej algorytmu heurystycznego skonstruowanego dla potrzeb niniejszej pracy i znanego z literatury algorytm LPT przedstawione zostały w Tab.1.

Tab. 1. Wyniki analizy porównawczej algorytmu heurystycznego i algorytmu LPT

n/m	Liczba instancji, dla których:			Δ^H	S^H	S^{LPT}
	$T_{zak}^H < T_{zak}^{LPT}$	$T_{zak}^H = T_{zak}^{LPT}$	$T_{zak}^H > T_{zak}^{LPT}$	%	sek	sek
30/2	16	2	12	2,3	2,1	1,8
30/4	15	2	13	2,9	2,5	2,1
30/6	15	3	12	3,3	3,8	3,2
30/8	16	1	13	3,8	4,4	4,1
30/10	17	1	12	4,1	5,6	5,1
60/2	15	2	13	1,7	2,9	2,6
60/4	16	2	12	2,5	3,9	3,4
60/6	16	0	14	3,4	4,6	3,9
60/8	16	1	13	4,4	5,7	4,9
60/10	16	2	12	4,8	6,8	5,5
90/2	14	3	13	2,6	3,7	3,2
90/4	15	1	14	2,9	5,3	4,9
90/6	16	0	14	3,6	6,6	5,8
90/8	17	0	13	4,5	7,9	6,9
90/10	17	1	12	5,0	8,9	7,9
120/2	15	2	13	2,8	4,9	4,6
120/4	16	0	14	2,9	5,7	5,2
120/6	16	2	12	3,1	7,9	6,8
120/8	17	2	11	4,8	8,9	7,9
120/10	17	3	10	5,9	10,5	9,7

W Tab. 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 .

6. Podsumowanie

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, pp. 275-280.
2. Boctor F. F.: A new and efficient heuristic for scheduling projects with resources restrictions and multiple execution models. *European Journal of Operational Research*, Vol. 90, 1996, pp. 349-361.
3. Janiak A.: Single machine scheduling problem with a common deadline and resource dependent release dates. *European Journal of Operational Research*, Vol. 53, 1991, pp. 317-325.
4. Janiak A., Kovalyov M.: Single machine scheduling subject to deadlines and resources dependent processing times. *European Journal of Operational Research*, 1996, Vol. 94, pp. 284-291.
5. Józefowska J., Węglarz J.: On a methodology for discrete-continuous scheduling. *European Journal of Operational Research*, Vol. 107, 1998, pp. 338-353.
6. 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, 2000, z. 129, s. 221-229.
7. Kubale M., Giaro K.: Złożoność zwarteo szeregowania zadań jednostkowych w systemie otwartym, przepływowym i mieszanym. Uczelniane Wydawnictwo Naukowo-Dydaktyczne AGH, seria-Automatyka, półrocznik, tom 5, zeszyt ½, Kraków, 2001, s. 329-334.
 8. Nowicki E., Smutnicki C.: The flow shop with parallel machines. A Tabu search approach. *European Journal of Operational Research* 106, 1998, pp. 226-253.
 9. Buchalski Z.: An heuristic solution procedure to minimize the total processing time of programs in multiprocessing computer system. *Information Systems Architecture and Technology ISAT 2005*, Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław, 2005, pp. 20-26.
 10. Buchalski Z.: A Program Scheduling Heuristic Algorithm in Multiprocessing Computer System with Limited Memory Pages. *Polish Journal of Environmental Studies*, Vol. 15, No. 4C, 2006, pp. 26-29.
 11. Buchalski Z.: An Heuristic Algorithm for Solving the Scheduling Problem in Multiprocessing Computer System. *Polish Journal of Environmental Studies*, Vol. 16, No. 4A, 2007, pp. 44-48.

Dr inż. Zbigniew BUCHALSKI
Instytut Informatyki, Automatyki i Robotyki
Politechnika Wroclawska
50-372 Wrocław, Janiszewskiego 11/17
tel.: (0-71) 320 32 92
e-mail: zbigniew.buchalski@pwr.wroc.pl