ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1994
Seria: AUTOMATYKA z.114 Nr kol. 1250

Ewa SKUBALSKA - RAFAJŁOWICZ
Politechnika Wrocławska

SZYBKI ALGORYTM SELEKCJI ZADAŃ

Streszczenie:W pracy pokażemy algorytm rozwiązywania problemu wyboru podzbioru zadań przeznaczonych do realizacji na jednej maszynie. Kryterium wyboru jest zysk związany z realizacją poszczególnych zadań. Zysk ten pomniejszany jest o kary związane z czasem oczekiwania zadania na wykonanie. Sformułowano model problemu w postaci specyficznego zadania programowania dynamicznego. W pracy pokażemy, że istnieje dokładny wielomianowy algorytm rozwiązania o złożoności $O(n^2)$.

FAST ALGORITHM FOR TASKS SELECTION

Summary: We show in the paper an algorithm for solving a problem of the choice of a subset of tasks waiting to be performed on a single machine. The objective is to maximize the total profit which consists of prizes we get for every completed task. The profit is decreased by penalties payed for waiting times of the chosen tasks. The model of the problem is stated as a specific dynamic programming problem. We present exact polynomial-time algorithm which solves the problem in $O(n^2)$.

SCHNELLER ALGORITHMUS FÜR WERKAUSWAHLPROBLEM

Zusammenfassung: In der Arbeit wird ein Algorithmus für Werkauswahlproblem auf einer Machine formuliert, bei denen nicht alle Werkstücke realisiert sollen. Die Auswahlkriterium ist Realisierungsprofit, der für jede ausgewähltes Werk erreicht wird und Reihenfolgerealisierungskosten die proportional mit Wartezeit für jedes Werk betragen werden. Maximisierung von Nettoprofit ist die Zielkriterium für aufstehende Optimierungsausgabe. Mathematisches Modell für dieses Problem mit der Anwendung der dynamischen Programmierung wird gegeben. Ein exakter, polynomialer Algorithmus mit Komplexität $O(n^2)$ wird presentiert.

1. Wstęp

W pracy rozważany jest następujący problem: n zadań oczekuje w kolejce do realizacji na pojedynczej maszynie. Nie wszystkie zadania muszą zostać zrealizowane. Zadanie j-te wymaga do zrealizowania użycia maszyny w czasie p_j. Za każde wykonane zadanie otrzymamy zysk w ustalonej wielkości z_j. Zysk ten zostanie pomniejszony o karę związaną z czasem pobytu zadania w systemie. Zatem najwięcej zyskujemy na realizacji zadań o krótkich czasach wykonania. Nie opłaca się natomiast wykonywanie zadań o długim czasie realizacji, szczególnie w sytuacji, gdy mamy do wykonania inne zadania o lepszym stosunku zysku z_j do czasu ich wykonania p_j. Chcemy zmaksymalizować zysk netto, który składa się z opłat (otrzymanych za wykonanie zadań) pomniejszonych o kary płacone proporcjonalnie do terminów zakończenia realizacji poszczególnych zadań. Naszym celem jest dokonanie wyboru podzbioru zadań przeznaczonych do wykonania oraz ustalenie kolejności ich realizacji. Po dokonaniu wyboru suma opłat otrzymanych za wykonanie zadań zostaje ustalona i wartość kryterium zależy jedynie od kolejności obsługiwania zadań.

Ze względu na prostotę sformułowania postawiony wyżej problem może znaleźć zastosowanie jako zagadnienie częściowe wchodzące w skład bardziej złożonych zadań szeregowania zarówno w przypadku konstrukcji algorytów dokładnych, jak i prostych obliczeniowo heurez.

Problem kolejnościowy z ustalonym z góry zbiorem zadań, które mają być wykonywane na jednej maszynie, tak by minimalizować sumaryczny ważony czas pobytu zadań w systemie (czas zakończenia realizacji poszczególnych zadań), został rozwiązany w roku 1956 przez Smitha [5]. Jak wiadomo, optymalne uszeregowanie otrzymujemy w tym przypadku poprzez uporządkowanie zadań zgodnie z nierosnącymi wartościami $\frac{w_j}{p_j}$.

W przypadku gdy wszystkie wagi w_j są takie same (powiedzmy równe 1), zadania należy wykonywać zgodnie z niemalejącym porządkiem czasów realizacji p_j. W konsekwencji istotnym problemem jest problem wyboru podzbioru zadań, które mają zostać zrealizowane, a nie problem ich uszeregowania.

Omawiany problem wyboru jest szczególnym przypadkiem szerszej klasy zadań zebranych i sformułowanych w pracy [3], przypadkiem, dla którego udało się uzyskać efektywne algorytmy rozwiązania. Przedstawiony problem wyboru zadań ma pewne formalne związki z modelami szeregowania zadań ze zmiennymi czasami realizacji [1], [6]. Można pokazać, że pewną (dość wąską zresztą) klasę problemów tego typu można rozwiązywać w bardziej efektywny sposób (to znaczy za pomocą algorytmu o złożoności $O(n^2)$, a nie jako ogólne zagadnienia przydziału).

W pracy sformułujemy algorytm ($O(n^2)$) rozwiązania problemu wyboru zadań przy wagach jednostkowych ($w_j = 1$) oparty na rozwiązaniu pewnego szczególnego zagadnienia

programowania dynamicznego. Pokażemy również, że w przypadku gdy zyski z_j są proporcjonalne do czasów wykonywania zadań, to znaczy:

$$z_j = v \cdot p_j; \quad j = 1, \ldots, n,$$

algorytm rozwiązania problemu daje się znacznie uprościć i ma złożoność $O(nlogn)$, przy czym większość czasu działania algorytmu pochłania porządkowanie zadań, natomiast sam problem wyboru wymaga wykonania co najwyżej $O(n)$ operacji.

Prostota proponowanego tu algorytmu jest na tyle duża, że mógłby on znaleźć zastosowanie do rozdziału zadań między procesory w sieciach o rozproszonej i niejednorodnej mocy obliczeniowej. Serwer sieci lokalnej dokonywałby selekcji zadań, których realizacji się sam podejmie, kierując się umownymi zyskami i czasami realizacji zależnymi od jego mocy obliczeniowej. Pozostałe zadania odsyłane byłyby do realizacji przez węzeł o większej mocy obliczeniowej. Dokładniejszy opis algorytmu przystosowany do przedstawionej wyżej sytuacji zamieszczony zostanie w innej pracy.

2. Sformułowanie problemu

Niech $J = \{1, 2, \ldots, n\}$ oznacza zbiór zadań do wykonania. Określone są czasy realizacji zadań $p_j \geq 0$, $j \in J$ oraz zyski związane z wykonaniem każdego zadania $z_j > 0$, $j \in J$. Zadania mogą być wykonywane w dowolnej kolejności, przy czym w każdym momencie czasowym wykonywane może być tylko jedno zadanie. Niech $S \subset J$ oznacza zbiór zadań, które zostaną zrealizowane. Wyznaczenie zbioru S jest podstawowym problemem decyzyjnym, który należy rozwiązać. W celu uproszczenia zapisu przyjmiemy, że zadania zostały ponumerowane zgodnie z niemalejącymi czasami wykonania, to znaczy $p_j \geq p_{j+1}$. Gdyby wszystkie zadania miały zostać zrealizowane, to zgodnie z regułą Smitha zadanie o numerze n powinno być realizowane jako pierwsze, a zadanie o numerze jeden jako ostatnie (pierwsze od końca).

Niech x_{ij} $1 \leq i, j \leq n$ będzie 0-1 zmienną decyzyjną o następującej interpretacji:

$$x_{ij} = \begin{cases} 1, & gdy\ j-te \text{ zadanie wybrano do wykonania jako } i-te \\ & \text{(licząc pozycje od końca)}, \ (j \in S), \\ 0, & w\ przeciwnym\ przypadku. \end{cases}$$

Model matematyczny problemu możemy wtedy sformułować w następujący sposób:

$$Y = \sum_{j=1}^{n} \left(\sum_{i=1}^{n} z_j \cdot x_{ij} - \sum_{i=1}^{n} i \cdot p_j \cdot x_{ij} \right) \rightarrow max \qquad (1)$$

przy ograniczeniach:

$$\sum_{i=1}^{n} x_{ij} \leq 1, \quad j = 1, \ldots, n, \tag{2}$$

$$\sum_{j=1}^{n} x_{ij} \leq 1, \quad i = 1, \ldots, n, \tag{3}$$

$$x_{ij} = 1 \Longrightarrow x_{ml} = 0, \quad m > i \quad l < j \quad i, j = 1, \ldots, n. \tag{4}$$

$$x_{ij} \in \{0, 1\}, \quad i, j = 1, \ldots, n, \tag{5}$$

Ograniczenie (4) oznacza, że jeśli zadanie j-te jest wykonywane jako i-te (od końca), to żadne z zadań o dłuższym czasie wykonywania ($l < j$) nie może go poprzedzać, czyli nie może zostać ustalone na pozycji $i + 1$, $i + 2, \ldots$. Zadanie optymalizacji (1)-(5) można rozbić na n zadań cząstkowych, w których liczba wybranych zadań jest z góry ustalona. W każdym z tych zadań zakładamy, że zrealizowano dokładnie k zadań ($k = 1, 2, \ldots, n$). Optymalna wartość funkcji celu jest wtedy równa:

$$Y = max_k Y_k \tag{6}$$

gdzie Y_k jest wartością funkcji celu dla problemu (1)-(5) z dodatkowym ograniczeniem:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} = k. \tag{7}$$

Prowadzi to do następującego schematu obliczeniowego bazującego na idei programowania dynamicznego.

Algorytm

$$Y_{i,j} = \max \{Y_{i,j-1}, Y_{i-1,j-1} + z_j - i \cdot p_j\}, \quad 0 \leq i \leq j \leq n,$$

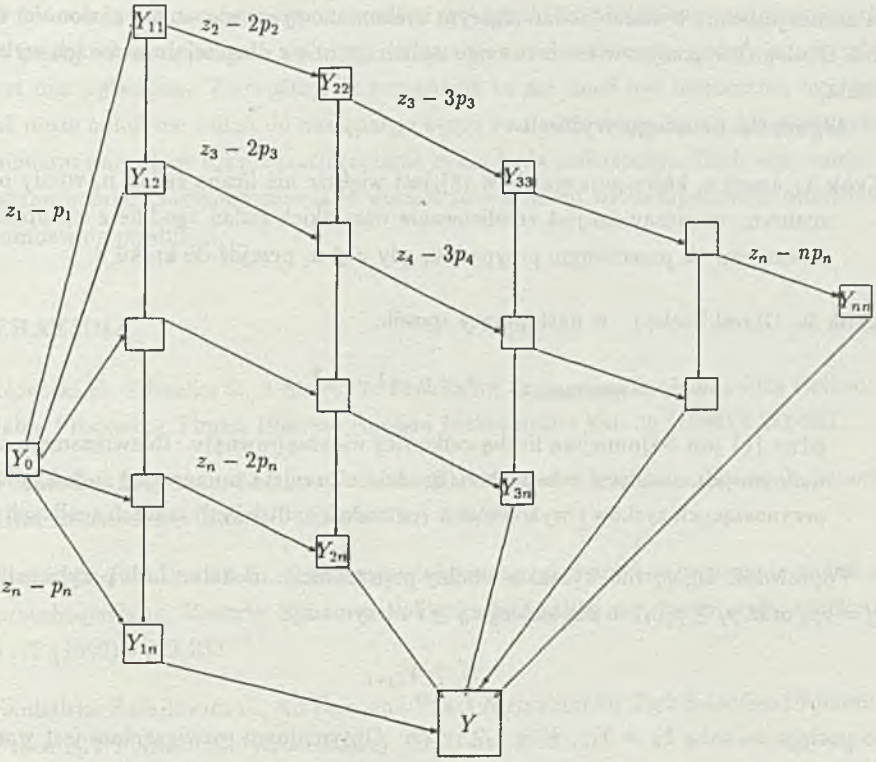
$$Y_{0,0} = 0, \quad Y_{ij} = 0, \quad n \geq i > j \geq 1$$

$$Y_k = Y_{k,n}, \quad k = 1, 2, \ldots, n.$$

$$Y = \max_{1 \leq k \leq n} Y_k$$

$Y_{1,j}$ oznacza zysk (netto) związany z umieszczeniem j-tego zadania na pierwszej pozycji (licząc od końca). $Y_{i,j}$ $i = 2, 3, \ldots, n$ oznacza maksymalny zysk (netto), jaki można uzyskać umieszczając na i-tej pozycji (licząc od końca) zadanie o numerze równym i, $i + 1, \ldots, j$. Zadania o numerach mniejszych niż i pomijamy, gdyż muszą one zajmować wcześniejsze (licząc od końca) pozycje.

Rozwiązanie zadania jest równoważne ze znalezieniem najdłuższej drogi w sieci, której strukturę przedstawiono schematycznie na rysunku 1. Wymaga to wykonania $O(n^2)$ operacji arytmetycznych.



Rys.1. Struktura obliczeniowa problemu selekcji zadań
Fig. 1. Computational structure of tasks selection problem

3. Zadanie z zyskami proporcjonalnymi do czasów realizacji

W przypadku gdy dla każdego $j \in J$

$$z_j = v \cdot p_j, \quad v > 0, \tag{8}$$

gdzie v jest pewną znaną stałą, problem selekcji zadań znacznie się upraszcza. Przy takim założeniu jesteśmy w stanie podać algorym wielomianowy rozwiązania o złożoności $O(n)$ (lub $O(nlogn)$ włączając w to sortowanie zadań zgodnie z długością czasów ich wykonywania).

Algorytm prostego wyboru

Krok 1. Jeżeli v, które pojawia się w (8),jest większe niż liczba zadań n, wtedy optymalnym rozwiązaniem jest zrealizowanie wszystkich zadań zgodnie z ich uporządkowaniem. W przeciwnym przypadku, gdy $v \leq n$, przejdź do kroku 2.

Krok 2. Określ liczbę r w następujący sposób:

$$r = n + 1 - \lceil v \rceil$$

gdzie $\lceil v \rceil$ jest najmniejszą liczbą całkowitą większą(równą)v. Rozwiązaniem optymalnym jest usunięcie ostatnich r (zgodnie z przyjętą numeracją) zadań jako nie przynoszących zysków i wykonanie $n - r$ zadań o dłuższych czasach realizacji.

Poprawność algorytmu wynika z analizy poprzedniego modelu. Jeśli przyjmiemy, że $z_j = vp_j$ oraz $p_j \geq p_{j+1}$, to dla każdego $j \geq i$ otrzymamy:

$$Y_{ij} \geq Y_{i,j+1}$$

co pociąga za sobą $Y_k = Y_{kk}$, $k = 1, 2, \ldots, n$. Optymalnym rozwiązaniem jest wartość $Y = Y_m$, takie że:

$$z_m - mp_m = (v - m)p_m > 0,$$

$$z_{m+1} - (m+1)p_{m+1} = (v - m - 1)p_{m+1} \leq 0$$

Inny dowód poprawności algorytmu przedstawiono w pracy [4].

4. Podsumowanie

Prostota proponowanego tu algorytmu daje szansę na stworzenie szybkiej metody selekcji nawet dużej liczby zadań, szczególnie jeśli zgodzimy się na zastosowanie przybliżonego losowego algorytmu sortowania [2].

Uogólnienie proponowanego modelu selekcji zadań do przypadku z kosztami uwzględniającymi ważone czasy oczekiwania zadań ($w_i \neq 1$) wydaje się łatwe i powinno prowadzić do dokladnego algorytmu o złożoności $O(n^3)$.

W tym miejscu warto zwrócić uwagę na ogólniejszy aspekt sformułowanego w pracy zadania, a mianowicie na negocjacyjny charakter problemu. W odróżnieniu od klasycznych zadań szeregowania, dopuszcza się tutaj rezygnację z realizacji części zadań, wtedy gdy nie jest ona opłacalna. Zauważmy, że rezygnacja ta nie musi być ostateczna, lecz oznaczać może odłożenie zadań do następnego etapu rozliczeniowego oraz może się wiązać ze zmianami warunków ich realizacji (zmiana zysku bądź realizatora). Takie spojrzenie na rozważane w pracy zadanie pozwala je widzieć jako element wielostopniowego problemu programowania produkcji.

LITERATURA

[1] Nowicki E., Zdrzałka S., A Survey of Results for Sequencing Problems with Controllable Processing Times, Discrete Applied Mathematics Vol. 26 (1990) s.271-287.

[2] Prabhakar Raghavan, Lecture Notes on Randomized Algorithms, Research Report IBM Research Division RC 15340 (1990) 1/9/90.

[3] Skubalska-Rafajłowicz E., Problemy selektywnego wyboru i szeregowania zadań z przezbrojeniami, Zeszyty Naukowe Politechniki Śląskiej ser.Automatyka z.109 nr 1175 (1992) s.223-232.

[4] Skubalska-Rafajlowicz E., An Easy and Exact Algorithm for Task Selection Problem, Prace ICT Politechniki Wrocławskiej (1992) nr.50/92

[5] Smith W.E., Various optimizers for single-state production, Naval Res. Logist. Quart. 3 (1956), pp 56-66.

[6] Vickson R.G., Two single machine sequencing problems involving controllable job processing times, AIIE Trans. 12 (1980), pp 258-262.

Recenzent: Prof. dr hab. inż. Tadeusz Sawik

Wpłynęło do Redakcji do 30.04.1994 r.

Abstract

In the paper the following situation is considered: n tasks wait to be processed on a single machine. Not all tasks must be perfomed. Task i-th ($i = 1, \ldots, n$) requires a positive processing time p_i. We get a prize z_i for every performed task and a penalty is paid for waiting time of the tasks. We want to maximize the net profit, which consists of the sum of prizes collected by processing tasks minus the sum of penalties payed proportionally to waiting times of the tasks. Our aim is to choose a subset of tasks to be performed and find a sequence of these tasks which maximize the net profit. The model of the problem is stated as a specific dynamic programming problem. We present exact polynomial-time algorithm which solves the problem in $O(n^2)$. We show also that there exists the polynomial time (O(nlogn)) algorithm which solves exactly the above problem under the assumption that the collected profits are proportional to the processing times of the performed tasks.