ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1982

Seria: AUTOMATYKA z. 63 Nr kol. 735

Jacek BŁAŻEWICZ

Politechnika Poznańska

WIELOMIANOWE ALGORYTMY SZEREGOWANIA ZADAŃ NA RÓWNOLEGŁYCH MASZYNACH PRZY OGRANICZONYCH ŻĄDANIACH ZASOBOWYCH

Streszczenie. W pracy rozpatrzono problem szeregowania zadań na równoległych maszynach przy ograniczeniach ze strony dodatkowych zasobów. Zbadano wpływ ograniczenia zbioru możliwych wartości parametrów, dotyczących maszyn i dodatkowych zasobów, na złożoność obliczeniową problemu. Wykazano istnienie dla pewnych przypadków algorytmów wielomianowych.

1. WSTĘP

Problematyka deterministycznego szeregowania zadań na maszynach (procesorach) stanowi przedmiot badań od około trzydziestu lat. Szczególnym zainteresowaniem cieszą się w ostatnim czasie zagadnienia szeregowania, z uwzględnieniem dodatkowych (oprócz maszyn) zasobów. Wynika to głównie z faktu, że model sformułowany na gruncie tej teorii jest bardzo użyteczny przy opisie zagadnień związanych z rozdziałem zasobów w systemach komputerowych. Konstruowane dla tych zastosowań algorytmy szeregowania winny być jak najprostsze, w sensie złożoności obliczeniowej, gdyż tylko takie mogą znaleźć bezpośrednie zastosowanie w systemach operacyjnych sterujących pracą systemów komputerowych.

Wykorzystując metody właściwe dla teorii złożoności obliczeniowej, można udowodnić [2, 5] przynależność ogólnego problemu szeregowania zadań z dodatkowymi zasobami do klasy problemów NP-zupełnych [1, 7, 8], to jest takich, dla których algorytmów efektywnych (o złożoności wielomianowej) najprawdopodobniej nie można skonstruować. Należy zatem zbadać, czy istnieją podproblemy rozwiązywalne w wielomianowym czasie, a także, które parametry problemu odgrywają istotną rolę przy zmianie charakteru problemu z nierozwiązywalnego na rozwiązywalny w czasie wielomianowym. W pracy tej pokażemy, że istotną rolę odgrywa w tym przypadku ograniczenie liczby dodatkowych zasobów oraz ograniczenie żądań zasobowych do określonego zbioru liczb.

Rozdział 2 zawiera niezbędne definicje. W rozdziale 3 przedstawimy natomiast trzy podproblemy ogólnego problemu szeregowania zadań o jednostkowych czasach wykonywania z dodatkowymi zasobami, które można rozwiązać

w wielomianowym czasie dzięki ograniczeniu niektórych parametrów związanych z żądaniami zasobowymi zadań.

2. DEFINICJE

Rozpatrywać będziemy zbiór n zadań $\{Z_1, Z_2, \ldots, Z_n\}$, zbiór m maszyn $\{M_1, M_2, \ldots, M_m\}$ oraz zbiór s rodzajów dodatkowych zasobów $\{R_1, R_2, \ldots, R_s\}$, dostępnych w liczbie odpowiednio $m_1, m_2, \ldots, m_s$ jednostek. O maszynach założymy, że są równoległe i identyczne, to znaczy spełniają te same funkcje, a ich prędkości wykonywania zadań są identyczne. Każda maszyna w danej chwili czasu może wykonywać tylko jedno zadanie. Zadanie Z_j scharakteryzowane jest przez następujące parametry:

- czas wykonywania $p_j > 0$,
- moment przybycia do systemu $r_j \geq 0$,
- wektor żądań zasobowych $R(Z_j) = [R_1(Z_j), R_2(Z_j), \ldots, R_s(Z_j)]$, gdzie $0 \leq R_l(Z_j) \leq m_l$, $l = 1,2,\ldots,s$, oznacza liczbę jednostek zasobu R_l potrzebną do wykonania zadania Z_j.

Do swego wykonania każde zadanie potrzebuje dowolnej maszyny oraz określonych przez wektor żądań zasobowych liczb jednostek poszczególnych zasobów. Zadania są niepodzielne, to znaczy od momentu rozpoczęcia do momentu zakończenia wykonywania zadania Z_j, $j = 1,2,\ldots,n$, upływa dokładnie p_j jednostek czasu. Rozpatrywane zbiory zadań zawierają zadania niezależne. Zadaniem aktywnym w chwili t nazywać będziemy zadanie, dla którego zachodzi $r_j \leq t$.

Zdefiniujemy teraz pojęcia uszeregowania oraz algorytmu szeregowania.

Uszeregowaniem będziemy nazywali takie przyporządkowanie w czasie maszyn i dodatkowych zasobów do zadań, dla którego spełnione są następujące warunki:

- w każdej chwili czasu każda maszyna wykonuje co najwyżej jedno zadanie a każde zadanie wykonywane jest przez co najwyżej jedną maszynę,
- wszystkie zadania zostaną wykonane,
- zadanie Z_j, $j = 1,2,\ldots,n$, jest wykonywane bez przerw w przedziale czasowym $[r_j, \infty]$,
- dla każdego $t \geq 0$ $\sum_{Z_j \in A(t)} R_l(Z_j) \leq m_l$, $l = 1,2,\ldots,s$, gdzie $A(t)$ jest zbiorem zadań wykonywanych w chwili t.

Uszeregowaniem optymalnym nazywać będziemy uszeregowanie minimalizujące długość uszeregowania $C_{max} = \max_j \{C_j\}$, gdzie C_j jest momentem zakończenia wykonywania zadania Z_j.

Algorytmem szeregowania dla problemu szeregowania (określonego przez zbiór parametrów) nazywać będziemy dowolną procedurę znajdującą uszerego-

wanie dla dowolnych danych (czyli ustalonych wartości wszystkich parametrów) tego problemu.

Optymalnym algorytmem szeregowania dla problemu szeregowania nazywać będziemy algorytm minimalizujący długość każdego skonstruowanego uszeregowania.

3. ALGORYTMY SZEREGOWANIA

Rozpatrzmy najpierw problem szeregowania na dwóch maszynach zadań o dowolnych momentach przybycia do systemu i jednostkowych czasach wykonywania. Oprócz maszyn w systemie znajduje się dodatkowy zasób jednego rodzaju, przy czym ograniczenia i żądania zasobowe są dowolne. Oznaczmy ten problem przez P1 i rozważmy następujący algorytm szeregowania.

Algorytm 1

1. Uporządkuj zadania na liście w kolejności nierosnących zadań zasobowych. Podstaw t:=0.

2. Przydziel do pierwszej maszyny pierwsze aktywne zadanie na liście. Do drugiej maszyny przydziel, jeśli to możliwe, kolejne aktywne zadanie na liście, tak by sumaryczne żądanie zasobowe obu zadań nie przekraczało wartości m_1.

3. Usuń z listy przydzielone zadania, podstaw t:=t+1 i jeśli znajdują się na liście zadania, to idź do punktu 2.

Optymalność tego algorytmu wynika z faktu, iż w każdej chwili czasu dąży on do maksymalizacji wykorzystania dodatkowego zasobu i maszyn, minimalizując w ten sposób czas wykonania zbioru zadań, czyli długość uszeregowania.

Złożoność algorytmu 1 zdominowana jest przez złożoność punktu 1. Uporządkowanie bowiem zbioru n elementów wymaga O(n log n) kroków, a łączna złożoność punktów 2 i 3 jest O(n). Złożoność całego algorytmu wynosi zatem O(n log n). Warto zauważyć, że w przypadku trzech maszyn problem P1 staje się NP - zupełny [6].

Rozpatrzmy teraz problem, w którym w stosunku do problemu P1 zachodzą następujące zmiany. Liczba maszyn jest dowolna, lecz żądania zasobowe zadań mogą dotyczyć co najwyżej jednostki dodatkowego zasobu. Oznaczmy ten problem przez P2.

W [3] wykazano, że poniższy algorytm jest optymalny dla problemu P2.

Algorytm 2

1. Podstaw t:=0, k:=0.

2. Przydziel w chwili t, do pierwszej wolnej maszyny aktywne i nie przydzielone jeszcze zadanie Z_j, dla którego $R_1(Z_j) = 1$, podstaw k:=k+1.

Powtarzaj ten punkt dopóty, dopóki albo $k = m_1$, albo nie będzie zadania o powyższych właściwościach.

3. Przydziel do pozostałych wolnych maszyn w momencie t aktywne i nie przydzielone jeszcze zadanie, dla których $R_1(Z_j) = 0$. Podstaw $t:=t+1$ i $k:=0$ i powtórz punkt 2, jeśli są jeszcze nie przydzielone zadania.

Złożoność powyższego algorytmu jest $O(n)$.

Kolejnym rozpatrywanym przez nas problemem jest problem P3, w którym założymy, że wszystkie $r_j = 0$, natomiast liczba rodzajów dodatkowych zasobów jest nie większa niż s, poszczególne zasoby dostępne są w liczbie co najwyżej p jednostek, a żądania zasobowe zadań są nie większe niż r, gdzie s, p oraz r są liczbami całkowitymi. Pozostałe parametry są takie same, jak w przypadku problemu P2. Opisane poniżej podejście 4 pozwala rozwiązać problem P3 w $O(n)$ krokach.

Zauważmy, że dla tak określonego problemu możemy każde zadanie Z_j przypisać do jednej z k klas, przy czym każda klasa jest charakteryzowana przez wektor żądań zasobowych $R(Z_j) \in \{0,1,\ldots,r\}^s$. Niech zależność pomiędzy tymi wektorami a klasami zadań będzie dana funkcją

$$f:\{0,1,\ldots,r\}^s \longrightarrow \{1,2,\ldots,k\},$$

gdzie k jest liczbą różnych wektorów żądań zasobowych, to znaczy $k=(r+1)^s$. Oznaczmy przez n_i liczbę zadań w klasie i, $i = 1,2,\ldots,k$. Zatem n_i jest liczbą zadań posiadających wektor żądań zasobowych równy $f^{-1}(i)$, $i = 1,2,\ldots,k$.

Korzystając z powyższych pojęć możemy teraz przedstawić dane problemu P3 w inny sposób. Zamiast zbioru zadań i ich wektorów żądań zasobowych wprowadzimy wektor $v = (v_1,v_2,\ldots,v_k) \in N_0^k$, gdzie v_i jest liczbą zadań posiadających wektor żądań zasobowych równy $f^{-1}(i)$, $i = 1,2,\ldots,k$.

Wprowadźmy teraz pojęcie elementarnego wektora $v \in N_0^k$ jako takiego, dla którego zachodzi $\sum_{j=1}^{n} R(Z_j) \leq (p,p,\ldots,p)$. Zauważmy, że elementarny wektor odpowiada danym problemu P3, dla których długość optymalnego uszeregowania jest równa 1.

Łatwo można sprawdzić, że liczba różnych elementarnych wektorów K zależy tylko od s, p oraz r. Przykładowe wartości przedstawiono poniżej.

s	p	r	K
2	1	1	4
3	1	1	14
4	1	1	48
...			
2	2	1	13
	...		
2	2	2	18

Oznaczmy teraz elementarne wektory (w dowolnej kolejności) przez $b_1, b_2, \ldots, b_k$. Zauważmy, że każdy wektor v można rozpatrywać jako sumę wektorów elementarnych, co wynika z faktu, iż każde uszeregowanie określa taką dekompozycję.

Ponieważ długość uszeregowania optymalnego odpowiadającego elementarnemu wektorowi jest równa 1, więc problem znalezienia uszeregowania optymalnego odpowiadającego nieelementarnemu wektorowi v sprowadza się do problemu znalezienia dekompozycji danego wektora v na nominalną liczbę wektorów elementarnych $b_1, b_2, \ldots, b_k$, to znaczy na liniową kombinację tych wektorów, dla której suma współczynników jest minimalna. Otrzymujemy zatem następujący problem:

znaleźć $\alpha_1, \alpha_2, \ldots, \alpha_k \in N_0$ takie, że

$$\sum_{i=1}^{k} \alpha_i b_i = v \qquad (1)$$

oraz

$$\sum_{i=1}^{K} \alpha_i \text{ jest minimalna.} \qquad (2)$$

Rozwiązując tak określony problem programowania liniowego całoliczbowego, znajdujemy żądaną dekompozycję wektora v, a co za tym idzie optymalne uszeregowanie.

Zbadajmy złożoność powyższego podejścia. Otóż, czas konstrukcji danych jest równy $2^s(K+\log n) = O(\log n)$. Aby określić złożoność algorytmu rozwiązania tego problemu wykorzystamy wynik podany w [9]. W pracy tej wykazano, że problem programowanie liniowego w liczbach całkowitych można dla ustalonych (to znaczy ograniczonych od góry przez stała) wartości liczb zmiennych i ograniczeń rozwiązać w czasie ograniczonym od góry przez wielomian zależny od liczby zmiennych i ograniczeń oraz od logarytmu z maksymalnej wartości wszystkich współczynników. Ponieważ w naszym przypadku liczba zmiennych i ograniczeń K jest ustalona dla ustalonych wartości s, p oraz r, a maksymalna wartość współczynnika w równaniu (1) wynosi n, więc otrzymujemy łączną złożoność podejścia $O(n)$.

4. WNIOSKI

W pracy przedstawiono kilka algorytmów o złożoności wielomianowej dla niektórych podproblemów ogólnego problemu szeregowania zadań jednostkowych, przy ograniczeniach zasobowych. Czynnikiem decydującym o możliwości konstrukcji algorytmów wielomianowych było wprowadzenie ograniczeń na zbiór

wartości, które mogą przyjmować następujące parametry: liczba maszyn, liczba rodzajów dodatkowych zasobów, liczby jednostek dodatkowych zasobów dostępne w systemie oraz żądanie zasobowe. Wydaje się, że podobne wyniki można uzyskać także dla innych kryteriów szeregowania, takich jak: średni czas przebywania zadania w systemie bądź maksymalne opóźnienie.

LITERATURA

[1] Aho, A.V., Hopcroft J.E., Ullman J.D.: The Design and Analysis of Computer Algorithms, Addison - Wesley, Reading, Mass. 1974.

[2] Błażewicz J.: Złożoność obliczeniowa algorytmów i problemów szeregowania zadań. Wyd. Politechniki Poznańskiej, Seria Rozprawy, No 104, 1979.

[3] Błażewicz J.: Complexity of computer scheduling algorithms under resource constraints, Roc. I Meeting AFCET-SMF on Applied Mathematics. Palaiseau (France), 1978, 169-178.

[4] Błażewicz J., Ecker K.: A polynomial in time algorithm for task scheduling under fixed resource constraints, Report, Instytut Automatyki, Politechnika Poznańska, 1981.

[5] Błażewicz J., Lenstra J.K., Rinnooy Kan A.H.G.: Scheduling subject to resource constraints: classification and complexity Report, Mathematisch Centrum, Amsterdam, 1980.

[6] Garey M.R., Johnson D.S.: Complexity results for multiprocessor scheduling under resource constraints, SIAM. J. on Computing 4, 1975, 397-411.

[7] Garey M.R., Johnson D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman, San Francisco, 1979.

[8] Karp R.M.: Reducibility among combinatorial problems, in R.E. Miller and J.W. Thatcher (eds.) Complexity of Computer Computation, Plenum Press, New York, 1972, 85-104.

[9] Lenstra H.W., Jr.: Integer programming with a fixed number of variables, Report, University of Amsterdam, 1981.

Recenzent: Doc. dr hab. inż. Andrzej GOŚCIŃSKI

Wpłynęło do Redakcji 15.05.1982 r.

ПОЛИНОМИАЛЬНЫЕ АЛГОРИТМЫ СОСТАВЛЕНИЯ РАСПИСАНИЯ ЗАДАЧ
НА ПАРАЛЛЕЛЬНЫХ МАШИНАХ С УЧЁТОМ ФИКСИРОВАННЫХ РЕСУРСНЫХ ОГРАНИЧЕНИЙ

Р е з ю м е

В работе рассмотрена проблема зависимости вычислительной сложности задачи составления расписания задач от фиксированного числа типов дополнительных ресурсов, а также от фиксированных ресурсных ограничений.

POLYNOMIAL - IN - TIME SCHEDULING ALGORITHMS FOR PARALLEL
MACHINES UNDER FIXED RESOURCE CONTRAINTS

S u m m a r y

In the paper the problem of task scheduling on parallel machines under resource constraints is considered. The impact of fixed resource limits and requirements on the computational complexity of the problem is shown. Algorithms polynomial in time are presented for some subproblems of the general problem.