ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 2000
Seria: AUTOMATYKA z. 129 Nr kol. 1474

Joanna JÓZEFOWSKA, Marek MIKA, Rafał RÓŻYCKI,
Grzegorz WALIGÓRA, Jan WĘGLARZ
Politechnika Poznańska

ROZWIĄZYWANIE DYSKRETNO-CIĄGŁYCH PROBLEMÓW ROZDZIAŁU ZASOBÓW PRZEZ DYSKRETYZACJĘ ZASOBU CIĄGŁEGO

Streszczenie. W artykule rozważa się dyskretno-ciągły problem rozdziału zasobów. W problemach tego typu do wykonania zadań konieczne są jednocześnie zasoby ciągłe i dyskretne, przy czym wszystkie zasoby są odnawialne. Szybkość wykonania każdej czynności zależy od przydzielonej jej w danej chwili ilości zasobu ciągłego. Czynności są niepodzielne, a celem jest minimalizacja długości uszeregowania. W pracy zaproponowano metodę rozwiązania polegającą na transformacji powyższego problemu do klasycznego problemu rozdziału zasobów z wieloma sposobami wykonywania czynności.

SOLVING A DISCRETE-CONTINUOUS PROJECT SCHEDULING PROBLEM VIA A CONTINUOUS RESOURCE DISCRETIZATION

Summary. In this paper a discrete-continuous project scheduling problem is considered. In this problem activities simultaneously require for their processing discrete and continuous resources. The processing rate of each activity depends on the amount of the continuous resource allotted to this activity at a time. All the resources are renewable ones. The activities are nonpreemtable and the objective is to minimize the makespan. Discretization of this problem leading to a classical (i.e. discrete) project scheduling problem in the multimode version is presented.

1. Wstęp

Dyskretno-ciągłe problemy rozdziału zasobów są uogólnieniem dyskretno-ciągłych problemów szeregowania zadań, których ogólną metodykę rozwiązywania przedstawiono w [5]. W problemach tego typu do wykonania zadań wymagane są jednocześnie zasoby dyskretne i ciągłe (tj. podzielne w sposób ciągły). Zasoby ciągłe mogą być przy tym przydzielone do zadań w dowolnych, nie znanych zawczasu ilościach z pewnego danego przedziału. W pracy rozważać będziemy wyłącznie zasoby odnawialne, tj. takie, co do których zakładamy, że ograniczona jest tylko ilość zasobu dostępna w danej chwili.

Uwzględnienie w rozważanym problemie zasobów ciągłych daje możliwość zastosowania modelu wykonania zadania, w którym czas jego wykonania byłby funkcją

przydzielonego mu zasobu ciągłego. W problemach dyskretno-ciągłych celowe jest jednak wykorzystanie ogólniejszego modelu wykonania zadania, w którym nie czas lecz chwilowa szybkość wykonania każdego zadania jest opisana ciągłą, rosnącą funkcją ilości zasobu ciągłego przydzielonego temu zadaniu w danej chwili. Taki model umożliwia głęboką analizę własności uszeregowań optymalnych. Ponadto w pewnych, ważnych w praktyce przypadkach, prowadzi on do rozwiązań analitycznych. Problemy dyskretno-ciągłe z modelem wykonania zadania typu szybkość/zasób ciągły są też bardziej naturalne w sytuacjach praktycznych, gdyż wprost odwzorowują chwilową naturę zasobów odnawialnych.

W niniejszej pracy rozważany jest dyskretno-ciągły problem rozdziału zasobów dla zadań (czynności) niepodzielnych, przedstawiony w pracy [6]. W sformułowaniu tym do wykonania czynności wymagane jest jednocześnie wiele typów zasobów dyskretnych i jednego zasobu ciągłego, przy czym wszystkie zasoby są odnawialne, a problem polega na znalezieniu uszeregowania o minimalnej długości. Łatwo zauważyć, że tak sformułowany problem jest uogólnieniem klasycznego problemu rozdziału zasobów RCPSP (*ang. Resource Constrained Project Scheduling Problem*) z uwzględnieniem zasobów podzielnych w sposób ciągły.

W ogólności dyskretno-ciągły problem rozdziału zasobów może być zdekomponowany na dwa powiązane podproblemy: (i) konstrukcję kolejnościowo i zasobowo (ze względu na zasoby dyskretne) dopuszczalnej sekwencji czynności oraz (ii) przydzielenie wcześniej uszeregowanym czynnościom stosownej ilości zasobu ciągłego. Dla znalezionej w kroku (i) dopuszczalnej sekwencji czynności może być wyznaczona w kroku (ii) optymalna ze względu na rozważane kryterium ilość zasobu ciągłego. W tym celu rozwiązuje się nieliniowy (wypukły) problem programowania matematycznego. W pracy [6] krok (i) realizowany był przez jedną z trzech procedur lokalnego przeszukiwania, w których do oceny bieżącej sekwencji czynności konieczne było wyznaczenie optymalnego rozdziału zasobu ciągłego. Oznaczało to konieczność użycia solwera nieliniowego w każdej iteracji takiej procedury. Takie podejście, choć ważne z teoretycznego punktu widzenia, jest jednak niepraktyczne ze względu na bardzo duży czas obliczeń wymagany do rozwiązania problemu wypukłego programowania matematycznego dla pojedynczej dopuszczalnej sekwencji czynności.

W tej pracy zaproponowano inne podejście, które polega na przekształceniu rozważanego problemu do dyskretnego problemu rozdziału zasobów z wieloma sposobami wykonania czynności MRCPSP (*ang. Multi Mode Resource Constrained Project Scheduling Problem*).

2. Sformułowanie problemu

Rozważmy następujący problem. Dany jest zbiór n kolejnościowo i zasobowo ograniczonych czynności niepodzielnych, do wykonania których wymagane są zasoby odnawialne dwóch kategorii: dyskretnych i ciągłych. Zakładamy, że dostępnych jest m zasobów dyskretnych i jeden zasób ciągły. Wektor $\mathbf{r}_i = [r_{i1}, r_{i2}, ..., r_{im}]$, $i = 1, 2,..., n$ określa ustalone dyskretne żądania zasobowe czynności i. Całkowita liczba jednostek zasobu dyskretnego j, $j = 1, 2,..., m$ jest ograniczona przez R_j. Zasób ciągły może być przydzielony czynności w dowolnej ilości z przedziału [0,1]. Nieznana z góry ilość zasobu ciągłego $u_i(t)$ przydzielona czynności i w chwili t określa szybkość wykonania czynności i zgodnie z następującą zależnością:

$$\dot{x}_i(t) = \frac{dx_i(t)}{dt} = f_i[u_i(t)], \qquad x_i(0) = 0, x_i(C_i) = \tilde{x}_i \ , \tag{1}$$

gdzie:

$x_i(t)$ - stan czynności i w chwili t,
$u_i(t)$ - ilość zasobu ciągłego przydzielona czynności i w chwili t,
f_i - ciągła, rosnąca funkcja, $f_i(0)=0$,
C_i - nie znany z góry czas zakończenia czynności i,
$\tilde{x}_i$ - rozmiar (stan końcowy) czynności i.

Należy znaleźć sekwencję czynności dopuszczalną ze względu na ograniczenia kolejnościowe i zasobowe (ale tylko dyskretne) oraz równocześnie przydział zasobu ciągłego, które minimalizują długość uszeregowania $M = \max_i\{C_i\}$. Ograniczenia kolejnościowe reprezentowane są przez graf ograniczeń kolejnościowych. Przydział zasobu ciągłego zdefiniowany jest jako odcinkami ciągła, nieujemna funkcja wektorowa $\mathbf{u}(t)=[u_1(t), u_2(t), ..., u_n(t)]$, której wartości $\mathbf{u}^*=[u_1^*, u_2^*,..., u_n^*]$ są optymalnymi ilościami przydzielonego zasobu ciągłego, odpowiadającego M^* - minimalnej wartości M. Wykonanie czynności i, $i = 1, 2,..., n$ wymaga spełnienia warunku:

$$x_i(C_i) = \int_{t=0}^{C_i} f_i[u_i(t)]dt = \tilde{x}_i \ . \tag{2}$$

Zauważmy, że dowolne uszeregowanie dopuszczalne, będące rozwiązaniem dyskretno-ciągłego problemu rozdziału zasobów, może być podzielone na $p \le n$ przedziałów o długości M_k, $k = 1, 2,..., p$, określone przez momenty zakończenia kolejnych czynności. Niech Z_k oznacza kombinację czynności odpowiadającą k-temu przedziałowi. Z każdym uszeregowaniem dopuszczalnym jest związana *sekwencja dopuszczalna* S kombinacji Z_k, $k = 1, 2,..., p$.

Dopuszczalność takiej sekwencji jest zachowana jeśli:

- liczba jednostek zasobu dyskretnego j, $j = 1,2,...,m$ przydzielonych czynności w kombinacji Z_k, $k = 1, 2,..., p$ nie przekracza R_j, tj.:

$$\sum_{i \in Z_k} r_{ij} \le R_j , \qquad j = 1, 2,..., m, \qquad k = 1, 2,..., p,$$

- każda czynność występuje co najmniej w jednej kombinacji,
- uwzględnione są ograniczenia kolejnościowe między czynnościami,
- zagwarantowana jest niepodzielność każdej czynności.

Ostatni warunek wymaga, by każda czynność występowała w dokładnie jednej bądź w kolejnych kombinacjach w S.

Z praktycznego punktu widzenia szczególnie uzasadnione jest rozważanie wklęsłych funkcji szybkości wykonywania czynności. W pracy [5] wykazano, że dla takich funkcji w uszeregowaniu optymalnym równolegle wykonywana jest możliwie największa liczba czynności. W tym przypadku dla danej sekwencji dopuszczalnej S można wyznaczyć optymalny podział rozmiarów czynności i, $i = 1, 2,..., n$ pomiędzy kombinacje w S, rozwiązując odpowiedni problem wypukłego programowania matematycznego. W problemie tym minimalizowana jest suma długości przedziałów odpowiadających kolejnym kombinacjom w S pod warunkiem, że każda czynność musi być wykonana. Rozwiązanie tego problemu umożliwia znalezienie uszeregowania optymalnego dla danej sekwencji dopuszczalnej. W związku z tym globalnie optymalne uszeregowanie mogłoby być znalezione poprzez optymalne rozwiązanie problemu rozdziału zasobu ciągłego dla wszystkich sekwencji dopuszczalnych i wybór najlepszego z nich. W tym przypadku proces znajdowania uszeregowania optymalnego może być traktowany jako problem przeszukiwania przestrzeni wszystkich sekwencji dopuszczalnych dla danej instancji problemu. Niestety, w ogólności liczba wszystkich sekwencji dopuszczalnych rośnie wykładniczo wraz ze wzrostem liczby czynności. Uzasadnione jest zatem zastosowanie algorytmów metaheurystycznych, np. symulowanego wyżarzania S.A. (*ang. Simulated Annealing*), przeszukiwania tabu TS (*ang. Tabu Search*) czy algorytmu genetycznego GA (*ang. Genetic Algorithm*). W [6] porównano efektywność powyższych metaheurystyk. Jednak czasy obliczeń wynikające z konieczności wykorzystania solwera nieliniowego do oceny każdego z potencjalnych rozwiązań problemu istotnie ograniczają praktyczną użyteczność takiego podejścia.

3. Dyskretyzacja ciągłych żądań zasobowych

Jak wspomniano, dla modelu (1) możliwe jest sformułowanie wypukłych problemów programowania matematycznego, których rozwiązanie dla danej sekwencji dopuszczalnej

pozwala znaleźć optymalny przydział zasobu ciągłego, co jednak wymaga dużych nakładów obliczeniowych. W związku z tym proponujemy inne podejście, które polega na dyskretyzacji żądań zasobowych odnośnie do zasobu ciągłego (ciągłych żądań zasobowych). Oznaczmy przez $r_i^{l_i} = u_i^{l_i} \in (0,1]$ zdyskretyzowane ciągłe żądanie zasobowe czynności i, $l_i = 1, 2, \ldots, L_i$, gdzie L_i jest dane. Po przekształceniu równania (1) czasy wykonania czynności i dla żądania zasobowego $r_i^{l_i}$ uzyskamy ze wzoru:

$$\tau_i^{l_i} = \frac{\mathfrak{X}_i}{f_i(r_i^{l_i})} \quad . \tag{3}$$

Traktując zasób ciągły jako dodatkowy, tj. $m+1$ zasób dyskretny, otrzymamy zatem problem rozdziału zasobów z wieloma sposobami wykonania czynności (MRCPSP), w którym żądania zasobowe czynności i, $i = 1, 2, \ldots, n$ wykonywanych l_i-tym sposobem, $l_i = 1, 2, \ldots, L_i$ są określone wektorem $\mathbf{r}_i^{l_i} = [r_{i1}, r_{i2}, \ldots, r_{im}, r_{im+1}^{l_i}]$, $\sum_i r_{im+1}^{l_i} \le 1$, a czasy wykonania czynności dane są wzorem (3).

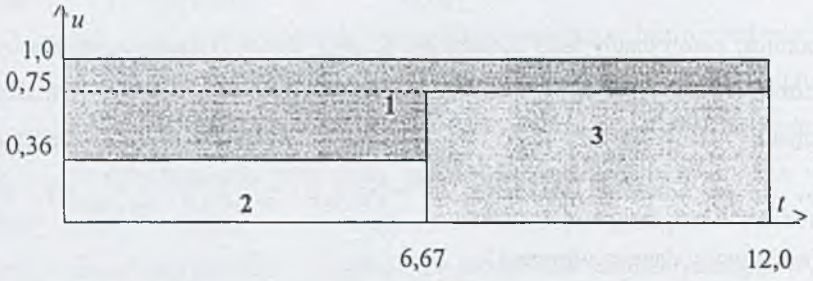
Idea dyskretyzacji może być atrakcyjna jako sposób na uniknięcie dużych nakładów obliczeniowych wymaganych do znalezienia optymalnego przydziału zasobów ciągłych. Istnieje wiele efektywnych algorytmów przybliżonych, rozwiązujących MRCPSP. Problemy tego typu szeroko rozważane były również w literaturze (por. [1], [2], [3], [4], [7]).

Należy w tym miejscu jednak zaznaczyć, że prezentowana tu dyskretyzacja ciągłych żądań zasobowych w problemie wyjściowym nie prowadzi do problemu mu równoważnego, to znaczy, optymalne rozwiązanie problemu MRCPSP nie gwarantuje optymalności uszeregowania odpowiadającego mu problemu dyskretno-ciągłego.

Dla ustalonej liczby L_i sposobów wykonania czynności i najbardziej naturalny wydaje się być równomierny podział przedziału [0,1], tzn. przyjęcie, że $r_i^{l_i} = \frac{l_i}{L_i}$, $l_i = 1, 2, \ldots, L_i$, $i = 1, 2, \ldots, n$. Przy takim założeniu zbiór rozwiązań dopuszczalnych problemu MRCPSP dla liczby sposobów wykonania czynności i równej L_i jest podzbiorem zbioru rozwiązań dopuszczalnych dla liczby sposobów równej $L_i' = k \cdot L_i$, gdzie k jest dowolną liczbą naturalną. Obserwacja ta jest zgodna z intuicją, że im większe L_i, tym lepiej przybliżamy rozwiązanie optymalne. Warto jednak zauważyć, że ta obserwacja nie dotyczy przypadków, kiedy $L_i' > L_i$ i $(L_i' \bmod L_i) \neq 0$. Poniżej przedstawiono przykład ilustrujący między innymi sytuację, w której zwiększenie liczby sposobów wykonania czynności nie prowadzi do zmniejszenia długości uszeregowania.

Przykład 1

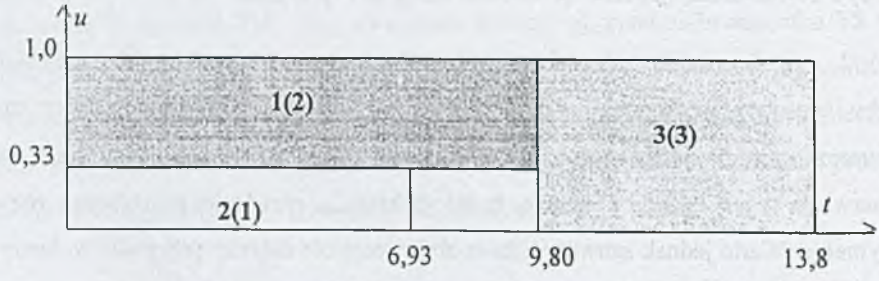
Niech będzie dany problem rozdziału zasobów dla $n = 3$ czynności o rozmiarach: $\tilde{x}_1 = 8$, $\tilde{x}_2 = 4$, $\tilde{x}_3 = 4$. W problemie występuje $m = 1$ zasób dyskretny dostępny w ilości $R_l = 2$ i jeden zasób ciągły. Jedynym ograniczeniem kolejnościowym jest wymóg wykonania czynności 3 po czynności 2 (2→3). Określone są funkcje szybkości wykonania poszczególnych czynności: $f_i = u_i^{1/2}$ dla i = 1, 2 i $f_3 = u_3$, oraz ich dyskretne żądania zasobowe $r_1 = r_2 = r_3 = 1$. Należy zminimalizować długość uszeregowania.



Rys.1. Uszeregowanie optymalne dla dyskretno-ciągłego problemu rozdziału zasobów
Fig.1. Optimal schedule for a discrete-continuous project scheduling problem

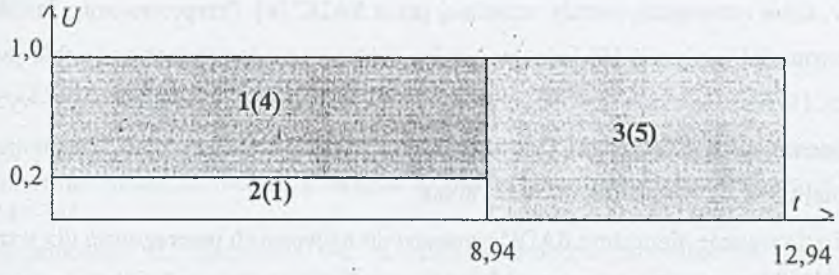
Na rys.1. pokazane jest uszeregowanie optymalne, w którym ilość zasobu ciągłego przydzielonego czynności 1 jest zmienna w czasie. Jeśli powyższy problem sprowadzimy do MRCPSP poprzez dyskretyzację ciągłych żądań zasobowych, to otrzymane rozwiązanie nie będzie lepsze od powyższego.

Niech $L_i = L = 3$ oraz $r_i^l = l_i / L$, $l_i = 1, 2, 3$, $i = 1, 2, 3$. Dla takiego przypadku uszeregowanie optymalne ma postać przedstawioną na rys.2 (w nawiasie podane są sposoby wykonania poszczególnych czynności).



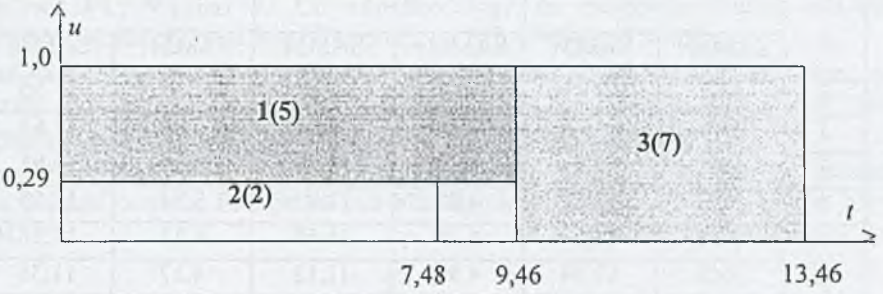
Rys.2. Uszeregowanie optymalne dla MRCPSP i $L_i = L = 3$
Fig.2. Optimal schedule for MRCPSP and $L_i = L = 3$

Dla $L_i = L = 5$ oraz $r_i^{l_i} = l_i / L$, $l_i = 1, 2, \ldots, 5$, $i = 1, 2, 3$ optymalne uszeregowanie (por. rys.3.) ma mniejszą długość. Oczywiście, wciąż jest to jednak długość większa od optymalnej dla oryginalnego dyskretno-ciągłego problemu rozdziału zasobów.



Rys.3. Uszeregowanie optymalne dla MRCPSP i $L_i = L = 5$
Fig.3. Optimal schedule for MRCPSP and $L_i = L = 5$

Zwiększenie liczby sposobów wykonywania czynności $L_i = L$ do $L = 7$ nie prowadzi do zmniejszenia długości odpowiadającego jej uszeregowania optymalnego, a wręcz przeciwnie, długość ta rośnie (por. rys.4.).



Rys.4. Uszeregowanie optymalne dla MRCPSP i $L_i = L = 7$
Fig.4. Optimal schedule for MRCPSP and $L_i = L = 7$

Praktyczna użyteczność zaproponowanej idei dyskretyzacji została sprawdzona przy zastosowaniu eksperymentu obliczeniowego. Racjonalną metodą weryfikacji tego podejścia jest porównanie wyników uzyskiwanych tą drogą z wynikami otrzymanymi przez algorytmy metaheurystyczne, wykorzystujące solwer nieliniowy w przypadku każdego odwiedzonego przez nie rozwiązania (badanymi w [6]). Założono przy tym, że punktem odniesienia będą wyniki uzyskane przez najlepszy z tych algorytmów, czyli algorytm symulowanego wyżarzania (oznaczmy go SADC). Do rozwiązywania problemu MRCPSP uzyskiwanego przez dyskretyzację problemu dyskretno-ciągłego użyto algorytmu symulowanego wyżarzania (SAMM) o takim samym schemacie chłodzenia oraz kryterium stopu. Różnice implementacyjne między SAMM a SADC, wynikające z konieczności przeszukiwania różnych przestrzeni rozwiązań dopuszczalnych, starano się ograniczyć do niezbędnego

minimum. Przez SAMM+ oznaczono wyniki otrzymane przez optymalne przydzielenie zasobu ciągłego w najlepszym rozwiązaniu znalezionym przez SAMM.

Eksperyment przeprowadzono dla instancji dyskretno-ciągłych problemów rozdziału zasobów, które rozwiązane zostały wcześniej przez SADC [6]. Przeprowadzono je zatem dla liczby czynności n równej 10, jednego zasobu dyskretnego dostępnego w liczbie jednostek $R = \{2, 5, 10\}$oraz jednego zasobu ciągłego. Oba algorytmy symulowanego wyżarzania zaimplementowano w języku C++, a eksperyment wykonano na komputerze SGI PowerChallenge XL z procesorami RISC 8000.

Zastosowanie algorytmu SADC prowadzi do najlepszych uszeregowań dla wszystkich instancji. Wyniki zamieszczone w tablicy 1 zawierają średnie odchylenia standardowe algorytmów SAMM i SAMM+ w stosunku do wyników uzyskanych przez SADC dla różnych L i R.

Tablica 1
Wyniki eksperymentu obliczeniowego – średnie odchylenia standardowe [%]

L	$R = 2$		$R = 5$		$R = 10$	
	SAMM+	SAMM	SAMM+	SAMM	SAMM+	SAMM
2	4.93	17.41	5.03	16.60	4.06	17.56
3	3.74	14.76	4.40	14.24	3.99	14.63
4	3.88	13.73	4.80	12.80	4.64	13.91
5	3.33	12.56	3.89	12.45	3.34	12.53
10	3.95	11.66	5.25	11.56	4.53	11.58
15	3.46	11.64	4.96	11.12	4.17	11.36
20	4.74	10.59	4.94	10.53	4.51	10.75
30	4.14	10.70	3.92	10.48	4.39	10.68
50	2.90	10.65	3.87	10.31	3.57	10.83
100	3.41	11.65	5.33	11.38	5.64	12.02

Średni czas obliczeń algorytmu SAMM wynosi 180 s, natomiast dla SADC - 13000 s. Uszeregowania znalezione przez SAMM są średnio gorsze o 10-18% od uszeregowań znalezionych przez SADC. Znaczną poprawę długości uszeregowań (średnie odchylenie standardowe: 2.9 – 5.7%) uzyskuje się przez zastosowanie algorytmu SAMM+. Wzrost czasu obliczeń wynikający z jednokrotnego zastosowania solwera nie jest w tym przypadku znaczący - średnio wynosi on 10 s.

4. Podsumowanie

W pracy pokazano metodę rozwiązywania dyskretno-ciągłych problemów rozdziału zasobów poprzez dyskretyzację ciągłych żądań zasobowych. W rezultacie dyskretyzacji

problem sprowadza się do dyskretnego problemu rozdziału zasobów z wieloma sposobami wykonania (MRCPSP). Na podstawie eksperymentu obliczeniowego pokazano, że metoda ta pozwala na znaczne (70 razy) skrócenie czasu obliczeń kosztem pogorszenia rozwiązania o 2.9 – 5.7%.

LITERATURA

1. Boctor F.F.: A new and efficient heuristic for scheduling projects with resource restrictions and multiple execution modes. European Journal of Operational Research, vol.90, 1996, pp. 349-361.
2. Demeulemeester E., Herroelen W.: A branch-and-bound procedure for the mulitple resource-constrained project scheduling problem. Management Science, vol.38(12), 1992, pp.1803-1818.
3. Drexl A., Gruenewald J.: Nonpreemptive multi-mode resource-constrained project scheduling. IIEE Transactions, vol.25, 1993, pp. 74-81.
4. Hartmann S., Drexl A.: Project scheduling with multiple modes: A comparison of exact algorithms. Networks, vol.32, 1998, pp. 283-297.
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.: Project scheduling under discrete and continuous resources. In Węglarz J.(ed.) Project scheduling: Recent Models, Algorithms and Applications. Kluwer, Dordrecht, 1999, pp. 289-307.
7. Sprecher A.: Resource-constrained project scheduling: Exact methods for the multi-mode case. Lecture notes in Economics and Mathematical Systems, Springer, Berlin, No 409, 1994.

Recenzent: Prof.dr hab.inż. K.Wala

Abstract

We consider a project scheduling problem in which nonpreemptive activities simultaneously require for their processing *m* discrete, renewable resources and one renewable continuous resource. Resource requirements concerning discrete resources are fixed, whereas the continuous resource allocation is not known in advance. Processing speed of activity *i* is an increasing function of the amount of the continuous resource allotted to this job at a time. The criterion is the project duration. We compare two approaches to solving this problem. The first one is based on a general methodology for discrete-continuous scheduling problems. The second one consists in discretization of the continuous resource leading to a Multimode Resource-Constrained Project Scheduling Problem. Simulated annealing algorithms are developed for both approaches. Results of a computational experiment comparing effectiveness of both algorithms are presented.