ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 2002
Seria: AUTOMATYKA z. 134 Nr kol. 1554

Jerzy JÓZEFCZYK, Wojciech THOMAS
Politechnika Wrocławska

WYKORZYSTANIE ALGORYTMÓW EWOLUCYJNYCH DO ROZWIĄZANIA WYBRANEGO PROBLEMU SZEREGOWANIA ZADAŃ Z RUCHOMYMI REALIZATORAMI*

Streszczenie. W pracy jest rozważane specyficzne, proste zagadnienie szeregowania zadań, w którym należy uwzględniać ruch realizatorów. Ograniczono się do szeregowania zadań niezależnych i niepodzielnych na realizatorach dowolnych w celu minimalizacji długości uszeregowania. Do rozwiązania sformułowanego problemu optymalizacyjnego zastosowano i przedstawiono dwie wersje algorytmu ewolucyjnego. Zaprezentowano wybrane rezultaty badań symulacyjnych, w których dokonano eksperymentalnej oceny rozpatrywanego algorytmu rozwiązania.

APPLICATION OF EVOLUTIONARY ALGORITHMS TO SCHEDULING OF TASKS ON MOVING EXECUTORS

Summary. Scheduling of manufacturing tasks on moving executors for a simple case is considered in the paper. The problem of independent and non-preemptive tasks as well as unrelated executors with makespan as the performance index is investigated. The corresponding optimization problem is formulated and solved using two versions of an evolutionary algorithm. The results of simulation experiments, which verify the quality of the solution algorithm, are given.

1. Wstęp

Rozpatrywanie coraz bardziej złożonych i bliższych zastosowaniom praktycznym problemów decyzyjnych z zakresu szeroko pojętych badań operacyjnych, w tym również dla dyskretnych systemów produkcyjnych, wymaga poszukiwania nowych, skutecznych metod i

* Praca została wykonana w ramach realizacji projektu badawczego nr 7 T11A 039 20 pt. „Algorytmy podejmowania decyzji i uczenia w systemach niepewnych i kompleksach operacji", finansowanego przez Komitet Badań Naukowych.

algorytmów rozwiązania. Metody i algorytmy sztucznej inteligencji, a także tzw. metaheurystyki są obecnie powszechnie stosowane do rozwiązywania takich problemów. Prezentowana praca odnosi się do systemu operacyjnego, nazywanego dalej kompleksem operacji. System taki składa się z operacji (zadań, czynności) powiązanych ze sobą na zasadzie kolejności czasowych. Są formułowane dla niego różne problemy decyzyjne związane m.in. z wykorzystaniem realizatorów (maszyn, procesorów) oraz innych zasobów niezbędnych do wykonania w określony sposób operacji z danego kompleksu. W pracy przedmiotem rozważań jest specyficzny problem szeregowania zadań, w którym czasy wykonywania zadań nie są dane a priori. Wykonywanie operacji przez realizatory wiąże się z koniecznością ich ruchu (jazdy), tzn. realizatory w celu wykonania zadania muszą dojechać do obiektu, na którym ma być ono wykonane. Ponieważ czas tego dojazdu nie może być z góry określony, stąd wspomniana nieznajomość czasów wykonania zadań. Dla takiego problemu rozwiązano szereg zagadnień szczegółowych, wyznaczając przy tym wiele algorytmów rozwiązania, zarówno dokładnych (problem jest NP–trudny), jak i heurystycznych, np. [2-5]. Rozważano m.in. zagadnienia szeregowania z takimi kryteriami jakości, jak: długość uszeregowania, maksymalne opóźnienie i suma momentów zakończenia zadań. Została również podjęta próba rozszerzenia dotychczasowych badań w kierunku zastosowania algorytmów ewolucyjnych do rozwiązania, powstałego w wyniku formalizacji zagadnienia, problemu optymalizacji dyskretnej. W rozdziale 2. jest krótko sformułowany rozpatrywany problem szeregowania. Następny rozdział jest poświęcony opisowi stosowanych algorytmów rozwiązania. W kolejnych rozdziałach są prezentowane odpowiednio: wyniki badań symulacyjnych i uwagi końcowe.

2. Sformułowanie problemu

Na początku wprowadźmy podstawowe określenia i oznaczenia. Ruch realizatorów powoduje, że operacje technologiczne, które odtąd będą określane jako zadania, są wykonywane w różnych miejscach – nazywanych dalej stanowiskami. Ponadto zakładamy, że na każdym stanowisku, a właściwie na zlokalizowanym tam obiekcie, realizator wykonuje tylko jedno zadanie. Oba zbiory, tj. zadania i stanowiska, są oznaczane jako $H = \{1,2,...,H\}$, gdzie H jest liczbą zadań bądź stanowisk. Dodatkowo wyróżnia się stanowisko, będące bazą dla realizatorów, z którego każdy realizator musi wyruszyć przed rozpoczęciem wykonywania pierwszego zadania i powrócić po wykonaniu ostatniego. Baza jest oznaczana jako $h = H+1$. Wtedy $\overline{H} = H \cup \{H+1\}$ jest zbiorem stanowisk wraz z bazą. Podobnie $\boldsymbol{R}$ i R są odpowiednio zbiorem i liczbą stanowisk. Najważniejszą charakterystyką zadań są czasy ich wykonania

przez poszczególne realizatory. Czas wykonania bieżącego zadania h przez bieżący realizator r, tj. $\tau_{r,h} = \overline{\tau}_{r,h} + \hat{\tau}_{r,g,h}$, $r = 1,2,...,R$ $h = 1,2,...,H$ $g = 1,2,...,H+1$, jest sumą czasu dojazdu tego realizatora ze stanowiska g, czyli $\hat{\tau}_{r,g,h}$, oraz czasu wykonania czynności na stanowisku h, czyli $\overline{\tau}_{r,h}$. Ze względu na założenie, że każdy realizator musi zjechać do bazy po zakończeniu wykonywania wszystkich zadań, dodatkowo przez $\hat{\tau}_{r,g,H+1}$ oznaczamy czas dojazdu realizatora r do bazy ze stanowiska g, g=1,2, ..., H.

Rozważany problem szeregowania można sformułować jako problem optymalizacji dyskretnej co najmniej na dwa równoważne sposoby. Jeden z nich prowadzi do tzw. problemu wyjściowego PW [2] z trójwymiarową macierzą binarną jako zmienną decyzyjną o postaci $\gamma = [\gamma_{r,g,h}]_{\substack{r=1,2,...,R \\ g,h=1,2,...,H+1}}$, gdzie $\gamma_{r,g,h} = 1(0)$, jeżeli realizator r wykonuje zadanie h po dojeździe ze stanowiska g (w przeciwnym przypadku). Jako wskaźnik jakości szeregowania przyjęto długość uszeregowania. Na macierz decyzyjną nałożono ograniczenia zapewniające, że każde zadanie będzie wykonane, trasy realizatorów są ciągłe, każdy realizator jeden raz wyjeżdża z bazy oraz trasy dla realizatorów są cyklami Hamiltona [5]. Długość uszeregowania możemy przedstawić w postaci następującego wyrażenia

$$Q(\gamma) = \max_{r=1,2,...,R} \{ \sum_{h=1}^{H+1} \sum_{g=1}^{H+1} \gamma_{r,g,h} (\overline{\tau}_{r,h} + \hat{\tau}_{r,g,h}) \}. \tag{1}$$

Szeregowanie zadań z ruchomymi realizatorami polega na wyznaczeniu dopuszczalnej macierzy γ tak, aby minimalizować kryterium (1) – dla danych: zbioru zadań (stanowisk) $\overline{H}$, zbioru realizatorów R oraz macierzy czasów dojazdów $\hat{\tau} = [\hat{\tau}_{r,g,h}]_{\substack{r=1,2,...,R \\ g,h=1,2,...,H+1}}$ i macierzy czasów wykonywania czynności na stanowiskach $\overline{\tau} = [\overline{\tau}_{r,h}]_{\substack{r=1,2,...,R \\ h=1,2,...,H}}$.

Faktycznymi wynikami szeregowania są trasy przejazdów realizatorów, które można określić na podstawie tych elementów macierzy γ, które są równe jeden. Tylko takie elementy dla ustalonego r są brane pod uwagę przy wyznaczaniu trasy dla realizatora r, $r = 1,2,...,R$. Trasy są wyrażone w postaci ciągów stanowisk o początkach i końcach w bazie. Są oznaczane jako $M_r = (m_r(j))_{j=1,2,...,M_r}$, gdzie $m_r(j)$ i M_r to odpowiednio j-ty element trasy i jej długość. Wartość bieżącego elementu trasy M_r, tj. $m_r(j) = h$, $j = 1,2,...,M_r$, jest równa numerowi stanowiska (zadania), dla którego $\gamma_{r,m_r(j-1),m_r(j)} = 1$,

gdzie $m_r(0) = H+1$. Jeżeli realizator dojeżdża do stanowiska $m_r(j)$, to musi je również opuścić. Wtedy $\gamma_{r,m_r(j-1),m_r(j)} = 1 \Rightarrow (\exists!\ m_r(j+1) \in \overline{H})(\gamma_{r,m_r(j),m_r(j+1)} = 1)$. Tak więc na podstawie macierzy γ trasy M_r mogą być wyznaczone w sposób iteracyjny.

3. Algorytmy rozwiązania

Oprócz opracowywania tradycyjnych algorytmów rozwiązujących problem optymalizacyjny sformułowany w poprzednim rozdziale, podjęto również próbę wykorzystania algorytmów ewolucyjnych. Mimo swoich oczywistych wad związanych głównie z czasem uzyskiwania rozwiązań są one ostatnio coraz częściej stosowanym narzędziem do wyznaczania optimów globalnych różnych problemów optymalizacyjnych – również z zakresu optymalizacji dyskretnej. W dalszym ciągu przedstawimy algorytm szeregowania oparty na podejściu ewolucyjnym. Wprowadzimy wpierw podstawowe pojęcia i oznaczenia właściwe dla takich algorytmów. Rozwiązanie dopuszczalne, będące w rozważanym przypadku trasami poszczególnych realizatorów, jest zapisywane w formie ciągu genów $(g_1, g_2, ..., g_M)$ o długości $M = H + R$. Ciąg ten jest podzielony na dwie części. Pierwsza składa się z H, a druga z R elementów. W drugiej części zapisano liczby zadań wykonywanych przez realizatory r, $r = 1,2,...,R$, czyli $g_{H+r} = M_r$. W pierwszej części kolejno dla poszczególnych realizatorów są zakodowane trasy przejazdów realizatorów w postaci numerów wykonywanych przez nie zadań. Bieżący element $m_r(j)$, $j = 1,2,...,M_r$ trasy M_r r-tego realizatora, będący numerem odpowiedniego stanowiska (zadania), jest zapisany jako g_m, gdzie $m = j + \sum_{i=0}^{r-1} M_i$ oraz $M_0 = 0$. Trasy M_r zawierają ciągi zadań wykonywanych przez realizatory i są uzupełnione przez bazę $H+1$, stanowiącą początek i koniec każdej trasy. Efektywność algorytmów ewolucyjnych, polegająca na osiąganiu ekstremów globalnych funkcji, silnie zależy od wartości parametrów tych algorytmów, takich jak: liczność populacji I, liczba iteracji (liczba pokoleń) J, a przede wszystkim prawdopodobieństwo krzyżowania p_1 i prawdopodobieństwo mutacji p_2. Zastosowanie podstawowego schematu algorytmu genetycznego polega na uruchamianiu w kolejnych iteracjach podstawowych operatorów genetycznych, tzn. selekcji, krzyżowania i mutacji. Do selekcji zastosowano metodę turniejową, a do krzyżowania procedurę PMX. Jako funkcję

przystosowania wprost przyjęto kryterium jakości szeregowania, czyli długość uszeregowania. Zastosowanie podstawowego schematu nie przyniosło zadowalających wyników, dlatego zaproponowano i opracowano nowe wersje algorytmu ewolucyjnego, w których w odmienny sposób w stosunku do klasycznego schematu algorytmu genetycznego, ustalano podstawowe jego parametry, jakimi są prawdopodobieństwo krzyżowania p_1 i prawdopodobieństwo mutacji p_2. W pierwszym przypadku dokonano adaptacji tych prawdopodobieństw w trakcie przebiegu algorytmu ewolucyjnego. Odpowiedni algorytm jest określany jako algorytm ewolucyjny z adaptacją. Ponadto do wyznaczania tych prawdopodobieństw zastosowano odpowiedni proces uczenia, a uzyskany algorytm rozwiązania nazwano algorytmem ewolucyjnym z uczeniem. Wprowadźmy wpierw następujące oznaczenia:

I_j – j-ta iteracja (populacja) algorytmu ewolucyjnego,

i, $i = 1,2,...,I$ – numer osobnika w populacji o liczności I (przyjmujemy, że wszystkie populacje są tak samo liczne),

$Q_j^{(i)}$ – wartość kryterium (funkcji przystosowania) dla i-tego osobnika w j-tej iteracji,

$\underline{Q}_j, \overline{Q}_j, \tilde{Q}_j$ – odpowiednio minimalna, maksymalna i średnia wartość kryterium w j-tej iteracji.

3.1. Algorytm ewolucyjny z adaptacją

W tym przypadku prawdopodobieństwa krzyżowania i mutacji są ustalane na podstawie wartości funkcji przystosowania w iteracjach bieżącej i poprzedniej, czyli

$$p_{l,j} = \Phi(Q_j^{(1)}, Q_j^{(2)}, ..., Q_j^{(I)}, Q_{j-1}^{(1)}, Q_{j-1}^{(2)}, ..., Q_{j-1}^{(I)}), \; l = 1,2 \;\; j = 2,3,...,J. \tag{2}$$

W pracy [4] podano kilka wersji ustalania tych prawdopodobieństw. Najskuteczniejsza z nich, wykorzystująca propozycję przedstawioną w pracy [6], jest następująca

$$p_{1,j} = \begin{cases} \rho_1 \dfrac{Q_j' - \underline{Q}_j}{\tilde{Q}_j - \underline{Q}_j}, \text{ dla } Q_j' \le \tilde{Q}_j, \\ \overline{\rho}_1, \text{ dla } Q_j' > \tilde{Q}_j, \end{cases} \tag{3}$$

gdzie Q'_j jest wartością kryterium dla rodziców $i_1, i_2 \in I_j$, obliczoną według jednego ze wzorów $Q'_j = 0{,}5(Q_j^{(i_1)} + Q_j^{(i_2)})$ lub $Q'_j = \min\{Q_j^{(i_1)}, Q_j^{(i_2)}\}$ oraz $\rho_1, \overline{\rho}_1 \in (0,1]$ są danymi współczynnikami. Prawdopodobieństwa mutacji dla osobników są wyznaczone według wzoru

$$p_{2,j} = \begin{cases} \rho_2 \dfrac{Q_j^{(i)} - \underline{Q}_j}{\tilde{Q}_j - \underline{Q}_j}, \text{ dla } Q_j^{(i)} \le \tilde{Q}_j, \\ \overline{\rho}_2, \text{ dla } Q_j^{(i)} > \tilde{Q}_j, \end{cases} \tag{4}$$

gdzie $\rho_2, \overline{\rho}_2 \in (0,1]$ są danymi współczynnikami. Proponowany sposób ustalania obu prawdopodobieństw zapewnia, że do następnej populacji są przenoszone osobniki (rozwiązania) o najlepszych wartościach kryterium. Pozostałe rozwiązania uczestniczą w poszukiwaniu rozwiązań w nowych obszarach zbioru rozwiązań dopuszczalnych.

3.2. Algorytm ewolucyjny z uczeniem

Innym sposobem wyznaczania obu prawdopodobieństw lub tylko jednego z nich jest zastosowanie uczenia. Punktem wyjścia przy uzasadnianiu celowości stosowania uczenia w rozważanym przypadku jest właściwość polegająca na tym, że wartości p_1 i p_2 mają z jednej strony wpływ na jakość uzyskiwanego wyniku optymalizacji, ale z drugiej strony zależą od danych problemu optymalizacyjnego i (lub) innych parametrów algorytmu ewolucyjnego. W przypadku szeregowania z ruchomymi realizatorami danymi tymi są przede wszystkim czasy dojazdów realizatorów do stanowisk, ale także czasy wykonywania czynności na stanowiskach. Parametrem algorytmu ewolucyjnego, mającym istotny wpływ na wartości p_1 i p_2, jest wielkość populacji I. Podamy teraz prosty przykład wyznaczania prawdopodobieństw krzyżowania i mutacji w procesie uczenia w wersji bez nauczyciela. W kolejnych cyklach uczenia n, $n = 0,1,\ldots$ wartości p_l, $l = 1,2$ są modyfikowane w sposób rekurencyjny na podstawie wartości kryterium jakości szeregowania, oceniającego kolejne zmiany p_1 i p_2. W każdym cyklu uczenia n jest uruchamiany algorytm ewolucyjny w celu wyznaczania wartości kryterium, ale za każdym razem dla nowych, generowanych losowo i niezależnych od wartości z poprzednich iteracji, danych problemu optymalizacyjnego i (lub) parametrów algorytmu ewolucyjnego. Można zaproponować różne algorytmy uczenia. W rozpatrywanym przypadku sprawdzono dwie ich wersje

$$p_l(n+1) = p_l(n) - \mu_l(n) d_l(n), \tag{5}$$

gdzie $\mu_l(n) = c/n^b$, $c, b > 0$ jest zmiennym współczynnikiem o własnościach $\mu_l(n) > 0$, $\sum_{n=0}^{\infty} \mu_l(n) = \infty$, $\sum_{n=0}^{\infty} \mu_l^2(n) < \infty$ oraz

$$p_l(n+1) = p_l(n) + \mu_l(n) d_l(n) \sigma_l(n), \tag{6}$$

gdzie $\mu_l(n) = c_l / n^{b_l}$ dla $c_l, b_l > 0$, $\sigma_l(n) = \mathrm{sgn}[d_l(n-1) - d_l(n)]$ oraz do wyznaczania $d_l(n)$ są wykorzystywane wartości funkcji przystosowania, tzn.

$$d_1(n) = \frac{1}{J} \sum_{j=1}^{J} \frac{\tilde{Q}_j(n)}{\overline{Q}_{\bar{j}}(n)}, \qquad d_2(n) = \frac{1}{J^2} \sum_{j=1}^{J} \frac{\underline{\tilde{Q}}_{\bar{j}}(n)}{\tilde{Q}_j(n)} \tag{7), (8}$$

oraz $\tilde{Q}_j(n) = \frac{1}{I} \sum_{i=1}^{I} Q_j^{(i)}(n)$, $j = 1,2,...,J$, $\overline{Q}_{\bar{j}}(n) = \max\{\overline{Q}_1(n), \overline{Q}_2(n), ..., \overline{Q}_m, ..., \overline{Q}_j(n)\}$,

gdzie $\overline{Q}_m(n) = \max_{i=1,2,...,I} \{Q_m^{(i)}(n)\}$, a także $\underline{Q}_{\bar{j}}(n) = \min\{\underline{Q}_1(n), \underline{Q}_2(n), ..., \underline{Q}_m, , \underline{Q}_j(n)\}$,

gdzie $\underline{Q}_m(n) = \min_{i=1,2,...,I} \{Q_m^{(i)}(n)\}$.

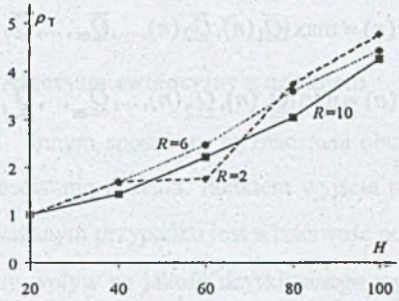
4. Badania symulacyjne

W celu weryfikacji proponowanych wersji algorytmów ewolucyjnych przeprowadzono badania symulacyjne. Dla algorytmu ewolucyjnego z uczeniem przeprowadzono dwa rodzaje badań: badanie procesu uczenia oraz badanie efektu uczenia. Wyniki badań przedstawiono m.in. w [4]. W pierwszym przypadku określano wpływ parametrów problemu szeregowania, np. liczby zadań H oraz liczby realizatorów R, a także parametrów algorytmu uczenia, np. współczynników c oraz b na liczbę cykli uczenia N, po której zostanie osiągnięte kryterium stopu

$$\frac{1}{L+1} \sum_{i=0}^{L} \{\zeta^i \sum_{k=1}^{2} [p_k(n-1) - p_k(n)]^2\} < \varepsilon, \tag{9}$$

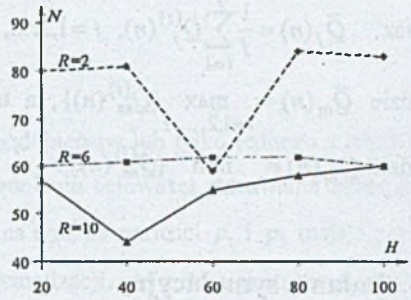
gdzie $\zeta \in (0,1]$, L i ε to odpowiednio zadany współczynnik, liczba cykli uczenia uwzględnionych przy obliczaniu dokładności i dokładność uczenia. Innym empirycznym wskaźnikiem oceniającym proces uczenia był względny czas wykonywania obliczeń $\rho_T = T/\overline{T}$, gdzie T – czas obliczeń, a $\overline{T}$ - czas obliczeń problemu o najmniejszym rozmiarze.

Zaobserwowano znaczący wpływ liczby zadań H na czas obliczeń T (rys. 1). Jest to spowodowane m.in. wydłużonym czasem selekcji dla problemów o większych rozmiarach. Okazało się również, że proces uczenia dla różnych rozmiarów problemu wymaga zbliżonej liczby cykli (rys. 2). Sprawdzono również wpływ parametrów algorytmu uczenia na jego wynik, czyli prawdopodobieństwa p_1 i p_2. Otrzymywano prawidłowe ich wartości dla szerokiego zakresu parametrów, np. rys. 3. Ocenę efektu uczenia, rozumianego w sensie kryterium jakości szeregowania, przeprowadzono wykorzystując losowo wygenerowany ciąg testujący o długości D. Elementem d, $d = 1,2,...,D$ takiego ciągu była instancja rozważanego problemu z określonymi losowo: liczbą zadań, liczbą realizatorów oraz macierzami czasów $\hat{\tau}$ i $\bar{\tau}$. Problem szeregowania dla każdej instancji rozwiązano dwukrotnie, tzn. z wykorzystaniem algorytmów ewolucyjnych z uczeniem oraz bez uczenia.



Rys. 1 . Zależność ρ_T od H dla różnych R

Fig. 1. Dependence of ρ_T on H for different R



Rys. 2. Zależność N od H dla różnych R

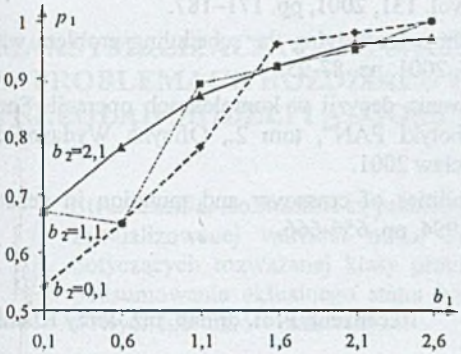
Fig. 2. Dependence of N on H for different R

W przypadku uczenia prawdopodobieństwa p_1 uzyskane wartości kryteriów jakości $Q(\gamma; p_1')$ oraz $Q(\gamma; p_1'')$ odpowiednio dla algorytmu z uczeniem oraz bez uczenia, gdzie p_1' i p_1'' to odpowiednio wartości prawdopodobieństwa krzyżowania uzyskane w wyniku uczenia oraz przyjęte a priori – zostały wykorzystane do wyznaczenia empirycznych wskaźników jakości oceny algorytmu ewolucyjnego z uczeniem. Zaproponowano trzy następujące formy takich wskaźników

$$\widetilde{W} = \frac{1}{D}\sum_{d=1}^{D} W_d, \quad \underline{W} = \min_{d=1,2,...,D}\{W_d\}, \quad \overline{W} = \max_{d=1,2,...,D}\{W_d\}, \tag{10}$$

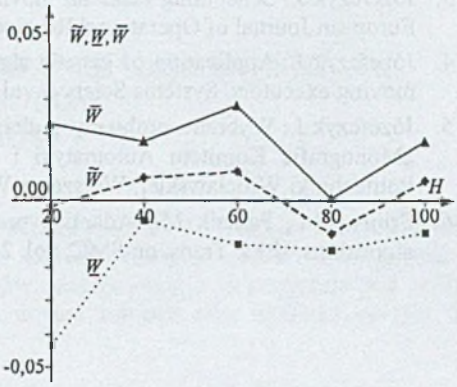
gdzie $W_d = (Q(\gamma; p_1') - Q(\gamma; p_1''))/Q(\gamma; p_1'')$. Podobnie jak w przypadku badania procesu uczenia, określano wpływ zmian parametrów problemu szeregowania i algorytmu uczenia na

wskaźnik wyrażający cel badania – w tym przypadku na (10). Przykładowe wyniki przestawiono na rys. 4. Zastosowanie uczenia można uznać za skuteczne, gdy wartości wskaźników w (10) są ujemne. Im mniejsze są to wartości, tym lepiej. Wyniki przedstawione na wykresie wskazują na możliwość kilkuprocentowej poprawy działania algorytmu ewolucyjnego w wyniku uczenia prawdopodobieństw krzyżowania.



Rys. 3. Zależność p_1 od b_1 dla różnych b_2

Fig. 3. Dependence of p_1 on b_1 for different b_2



Rys. 4. Zależność $\tilde{W}, \underline{W}, \overline{W}$ od H

Fig. 4. Dependence of $\tilde{W}, \underline{W}, \overline{W}$ on H

5. Uwagi końcowe

W pracy przedstawiono zastosowanie algorytmu ewolucyjnego do rozwiązania trudnego problemu optymalizacji dyskretnej, powstałego w wyniku formalizacji nieklasycznego zagadnienia szeregowania zadań, w którym jest konieczne uwzględnianie przemieszczania się realizatorów, traktowanych jako podmioty wykonujące zadania. Wstępne wyniki badania algorytmu rozwiązania wskazały na jego przydatność, ale z drugiej strony na konieczność dalszych modyfikacji, przede wszystkim w celu skrócenia czasu obliczeń. Prace takie, w których jest stosowany inny sposób kodowania oraz udoskonalone operatory reprodukcji, są obecnie prowadzone. Stosowanie takich technik rozwiązywania trudnych problemów optymalizacyjnych jak algorytmy ewolucyjne, czy algorytmy hybrydowe, powstałe z połączenia różnych szczegółowych podejść, na przykład algorytmów ewolucyjnych i algorytmów symulowanego wyżarzania – jest szczególnie przydatne w przypadku rozpatrywania bardziej złożonych problemów szeregowania zadań z uwzględnieniem ruchu realizatorów.

LITERATURA

1. Baeck T.: Evolutionary Algorithms in Theory and Practice. Oxford University Press, New York 1996.
2. Józefczyk J.: Szeregowanie zadań w kompleksie operacji z uwzględnieniem ruchu realizatorów. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 1996.
3. Józefczyk J.: Scheduling tasks on moving executors to minimise the maximum lateness. European Journal of Operational Research, vol. 131, 2001, pp. 171–187.
4. Józefczyk J.: Application of genetic algorithms for solving the scheduling problem with moving executors. Systems Science, vol. 27, 2001, pp. 87-95.
5. Józefczyk J.: Wybrane problemy podejmowania decyzji w kompleksach operacji. Seria „Monografie Komitetu Automatyki i Robotyki PAN", tom 2., Oficyna Wydawnicza Politechniki Wrocławskiej, Warszawa-Wrocław 2001.
6. Srinivas M., Patnaik M.: Adaptive probabilities of crossover and mutation in genetic algorithms. IEEE Trans. on SMC, vol. 24, 1994, pp. 656-666.

Recenzent: Prof. dr hab. inż. Jerzy Klamka

Abstract

The paper concerns the decision making in a complex operation system with application to discrete manufacturing systems. The simple problem of scheduling of tasks on moving executors is considered. The case of independent non-preemptive tasks as well as unrelated executors with makespan as the performance index is investigated. The scheduling problem is stated as the discrete optimization problem. For solving this problem the approach based on evolutionary computations is used. Two versions of modified genetic algorithm called as evolutionary algorithms are presented. For both cases, standard genetic operators are applied, i.e. the selection with the tournament method and PMX procedure for the crossover. The first evolutionary algorithm uses during the calculations the different values of crossover and mutation probabilities, which are obtained via the process of their adaptation. For the second version, these probabilities are results of the learning process performed before starting the optimization procedure. The solution algorithms are verified during simulation experiments. The results concerning the second version are presented in the paper. The learning process and the results of learning are evaluated. The second ones are understood as makespan and both probabilities. The parameters of the scheduling problem, i.e. the number of tasks and the number of executors as well as parameters of the learning process are the basis for the evaluation