Jarosław PEMPERA, Czesław SMUTNICKI
Politechnika Wrocławska

HARMONOGRAMOWANIE CYKLICZNE W PRZEPŁYWOWYM SYSTEMIE PRODUKCYJNYM Z OGRANICZENIEM BEZ CZEKANIA

Streszczenie. Praca dotyczy poszukiwania optymalnego harmonogramu cyklicznego w tzw. przepływowym systemie wytwórczym z ograniczeniem bez czekania dla zadań o różnych dedykowanych marszrutach. Zaproponowano oryginalny model grafowy rozważanego problemu pozwalający na wyznaczenie harmonogramu oraz minimalnego czasu cyklu w czasie wielomianowym. Sformułowano pewne własności problemu, które zostały wykorzystane w konstrukcji dwu-poziomowego metaheurystycznego algorytmu optymalizacyjnego. Wysoka efektywność algorytmu została potwierdzona podczas badań eksperymentalnych.

CYCLIC SCHEDULING IN FLOW SHOP SCHEDULING PROBLEM WITH NO-WAIT CONSTRAINTS

Summary. The paper deals with the problem of seeking optimal cyclic schedule in so called flow manufacturing system with no-wait constraints and missing operations. There were proposed an original graph model of the considered problem allowing for the effective determination of the schedule as well as the minimal cycle time. Some properties of the problem were formulated, which were used in the construction of the two-level metaheuristic optimization algorithm. The high efficiency of the algorithm was confirmed in an experimental research.

1. Wprowadzenie

W teorii i praktyce szeregowania zadań dla potrzeb systemów wytwarzania rozważa się co najmniej dwa różne rodzaje harmonogramów: wsadowe (batch), cykliczne (cyclic). Pełna taxonomia uwzględnia, między innymi, struktury systemów wytwórczych, dodatkowe ograniczenia, kryteria optymalizacji. Systemy o strukturze szeregowej, przepływowe (tzw. flow-shop) prowadzą do problemów optymalizacyjnych o umiarkowanym stopniu skomplikowania, pozwalając tym samym na przystępne opisanie proponowanego podejścia. W tej pracy rozważamy cykliczny przepływowy system wytwórczy, w którym pewien zestaw produktów (mieszanka) wytwarzany jest w powtarzalnych, następujących po sobie, seriach. W niektórych systemach tego typu występują pewne dodatkowe ograniczenia technologiczne, które powodują, że problem staje się nietrywialny. Niniejsza praca uwzględnia ograniczenia znane jako "bez czekania" oraz indywidualne marszruty technologiczne zadań.

Znalezienie cyklicznego harmonogramu dla podstawowego problemu przepły-

wowego jest stosunkowo łatwe. Z kolei, harmonogram cykliczny dla problemu przepływowego z ograniczeniem „bez czekania" można wyznaczyć w sposób dokładny tylko dla dwóch stanowisk (maszyn) [2], zaś dla większej liczby maszyn, ze względu na silną NP-trudność problemu i jego związek z TSP, polecana jest szeroka gama algorytmów dedykowanych dla asymetrycznego problemu komiwojażera (A-TSP).

Pomijanie maszyn w przepływowym problemie z ograniczeniem bez czekania znacznie komplikuje proces modelowania i projektowania algorytmów. Istotnie, niewiele jest publikacji, w których autorzy zmierzyli się z tym wyzwaniem. W pracy [3] rozważano przepływowy systemem produkcyjny z ograniczeniem bez czekania, możliwością pominięcia drugiej maszyny i z kryterium długości uszeregowania. Wobec NP-trudności problemu, ostatecznie w w/w pracy zaproponowano pewne algorytmy heurystyczne. Pierwszy algorytm szeregowania cyklicznego w systemie przepływowym z ograniczeniami "bez bufora/magazynu" pochodzi z publikacji [7]. Zaproponowano tam też pewną metodę wyznaczania minimalnego czasu cyklu w oparciu o przepływy w sieciach.

W nowoczesnych systemach produkcyjnych ukierunkowanych na realizację spersonalizowanych żądań klientów nie wszystkie etapy produkcyjne muszą być realizowane w całości, przykładowo produkt dostarczany jest do podwykonawcy w celu wykończenia zgodnego z życzeniem odbiorcy. W takim przypadku pojawia się pomijanie niektórych stanowisk obsługi (ang. system with missing operations). Chociaż modele tego typu pojawiają się w literaturze najczęściej w kontekście systemów przepływowych, to ich analiza jest prowadzona powszechnie przy pomocy narzędzi charakterystycznych dla problemów gniazdowych (ang. job-shop), zatem bardziej zaawansowanych.

2. Opis problemu oraz jego model matematyczny

W przepływowym systemie wytwarzania park maszynowy składa się z m maszyn ze zbioru $M = \{1, \ldots, m\}$, w którym należy wykonać n zadań ze zbioru $J = \{1, \ldots, n\}$. W systemie tego typu z możliwością pomijania maszyn, zadania wykonywane są na maszynach w kolejności zgodnej z ich numeracją przy czym mogą one nie być przetwarzane na niektórych maszynach, ogólnie różnych dla różnych zadań. Każde zadanie składa się z $n_i \leqslant m$ operacji, czas wykonania operacji zadania $i \in J$ na maszynie $k \in M$ wynosi $p_{ik} > 0$, w przypadku gdy zadanie pomija maszynę czas ten nie jest określony. Ograniczenie "bez czekania" oznacza, że wykonanie zadania na danej maszynie musi rozpocząć się niezwłocznie po zakończeniu wykonania zadania na maszynie poprzedniej (tj. dokładnie w momencie jego zakończenia). Wykonywania zadań na maszynach nie można przerywać, w tym samym czasie każda maszyna może wykonywać jedno zadanie i zadanie może być wykonywane tylko na jednej maszynie.

Zadania są wprowadzane do systemu w pewnej kolejności (nazywanej dalej kolejnością ładującą) π, $\pi \in \Pi$, gdzie Π jest zbiorem wszystkich permutacji określonych na zbiorze J i przetwarzane w tej kolejności na maszynach. Zadania wykonywane na tej samej maszynie muszą być wykonywane zgodnie z kolejnością π, tj. jeżeli pewne zadanie znajduje się przed innym w kolejności π, to na wszystkich maszynach, na których wykonywane są te zadania zasada ta obowiązuje. Pośrednio oznacza to, że jedyną zmienna decyzyjną w problemie może być π.

Oznaczmy przez $O = \{1, ..., o\}$, gdzie $o = \sum_{s=1}^{n} n_s$ zbiór wszystkich operacji. Operacje o indeksach $l_i + 1, \ldots, l_i + n_i$, gdzie $l_i = \sum_{s=1}^{i-1} n_s$ należą do zadania $i \in J$.

Operacja $j = l_i + k \in O$ wykonywana jest na maszynie $\mu_j \in M$ przez czas $p_j = p_{ik}$. Niech $\overline{t}_j$ ($\underline{t}_j$) będzie następnikiem (poprzednikiem) technologicznym operacji j, tj. $\overline{t}_j = j + 1$ oraz $\overline{t}_j = 0$ gdy operacja j jest ostatnią operacją pewnego zadania, natomiast $\underline{t}_j = j - 1$ oraz $\underline{t}_j = 0$ gdy operacja j jest pierwszą operacją pewnego zadania. W podobny sposób definiujemy następnika ($\overline{s}_j$) i poprzednika ($\underline{s}_j$) maszynowego operacji $i \in O$. Następnicy $\overline{s}_j$ i poprzednicy $\underline{s}_j$ wynikają z kolejności π.

Harmonogram wykonywania operacji na maszynach możemy opisać przy pomocy momentów rozpoczęcia $[S_j]_{1\times o}$ i zakończenia $[C_j]_{1\times o}$ ich wykonywania. Dla zadanej kolejności π wartości te muszą spełniać następujące ograniczenia

$$S_j \geqslant 0, \quad j = 1, \ldots, o, \tag{1}$$

$$C_j = S_j + p_j, \quad j = 1, \ldots, o, \tag{2}$$

$$S_j \geqslant C_{\underline{s}_j} \quad \underline{s}_j \neq 0, \; j = 1, \ldots, o, \tag{3}$$

$$S_j = C_{\underline{t}_j} \quad \underline{t}_j \neq 0, \; j = 1, \ldots, o. \tag{4}$$

Ograniczenie (1) "lokuje" harmonogram na dodatniej półosi czasu, równanie (2) wiąże moment zakończenia operacji z momentem jej rozpoczęcia. Ograniczenie (3) oznacza, że rozpoczęcie wykonywania operacji na maszynie może nastąpić dopiero po zakończeniu wykonywania operacji poprzednio wykonywanej na tej maszynie. Ostatnie ograniczenie modeluje wymaganie bez czekania, gdzie moment rozpoczęcia wykonywania operacji równy jest momentowi zakończenia wykonywania poprzedniej operacji technologicznej.

W cyklicznym systemie produkcyjnym zbiór zadań wykonywany jest wielokrotne w tzw. cyklach produkcyjnych. W każdym cyklu zadania wykonywane są na tych samych maszynach, w takiej samej kolejności i zgodnie z tym samym harmonogramem. Harmonogram wykonywania zadań w cyklach przesunięty jest o wielokrotność pewnego okresu T. Minimalny okres dla którego spełnione są ograniczenia technologiczne i ograniczenia związane z kolejnością wykonywania operacji na maszynach w ramach cykli oraz pomiędzy nimi będziemy nazywali czasem cyklu i oznaczali symbolem $T(\pi)$. Innymi słowy w harmonogramie cyklicznym zachodzi

$$S_j^x = S_j^1 + (x-1)T \quad j = 1, \ldots, o, \; x = 1, \ldots. \tag{5}$$

gdzie S_j^x jest momentem rozpoczęcia operacji $j \in O$ w x-tym cyklu zaś $T \geqslant T(\pi)$.

Czas cyklu dla ustalonej kolejności wykonywania zadań produkcyjnych można wyznaczyć różnymi metodami: używając mieszanego programowania całkowitoliczbowego [1], poprzez maksymalny przepływ w sieci [7], metodą Howard'a [6]. Każda z metod ma inną złożoność obliczeniową i wymaga specyficznego sposobu modelowania. Z analizy teoretycznej oraz z rezultatów badań eksperymentalnych dla wielu problemów szeregowania zadań wynika, że wszystkie w/w metody są znacznie mniej efektywne czasowo od metody opartej na analizie długości najdłuższych dróg w odpowiednio skonstruowanym grafie skierowanym. Podejście to zostało zastosowane, między innymi, do wyznaczenia czasu cyklu w otwartym systemie obsługi zadań produkcyjnych [11], ang. open-shop scheduling problem.

3. Model grafowy i własności problemu

Przed przystąpieniem do zaprezentowania modelu grafowego przekształcimy ograniczenia (1)–(5) do następującej postaci

$$S_j \geqslant 0, \;\; j = 1, \ldots, n, \tag{6}$$

$$S_j \geqslant S_{\underline{s}_j} + p_{\underline{s}_j}, \;\; \underline{s}_j \neq 0, \;\; j = 1, \ldots, n, \tag{7}$$

$$S_j \geqslant S_{\underline{t}_j} + p_{\underline{t}_j} \;\; \underline{t}_j \neq 0, \;\; j = 1, \ldots, n, \tag{8}$$

$$S_j \geqslant S_{\overline{t}_j} + p_{\overline{t}_j} - p_{\overline{t}_j} - p_j, \;\; \overline{t}_j \neq 0, \;\; j = 1, \ldots, n. \tag{9}$$

W układzie nierówności (6)–(9) usunięto terminy zakończenia operacji korzystając z (2) oraz ograniczenie (4) zastąpiono dwoma ograniczeniami (8) oraz (9).

Dla kolejności π konstruujemy elementarny graf skierowany $G(\pi) = (O, E(\pi))$ ze zbiorem węzłów O oraz zbiorem łuków $E(\pi) \subset O \times O$. Węzeł $i \in O$ reprezentuje operację i oraz jest obciążony wagą p_i. Zbiór łuków $E(\pi) = P \cup R \cup Q(\pi)$ składa się z trzech podzbiorów, określonych odpowiednio:

$$P = \{(j, \overline{t}_j) : \; \overline{t}_j \neq 0, \; j \in O\}, \tag{10}$$

gdzie łuki ze zbioru P obciążone są wagą 0, modelują ograniczenia technologiczne i odpowiadają nierówności (8), dalej

$$R = \{(\underline{t}_j, j) : \; \underline{t}_j \neq 0, \; j \in O\}, \tag{11}$$

gdzie każdy łuk $(\underline{t}_j, j) \in R$ modeluje ograniczenie bez czekania, obciążony jest wagą $-p_{\overline{t}_j} - p_j$ i odpowiada ograniczeniu (9), zaś

$$Q(\pi) = \{(j, \overline{s}_j) : \; \overline{s}_j \neq 0, \; j \in O\}, \tag{12}$$

gdzie łuki ze zbioru $Q(\pi)$ obciążone są wagą 0 i modelują ograniczenia wynikające z kolejności wykonywania zadań na maszynach odpowiadające nierównościom (7). Zauważmy, że graf $G(\pi)$ jest właściwy do wyznaczenia długości uszeregowania dla pojedynczego cyklu w tzw. wsadowym modelu szeregowania.

Na bazie grafu $G(\pi)$ konstruujemy rodzinę grafów $G^*(\pi, k)$ z parametrem k, reprezentujących k początkowych cykli produkcyjnych dla $x = 1, 2, \ldots, k$, które będziemy nazywali k-krotnym cyklicznym rozszerzeniem $G(\pi)$ lub krótko rozszerzeniem cyklicznym. Graf $G^*(\pi, k)$ jest połączeniem k kopi grafu $G(\pi)$ (każdy z nich reprezentuje jeden cykl) przy wykorzystaniu łuków modelującymi ograniczenia maszynowe pomiędzy ostatnimi operacjami wykonywanymi na wszystkich maszynach w pewnym cyklu, a odpowiadającymi im pierwszymi operacjami na tych maszynach w następnym cyklu. Opiszemy konstrukcję tej rodziny bardziej precyzyjnie. Graf

$$G^*(\pi, k) = (O^*, E^*(\pi) \cup W^*(\pi)), \tag{13}$$

posiada zbiór węzłów

$$O^* = \bigcup_{s=1}^{k} O^s, \;\; O^s = \{j^s : j \in O\}, \tag{14}$$

gdzie O^s jest $s-$tą kopią zbioru O. Zauważmy, że operacja j w s-tym cyklu jest oznaczana przez j^s. Analogicznie definiujemy zbiór łuków

$$E^*(\pi) = \bigcup_{s=1}^{k} E^s(\pi). \tag{15}$$

Niech a_l (b_l) będzie ostatnią (pierwszą) operacją wykonywaną na maszynie $l \in M$. Zbiór łuków łączących kopie grafów zdefiniowany jest następująco

$$W^*(\pi) = \bigcup_{s=1}^{k-1} W^s(\pi), \;\; W^s(\pi) = \{(a_l^s, b_l^{s+1}) : l \in M\}, \tag{16}$$

przy czym łuki ze zbioru $W^*(\pi)$ nie są obciążone.

Niech $L(i,j)$ oznacza długość najdłuższej drogi od węzła i do węzła j, gdzie $i,j \in O^*$ w grafie $G^*(\pi,k)$. Dla dowolnej operacji $j \in O$ odstęp czasowy pomiędzy momentem jej rozpoczęcia w pierwszym cyklu i k-tym cyklu nie może być mniejszy niż $L(j^1, j^k)$, tj.

$$S_{j^k} - S_{j^1} \geqslant L(j^1, j^k). \tag{17}$$

Podobnie jak w pracy [11] można pokazać, że

$$T(\pi) = \max_{y=2,\ldots,m} \max_{1 \leqslant k \leqslant m} \frac{L_{a_l^{(1)}, a_l^{(r)}}}{y-1} \tag{18}$$

Wyznaczenie $T(\pi)$ w oparciu o wyrażenie (18) ma złożoność obliczeniową $O(om^2)$.

Dowolnie wybraną ścieżkę dla której wyrażenie (18) przyjmuje wartość maksymalną będziemy nazywali ścieżką krytyczną. Maksymalna podsekwencja tej ścieżki zawierająca węzły reprezentujące operacje wykonywane na tej samej maszynie będziemy nazywali blokiem. W każdym bloku wyróżniamy pierwszą i ostatnią operację; pozostałe operacje nazywane są operacjami wewnętrznymi i tworzą tzw. wewnętrzny blok.

Własność 1. *Niech π' będzie nową kolejnością ładującą otrzymaną z π. Jeśli $T(\pi') < T(\pi)$ to co najmniej jedna operacja z co najmniej jednego bloku wewnętrznego jest wykonywana przed pierwszą lub za ostatnią operacją tego bloku.*

Własność ta jest nietrywialnym rozszerzeniem tzw. własności blokowej, m.in. [5], [9].

4. Przykład

W systemie produkcyjnym składającym się z $m = 3$ maszyn należy wykonać $n = 4$ zadania. Czasy wykonania zadań oraz maszyny, na których są wykonywane, podano w tabeli 1. Obliczenia prowadzące do otrzymania $T(\pi)$ przedstawiono w tabeli 2 zaś na rysunku 1 zilustrowano harmonogram cykliczny. Dodatkowymi liniami zobrazowano przebieg ścieżki krytycznej, która obejmuje dwa cykle. $T(\pi) = 13$.

5. Algorytm optymalizacyjny TS

Algorytm optymalizacyjny dla rozważanego problemu oparto na jednej z najefektywniejszych metod konstruowania algorytmów przeszukujących przestrzeń rozwiązań

Tabela 1

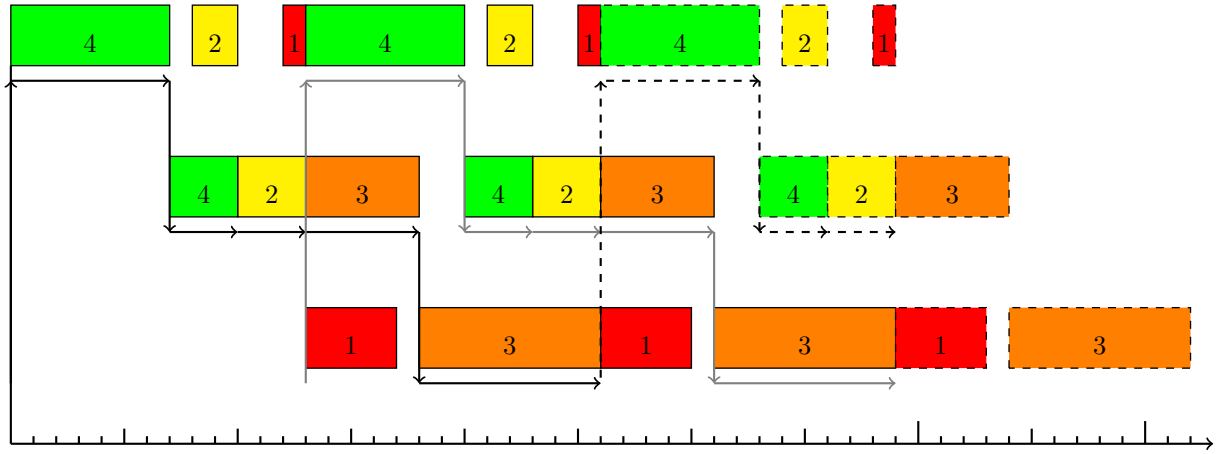
Czasy wykonywania zadań na maszynach

	zadanie			
maszyna	1	2	3	4
1	1	2	-	7
2	-	3	5	3
3	4	-	8	-

Tabela 2

Wyznaczanie $T(\pi)$ poprzez ścieżki w grafie $G^*(\pi, k)$

		Cykl 1				Cykl 2				Cykl 3			
maszyna		4	2	1	3	4	2	1	3	4	2	1	3
1	S	0	8	12	-	13	21	25	-	26	34	38	-
	C	7	10	13	-	20	23	26	-	33	36	39	-
2	S	7	10	-	13	20	23	-	26	33	36	-	39
	C	10	13	-	18	23	26	-	31	36	39	-	44
3	S	-	-	13	18	-	-	26	31	-	-	39	44
	C	-	-	17	26	-	-	30	39	-	-	43	52



Rys. 1. Cykliczny harmonogram wykonywania zadań produkcyjnych

jakim jest przeszukiwanie z zabronieniami [4]. Ze względu na duże podobieństwo sposobu modelowania grafowego oraz własności problemu zdecydowano się na zmodyfikowanie jednego z najlepszych algorytmów opartych na tej metodzie dla klasycznego problemu przepływowego [9], [5]. Działanie proponowanego algorytmu składa się z dwóch etapów: (*i*) wyznaczenie najlepszego rozwiązania z funkcją celu $\min_\pi C_{\max}(\pi)$, (*ii*) wyznaczenie najlepszego rozwiązania z funkcją celu $\min_\pi T(\pi)$. Najlepsze rozwiązanie otrzymane w pierwszej fazie było rozwiązaniem początkowym dla fazy drugiej. W opisie algorytmu ograniczymy się do najistotniejszych jego komponentów.

5.1. Sąsiedztwo

Proces optymalizacyjny w algorytmie TS opiera się na przeszukiwaniu lokalnego sąsiedztwa bieżącego rozwiązania. Z tego sąsiedztwa wybiera się najlepsze niezabronione rozwiązanie, które staje się rozwiązaniem bazowym w kolejnej iteracji. Rozwiązania

sąsiednie generowane są z bazowego rozwiązania przez jego modyfikację. Modyfikację (ruch) polegający na usunięciu zadania z pozycji x w permutacji ładującej π i wstawieniu na pozycję y ($y \neq x$) będziemy nazywali ruchem wstaw (ang. Insert) i oznaczali parą $v = (x, y)$. Permutację ładującą otrzymaną z π przez modyfikację v będziemy oznaczali symbolem π_v. Zbiór ruchów $I(\pi)$ zdefiniujemy następująco:

$$I(\pi) = \{(x, y) : \ y \neq x, \ x, y = 1, \cdots, n\} \tag{19}$$

Zbior ten można zmniejszyć eliminując ruchy nieperspektywiczne tj. ruchy, o których wiemy a priori, że nie spowodują poprawy wartości funkcji celu. Precyzyjniej, zbiór ruchów perspektywicznych $B(\pi)$ jest podzbiorem zbioru $I(\pi)$ składającym się z ruchów spełniających warunki rozszerzonego twierdzenia blokowego, patrz Własność 1.

5.2. Pozostałe elementy algorytmu

Do realizacji mechanizmu tabu została użyta cykliczna lista TL o długości L. W przypadku wykonania ruchu $v = (x, y)$ (podobnie jak w pracy [9]) do listy TL dodawane jest: $(\pi(x), \pi(x + 1))$ gdy zadanie przesuwane jest w prawo ($y > x$) oraz $(\pi(x - 1), \pi(x))$ gdy zadanie przesuwane jest w lewo ($y < x$). Ruch $v = (x, y)$ jest zabroniony, jeżeli istnieje w TL element (a, b) taki, że zadanie a znajduje się przed zadaniem b w π_v. Algorytm kończy działanie po wykonaniu $maxiter$ iteracji.

6. Badania eksperymentalne

Głównym celem badań była praktyczna ocena jakości oraz czasu działania zaproponowanych algorytmów i ich wariantów. Zostały zaimplementowane 3 algorytmy typu TS, mianowicie: TS(I+I), TS(I+B), TS(B+B). Algorytmy te różnią się otoczeniami wykorzystywanymi w pierwszej i drugiej fazie. Algorytmy zostały zakodowane w C++ i były testowane na komputerze z procesorem I7, 2.4 GHz.

Badania testowe przeprowadzono na pierwszych 5-ciu grupach przykładów Taillard'a [12]. Na potrzeby rozważanego problemu, w każdym zadaniu usunięto jedną operację (dokładniej, w i-tym zadaniu usunięto operację $(i - 1) \bmod m + 1$). Algorytmy TS uruchomiono z rozwiązania startowego otrzymanego algorytmem NEH [8], który jest jednym z najlepszych algorytmów konstrukcyjnych dla problemu przepływowego, w szczególności funkcji kryterialnej $C_{\max}$. W pierwszej fazie algorytm TS wykonywał $maxiter = 3000$ iteracji, natomiast w fazie drugiej $maxiter = 1000$ iteracji.

Dla każdego przykładu oraz każdego badanego algorytmu wyznaczono następujące wielkości :

- T - czas cyklu dla najlepszego rozwiązania wygenerowanego algorytmem,
- CPU - czas wykonania algorytmu.

Następnie obliczono $PRD = 100 \cdot (T - T^*)/T^*$ tj. względny procentowy błąd rozwiązania otrzymanego algorytmem, gdzie T^* jest najlepszym rozwiązaniem znalezionym dla danej instancji podczas badań. W tabeli 3 zebrano wartości średnie PRD algorytmów dla wszystkich grup przykładów testowych.

Z analizy rezultatów badań wynika, że algorytmy TS znacznie przewyższają swoją efektywnością algorytm NEH. Średni błąd algorytmu wynosi aż 7,8% podczas gdy algorytmu TS waha się od 0,29% do 0,35% w zależności od wersji. Porównując efektywność różnych wersji algorytmu TS można zauważyć, że zastosowanie otoczenia blo-

kowego w drugiej fazie zmniejsza średni błąd o 0,03% (z 0,35 do 0,32), natomiast w obu fazach o 0,06% (z 0,35 do 0,29).

Tabela 3

Średnie wartości PRD dla grup instancji

		$C_{\max}$		Czas cyklu T			
Grupa	$n \times m$	NEH	TS(I)	NEH	TS(I+I)	TS(I+B)	TS(B+B)
1-10	20×5	41,4	3,87	9,5	0,48	0,13	0,33
11-20	20×10	30,7	2,69	5,7	0,04	0,12	0,02
21-30	20×20	29,8	6,32	6,2	0,03	0,06	0,03
31-40	50×5	51,0	1,88	10,2	0,83	0,72	0,64
41-50	50×10	38,0	1,47	7,6	0,35	0,55	0,43
średnio		38,2	3,25	7,8	0,35	0,32	0,29

Tabela 4

Średni czas obliczeń

		$C_{\max}$	Czas cyklu T		
Grupa	$n \times m$	TS(I)	TS(I+I)	TS(I+B)	TS(B+B)
1-10	20×5	0,6	5,6	3,4	3,5
11-20	20×10	1,2	38,4	33,0	34,2
21-30	20×20	2,4	359,8	358,6	338,5
31-40	50×5	8,4	88,0	47,5	49,9
41-50	50×10	19,2	638,1	539,9	510,1
średnio		6,4	226,0	196,5	187,2

Drugą cechą pozytywną otoczenia blokowego jest zmniejszenie czasu obliczeń, które wynika z redukcji liczby przeglądanych rozwiązań. Średni czas działania najlepszej wersji algorytmu TS(B+B) wynosi 187 sekund i jest w przybliżeniu o 5% krótszy od wersji TS(I+B) i aż o 20% krótszy od wersji TS(I+I).

W wielu pracach naukowych poświęconych problemom cyklicznym, konstruuje się harmonogram cykliczny powielając harmonogram wyznaczony dla funkcji $C_{\max}$. Wynika to między innymi z faktu, że złożoność obliczeniowa wyznaczenia wartości funkcji $C_{\max}$ jest znacznie mniejsza niż w przypadku czasu cyklu, nawet po zastosowaniu prezentowanych w pracy własności. Średni czas działania algorytmu TS dla funkcji $C_{\max}$ wynosi 6,4 sekundy i jest przeszło 30 razy krótszy od najszybciej działającej wersji algorytmu TS dla kryterium czasu cyklu. Niestety jakość generowanych rozwiązań, w przypadku rozważanego problemu, jest znacznie gorsza. W 3 i 4 kolumnie tabeli 3 wyznaczono średni błąd algorytmu NEH i TS działających zgodnie z przedstawioną ideą. Średni błąd algorytmu NEH przekracza aż 38%, natomiast algorytmu TS 3%.

7. Zakończenie

Zgodnie z naszą wiedzą, zaproponowany algorytm jest pierwszym algorytmem przeszukiwań lokalnych dla rozważanego problemu. W takim wypadku nie można bezpośrednio porównać jego efektywności z innymi algorytmami, bowiem relacja pomiędzy kosztem obliczeń a jakością generowanych rozwiązań jest odmienna. O wysokiej

skuteczności zaproponowanego algorytmu może pośrednio świadczyć znaczna przewaga nad algorytmem NEH, będącego championem wśród metod konstrukcyjnych dla problemów przepływowych. Ponadto algorytmy autorów oparte na podobnej konstrukcji generują wysokiej jakości rozwiązania np. [10].

Zaproponowana metoda modelowania, pozwala nie tylko na efektywne wyznaczanie minimalnego czasu cyklu dla danej kolejności, ale także dostarcza własności szczególnych dla metody poszukiwań lokalnych. Zaprezentowane podejście może zostać rozszerzone na inne bardziej złożone problemy szeregowania cyklicznego.

Praca powstała w wyniku realizacji projektu badawczego DEC 2017/25/B/ST7/02181 finansowanego ze środków Narodowego Centrum Nauki.

LITERATURA

1. Brucker P., Kampmeyer T.: A general model for cyclic machine scheduling problems. Discrete Applied Mathematics, 156/13, 2008, 2561–2572.
2. Gilmore P.C. and Gomory R.E.: Sequencing a One State-Variable Machine: A Solvable Case of the Traveling Salesman Problem. Operations Research, 12/5, 1964, 655–679.
3. Glass C.A., Gupta J.N.D., Potts C.N.: Two-Machine No-Wait Flow Shop Scheduling with Missing Operations. Mathematics of Operations Research, 24/4, 1999, 911–924.
4. Glover F.: Tabu Search—Part I. ORSA Journal on Computing, 1/3, 1989, 190–206.
5. Grabowski J., Pempera J.: New block properties for the permutation flow shop problem with application in tabu search. Journal of the Operational Research Society, 52/2, 2001, 210–220.
6. Howard R.A.: Dynamic Programming and Markov Processes. Technology Press and Wiley, New York, NY, 1960.
7. McCormick S.T., Pinedo M.L., Shenker S., Wolf B.: Sequencing in an Assembly Line with Blocking to Minimize Cycle Time. Operations Research, 37/6, 1989, 925–935.
8. Nawaz M., Enscore E., Ham I.: A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. Omega, 11/1, 1983, 91–95.
9. Nowicki E., Smutnicki C.: A fast tabu search algorithm for the permutation flow-shop problem. European Journal of Operational Research, 91/1, 1996, 160–175.
10. Pan Q., Wang L., Zhao B.H.: An improved iterated greedy algorithm for the no-wait flow shop scheduling problem with makespan criterion. The International Journal of Advanced Manufacturing Technology, 38, 2008, 778–786.
11. Pempera J., Smutnicki C.: Open shop cyclic scheduling. European Journal of Operational Research, 269/2, 2018, 773–781.
12. Taillard E.: Some efficient heuristic methods for the flow shop sequencing problem. European Journal of Operational Research, 47/1, 1990, 65–74.