Prof. dr hab. inż. Jerzy Józefczyk
Instytut Informatyki Technicznej
Politechnika Wrocławska

PODEJMOWANIE DECYZJI W DWUPOZIOMOWYM KOMPLEKSIE OPERACJI PRODUKCYJNYCH

Streszczenie. Jest rozpatrywany dwupoziomowy kompleks operacji produkcyjnych, w którym łącznie są rozważane podproblemy decyzyjne szeregowania zadań na ruchomych realizatorach oraz sterowania jazdą grupy realizatorów. Przykładami zastosowań są dyskretne systemy produkcji, w których ruch wytwarzanych obiektów jest niemożliwy lub zbyt kosztowny i należy przemieszczać realizatory. Podproblemy decyzyjne występujące w systemie dwupoziomowym są ze sobą powiązane i należy je rozpatrywać łącznie. Dlatego istnieje potrzeba opracowania nowych algorytmów ich rozwiązania. Przedstawiono trzy heurystyczne algorytmy, działające na bieżąco lub w trybie off-line oraz zaprezentowano wybrane wyniki ich eksperymentalnej oceny. Podano wstępne uwagi dotyczące niepewnego systemu decyzyjnego, który zastępuje złożony system dwupoziomowy oraz przedstawiono przykładowy algorytm rozwiązania

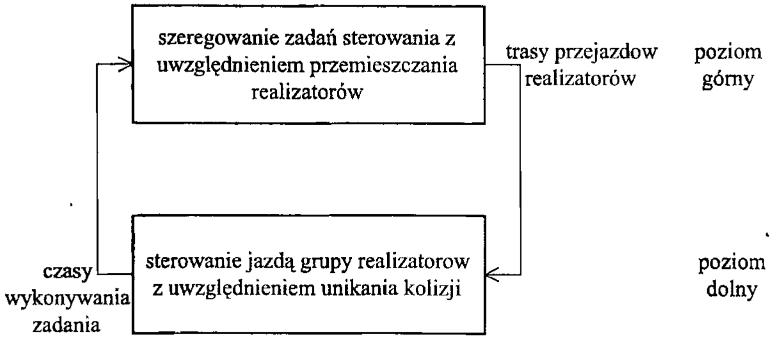
DECISION MAKING IN TWO-LEVEL COMPLEX MANUFACTURING SYSTEM

Abstract. A task scheduling problem on moving executors being a combined routing-scheduling problem is considered. An assumption concerning the movement of executors performing the tasks leads to a new discrete optimization problem which is formulated in the form of the decision making in a two-level system, where a motion control of a group of moving executors performing the tasks is also investigated. Discrete manufacturing systems are the main area of applications of such a complex problem which deal with the cases when the movement of plants to be produced is impossible or too expensive and it is necessary to move executors. Three heuristic on-line and off-line solution algorithms for the two-level system are given as well as the comparison of the algorithms via computer simulation are also presented. An uncertain decision making system which replaces the two-level deterministic system along with the example of a solution algorithm for such a system are also introduced.

1. WSTĘP

Szeregowanie i alokacja zadań należą do istotnych problemów podejmowania decyzji, które są formułowane i rozwiązywane dla systemów typu kompleks operacji, czyli dla takich złożonych obiektów, których elementy, tzn. operacje (zadania, czynności) są ze sobą powiązane na zasadzie kolejności czasowych. Dyskretne systemy produkcyjne, w tym elastyczne systemy produkcji i systemy komputerowo zintegrowanego wytwarzania, są ważnymi przykładami takich systemów. W pracy jest przyjmowane założenie, że czasy wykonywania zadań, które są głównymi charakterystykami kompleksów operacji oraz istotnymi danymi dla wszystkich problemów podejmowania decyzji, nie są dane a priori, ale są pośrednimi wynikami sterowania wykonywaniem zadań. Sytuacje takie można spotkać, gdy rozpatruje się ruch różnych elementów systemów produkcyjnych. Wówczas czasy wykonywania zadań produkcyjnych są pośrednimi wynikami sterowania. Uwzględnienie sterowania wykonywaniem zadań w problemach podejmowania decyzji w kompleksach operacji z jednej strony komplikuje te problemy, ale z drugiej – dotyczy zagadnień bliższych zastosowaniom praktycznym. Ruch w systemie produkcyjnym jest rozumiany w różny sposób i może dotyczyć różnych elementów systemu. Prezentowane rozważania są ograniczone do przypadku, gdy poruszają się podmioty wykonujące zadania produkcyjne, nazywane dalej realizatorami, a obiekty, na których zadania są wykonywane, pozostają nieruchome. Ruchome realizatory są stosowane przy produkcji obiektów, których przemieszczanie jest niewskazane lub niemożliwe, np. dużych silników lub elementów wagonów, ale także przy wykonywaniu zadań obsługowych lub transportowych. Okazuje się, że podejmowanie decyzji w kompleksie operacji oraz sterowanie wykonaniem zadań są ze sobą powiązane i nie mogą być rozpatrywane niezależnie. Wygodnie jest przedstawić je jako dwa połączone podproblemy tworzące tzw. dwupoziomowy system podejmowania decyzji w kompleksie operacji produkcyjnych (system dwupoziomowy). Pierwszy z nich jako bardziej strategiczny jest umieszczony na poziomie górnym. W ramach tak ogólnie zarysowanego problemu decyzyjnego mogą być rozpatrywane konkretne zagadnienia decyzyjne, rozwiązywane przez różne algorytmy. Koncepcja systemu dwupoziomowego została wprowadzona w [2] i nawiązuje do innych prac, np. [1,11–13].

W dalszym ciągu rozważania zostaną ograniczone do przypadku, gdy na poziomie górnym jest rozważany prosty problem szeregowania zadań na ruchomych realizatorach równoległych, a sterowanie dotyczy tylko jazdy realizatorów (Rys. 1). Podproblemy z



Rys. 1. Dwupoziomowy system podejmowania decyzji w kompleksie operacji produkcyjnych

obu poziomów były rozpatrywane odrębnie, a algorytmy ich rozwiązania zarówno dokładne, jak i przybliżone oraz heurystyczne są zamieszczone m.in. w [2–5,7–9].

W pracy skoncentrowano się na przedstawieniu oraz eksperymentalnej ocenie heurystycznych algorytmów rozwiązania dla systemu dwupoziomowego. Opisy algorytmów rozwiązania podproblemów zostały pominięte. W następnym rozdziale sformułowano problem, a w rozdziale 3. zaprezentowano algorytmy rozwiązania. Rozdział 4. zawiera wybrane wyniki badań symulacyjnych algorytmów, a w rozdziale 5. wskazano na możliwość sprowadzenia systemu dwupoziomowego do niepewnego systemu jednopoziomowego.

2. SFORMUŁOWANIE PROBLEMU

Problem podejmowania decyzji w systemie dwupoziomowym będzie sformułowany jako problem optymalizacji dyskretnej po krótkiej prezentacji obu podproblemów.

2.1. Podproblem na poziomie górnym

Na poziomie górnym jest rozwiązywany podproblem szeregowania zadań na ruchomych realizatorach, który jest rozwinięciem klasycznego problemu szeregowania, polegający na uwzględnieniu ruchu realizatorów w procedurze przydziału zadań do realizatorów. W celu wyjaśnienia koncepcji szeregowania na ruchomych realizatorach zostaną wprowadzone najważniejsze oznaczenia. *Zadanie* rozumiane tak jak w teorii szeregowania, ma swoją specyfikę wynikającą z ruchu realizatorów. *Realizator* jest zwykle urządzeniem technicznym, wykonującym zadania na nieruchomych obiektach, znajdujących się na *stanowiskach* rozmieszczonych na płaszczyźnie lub w przestrzeni. Wśród stanowisk jest wyróżniona baza, jako miejsce, w którym każdy realizator rozpoczyna i kończy swoje działanie. Główna idea szeregowania na ruchomych realizatorach polega na tym, że realizator w celu wykonania zadania musi dojechać do stanowiska. Dlatego każde zadanie składa się z dwóch części: z dojazdu realizatora do stanowiska oraz z wykonania *czynności* na stanowisku, przy czym w bazie nie jest wykonywana żadna czynność. W konsekwencji, *czas wykonania zadania* jest sumą *czasu dojazdu* do stanowiska oraz *czasu wykonania czynności* na obiekcie umieszczonym na stanowisku. Takie uogólnienie prowadzi do nowych problemów szeregowania, które nie mogą być rozwiązywane za pomocą znanych algorytmów, opracowanych dla wersji klasycznych, tzn. bez ruchu realizatorów. Wersje z ruchomymi realizatorami mogą być rozpatrywane dla wszystkich szczegółowych zagadnień szeregowania. W pracy jest badany prosty problem szeregowania zadań niezależnych i niepodzielnych na realizatorach dowolnych z równymi momentami gotowości w celu minimalizacji długości uszeregowania. Ponadto przyjęto założenie, że na każdym stanowisku może być wykonana tylko jedna czynność, oraz że każdy realizator rozpoczyna i kończy pracę w bazie. W konsekwencji, w notacji brak jest rozróżnienia między zbiorami zadań i stanowisk, oba są oznaczane jako $\boldsymbol{H} = \{1,2,...,H\}$, gdzie H jest liczbą zadań oraz stanowisk. Baza jest oznaczana jako $h = H+1$, stąd $\overline{\boldsymbol{H}} = \boldsymbol{H} \cup \{H+1\}$ jest zbiorem stanowisk wraz z bazą. Podobnie $\boldsymbol{R}$ i R to odpowiednio zbiór oraz liczba realizatorów. Czas $\tau_{r,g,h}$ wykonania zadania h przez realizator r po dojeździe ze stanowiska g jest sumą czasu $\bar{\tau}_{r,h}$ wykonania czynności h na stanowisku h przez realizator r oraz czasu $\hat{\tau}_{r,g,h}$ dojazdu realizatora r do stanowiska h ze stanowiska g, czyli $\tau_{r,g,h} = \bar{\tau}_{r,h} + \hat{\tau}_{r,g,h}$. Niech zmienna optymalizacyjna będzie trójwymiarową macierzą binarną $\gamma = [\gamma_{r,g,h}]_{\substack{r=1,2,\ldots,R \\ g,h=1,2,\ldots,H+1}}$, gdzie

$$\gamma_{r,g,h} = \begin{cases} 1, & \text{jeśli realizator } r \text{ wykonuje zadanie } h \text{ po dojeździe ze stanowiska } g, \\ 0, & \text{w przeciwnym przypadku.} \end{cases}$$

Wówczas długość uszeregowania można zapisać jako

$$Q(\gamma) = \max_{r=1,2,\ldots,R} \left\{ \sum_{h=1}^{H+1} \sum_{g=1}^{H+1} \gamma_{r,g,h} (\bar{\tau}_{r,h} + \hat{\tau}_{r,g,h}) \right\}. \qquad (1)'$$

Na macierz γ są nałożone ograniczenia, które zapewniają wyznaczenie dopuszczalnych rozwiązań, czyli wykonanie każdego zadania oraz określenie ciągłych tras dla realizatorów o początku i końcu w bazie.

Dopuszczalna macierz γ zawiera informacje o trasach przejàzdów realizatorów, które będąc faktycznymi rezultatami rozwiązania podproblemu z górnego poziomu, mogą być wyznaczone na podstawie tych elementów macierzy γ, które są równe 1 ([9]). Elementy takie dla ustalonego r, $r = 1,2,\ldots, R$ są brane pod uwagę przy ustalaniu trasy odpowiedniego realizatora. Trasy, czyli sekwencje stanowisk o początku i końcu w bazie są oznaczane jako $\boldsymbol{M}_r = (m_r(j))_{j=1,2,\ldots,M_r}$, gdzie $m_r(j)$ i M_r to odpowiednio j-ty element oraz długość trasy r-tego realizatora. Wartość bieżącego elementu $\boldsymbol{M}_r$, tzn. $m_r(j) = h$, $j = 1,2,\ldots, M_r$ jest równa indeksowi stanowiska (zadania), dla którego $\gamma_{r,m_r(j-1),m_r(j)} = 1$, gdzie $m_r(0) = H+1$. Ponadto, jeśli realizator przybywa do stanowiska $m_r(j)$, musi je również opuścić, stąd

$$\gamma_{r,m_r(j-1),m_r(j)} = 1 \Rightarrow (\exists!\ m_r(j+1) \in \overline{\boldsymbol{H}})(\gamma_{r,m_r(j),m_r(j-1)} = 1).$$

Tak więc, na podstawie macierzy γ, trasy $\boldsymbol{M}_r$ mogą być wyznaczone w sposób iteracyjny.

Ostatecznie, podproblem z poziomu górnego może być sformułowany następująco. Dla danych macierzy czasów $\bar{\tau} = [\bar{\tau}_{r,h}]_{\substack{r=1,2,\ldots,R \\ h=1,2,\ldots,H+1}}$, $\hat{\tau} = [\hat{\tau}_{r,g,h}]_{\substack{r=1,2,\ldots,R \\ g,h=1,2,\ldots,H+1}}$ oraz zbiorów $\overline{\boldsymbol{H}}$, $\boldsymbol{R}$, należy wyznaczyć dopuszczalną macierz γ, a w konsekwencji trasy $\boldsymbol{M}_r$, $r = 1,2,\ldots, R$ tak, aby minimalizować (1).

2.2. Podproblem na poziomie dolnym

Sterowanie grupą realizatorów faktycznie polega na sterowaniu jazdą poszczególnych realizatorów oraz koordynacji ich jazdy w celu uniknięcia ewentualnych kolizji z nieruchomymi i stacjonarnymi przeszkodami. Jest oczywiste, że sterowanie jazdą pojedynczego realizatora, które polega na doprowadzeniu go do stanu końcowego, dla danego stanu początkowego, nie może być przeprowadzone niezależnie od procedur sterowania innymi realizatorami, ponieważ algorytm sterowania danym realizatorem musi znać położenia (ogólnie – stany) pozostałych.

Jest rozważana grupa R ruchomych realizatorów (np. ruchomych robotów), poruszających się we wspólnej przestrzeni roboczej. Niech $x_r(t) = [y_r(t)^{\mathrm{T}}, \dot{y}_r(t)]^{\mathrm{T}}$,

$r = 1,2,...,R$ będzie wektorem stanu dla realizatora r, $y_r(t)$ i $\dot{y}_r(t)$ są odpowiednio l-wymiarowymi wektorami położeń i szybkości. Zmienne sterujące tworzą l-wymiarowy wektor $u_r(t)$. Stosując znane metody tworzenia modeli mechanizmów jazdy, po dyskretyzacji można przedstawić jego model w postaci następującego układu równań różnicowych

$$x_{r,\nu+1}^{(i)} = f_r^{(i)}(x_{r,\nu}, u_{r,\nu}), \; i = 1,2,.., 2l, \; \nu = 0,1,2,.. \tag{2}$$

ze stanem początkowym $x_{r,0}^{(i)} \triangleq x_r^{(i)}(\underline{t}_{m_r(j)})$, $j = 1,2,..., M_r$, gdzie ν oznacza takt sterowania, a $\underline{t}_{m_r(j)}$ jest momentem rozpoczęcia pokonywania $m_r(j)$-tego elementu trasy $\boldsymbol{M}_r$ lub równoważnie rozpoczęcia wykonywania zadania $h = m_r(j)$. Oznaczając przez $\bar{t}_{m_r(j)}$ moment osiągnięcia przez realizator r stanowiska $h = m_r(j)$, można podać związek między stanami końcowym i początkowym dwóch kolejnych elementów trasy $\boldsymbol{M}_r$ jako $x_r^{(i)}(\bar{t}_{m_r(j-1)}) = x_r^{(i)}(\underline{t}_{m_r(j)})$, $j = 1,2,..., M_r$. Ponadto zachodzi następująca równość $\underline{t}_{m_r(j+1)} = \bar{t}_{m_r(j)} + \bar{\tau}_{r,m_r(j)}$, która oznacza, że po osiągnięciu bieżącego stanowiska $h = m_r(j)$ realizator może rozpocząć pokonywanie następnego odcinka trasy dopiero po upływie czasu $\bar{\tau}_{r,h}$.

Dla rozpatrywanego podproblemu sterowania bardzo ważne są ograniczenia narzucone na zmienne stanu i zmienne sterujące. Dla zmiennych sterujących i dla zmiennych stanu będących szybkościami mają one najczęściej postać przedziałów o znanych końcach, tzn.

$$u_{r,\nu}^{(i)} \in \boldsymbol{U}_r^{(i)} = [\underline{u}_r^{(i)}, \bar{u}_r^{(i)}], \; \nu = 0,1,2,..., \; i = 1,2,..., s, \tag{3}$$

oraz

$$x_{r,\nu}^{(i)} \in \boldsymbol{X}_r^{(i)} = [\underline{x}_r^{(i)}, \bar{x}_r^{(i)}], \; \nu = 0,1,2,..., \; i = l+1, l+2,..., 2l. \tag{4}$$

Na pozostałą część zmiennych stanu, będących położeniami opisywanego mechanizmu, jest nałożone wspólne ograniczenie w postaci

$$y_{r,\nu} \in \boldsymbol{Y}_{r,\nu}, \tag{5}$$

tj. dla każdego taktu sterowania ν położenie r-tego realizatora powinno zawierać się w dopuszczalnym obszarze, określonym jako

$$\boldsymbol{Y}_{r,\nu} = \{y_{r,\nu} = [x_{r,\nu}^{(1)},..,x_{r,\nu}^{(l)}]^{\mathrm{T}} : y_{r,\nu} \in \boldsymbol{X}^{(1)} \times \boldsymbol{X}^{(2)} \times ... \times \boldsymbol{X}^{(l)} - (\bigcup_{\substack{s=1 \\ s \neq r}}^{R} \boldsymbol{D}_{s,\nu} \cup \bigcup_{h=1}^{\bar{H}} \overline{\boldsymbol{D}}_h)\},$$

gdzie $\boldsymbol{X}^{(i)} = [\underline{x}^{(i)}, \bar{x}^{(i)}]$, $i = 1,2,...,l$, $\boldsymbol{D}_{s,\nu}$ jest zmiennym w czasie obszarem zabronionym dla s-tego realizatora w takcie ν, $\overline{\boldsymbol{D}}_h$, $h = 1,2,...,\bar{H}$ są niezależnymi od czasu obszarami zajmowanymi przez przeszkody stacjonarne. Ograniczenie (5) jest dynamiczne, ponieważ wiąże się z koniecznością unikania kolizji z innymi realizatorami, traktowanymi jako przeszkody ruchome. Dla oceny jazdy r-tego realizatora pokonującego $m_r(j)$ -ty odcinek trasy, jest proponowany lokalny wskaźnik jakości sterowania, który umożliwia wyznaczenie w każdym takcie ν takiego sterowania $u_{r,\nu}$, aby maksymalnie zbliżyć się do stanu końcowego, czyli

$$Q_{r,\nu}(u_{r,\nu}) = \sum_{i=1}^{2l} \beta_i \left[x_r^{(i)}(\bar{t}_{m_r(j)}) - x_{r,\nu+1}^{(i)}\right]^2 = \sum_{i=1}^{2l} \beta_i \left[x_r^{(i)}(\bar{t}_{m_r(j)}) - f_r^{(i)}(x_{r,\nu}, u_{r,\nu})\right]^2, \quad (6)$$

gdzie $\beta_i, i = 1,2,..., 2l$ – dane nieujemne współczynniki. W celu zakończenia procedury sterowania wprowadzony jest warunek stopu

$$\left[\sum_{i=1}^{2l} (x_r^{(i)}(\bar{t}_{m_r(j)}) - x_{r,\nu}^{(i)})^2\right]^{\frac{1}{2}} < \varepsilon_r, \quad (7)$$

gdzie ε_r jest zadaną dokładnością osiągnięcia stanu końcowego. Wskaźnik jakości (6) oraz warunek stopu (7) wyrażają strategię sterowania „point-to-point".

Sformułowanie podproblemu sterowania jazdą r-tego realizatora pokonującego bieżący tj. $m_r(j)$ -ty element trasy $\boldsymbol{M}_r$ jest następujące. Dla danych: modelu (2) mechanizmu jazdy realizatora, stanu początkowego i końcowego, położeń wszystkich stanowisk należy wyznaczyć taki ciąg decyzji sterujących $(u_{r,\nu})_{\nu=0,1,2,...}$, dopuszczalnych w sensie ograniczeń (3)–(5), aby minimalizować (6), aż do spełnienia warunku stopu (7). Aby rozwiązać ten podproblem, muszą być znane obszary $\boldsymbol{Y}_{r,\nu}$, które z kolei mogą być wyznaczone w wyniku rozwiązania procedury koordynacji jazdy wszystkich realizatorów. Różne sposoby koordynacji jazdy realizatorów są podane w [6,9,10].

2.3 Problem decyzyjny dla systemu dwupoziomowego

Jako wskaźnik jakości dla systemu dwupoziomowego przyjmujemy kryterium z poziomu górnego, co oznacza, że jest rozwiązywany problem czasowo-optymalny w sensie długości uszeregowania. Na wyznaczanie decyzji na poziomie dolnym są nałożone tylko lokalne wymagania (6), a czasy dojazdu $\hat{\tau}_{r,g,h}$ są pośrednim wynikiem decyzji podejmowanych na tym poziomie. Dlatego macierz $\hat{\boldsymbol{\tau}}$ zależy od $\boldsymbol{u} \triangleq \{(u_{r,\nu})_{\nu=0,1,2,...}, r = 1,2,.., R\}$, gdzie $\boldsymbol{u}$ jest zbiorem zmiennych sterujących. Stąd otrzymujemy zależności $\hat{\boldsymbol{\tau}} = G(\boldsymbol{u})$, $Q(\boldsymbol{\gamma}, \hat{\boldsymbol{\tau}}) = Q(\boldsymbol{\gamma}, G(\boldsymbol{u})) = Q(\boldsymbol{M}, \boldsymbol{u}) \triangleq Q_{\mathrm{M,u}}$, gdzie $\boldsymbol{M} \triangleq \{\boldsymbol{M}_r : r = 1,2,..., R\}$.

Ostatecznie, dla danych z obu podproblemów problem podejmowania decyzji w systemie dwupoziomowym polega na wyznaczeniu tras przejazdów realizatorów $\boldsymbol{M}$ tak, aby minimalizować (1) oraz zmiennych sterujących $\boldsymbol{u}$ dla poszczególnych realizatorów, które zapewniają realizację strategii „point-to-point".

3. ALGORYTMY ROZWIĄZANIA

Podproblemy z obu poziomów są powiązane, co oznacza, że wyniki jednego z nich są danymi dla drugiego. I tak, trasy przejazdów realizatorów są danymi dla podproblemu sterowania bezkolizyjną jazdą grupy realizatorów, a czasy przejazdów, będące pośrednimi rezultatami generowanymi na poziomie dolnym są danymi dla podproblemu szeregowania zadań z poziomu górnego. Aby rozwiązać te powiązane podproblemy, opracowano kilka algorytmów heurystycznych. Trzy z nich są prezentowane w pracy. Dwa działają w trybie on-line. Pierwszy, czyli *algorytm adaptacyjny*, wykorzystuje tzw. dekompozycję czasową i zakłada bieżąca modyfikację tras przejazdów realizatorów. *Algorytm dyspozytorski* umożliwia wyznaczanie decyzji najlepszych tylko w bieżącym takcie podejmowania decyzji i nie obejmuje – tak jak to jest w przypadku algorytmu adaptacyjnego– horyzontu czasu pozostałego do końca działania algorytmu. Decyzje generowane przez *algorytm iteracyjny* są wyznaczane w trybie off-line przed rozpoczęciem procesu produkcyjnego. Algorytm ten polega na kolejnym rozwiązywaniu obu podproblemów, aż do osiągnięcia warunku stopu.

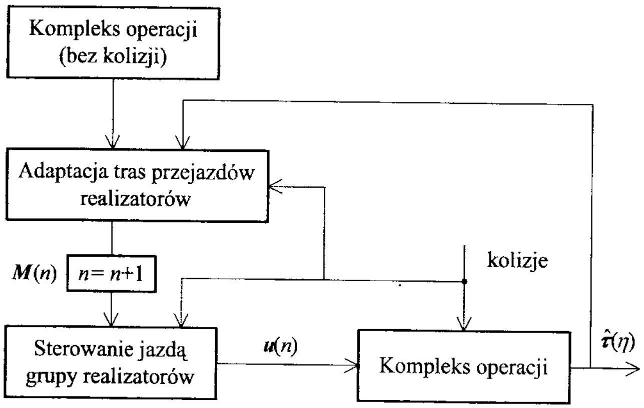
3.1. Algorytm adaptacyjny

Algorytm adaptacyjny umożliwia modyfikację tras przejazdów realizatorów w trakcie działania procesu produkcyjnego i wykorzystuje pojęcie *zdarzenia,* które polega na zakończeniu wykonania co najmniej jednego zadania. Wówczas jest podejmowana decyzja, czy należy zmodyfikować dotychczasowe trasy, czy też należy je utrzymać do następnego zdarzenia. Oznaczmy przez n i $\boldsymbol{M}_r(n)$ numer bieżącego zdarzenia (taktu podejmowania decyzji) i trasę r-tego realizatora w takcie n oraz określmy następujące podzbiory zbioru zadań $\boldsymbol{H}$: $\boldsymbol{H}(n)$ – niezakończone do taktu n ($\boldsymbol{H}(0) = \boldsymbol{H}$), $\boldsymbol{H}'(n)$ – w trakcie wykonywania w takcie n zgodnie z algorytmem szeregowania z poprzedniego taktu, $\boldsymbol{H}''(n)$ – nierozpoczęte do taktu n, $\boldsymbol{H}'''(n)$ – zakończone w takcie n. Niech $\Delta(n)$ będzie czasem trwania taktu n. Stąd, $\boldsymbol{H}''(n) = \boldsymbol{H}''(n-1) - (\boldsymbol{H}'(n) - \boldsymbol{H}'''(n))$. Dla zadań z podzbiorów $\boldsymbol{H}'(n)$ i $\boldsymbol{H}'''(n)$ są obliczane nowe czasy wykonywania według formuły $\tau_{r,h}(n) = \tau_{r,h}(n-1) - \Delta(n)$, a dla zadań należących do podzbioru $\boldsymbol{H}''(n)$ czasy pozostają bez zmian, czyli $\tau_{r,h}(n) = \tau_{r,h}(n-1)$. Wszystkie czasy $\tau_{r,h}(n)$ tworzą macierz czasów $\boldsymbol{\tau}(n) = \bar{\boldsymbol{\tau}} + \hat{\boldsymbol{\tau}}(n)$. Zmieniane są również ograniczenia ([8]). Zadania z podzbioru $\boldsymbol{H}'(n)$ nie są brane pod uwagę, bo właśnie się zakończyły, a stanowiska odpowiadające zadaniom z podzbiorów $\boldsymbol{H}'(n)$ lub $\boldsymbol{H}'''(n)$ są początkami nowych tras. Wówczas podproblem szeregowania polega na minimalizacji następującego wskaźnika jakości

$$Q(\gamma(n)) = \max_{r=1,2,\dots,R} \left\{ \sum_{h=1}^{H''(n)+1} \sum_{g=1}^{H''(n)+1} \gamma_{r,g,h}(n)(\bar{\tau}_{r,h} + \hat{\tau}_{r,g,h}(n)) \right\}, \tag{8}$$

ze zmodyfikowanymi ograničzeniami. Rozwiązaniami są trasy $\boldsymbol{M}_r(n)$ o początkach w $h'(n)$ i końcach w $h = H+1$, tzn. $\boldsymbol{M}_r(n) = (m_r(1,n), m_r(2,n), .., m_r(M_r(n),n))$, gdzie

$m_r(1,n) = h'(n)$ dla $h'(n) \in \boldsymbol{M}_r(n-1)$ i $m_r(M_r(n),n) = H+1$. Moment $t^h(n)$ zajścia zdarzenia (zakończenia wykonywania zadania) może być obliczony zgodnie z zależnością $t^h(n) = \bar{t}^{m_r(j,n)} = \underline{t}^{m_r(j-1,n)} + \tau_{r,h} = \underline{t}^{m_r(j-1,n)} + \bar{\tau}_{r,h} + \hat{\tau}_{r,m_r(j-1,n),m_r(j,n)}$, $j = 1,..,M_r(n)$, gdzie $m_r(j,n)$ dotyczy wykonania zadania h oraz $m_r(0,n)$ jest równe 0. Liczba zdarzeń $Z(n)$ jest nie większa niż $H+R$. Po uporządkowaniu, momenty $t^h(n)$ tworzą ciąg $\boldsymbol{T}(n)$. W każdym takcie jest sprawdzany warunek $\boldsymbol{T}(n) = \boldsymbol{T}(n-1)$, $n = 1,2,...$, czyli $t^z(n) = t^z(n-1)$, $z = 1,2,..., Z$. Jeśli warunek jest spełniony, to są kontynuowane trasy z poprzedniego taktu, w przeciwnym przypadku jest rozwiązywany nowy podproblem szeregowania zdań. Algorytm wymaga warunku początkowego w postaci macierzy czasów dojazdów $\hat{\boldsymbol{\tau}}(0)$, które są obliczane w trakcie bezkolizyjnych przejazdów między każdą parą stanowisk. Algorytm adaptacyjny podejmowania decyzji w systemie dwupoziomowym można przedstawić w postaci pięciu kroków (Rys. 2).



Rys. 2. Schemat blokowy algorytmu adaptacyjnego

Dane: $\overline{\boldsymbol{H}}(0)$, $\boldsymbol{R}$, $\hat{\boldsymbol{\tau}}(0)$, $\bar{\boldsymbol{\tau}}$, $n = 0$.

1. W wyniku rozwiązania podproblemu szeregownia zadań na ruchomych realizatorach wyznacz macierz $\gamma(n)$ lub równoważnie trasy $\boldsymbol{M}(n)$, a następnie określ ciąg $\boldsymbol{T}(n)$ i liczbę $Z(n)$.

2. Sprawdź, czy $H(n)$>0. Jeśli tak, to przejdź do następnego kroku, w przeciwnym przypadku zakończ algorytm.

3. W wyniku rozwiązania podproblemu sterowania jazdą realizatorów wyznacz $\boldsymbol{u}(n)$.

4. Zaobserwuj momenty $t^z(n)$, oblicz $\hat{\boldsymbol{\tau}}(n)$, $\boldsymbol{\tau}(n)$ i wyznacz $\boldsymbol{H}(n)$.

5. Sprawdź, czy $(\forall h'(n) \in \boldsymbol{H}'''(n))$ $(t^z(n) = t^{h'(n)}(n-1))$. Jeśli tak, podstaw $\boldsymbol{n} = n+1$ i przejdź do kroku 2, w przeciwnym przypadku przejdź do kroku 1.

3.2. Algorytm dyspozytorski

Algorytm działa również w trybie on-line, ale decyzje są podejmowane odrębnie dla każdego taktu. Polegają one na przyporządkowaniu, według wybranej heurystycznej reguły, zadań z podzbioru $\boldsymbol{H}''(n)$ do wolnych w danym takcie realizatorów, które tworzą podzbiór $\boldsymbol{R}''(n)$. Indeks stanowiska, na którym w takcie n znajduje się wolny realizator r z podzbioru $\boldsymbol{R}''(n)$, jest oznaczany jako $h_r(n)$. Algorytm składa się z pięciu następujących kroków (Rys. 3):

1. Ustal $n=0$, $\boldsymbol{H}''(0)=\boldsymbol{H}$, $\boldsymbol{R}''(0)=\boldsymbol{R}$, $h_r(0)=H+1$, $r=1,2,\dots,R$.

2. Kolejno, dla realizatorów od 1 do $R''(n)$:

   a. Wybierz zadanie h^* o najmniejszym numerze, dla którego jest spełniona nierówność $\bar{\tau}_{r,h^*}+\hat{\tau}_{r,h_r(n),h^*}\le\bar{\tau}_{r,h}+\hat{\tau}_{r,h_r(n),h}$, $h\in\boldsymbol{H}''(n)$, $h\ne h^*$ i przyporządkuj je do realizatora r,

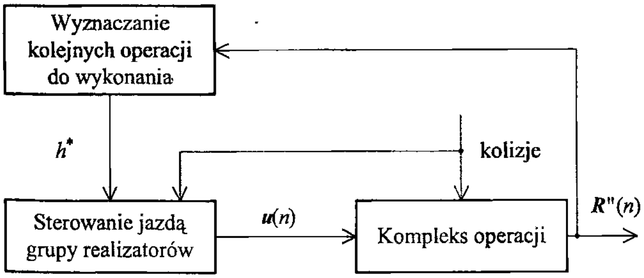
   b. Podstaw $\boldsymbol{H}''(n)=\boldsymbol{H}''(n)-\{h^*\}$.

Jeśli $n>0$ przejdź do kroku 4, w przeciwnym przypadku – do kroku 3.

3. Uruchom algorytm sterowania jazdą realizatorów na poziomie dolnym.

4. Zaobserwuj zdarzenie n, wyznacz podzbiór $\boldsymbol{R}''(n)$ oraz stanowiska $h_r(n)$, $r=1,2,\dots,R''(n)$ oraz podstaw $n=n+1$. jeśli $H''(n)>0$, tzn. jeśli podzbiór $\boldsymbol{H}''(n)$ jest niepusty, przejdź do kroku 2, w przeciwnym przypadku – do kroku 5.

5. Jeśli $R''(n)=R$ zakończ algorytm sterowania jazdą. W przeciwnym przypadku przyporządkuj zadanie $h=H+1$ do wszystkich realizatorów należących do podzbioru $\boldsymbol{R}''(n)$ i przejdź do kroku 4.



Rys. 3. Schemat blokowy algorytmu dyspozytorskiego

Heurystyczna reguła zastosowana w kroku 2. umożliwia znalezienie dla kolejnych realizatorów zadań o najkrótszych czasach wykonania.

3.3. Algorytm iteracyjny

W algorytmie iteracyjnym jedna iteracja, oznaczana jako η, polega na rozwiązaniu obu podproblemów decyzyjnych, tj, szeregowania zadań w celu uzyskania tras $\boldsymbol{M}$ i sterowania jazdą realizatorów, a w konsekwencji obliczenia czasów przejazdów $\hat{\tau}$.

Trasy i czasy przejazdów, które zostały wyznaczone w iteracji η, $\eta = 0,1,\ldots$, są oznaczane jako $\boldsymbol{M}(\eta)$ i $\hat{\tau}(\eta)$. Wartości $\hat{\tau}(\eta)$ oraz punkt startowy algorytmu w postaci czasów $\hat{\tau}(0)$ w przypadku bezkolizyjnej jazdy realizatorów mogą być uzyskane z modelu symulacyjnego – po zastosowaniu sterowań $\boldsymbol{u}(\eta)$. Warto zauważyć, że potencjalne kolizje między realizatorami mogą być traktowane jako zakłócenia działające na obiekt sterowania, czyli grupę ruchomych realizatorów. Konieczność unikania kolizji jest przyczyną uzyskiwania różnych wyników podejmowania decyzji w kolejnych iteracjach. Struktura systemu decyzyjnego, w którym zastosowano algorytm iteracyjny, jest przedstawiona na Rys. 4. Są wprowadzone dwa warunki stopu umożliwiające zakończenie algorytmu. W pierwszym z nich są brane pod uwagę wartości kryterium $Q_{\mathrm{M,u}}(\eta)$ z η_0 ostatnich iteracji, czyli

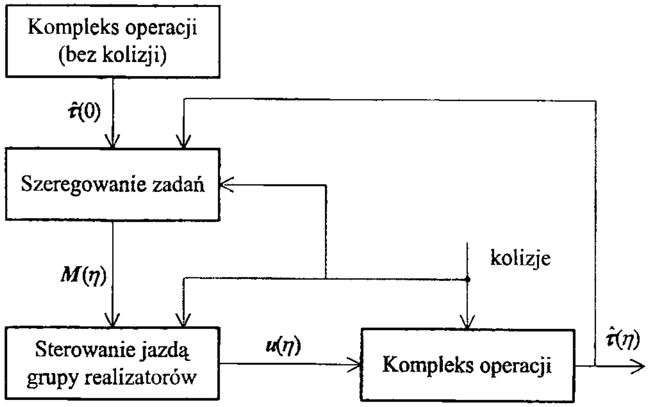
$$\sum_{k=\eta-\eta_0}^{\eta} \zeta^{\eta-k}(Q_{\mathrm{M,u}}(k-1)-Q_{\mathrm{M,u}}(k)) < \bar{\varepsilon}, \tag{9}$$

gdzie $\zeta \in (0,1]$ – współczynnik pamięci, $\bar{\varepsilon}$ – zadana dokładność algorytmu iteracyjnego.

W drugim warunku stopu są porównywane czasy przejazdów z dwóch kolejnych iteracji

$$\hat{\tau}(\eta) = \hat{\tau}(\eta-1),\ \eta = 1,2,\ldots. \tag{10}$$

Równanie macierzowe (10) należy rozumieć w sensie równości wszystkich elementów macierzy, tj. $\hat{\tau}_{r,g,h}(\eta) = \hat{\tau}_{r,g,h}(\eta-1)$.



Rys. 4. Schemat blokowy algorytmu iteracyjnego

Algorytm dla iteracji η, $\eta = 0,1,\ldots$ składa się z czterech następujących kroków.

Dane: $\overline{\boldsymbol{H}}, \boldsymbol{R}, \hat{\tau}(0), \bar{\tau}$, $\eta = 1$.

1. W wyniku rozwiązania podproblemu sterowania jazdą realizatorów wyznacz $\boldsymbol{u}(\eta)$.

2. Oblicz $\hat{\tau}(\eta)$ korzystając z modelu symulacyjnego.

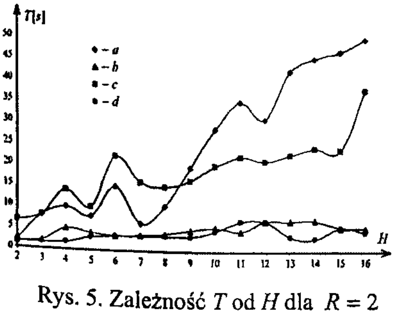
3. W wyniku rozwiązania podproblemu szeregownia zadań na ruchomych realizatorach wyznacz macierz $\gamma(\eta)$ lub równoważnie trasy $M(\eta)$.

4. Sprawdź warunek stopu (9) i (lub) warunek stopu (10). Jeśli co najmniej jeden z nich nie jest spełniony, podstaw $\eta = \eta + 1$ i przejdź do kroku 1. W przeciwnym przypadku zakończ algorytm i zapamiętaj rozwiązanie problemu decyzyjnego w postaci tras przejazdów $M(\eta)$ oraz wartość kryterium $Q_{\mathrm{M,u}}(\eta)$.

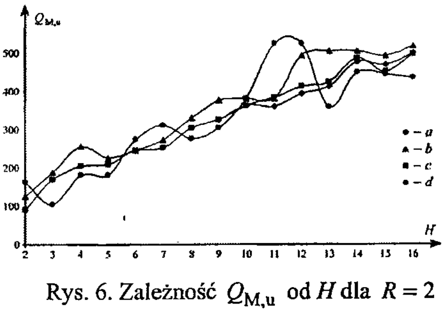
Wartości elementów macierzy $\hat{\tau}(0)$ są obliczane z modelu symulacyjnego w przypadku braku kolizji tak, jak to opisano w p. 3.1 (Rys. 4). Realizacja kroków 1. i 3. wymaga zastosowania odrębnych algorytmów, które są szczegółowo opisane np. w [4,5,8].

4. BADANIA SYMULACYJNE

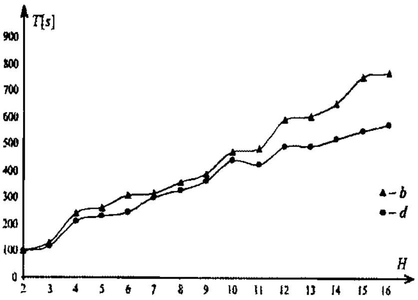
Zaproponowane heurystyczne algorytmy rozwiązania oceniono w sposób eksperymentalny, z użyciem napisanego w tym celu programu komputerowego, w którym do rozwiązania podproblemów z obu poziomów zastosowano algorytmy przybliżone. Obliczenia przeprowadzono dla $R \geq 2$ i $H \leq 16$. Macierz $\overline{\tau}$ czasów wykonania czynności na stanowiskach była generowana losowo według rozkładu prostokątnego. Stanowiska rozmieszczono równomiernie na kwadratowej przestrzeni roboczej. Przykładowe wyniki pokazano na Rys. 5–8 (kolejne litery na legendach wykresów oznaczają: a-algorytm iteracyjny z warunkiem stopu (9), b-algorytm adaptacyjny, c-algorytm iteracyjny z warunkiem stopu (10), d-algorytm dyspozytorski). Podstawą oceny były: wskaźnik jakości $Q_{\mathrm{M,u}}$ i czas realizacji algorytmu T. Badania przeprowadzono dla różnej liczby zadań H i realizatorów R. Analizując wartość wskaźnika jakości $Q_{\mathrm{M,u}}$ dla $R = 2$, można zauważyć, że algorytmy różnią się niewiele, tzn. oba algorytmy iteracyjne są nieznacznie lepsze od pozostałych, a algorytm dyspozytorski zachowuje się w sposób nieregularny. Podobne wnioski są słuszne dla $R = 4$. Na Rys. 8 przedstawiono wyniki tylko dla dwóch algorytmów, algorytm dyspozytorski jest nieznacznie lepszy niż adaptacyjny. W konsekwencji najlepszy jest algorytm iteracyjny z kryterium stopu (9), a najgorszy – algorytm adapta-



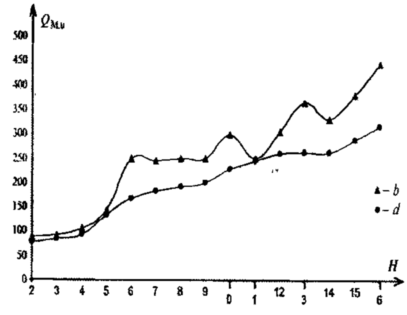
Rys. 5. Zależność T od H dla $R = 2$



Rys. 6. Zależność $Q_{\mathrm{M,u}}$ od H dla $R = 2$



Rys. 7. Zależność T od H dla $R = 4$



Rys. 8. Zależność $Q_{\mathrm{M,u}}$ od H dla $R = 4$

cyjny. Biorąc pod uwagę czasy działania algorytmów (Rys. 5 i 7), najlepszy jest algorytm dyspozytorski, a obie wersje algorytmu iteracyjnego są znacznie wolniejsze, zwłaszcza wersja z warunkiem stopu (9). Ostateczna rekomendacja jest następująca. Należy wybrać algorytm iteracyjny z warunkiem stopu (9), jeśli czas obliczeń nie jest ograniczony, a algorytm dyspozytorski – w przeciwnym przypadku.

5. NIEPEWNY SYSTEM DECYZYJNY

Wyznaczanie decyzji w systemie dwupoziomowym jest zagadnieniem złożonym i pracochłonnym, w szczególności dotyczy to podproblemu na poziomie dolnym. Tak jak to przedstawiono w poprzednich rozdziałach, celem nadrzędnym jest uzyskanie uszeregowania optymalnego w sensie kryterium (1). Konieczność rozpatrywania systemu dwupoziomowego wynika z braku dokładnych (deterministycznych) informacji o czasach przejazdów realizatorów, co z kolei jest spowodowane koniecznością unikania kolizji, które powodują opóźnienia w poruszaniu się realizatorów. Jeśli przyjąć założenie, że opóźnienie związane z uniknięciem pojedynczej kolizji jest możliwe do określenia, a maksymalna liczba kolizji, których musi uniknąć pojedynczy realizator jest równa liczbie pozostałych realizatorów (tzn. realizator może być w tzw. sytuacji kolizyjnej z innym realizatorem co najwyżej raz), to można wyznaczyć przedział czasu przejazdu realizatora. Granicami tego przedziału są: czas przejazdu bez konieczności unikania kolizji oraz czas przejazdu z $R-1$ opóźnieniami spowodowanymi unikaniem kolizji, w konsekwencji

$$\tau_{r,g,h} \in [\underline{\tau}_{r,g,h}, \overline{\tau}_{r,g,h}]. \qquad (11)$$

Wówczas nie ma potrzeby rozpatrywania podproblemu na poziomie dolnym. W zamian należy rozwiązać niepewny problem szeregowania zadań z poziomu górnego, przy czym niepewność dotyczy czasów wykonania zadań.

W dalszym ciągu zostanie krótko scharakteryzowane jedno z możliwych podejść do rozwiązania takiego niepewnego zagadnienia decyzyjnego, przy założeniu, że jedyna informacja o czasach wykonania zadań sprowadza się do znajomości granic przedziałów w (11). Polega ono na podejmowaniu decyzji odpornej, najlepszej w sensie tzw. względnej funkcji żalu w wersji pesymistycznej (ang. *worst-case relative regret function*). Należy wyznaczyć decyzje minimalizujące kryterium w postaci

$$z(\gamma) = \max_{\tau \in T} \frac{Q(\tau,\gamma) - Q^*(\tau)}{Q^*(\tau)}, \qquad (12)$$

gdzie $Q(\tau,\gamma)$ jest innym zapisem wskaźnika jakości (1), w którym podkreślono, że jego wartość zależy nie tylko od macierzy decyzyjnej γ, ale również od wektora wszystkich czasów wykonania czynności zapisanych w formie wektora τ, a $Q^*(\tau)$ oznacza optymalną wartość $Q(\tau,\gamma)$ dla ustalonego τ. Wartość wskaźnika (12) informuje, jaka jest największa ze względu na τ względna różnica między wartością kryterium $Q(\tau,\gamma)$ a jej wartością optymalną i wyraża żal (ang. *regret*), że $Q(\tau,\gamma)$ różni się od $Q^*(\tau)$. Okazuje się ([11]), że istnieje macierz γ optymalna ze względu na (12), którą można wyznaczyć według następującego algorytmu.

1. Rozwiąż, dla każdego realizatora osobno, problem deterministyczny, przyjmując $\tau = \tau^r$, gdzie elementy wektora τ^r są następujące: $\tau_{r,g,h} = \bar{\tau}_{r,g,h}$, $\tau_{s,g,h} = \underline{\tau}_{s,g,h}$ dla $s \neq r$, czyli czasy dla realizatora r przyjmuje się jako maksymalne, a dla pozostałych realizatorów – jako minimalne. Następnie oblicz wartości kryterium $Q^*(\tau^r)$.
2. Rozwiąż problem deterministyczny dla czasów wykonania zadań ustalonych jako $\tau_{r,g,h} = \bar{\tau}_{r,g,h} / Q^*(\tau^r)$.

Tak więc, rozwiązanie niepewnego problemu decyzyjnego zostało sprowadzone do rozwiązania $R+1$ problemów deterministycznych. Szczegóły można znaleźć w [11,12].

6. UWAGI KOŃCOWE

W pracy rozważono globalne podejście do rozwiązania złożonego problemu decyzyjnego w kompleksie operacji produkcyjnych, w którym łącznie potraktowano wybrany dyskretny problem decyzyjny (proste zagadnienie szeregowania zadań) oraz tradycyjny problem sterowania obiektem ciągłym (mechanizmem jazdy realizatora). Duża złożoność problemu nie doprowadziła, jak dotychczas, do uzyskania optymalnych, czy nawet przybliżonych algorytmów rozwiązania. Wyznaczono natomiast algorytmy heurystyczne, a trzy z nich omówiono w pracy. Przedstawiono również ich ocenę oraz porównanie, uzyskane w jako wynik przeprowadzonych badań symulacyjnych. Przedyskutowano również wstępnie możliwość zastąpienia deterministycznego problemu podejmowania decyzji w systemie dwupoziomowym odpowiednim problemem decyzyjnym dla systemu niepewnego.

Warto wspomnieć co najmniej o dwóch kierunkach dalszych prac. Po pierwsze, na obu poziomach mogą być rozpatrywane inne podproblemy decyzyjne niż te, które zaprezentowano w pracy. Dotyczy to zwłaszcza poziomu górnego, gdzie istnieje potrzeba badania innych zagadnień z zakresu szeroko rozumianych badań operacyjnych. Po drugie, można w różny sposób formułować problem decyzyjny dla systemu dwupoziomowego, m.in. różną postać mogą mieć wskaźniki jakości dla tego systemu.

Praca została sfinansowana ze środków budżetowych na naukę w latach 2005-2007 w ramach projektu badawczego nr 3 T11A 031 28.

LITERATURA

[1] Dumas Y, Desrosiers J., Gelinas E., Solomon M.M., An optimal algorithm for the travelling salesman problem with time windows, Operations Res. 43, No. 2, 1995.

[2] Józefczyk J., Two-level control algorithm for mobile executors in flexible manufacturing systems, W: Proc. of 10th International Conference on Systems Engineering, Coventry University, Coventry, UK 1994.

[3] Józefczyk J., On the functional decomposition approach to the problem of tasks scheduling on moving executors, W: Proc. of 11th International Conference on Systems Engineering, University of Nevada Las Vegas, USA 1996.

[4] Józefczyk J., An algorithm for scheduling tasks on moving executors, W: Proceedings of 1st IFAC Workshop on Manufacturing Systems: Modelling, Management and Control, Vienna, Austria 1997.

[5] Józefczyk J., Knowledge based motion control of a group of mobile executors in the two-level complex operation system, W: Proc. of 2nd World Manufacturing Congress WMC 1999 (Eds.: S.Nahavandi, M.Saadat), International Computer Science Conventions, Canada/Switzerland, 1999.

[6] Józefczyk J., Application of knowledge based pattern recognition in the control system of a group of executors, W: Proc. of 11th Int. Congress of Cybernetics and Systems (Eds.:R.Vallee, J.Rose), Brunel University, Uxbridge, UK 1999, s. 330-333.

[7] Józefczyk J., Scheduling tasks on moving executors to minimise the maximum lateness, European Journal of Operational Research, vol.131 2001, s. 171–187.

[8] Józefczyk J., Wybrane Problemy Podejmowania Decyzji w Kompleksach Operacji, seria „Monografie Komitetu Automatyki i Robotyki PAN", tom 2., Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 2001.

[9] Józefczyk J., Knowledge based two-level control of a group of mobile executors, Integrated Computer-Aided Engineering vol. 10, No. 2, 2003, s. 191-202.

[10] Józefczyk J., Application of knowledge based pattern recognition to movement control of a group of vehicles, International Journal of Knowledge-Based Intelligent Engineering Systems, 6, No.4, 2002, s. 192-198.

[11] Józefczyk J., Robust algorithm for task scheduling on moving executors with uncertain processing times, Proc. of 15th Mini-EURO Conf. "Managing Uncertainty in Decision Support Models MUDSM 2004", Coimbra, Portugal 2004, [cd-rom].

[12] Józefczyk J., Robust algorithm for the uncertain scheduling problem with moving executors. 16th IFAC World Congress, Praga, 4-8.07.2005 r. (referat przyjęty do wygłoszenia na kongresie).

[13] Langevin A., Desrochers M., Desrosiers J., Gelinas S., Soumis F., A two–commodity flow formulation for the traveling salesman and the makespan problems with time windows, Networks 23 (1993).

[14] Mingozzi A., Bianco L., Ricciardelli S., Dynamic programming strategies for the travelling salesman problem with time window and precedence constraints, Operations Research 45, No. 3, 1997.

[15] Nelson R.T., Sarin R.T., Daniels R.L., Scheduling with multiple performance measures: the one machine case, Management Science 32, 1986.