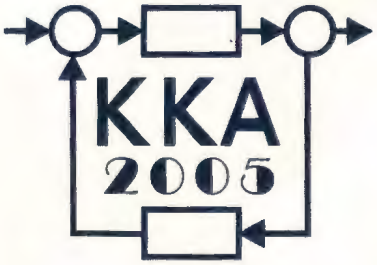
XV Krajowa Konferencja Automatyki

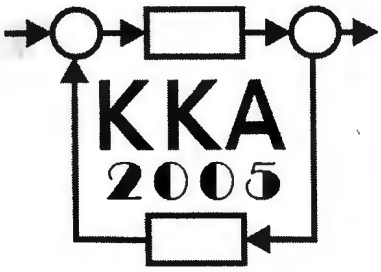
Tom II



Redaktorzy:
Zdzisław Bubnicki
Roman Kulikowski
Janusz Kacprzyk

XV Krajowa Konferencja Automatyki

Tom II



Redaktorzy:
Zdzisław BUBNICKI
Roman KULIKOWSKI
Janusz KACPRZYK

ORGANIZATOR
Komitet Automatyki i Robotyki Polskiej Akademii Nauk
Instytut Badań Systemowych Polskiej Akademii Nauk
WSPÓŁORGANIZATORZY
Politechnika Warszawska
Przemysłowy Instytut Automatyki i Pomiarów
Polskie Stowarzyszenie Pomiarów, Automatyki i Robotyki

ORGANIZATOR

Komitet Automatyki i Robotyki Polskiej Akademii Nauk
Instytut Badań Systemowych Polskiej Akademii Nauk

WSPÓŁORGANIZATORZY

Politechnika Warszawska
Przemysłowy Instytut Automatyki i Pomiarów
Polskie Stowarzyszenie Pomiarów, Automatyki i Robotyki

KOMITET PROGRAMOWY

Przewodniczący	Zdzisław BUBNICKI
Zastępca Przewodniczącego	Roman KULIKOWSKI

CZŁONKOWIE

Stanisław BAŃKA	Michał BIAŁKO
Mikołaj BUSŁOWICZ	Władysław FINDEISEN
Ryszard GESSING	Henryk GÓRECKI
Jakub GUTENBAUM	Jerzy JÓZEFCZYK
Stanisław KACZANOWSKI	Tadeusz KACZOREK
Janusz KACPRZYK	Jerzy KLAMKA
Józef KORBICZ	Zbigniew KOWALSKI
Krzysztof KOZŁOWSKI	Juliusz L. KULIKOWSKI
Krzysztof KUŹMIŃSKI	Kazimierz MALANOWSKI
Krzysztof MALINOWSKI	Wojciech MITKOWSKI
Antoni NIEDERLIŃSKI	Władysław PEŁCZEWSKI
Tadeusz PUCHAŁKA	Leszek RUTKOWSKI
Stanisław SKOCZOWSKI	Roman SŁOWIŃSKI
Jerzy ŚWIĄTEK	Andrzej ŚWIERNIAK
Ryszard TADEUSIEWICZ	Piotr TATJEWSKI
Krzysztof TCHOŃ	Leszek TRYBUS
Jan WĘGLARZ	Andrzej P. WIERZBICKI

KOMITET ORGANIZACYJNY

Przewodniczący	Roman KULIKOWSKI
Zastępcy Przewodniczącego	Janusz KACPRZYK
	Stanisław KACZANOWSKI
	Tadeusz KACZOREK
	Krzysztof MALINOWSKI
Członkowie	Roman OSTROWSKI
	Tadeusz PUCHAŁKA
	Dariusz WAGNER
Sekretarze naukowi	Jan STUDZIŃSKI
	Jan W. OWSIŃSKI

ISBN 83-89475-01-4

Druk: ARGRAF, Warszawa

STEROWANIE KOMPLEKSAMI OPERACJI

HYBRYDOWE ALGORYTMY SZEREGOWANIA ZADAŃ NA RUCHOMYCH REALIZATORACH DLA KRYTERIUM ŚREDNIEGO CZASU PRZEPŁYWU

Wojciech THOMAS

Politechnika Wrocławska, Instytut Informatyki Technicznej
ul. Janiszewskiego 11/17, 50-370 Wrocław, e-mail:wojciech.thomas@pwr.wroc.pl

Streszczenie: Przedstawiono wybrany problem szeregowania zadań na ruchomych realizatorach z kryterium w postaci średniego czasu przepływu, w zastosowaniu do nowoczesnych systemów produkcyjnych, w których realizatory (np. roboty) poruszają się między stanowiskami, na których są wykonywane zadania. W artykule omówiono krótko dwa algorytmy heurystyczne wykorzystywane do rozwiązania tego problemu: ewolucyjny oraz symulowanego wyżarzania. Na ich bazie zaproponowano również dwa algorytmy hybrydowe łączące zalety obu algorytmów składowych. Zamieszczono również wyniki badań symulacyjnych porównujące jakość rozwiązań i czasy działania przedstawionych algorytmów.

Słowa kluczowe: Algorytm symulowanego wyżarzania, algorytmy ewolucyjne, algorytmy hybrydowe, szeregowanie zadań, ruchome realizatory

1. WSTĘP

Problem szeregowania zadań na ruchomych realizatorach występuje w nowoczesnych systemach produkcyjnych, jeśli podmioty czynności są zbyt duże lub zbyt ciężkie, aby przemieszczać je pomiędzy wykonującymi je realizatorami, którymi mogą być np. ruchome roboty. Szczegółowo problem szeregowania zadań na ruchomych realizatorach omówiono w wielu pracach, m.in.: [4], [5], [6], gdzie rozpatrywano problemy szeregowania zadań z kryterium w postaci długości uszeregowania i maksymalnego opóźnienia. W pracy jest rozważane kryterium średniego czasu przebywania w systemie, definiowane jako suma momentów zakończeń wykonania zadań. Kryterium to jest istotne w wykorzystywanych współcześnie rozwiązaniach, ponieważ stawia na pierwszym miejscu liczbę wykonanych zadań. Współczesne firmy często pracują bez przerw, dlatego możliwość szybszego zakończenia pracy realizatorów nie ma tak dużego znaczenia, jak maksymalnie szybka obsługa jak największej liczby klientów.

Ze względu na złożoność obliczeniową systemu (problem jest NP-trudny) [8], istotne jest znalezienie algorytmów heurystycznych pozwalających efektywnie poszukiwać rozwiązań badanego problemu.

2. SFORMUŁOWANIE PROBLEMU

Zbiory zadań i realizatorów są odpowiednio oznaczane przez $\mathcal{H} = \{1, 2, \ldots, H\}$ oraz $\mathcal{R} = \{1, 2, \ldots, R\}$. Baza, z której wszystkie realizatory wyruszają oraz do której powracają po wykonaniu przydzielonych zadań jest oznaczana przez $H + 1$, zaś zbiór zadań z uwzględnieniem bazy jest oznaczany przez $\bar{\mathcal{H}}$.

Każde zadanie składa się z dojazdu realizatora do stanowiska oraz wykonania czynności na tym stanowisku. Na każdym stanowisku jest wykonywana dokładnie jedna czynność, dlatego zbiory stanowisk, czynności i zadań są takie same i nie wprowadza się dla nich osobnych oznaczeń. Dla uproszczenia przyjmuje się, że na stanowisku $H + 1$ (w bazie) jest wykonywana czynność pozorna o zerowym czasie trwania. Czas wykonania zadania h przez realizator r wynosi więc:

$$\tau_{r,h} = \hat{\tau}_{r,g,h} + \bar{\tau}_{r,h}, \tag{1}$$

przy czym $\hat{\tau}_{r,g,h}$ oznacza czas przejazdu realizatora r ze stanowiska g na stanowisko h, a $\bar{\tau}_{r,h}$ oznacza czas wykonania czynności na stanowisku h przez realizator r. Takie sformułowanie problemu sprawia, że czas wykonania zadania nie jest znany, dopóki nie są znane czasy przejazdu realizatorów.

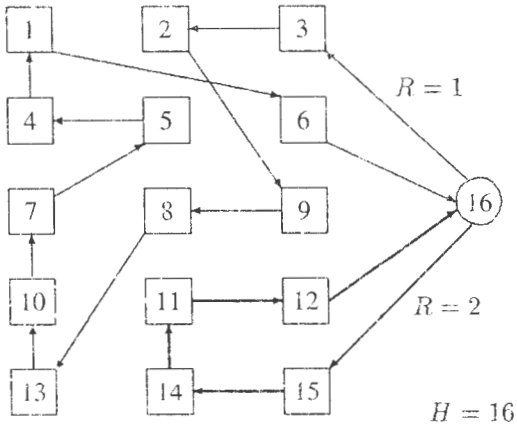
Rozwiązanie problemu jest reprezentowane przez binarną macierz decyzyjną γ:

$$\gamma = [\gamma_{r,h,n}], \tag{2}$$

gdzie $r = 1, 2, \ldots, R$, $h, n = 1, 2, \ldots, H$. Elementy macierzy mają następującą interpretację:

$$\gamma_{r,h,n} = \begin{cases} 1, & \text{jeśli realizator } r \text{ wykonuje zadanie } h \\ & \text{jako } (H_r - n + 1)\text{-te z kolei,} \\ 0, & \text{w pozostałych przypadkach} \end{cases}$$

gdzie H_r oznacza liczbę zadań wykonywanych przez realizator r. Na ich podstawie można wyznaczyć trasy przejazdów. Szczegóły dotyczące reprezentacji rozwiązania, w tym ograniczenia sprawiające, że macierz γ reprezentuje poprawne rozwiązanie problemu można znaleźć w pracy [9].



Rys. 1. Przykładowe trasy przejazdów dla 2 realizatorów i 15 zadań.

W omawianym problemie kryterium jakości znalezionego rozwiązania jest średni czas przebywania zadania w systemie, definiowany jako:

$$F = \sum_{h=1}^{H} C_h, \tag{3}$$

gdzie C_h jest momentem zakończenia wykonywania zadania.

Problem szeregowania zadań z ruchomymi realizatorami można więc sformułować jako problem optymalizacji dyskretnej w następujący sposób: dla danych R, H, τ znaleźć taką macierz decyzyjną γ, która minimalizuje wartość kryterium F.

3. ALGORYTM EWOLUCYJNY (EA)

Pierwszym z algorytmów heurystycznych wykorzystanych do rozwiązania problemu był algorytm ewolucyjny [9]. Rozwiązanie dopuszczalne jest w nim reprezentowane przez dwuwymiarową macierz g:

$$g = [g^{k,l}] \tag{4}$$

gdzie $k = 1, 2$ i $l = 1, 2, \ldots, H$. Elementy $g^{2,l}$ oznaczają numery zadań, zaś elementy $g^{1,l}$ oznaczają numery realizatorów do których przypisano odpowiednie zadania. Zadania przypisane do realizatora są wykonywane w takiej kolejności, w jakiej znajdują się w macierzy g. Poniżej przedstawiono fragment macierzy reprezentującej rozwiązanie przedstawione na rys. 1:

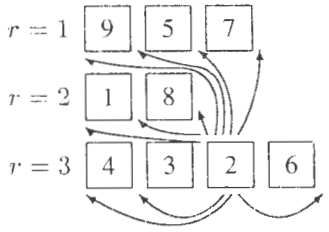
$$g = \begin{bmatrix} 1 & 1 & 1 & 1 & 2 & 1 & 2 & 1 & 1 & 1 & \cdots \\ 3 & 2 & 9 & 8 & 15 & 13 & 14 & 10 & 7 & 5 & \cdots \end{bmatrix}$$

Pomiędzy reprezentacją (4) a macierzą decyzyjną γ można dokonać przejścia w następujący sposób:

$$g^l = \begin{bmatrix} g^{1,l} \\ g^{2,l} \end{bmatrix} \Longrightarrow \gamma_{g^{1,l},g^{2,l},n} = 1$$

gdzie

$$n = \sum_{m=1}^{H} \delta(g^{1,l} - g^{1,m})$$



Rys. 2. Generowanie otoczenia bieżącego rozwiązania w algorytmie SA.

oraz

$$\delta(x) = \begin{cases} 1, & \text{jeżeli } x = 0 \\ 0, & \text{w przeciwnym wypadku} \end{cases}$$

Ze względu na możliwość powstania nieprawidłowych rozwiązań [2], w algorytmie zrezygnowano z klasycznego sposobu krzyżowania (z cięciem w jednym lub wielu punktach) i zastosowano zmodyfikowany operator krzyżowania, który wymienia między rozwiązaniami jedynie numery realizatorów. Z powodu modyfikacji operatora krzyżowania, zmianie uległ również operatory mutacji – wprowadzono kilka różnych operatorów mutacji.

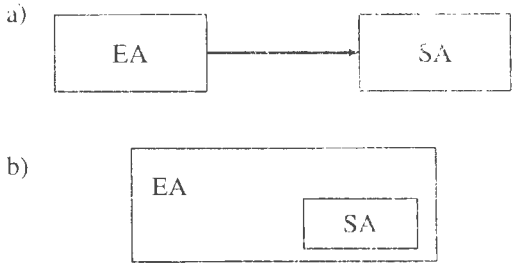
4. ALGORYTM SYMULOWANEGO WYŻARZANIA (SA)

Algorytm symulowanego wyżarzania (SA) należy do grupy algorytmów lokalnego przeszukiwania. Istotą algorytmu jest przeszukiwanie otoczenia bieżącego rozwiązania i wybór najlepszego z rozwiązań znajdujących się w sąsiedztwie. W przeciwieństwie do klasycznego algorytmu przeszukiwania lokalnego, w algorytmie SA istnieje możliwość przyjęcia jako nowego rozwiązania bieżącego rozwiązania gorszego od bieżącego, przy czym prawdopodobieństwo takiej sytuacji maleje w czasie działania algorytmu w sposób opisany tzw. schematem schładzania.

W przypadku systemów dyskretnych otoczenie bieżącego rozwiązania nie jest oczywiste tak jak w przypadku systemów ciągłych. W zastosowanym rozwiązaniu przyjęto, że otoczenie bieżącego rozwiązania stanowią takie rozwiązania problemu, w których losowo wybrane zadanie (na rys. 2 zadanie nr 2) jest umieszczane na wszystkich możliwych pozycjach, pomiędzy pozostałymi uszeregowanymi zadaniami. W algorytmie symulowanego wyżarzania zastosowano schemat schładzania zaproponowany w pracy [3].

5. ALGORYTMY HYBRYDOWE

Oba przedstawione algorytmy dla prezentowanego problemu mają swoje mocne i słabe strony. Algorytm ewolucyjny znajduje dobre rozwiązania, ale nawet wydłużenie czasu działania algorytmu nie powoduje znaczącej poprawy wartości kryterium. Algorytm symulowanego wyżarzania znajduje rozwiązania lepsze od algorytmu ewolu-



Rys. 3. Schematy blokowe proponowanych algorytmów hybrydowych: a) EASA i b) EA(SA).

cyjnego, jednak dzieje się to kosztem znacznego wydłużenia czasu działania algorytmu.

Jako rozwiązanie tego problemu zaproponowano algorytmy hybrydowe, łączące zalety obu rozwiązań oraz unikające ich słabości. Algorytm EASA (rys. 3a) stanowi szeregowe połączenie algorytmu ewolucyjnego i symulowanego wyżarzania. W tym algorytmie celem działania części ewolucyjnej jest przegląd przestrzeni rozwiązań i znalezienie rozwiązania możliwie bliskiego optimum globalnemu. Po znalezieniu takiego rozwiązania moduł realizujący algorytm symulowanego wyżarzania wykonuje "strojenie" rozwiązania.

W drugim przypadku oznaczonym jako EA(SA) (rys. 3b), algorytm symulowanego wyżarzania jest traktowany jako dodatkowy operator algorytmu ewolucyjnego. Operator ten jest wywoływany co określoną liczbę iteracji algorytmu ewolucyjnego dla najlepszego rozwiązania w bieżącej iteracji. Algorytm symulowanego wyżarzania **nie jest** wywoływany:

- dla każdego rozwiązania w populacji – ponieważ populacja nie byłaby wtedy zróżnicowana [1],
- w każdej iteracji algorytmu – ponieważ czas wykonania takiego algorytmu uległby znacznemu wydłużeniu

6. WYNIKI

W celu porównania zaproponowanych algorytmów przeprowadzono porównawcze badania eksperymentalne. W trakcie badań oceniono wartości kryterium dla poszczególnych algorytmów oraz czasy ich działania.

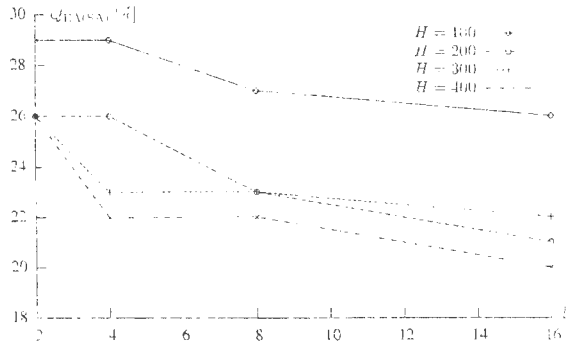
Wprowadzono następujące wskaźniki jakości umożliwiające ocenę jakości algorytmów hybrydowych wyrażonych poprzez wskaźnik jakości szeregowania (3), w stosunku do algorytmu ewolucyjnego EA:

$$Q_{\mathrm{EA(SA)}} = \frac{F_{\mathrm{EA}} - F_{\mathrm{EA(SA)}}}{F_{\mathrm{EA}}} \cdot 100\%, \qquad (5)$$

gdzie $F_{\mathrm{EA(SA)}}$ i F_{EA} oznaczają odpowiednio wartość kryterium rozwiązania znalezionego przez algorytm EA(SA) i EA oraz

$$Q_{\mathrm{EASA}} = \frac{F_{\mathrm{EA}} - F_{\mathrm{EASA}}}{F_{\mathrm{EA}}} \cdot 100\%. \qquad (6)$$

gdzie F_{EASA} oznacza wartość kryterium rozwiązania znalezionego przez algorytm EASA.



Rys. 4. Zależność $Q_{\mathrm{EA(SA)}}$ od l dla różnych H.

W badaniach symulacyjnych przyjęto następujące wartości parametrów:

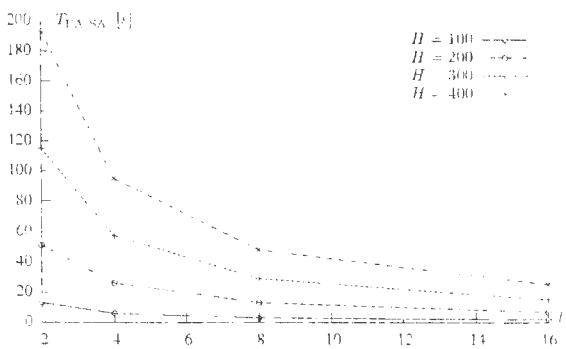
- dla algorytmu ewolucyjnego: $R = 3$, $J = 8000$ (liczba iteracji), $l = 50$ (liczebność populacji), $p_{\mathrm{M}} = 0.01$ (częstość mutacji), $p_{\mathrm{C}} = 0,95$ (częstość krzyżowania);
- dla algorytmu symulowanego wyżarzania: $R = 3$, $t_0 = 100000$ (temperatura początkowa), $t_k = 1$ (temperatura końcowa), $k = 5000$ (liczba kroków).

Prezentowane wybrane wyniki stanowią uśrednienie z 20 uruchomień każdego z algorytmów.

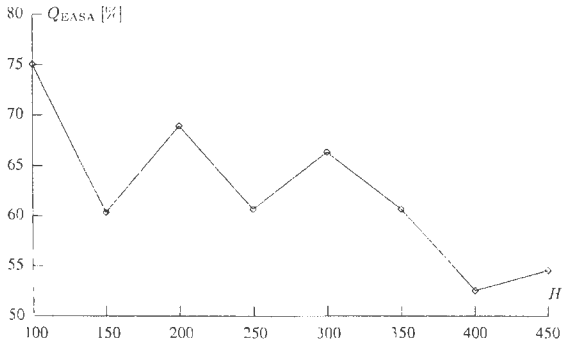
Na rys. 4 przestawiono wpływ częstości uruchamiania algorytmu symulowanego wyżarzania (co l iteracji algorytmu ewolucyjnego) na jakość rozwiązania znajdowanego przez algorytm hybrydowy. Jak wykazały badania częstość uruchamiania algorytmu SA ma niewielki (kilkuprocentowy) wpływ na zmianę wartości kryterium. Warto jednak zauważyć, że nawet rzadkie wywoływanie algorytmu SA powoduje poprawę jakości znajdowanego rozwiązania o ponad dwadzieścia procent.

Na rys. 6 przedstawiono procentową poprawę jakości znalezionego rozwiązania dla algorytmu EASA. Porównanie wyników wskazuje, że algorytm EASA zapewnia znacznie lepszą (o ponad 50%) poprawę wartości kryterium w stosunku do algorytmu EA. Przebieg Q_{EASA} dla wzrastających H jest nieregularny, ale zachowuje charakter malejący.

Porównując wykresy na rys. 5 i 7 można zauważyć, że algorytm EASA znajduje rozwiązania w czasie znacznie

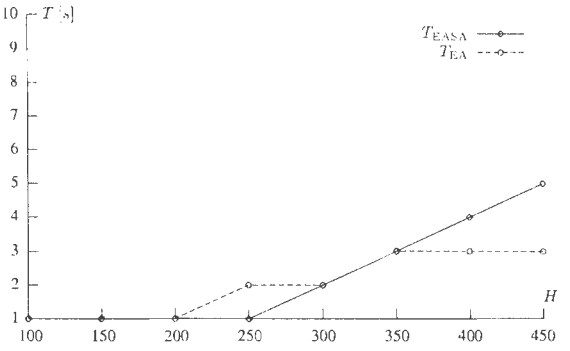


Rys. 5. Zależność $T_{\mathrm{EA(SA)}}$ od l dla różnych H.



Rys. 6. Zależność Q_{EASA} od H.

krótszym niż gorszy od niego algorytm EA(SA). Przyczyna takiego zachowania algorytmów wynika z rozwiązania, które jest przekazywane do algorytmu SA. W przypadku algorytmu EA(SA) jest to najlepsze rozwiązanie w bieżącej populacji, natomiast nie musi być ono najlepszym z rozwiązań znalezionych we wcześniejszych iteracjach (które jest pamiętane niezależnie od populacji). W przypadku algorytmu EASA, algorytm SA zaczyna przeszukiwanie przestrzeni rozwiązań od najlepszego znalezionego przez algorytm EA (które jednakże nie musiało znajdować się w populacji ostatniej iteracji).



Rys. 7. Zależność T_{EASA} oraz T_{EA} od H.

7. PODSUMOWANIE

W pracy zaproponowano dwa algorytmy hybrydowe rozwiązujące problem szeregowania zadań na ruchomych realizatorach. Z badań symulacyjnych wynika, że lepszym z dwóch proponowanych rozwiązań jest szeregowe połączenie algorytmów: ewolucyjnego i symulowanego wyżarzania, czyli algorytm EASA. Dalsze prace dotyczące tego zagadnienia powinny dotyczyć poprawy jakości działania samego algorytmu ewolucyjnego. Pewnym sposobem na polepszenie działania algorytmu EA(SA) mogłoby być uwzględnienie w algorytmie ewolucyjnym elity, czyli grupy najlepszych znalezionych w danej iteracji rozwiązań nie podlegających działaniu operatorów ewolucyjnych, dla których można by uruchamiać algorytm SA.

HYBRID TASK SCHEDULING ALGORITHMS FOR MOVING EXECUTORS AND MEAN FLOW TIME CRITERION

Abstract:

In the paper the task scheduling problem with moving executors is presented. This problem concerns modern discrete manufacturing systems. Moving executors (e.g. robots) move between workstations. The job is defined as the drive-up to a workstation and performing a task there. The each task is performed at the separate workstation. The execution times for jobs are unknown until moving executors routes are known. In the paper two heuristic algorithms are briefly presented: evolutionary algorithm and simulated annealing algorithm. Hybrid algorithms were also proposed. In the simulation experiments both algorithm were compared with simply evolutionary algorithm.

Literatura

[1] Arabas J. (2001) Wykłady z algorytmow ewolucyjnych. WNT, Warszawa.

[2] Czaja J., Józefczyk J. (2003) Zastosowanie algorytmów ewolucyjnych z wielokrotnym krzyżowaniem do rozwiązania problemu szeregowania zadań z uwzględnieniem ruchu realizatorów. Materialy konferencyjne Inżynieria Wiedzy i Systemy Ekspertowe, Wrocław.

[3] Janiak A. (1999) Wybrane problemy i algorytmy szeregowania zadań i rozdziału zasobów. PLJ, Warszawa.

[4] Józefczyk J. (1996) Szeregowanie zadań w kompleksie operacji z uwzględnieniem ruchu realizatorów. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław.

[5] Józefczyk J. (2001) Scheduling tasks on moving executors to minimise the maximum lateness, European Journal of Operational Research, vol. 131, no. 1, 171–187.

[6] Józefczyk J. (2001) Wybrane problemy podejmowania decyzji w kompleksach operacji. Monografie Komitetu Automatyki i Robotyki PAN, vol. 2, Oficyna Wydawnicza Politechniki Wrocławskiej, Warszawa–Wrocław.

[7] Józefczyk J., Thomas W. (2002) Wykorzystanie algorytmów ewolucyjnych do rozwiązania wybranego problemu szeregowania zadań z ruchomymi realizatorami. Zeszyty Naukowe Politechniki Śląskiej. seria: Automatyka. Gliwice, 233–242.

[8] Thomas W. (2004) Algorytm dokładny rozwiązania problemu szeregowania zadań z ruchomymi realizatorami dla kryterium MFT, Materiały konferencyjne KKAPD, Zakopane.

[9] Thomas W. (2003) Application of the genetic algorithm to solve the task scheduling problem with moving executors, Proceedings of AI-Meth, Gliwice.

Instytut Badań Systemowych
Polskiej Akademii Nauk

ISBN 83-89475-01-4