ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1996
Seria: AUTOMATYKA z.117 Nr kol. 1337

Krzysztof GIARO, Marek KUBALE, Michał MAŁAFIEJSKI
Politechnika Gdańska

SZEREGOWANIE ZADAŃ JEDNOSTKOWYCH W SYSTEMIE OTWARTYM BEZ OBUSTRONNYCH PRZESTOJÓW

Streszczenie. W pracy rozważamy problem szeregowania zadań jednostkowych w systemie otwartym bez obustronnych przestojów w celu minimalizacji długości uszeregowania, symbolicznie $O|p_{ij}\in\{0,1\},no\text{-}wait|idle|C_{max}$. Pokazujemy, że problem ten sprowadza się do problemu zwartego kolorowania krawędzi grafu dwudzielnego, który jest NP-trudny. Dlatego koncentrujemy się na identyfikacji tych przypadków szczególnych, które mogą być rozwiązane w czasie wielomianowym.

NO-WAIT OPEN SHOP SCHEDULING OF ZERO-ONE TIME OPERATIONS WITHOUT INSERTED IDLE TIMES

Summary. In the paper we consider the problem of no-wait scheduling of zero-one execution time tasks in an open shop without inserted idle times, in symbols $O|p_{ij}\in\{0,1\},no\text{-}wait|idle|C_{max}$. We show that this problem is equivalent to the problem of consecutive coloring of the edges of a bipartite graph, which is NP-hard. Therefore, we concentrate on finding special cases which can be solved in polynomial time.

1. Wprowadzenie

W pracy rozważamy następujący problem szeregowania zadań: Danych jest m specjalizowanych maszyn i n zadań. Każde zadanie składa się z m rodzajów operacji, zaś każda operacja musi odbywać się na dokładnie jednej przeznaczonej dla niej maszynie. Czas działania pojedynczej operacji jest zerowy lub jednostkowy, zaś kolejność realizacji operacji jest dowolna (system otwarty). Zakładamy ponadto, że zarówno operacje tego samego zadania wykonują się bez przestojów, jak i maszyny pracują w sposób ciągły. Celem naszym jest znalezienie uszeregowania o minimalnej długości. Zatem problem może być symbolicznie zapisany jako $O|p_{ij}\in\{0,1\},no\text{-}wait|idle|C_{max}$.

Dobrą ilustracją takiego problemu jest układanie rozkładów zajęć szkolnych. W tym przykładzie klasy odpowiadają zadaniom, nauczyciele - procesorom, zaś lekcje - operacjom.

Zadanie polega na ułożeniu takiego harmonogramu zajęć, że żadna klasa i żaden nauczyciel nie ma okienek, zaś ostatnia lekcja odbywa się tak wcześnie jak to możliwe. Zauważmy, że ze względu na wymaganie bezokienkowości rozkładu zajęć długość harmonogramu może być większa niż bez tego ograniczenia, a nawet rozwiązanie dopuszczalne może nie istnieć.

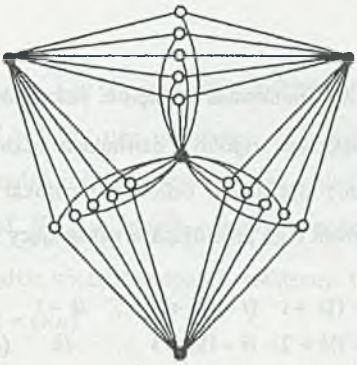
Rozważany problem może być zamodelowany w postaci grafu dwudzielnego, w którym wierzchołki odpowiadają zadaniom i maszynom, zaś krawędzie - operacjom jednostkowym. Wówczas rozwiązanie dopuszczalne odpowiada zwartemu pokolorowaniu krawędzi tego grafu. Ponieważ problem zwartego kolorowania krawędzi grafu dwudzielnego jest NP-trudny, w pracy skoncentrujemy się na identyfikacji tych przypadków szczególnych, które mogą być rozwiązane w czasie wielomianowym względem m i n. Chodzi tutaj o przypadki uproszczonej struktury grafu dwudzielnego dopuszczającej takie rozwiązania, np. pełne grafy dwudzielne, drzewa, grafy regularne, cukierki, kaktusy, grafy podkubiczne itp.

2. Model matematyczny

W poprzednim punkcie wspomnieliśmy, że rozważany problem może być przedstawiony w postaci grafu dwudzielnego $G = (V_1, V_2, E)$, gdzie V_1 jest zbiorem n wierzchołków odpowiadających zadaniom, V_2 jest zbiorem m wierzchołków odpowiadających maszynom, zaś E jest zbiorem krawędzi pomiędzy elementami zbioru V_1 i elementami zbioru V_2 odpowiadającymi operacjom jednostkowym. W dalszym ciągu graf G nazywać będziemy *grafem szeregowania*. Zauważmy, że problem szeregowania bez obustronnych przestojów polega na kolorowaniu krawędzi grafu G w taki sposób, iż każdy wierzchołek „widzi" zwarty przedział kolorów, zaś liczba użytych kolorów jest minimalna możliwa. Poniżej podamy kilka istotnych faktów zaczerpniętych z teorii kolorowania grafów.

Po pierwsze, nie wszystkie grafy dwudzielne mają takie pokolorowania. Najmniejszy, w sensie liczby wierzchołków, znany graf dwudzielny, który nie może być pokolorowany zwarcie, to tzw. rozetka Małafiejskiego [3]. Na rysunku 1 pokazujemy ją jako graf szeregowania z $m = 4$ maszynami (wierzchołki zaczernione) i $n = 15$ zadaniami.

Po drugie, problem rozstrzygnięcia, czy dany graf dwudzielny posiada takie pokolorowanie, jest silnie NP-zupełny [7].



Rys. 1. Rozetka M_5
Fig. 1. Rossete M_5

Po trzecie, w odróżnieniu od klasycznego kolorowania krawędzi minimalna *rozpiętość* pokolorowania zwartego, czyli liczba użytych kolorów, nie jest uzależniona od maksymalnego stopnia grafu Δ. W rzeczywistości autorzy pokazali w [4], że istnieje ciąg grafów dwudzielnych stopnia $\Delta = 32$, dla których liczba kolorów niezbędnych do pokolorowania zwartego rośnie do nieskończoności. Jednakże liczba ta nie przekracza nigdy $diam(G)(\Delta-1)+1$, gdzie $diam(G)$ jest średnicą grafu G [1].

Po czwarte, pesymistyczna *deficytowość* grafów nie mających zwartego pokolorowania, czyli minimalna liczba krawędzi wiszących, których dołączenie do G czyni go zwarcie kolorowalnym, rośnie przynajmniej tak szybko jak liczba wierzchołków. W [3] autorzy pokazali ciąg grafów dwudzielnych, dla których deficytowość zbliża się asymptotycznie do liczby wierzchołków.

Po piąte, problem decyzyjny $Om|p_{ij}\in\{0,1\},no\text{-}wait|idle|$- ma zawsze rozwiązanie pozytywne, gdy $m \leq 3$. Przypadek dwumaszynowy rozważamy w punkcie 3.2. Brak takiego rozwiązania dla $m = 4$ ilustruje rysunek 1.

3. Przypadki wielomianowo rozwiązywalne

W niniejszym punkcie dokonamy przeglądu tych przypadków szczególnych, dla których znalezienie pokolorowania zwartego o minimalnej rozpiętości nie przedstawia większych trudności obliczeniowych.

3.1. Pełne grafy dwudzielne

W przypadku grafów $K_{n,m}$ minimalna rozpiętość pokolorowania wynosi $C^*_{max} = m+n-k$, gdzie $k = gcd(m,n)$ jest największym wspólnym dzielnikiem liczb m i n [6]. Zauważmy, że w tym przypadku pokolorowanie optymalne definiuje prostokąt łaciński L rozmiaru $m \times n$. Zdefiniujmy blok B_i jako fragment tego prostokąta w następujący sposób:

$$B_i = \begin{bmatrix} (i-1)k+1 & (i-1)k+2 & \dots & ik-1 & ik \\ (i-1)k+2 & (i-1)k+3 & \dots & ik & (i-1)k+1 \\ \dots & \dots & \dots & \dots & \dots \\ ik-1 & ik & \dots & ik-3 & ik-2 \\ ik & (i-1)k+1 & \dots & ik-2 & ik-1 \end{bmatrix}$$

Zatem prostokąt łaciński ma postać

$$L = \begin{bmatrix} B_1 & B_2 & \dots & B_{\frac{n}{k}} \\ B_2 & B_3 & \dots & B_{\frac{n}{k}+1} \\ \dots & \dots & \dots & \dots \\ B_{\frac{n}{k}} & B_{\frac{n}{k}+1} & \dots & B_{\frac{n+m}{k}-1} \end{bmatrix}$$

Wobec tego, że kolor maksymalny $c_{max}(B_i) = ik$, natychmiast dostajemy

$$c_{max}(K_{n,m}) = c_{max}(B_{(n+m)/k-1}) = m+n-k.$$

Na przykład dla grafu $K_{6,4}$ pojedynczy blok B_i ma postać

$$B_i = \begin{bmatrix} 2i-1 & 2i \\ 2i & 2i-1 \end{bmatrix}$$

natomiast cała tablica

$$L = \begin{bmatrix} B_1 & B_2 & B_3 \\ B_2 & B_3 & B_4 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 1 & 4 & 3 & 6 & 5 \\ 3 & 4 & 5 & 6 & 7 & 8 \\ 4 & 3 & 6 & 5 & 8 & 7 \end{bmatrix}$$

Oczywiście, złożoność tej metody wynosi $O(mn)$.

3.2. Cukierki

Pojęcie cukierka wprowadzili de Werra i Solot [8]. Z takim grafem szeregowania mamy do czynienia, gdy $m = 2$. W tym przypadku zbiór zadań można podzielić na dwa podzbiory, mianowicie: n_1 zadań jednooperacyjnych i n_2 zadań dwuoperacyjnych. Wówczas kolorujemy najpierw podgraf $K_{n_2,2}$ w sposób opisany w poprzednim punkcie, a następnie kolorujemy pozostałe krawędzie wiszące w sposób zachłanny. Oczywiście złożoność takiego algorytmu wynosi $O(2n_2+n_1) = O(n)$.

3.3. Grafy regularne

Indeks chromatyczny grafu dwudzielnego Δ-regularnego wynosi $\chi'(G) = \Delta$. Każde Δ-kolorowanie krawędzi jest, rzecz jasna, pokolorowaniem zwartym. Najlepszy znany algorytm kolorowania krawędzi takiego grafu $G = (V_1,V_2,E)$ ma złożoność $O(|E|\log(|V_1|+|V_1|)) = O(mn\log(m+n))$ [2].

3.4. Drzewa

Jeżeli G jest drzewem o maksymalnym stopniu Δ, to G można zwarcie pokolorować Δ kolorami tak, aby dowolna wybrana krawędź uzyskała ustaloną barwę $k \leq \Delta$. Rzeczywiście, niech e będzie krawędzią, która ma otrzymać kolor k. Po pokolorowaniu e wybieramy dowolną krawędź $\bar{e}$ łączącą się z już pokolorowaną krawędzią w wierzchołku v. W ogólności, kolory już użyte przy v tworzą przedział $[a,..., b]$, dlatego kolorujemy $\bar{e}$ przez a-1 lub b+1 w zależności od tego, która z tych liczb należy do $\{1,...,\Delta\}$. Powtarzając tę czynność otrzymujemy żądane pokolorowanie w czasie $O(m+n)$.

Poprawność algorytmu wynika z faktu, że za każdym razem, gdy kolorujemy nową krawędź, krawędzie już pomalowane tworzą zwarcie pokolorowane i spójne poddrzewo w G. Zatem $\bar{e}$ łączy się z krawędzią już pokolorowaną tylko w jednym wierzchołku v.

3.5. Kaktusy

Naturalnym uogólnieniem drzew są kaktusy. Formalnie, *kaktus* jest grafem spójnym, w którym każdy blok (czyli maksymalny podgraf bez przegubu) jest albo krawędzią, albo bezcięciwowym cyklem prostym. Niech będzie dany kaktus C. Usuńmy krok po kroku wszystkie wierzchołki stopnia 1 do momentu, gdy nie będzie już takich wierzchołków. Graf, który pozostanie, nazywamy *rdzeniem* kaktusa. Kaktus C jest *parzysty*, jeśli każdy cykl jego rdzenia jest parzysty. C jest *silnie spójny*, gdy rdzeń kaktusa jest silnie spójny, czyli nie zawiera mostów. Poniżej pokażemy, że rdzeń silnie spójnego kaktusa parzystego może być zwarcie pokolorowany Δ kolorami, gdzie Δ jest parzystym stopniem takiego kaktusa.

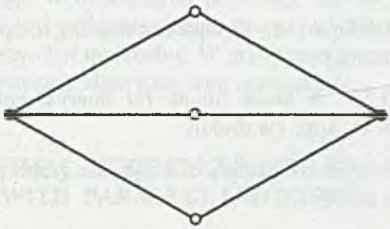
Niech R będzie rdzeniem silnie spójnego kaktusa parzystego. Wtedy R można zwarcie pokolorować kolorami $1,\ldots,\Delta$ tak, by w dowolnym cyklu krawędzie były pomalowane naprzemiennie przez parę sąsiednich kolorów, z których niższy jest nieparzysty. Para kolorów dla dowolnego cyklu może być z góry ustalona. Sposób kolorowania jest analogiczny do sposobu kolorowania drzew. Najpierw malujemy krawędzie wybranego cyklu naprzemiennie wybraną parą kolorów. Następnie wybieramy krawędź bezbarwną $\bar{e}$ łączącą się z już pokolorowaną krawędzią w wierzchołku v i kolorujemy cykl zawierający $\bar{e}$ w następujący sposób. Niech kolory użyte przy v tworzą przedział $[a,\ldots, b]$. Wówczas kolorujemy krawędzie cyklu naprzemiennie za pomocą a-2, a-1 lub b+2, b+1 w zależności od tego, która para zawiera się w $\{1,\ldots, \Delta\}$ i tak dalej dla pozostałych cykli. Łatwo zauważyć, że pokolorowania uzyskane tą drogą są zwarte, zaś liczba kroków algorytmu jest proporcjonalna do liczby krawędzi.

Podany sposób można łatwo uogólnić na wszystkie silnie spójne kaktusy parzyste. Mianowicie, najpierw kolorujemy rdzeń R, a następnie kolorujemy drzewa zakorzenione w jego cyklach poczynając od korzenia w sposób opisany w punkcie 3.4. Złożoność obliczeniowa tego algorytmu jest proporcjonalna do łącznej liczby krawędzi. Ponieważ w przypadku silnie spójnych kaktusów parzystych $G = (V_1,V_2,E)$ mamy $|E| < 4(|V_1|+|V_2|)/3$, więc algorytm ten daje uszeregowanie długości $C^*_{\max} = \Delta$ w czasie $O(m+n)$.

Na marginesie tych rozważań zauważmy, że dowolny kaktus parzysty może być zwarcie pomalowany co najwyżej Δ+1 kolorami w wielomianowym czasie.

3.6. Grafy podkubiczne

Graf nazywamy *kubicznym*, gdy jest regularny stopnia 3. Graf, którego maksymalny stopień $\Delta \leq 3$, nazwiemy *podkubicznym*. Oczywiście, każdy dwudzielny graf kubiczny, jako regularny, ma 3-pokolorowanie zwarte. Można udowodnić, że wszystkie dwudzielne grafy podkubiczne mają zwarte pokolorowania, aczkolwiek niekiedy liczba niezbędnych kolorów może wzrosnąć do 4. Najmniejszym takim grafem nie posiadającym zwartego 3-pokolorowania jest $K_{3,2}$ pokazany na rysunku 2.



Rys. 2. Najmniejszy podkubiczny graf dwudzielny bez zwartego 3-pokolorowania
Fig. 2. The smallest subcubic bipartite graph having no consecutive 3-coloring

Zatem rodzinę wszystkich dwudzielnych grafów podkubicznych można podzielić na dwie klasy: tych, które mają zwarte 3-pokolorowania, i tych, które mają wyłącznie zwarte 4-pokolorowania, przy czym istnieje wielomianowe kryterium pozwalające stwierdzić, z którego rodzaju grafem mamy do czynienia. Przykładami grafów pierwszej klasy są grafy hamiltonowskie (np. drabiny i wiatraki) i tzw. „nieparzyste usta" oraz wszystkie grafy, których rdzeń jest jednym z wyżej wymienionych (np. jednocykliczne). Zwarte kolorowanie takich grafów nie przedstawia większych problemów obliczeniowych. Na przykład, w przypadku grafów hamiltonowskich algorytm rozpoczynamy od znalezienia maksymalnego skojarzenia spinającego wierzchołki stopnia 3, które malujemy kolorem 1, a następnie malujemy pozostałe cykle parzyste kolorami 2 i 3. Jeśli do takiego grafu podoczepiane są drzewa binarne, to malujemy je nadając krawędzi przy korzeniu kolor 1 i tak dalej w sposób opisany w punkcie 3.4. W przypadku nieparzystych ust najpierw kolorujemy jeden z cykli kolorami 1 i 2, a następnie ścieżkę cięciwową kolorami 3, 2, 3 itd. Jeśli w ustach zakorzenione są drzewa binarne, to malujemy je zwarcie poczynając od korzeni.

LITERATURA

1. Asratian A.S., Kamalian R.R.: Investigation on interval edge-colorings of graphs, J. Combin. Theory, Ser. B., vol. 62, 1994, pp. 34-43.
2. Cole R., Hopcroft J.E.: On edge coloring bipartite graphs, SIAM J. Comput., vol. 11, 1982, pp. 540-546.
3. Giaro K., Kubale M., Małafiejski M.: The deficiency of bipartite graphs (wysłane na konferencję 22nd Int. Workshop on Graph-Theoretic Concepts in Computer Science).
4. Giaro K., Kubale M., Małafiejski M.: Consecutive coloring the edges of a general graph, Tech. Rep. 28/95, Wydział ETI, Politechnika Gdańska, 1995.
5. Giaro K., Kubale M., Małafiejski M.: Compact scheduling in open shop with zero-one time operations (w przygotowaniu).
6. Hanson D., Loten C.O.M.: A lower bound for interval colouring bi-regular bipartite graphs, Bull. Inst. Comb. & Appl. (w druku).
7. Sevastianow S.V.: On interval colorability of a bipartite graph (po rosyjsku), Mat. Diskret. Anal., vol. 50, 1990, pp. 61-72.
8. Werra de D., Solot P.: Some graph-theoretical models for scheduling in automated production systems, Networks, vol. 23, 1993, pp. 651-660.

Recenzent: Prof.dr hab.inż. Jan Węglarz

Wpłynęło do Redakcji do 30.06.1996r.

Abstract

In the paper we consider the following scheduling problem: Given m specialized machines and n jobs. Each job consists of m operations and each operation must be performed by precisely one dedicated machine. Execution time of each operation is either zero or one and the order in which operations of a job are performed is immaterial (open shop). Moreover, we assume that operations of a job are performed in a no-wait manner and machines work without inserted idle times. Our aim is to find a schedule with minimum possible length. Thus the problem can be described in symbols as $O|p_{ij}\in\{0,1\},no\text{-}wait|idle|C_{max}$. Obviously, not always such a solution is possible.

The problem considered herein can be modeled as a bipartite graph in which the vertices correspond to both jobs and machines while the edges correspond to unit execution time operations. Then a solution to the scheduling problem corresponds to a consecutive coloring of the edges of the associated bipartite graph. Since deciding the latter is NP-complete, in the paper we focus on identifying special cases which can be solved in polynomial time with respect to m and n. This concerns highly structured graphs such as: complete bipartite graphs, trees, regular graphs, cacti, candies, etc.