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

Mariusz MAKUCHOWSKI, Eugeniusz NOWICKI
Politechnika Wrocławska

ALGORYTM TABU DLA PROBLEMU GNIAZDOWEGO Z OPERACJAMI WIELOMASZYNOWYMI NIERÓWNOCZEŚNIE WYKORZYSTUJĄCYMI MASZYNY

Streszczenie. W pracy rozważa się problem gniazdowy z operacjami wielomaszynowymi, nierównocześnie wykorzystującymi maszyny. Przedstawia się jego model permutacyjno-grafowy. Następnie prezentuje się algorytm przybliżony typu tabu zwracając szczególną uwagę na definicję sąsiedztwa oraz atrybuty zapisywane na listę tabu. Przeprowadza się badania eksperymentalne zaproponowanego algorytmu na przykładach testowych powszechnie stosowanych w literaturze.

TABU SEARCH ALGORITHM FOR JOB SHOP MULTIMACHINE OPERATIONS PROBLEM WITH NON-SIMULTANEOUSLY USED MACHINES

Summary. The paper discusses the job shop multimachine operations problem with non-simultaneously used machines. Its permutation-graph model is shown. The algorithm based on a tabu search technique with a specific neighborhood definition and suitable attributes written to taboo list is presented. Moreover, the algorithm is examined on well-known literature examples.

1. Wprowadzenie

Jednym z istotniejszych zagadnień w teorii harmonogramowania jest problem gniazdowy z kryterium będącym momentem wykonania wszystkich zadań. Waga tego zagadnienia wynika z faktu, że już jego klasyczne sformułowanie modeluje szereg praktycznych procesów produkcyjnych. Różnorodne uogólnienia, takie jak: wprowadzenie czasów przezbrojeń, ograniczonej pojemności buforów, ograniczonej liczby palet, uwzględnienie transportu itp., powodują, że obszar jego stosowalności stale się powiększa. Jednym z ważnych kierunków uogólnień jest wprowadzenie operacji wielomaszynowych modelujących procesy, w których do wykonywania poszczególnych operacji zaangażowana

jest nie jedna (tak jak w modelu klasycznym), ale kilka maszyn, [4, 2]. Odpowiednie algorytmy konstrukcyjne oraz algorytmy popraw (oparte na technice tabu) zostały podane w [5, 1, 8, 7].

Dalsze uogólnienie operacji wielomaszynowych – wynikające z praktyki przemysłowej – polega na uwzględnieniu sytuacji, w których nie wszystkie maszyny zaangażowane w proces wykonywania danej operacji są jednocześnie wykorzystywane, patrz np. [9]. Co więcej, dla każdej zaangażowanej maszyny określony jest jeden przedział jej wykorzystywania oraz przedziały te są względem siebie ustalone. Ten typ operacji będziemy nazywać operacjami wielomaszynowymi, nierównocześnie wykorzystującymi maszyny. Pierwsze wyniki badań nad algorytmami konstrukcyjnymi dla tego typu operacji zamieszczono w [9, 6]. W tej pracy formułujemy permutacyjno-grafowy model badanego zagadnienia, a następnie podajemy algorytm typu „popraw" oparty na technice tabu. Wyczerpujące badania testowe pokazują, że proponowany algorytm poprawia rozwiązania dostarczone przez algorytmy konstrukcyjne o około 10%, w średnim czasie kilku minut na komputerach klasy PC.

2. Matematyczne sformułowanie problemu i model permutacyjno-grafowy

Dany jest zbiór maszyn $M = \{1, \ldots, |M|\}$, zbiór zadań $= \{1, \ldots, |J|\}$ oraz zbiór operacji $O = \{1, \ldots, |O|\}$. Zadanie $k \in J$ składa się z ciągu $o_k \geq 1$ operacji indeksowanych kolejno przez $j_k + 1, j_k + 2, \ldots, j_k + o_k$, gdzie $j_k = \sum_{i=1}^{k-1} o_i$. Wykonanie zadania polega na wykonaniu w powyższej kolejności wszystkich jego operacji. Dla każdej operacji $j \in O$ określony jest zbiór maszyn $M_j \subset M$, na których jest ona wykonywana. Proces wykonywania operacji j na maszynie $l \in M_j$ nazywamy czynnością i notujemy jako parę (j, l). Czynność (j, l) jest scharakteryzowana przez dwie wielkości: głowę $\tau_{j,l}$ i ciało $p_{j,l}$. Głowa $\tau_{j,l}$ określa, po jakim czasie od pewnej chwili czasowej t_j (nieustalonej ale identycznej dla wszystkich czynności danej operacji) ma się rozpocząć wykonywanie czynności (j, l) na maszynie l, zaś ciało $p_{j,l}$ jest długością trwania tej czynności; przyjmujemy, że przynajmniej jedna z wartości $\tau_{j,l}$, $l \in M_j$ jest równa zero. Wykonanie operacji polega na wykonaniu wszystkich jej czynności zgodnie z przedstawionymi ograniczeniami. Niech $S(j, l)$ oznacza moment rozpoczęcia wykonywania czynności (j, l). Wtedy $S(j, l) - \tau_{j,l} = const \stackrel{\text{def}}{=} t_j$ dla $l \in M_j$ oraz moment $S(j) = t_j$ jest traktowany jako moment rozpoczęcia wykonywania

operacji j, zaś $C(j) = \max_{l \in M_j}\{S(j,l) + p_{j,l}\}$ – jako moment zakończenia wykonywania tej operacji.

Podobnie jak w klasycznym problemie gniazdowym przyjmuje się, że: (i) żadna z maszyn nie może wykonywać jednocześnie kilku czynności, (ii) nie można wykonywać jednocześnie więcej niż jedną operację danego zadania oraz (iii) wykonywanie czynności na maszynie nie może być przerywane. Uszeregowanie dopuszczalne definiowane jest przez takie momenty rozpoczęcia wykonywania czynności $S(j,l) \geq 0$, $l \in M_j$, $j \in O$, że spełnione są powyższe ograniczenia. Problem polega na znalezieniu dopuszczalnego uszeregowania minimalizującego moment wykonania wszystkich operacji $\max_{j \in O} C(j)$.

Przejdziemy teraz do sformułowania modelu permutacyjno-grafowego. Jako zmienną decyzyjną przyjmujemy zestaw permutacji $\pi = (\pi_1, \pi_2 \ldots, \pi_m)$, gdzie $m = |M|$ oraz $\pi_l = (\pi_l(1), \ldots, \pi_l(|\pi_l|))$ jest permutacją zbioru czynności $\mathcal{O}_l = \{(j,l) : l \in M_j, j \in O\}$ wykonywanych na maszynie l, $|\pi_l| = |\mathcal{O}_l|$, $l \in M$; zbiór wszystkich takich zestawów, zwanych krótko permutacjami, oznaczamy przez Π. Dla permutacji $\pi \in \Pi$ definiujemy graf $G(\pi) = (\mathcal{O}, E^T \cup E^O \cup E^K(\pi))$ ze zbiorem obciążonych wierzchołków $\mathcal{O} = \{(j,l) : l \in M_j, j \in O\}$ reprezentujących poszczególne czynności, zbiorem nieobciążonych łuków technologicznych (wynikających z kolejności wykonywania operacji w zadaniu)

$$E^T = \bigcup_{k \in J} \bigcup_{j=j_k+2}^{j_k+o_k} \{((j-1, l^+_{j-1}), (j, l^-_j))\}, \tag{1}$$

kolejnościowych (wynikających z permutacji π)

$$E^K(\pi) = \bigcup_{l \in M} \bigcup_{i=2}^{|\pi_l|} \{(\pi_l(i-1), \pi_l(i))\} \tag{2}$$

oraz zbiorem obciążonych łuków operacyjnych (odzwierciedlających ustalone względne położenie czynności danej operacji)

$$E^O = \bigcup_{j \in O} \bigcup_{i=2}^{|M_j|} \{((j, m_j(i-1)), (j, m_j(i))), ((j, m_j(i)), (j, m_j(i-1)))\}, \tag{3}$$

gdzie $l^-_j = \arg\min_{l \in M_j} \tau_{j,l}$ oraz $l^+_j = \arg\max_{l \in M_j}(\tau_{j,l} + p_{j,l})$ oznacza maszynę, na której odpowiednio rozpoczyna się oraz kończy wykonywanie operacji j, zaś $M_j = \{m_j(1), m_j(2), \ldots, m_j(|M_j|)\}$, $j \in O$. Wierzchołek reprezentujący czynność (j,l) obciążony jest wartością $p_{j,l}$, zaś łuk $((j,a),(j,b)) \in E^O$ (z wierzchołka (j,a) do wierzchołka (j,b)) – wartością $w_{j,a,b} = \tau_{j,b} - \tau_{j,a} - p_{j,a}$.

Z definicji grafu $G(\pi)$ wynika, że zawiera on cykle; długość tych cykli może być dowolna (ujemna, zerowa lub dodatnia). W przypadku kiedy w grafie $G(\pi)$ nie ma cykli o dodatniej długości, niech $r^{\pi}((j,l))$ $(q^{\pi}((j,l)))$ oznacza wartość najdłuższej drogi dochodzącej do (wychodzącej z) wierzchołka (j,l), łącznie z wagą tego wierzchołka. Wtedy $S(j,l) = r^{\pi}((j,l)) - p_{j,l}$, $l \in M_j$, $j \in O$ jest uszeregowaniem dopuszczalnym. W dowodzie ograniczymy się tylko do pokazania, że $S(j,l) - \tau_{j,l} = const$ dla wszystkich czynności operacji j; pozostałe warunki wynikają z rozumowania analogicznego jak dla klasycznego problemu gniazdowego. Załóżmy, że dwie czynności (j,a) i (j,b) operacji j połączone są łukami operacyjnymi. Wtedy z zależności $S(j,b) \geq S(j,a)+p_{j,a}+w_{j,a,b} = S(j,a)+\tau_{j,b}-\tau_{j,a}$ oraz $S(j,a) \geq S(j,b)+p_{j,b}+w_{j,b,a} = S(j,b)+\tau_{j,a}-\tau_{j,b}$ wynika, że $S(j,a) = S(j,b)+\tau_{j,a}-\tau_{j,b}$, czyli $S(j,a) - \tau_{j,a} = S(j,b) - \tau_{j,b}$. Podobne rozumowanie dla pozostałych par czynności zadania j kończy dowód.

Zdefiniowany graf pozwala na proste sformułowanie badanego problemu. Niech $C_{\max}(\pi)$ oznacza długość najdłuższej ścieżki (ścieżka krytyczna) w grafie $G(\pi)$ dla permutacji $\pi \in \Pi$; w przypadku gdy graf $G(\pi)$ posiada cykle o dodatniej długości, przyjmujemy, że $C_{\max}(\pi) = \infty$. Rozważany problem polega na znalezieniu takiej permutacji $\pi^* \in \Pi$, że odpowiadający jej graf $G(\pi^*)$ ma najmniejszą długość ścieżki krytycznej $C_{\max}(\pi^*)$.

3. Algorytm popraw typu tabu

Proponowany algorytm (zwany dalej algorytmem TMN) bazuje na technice tabu, której ogólny opis można znaleźć np. w pracy [3]. W poniższych podrozdziałach przedstawiamy szczegółowo najważniejsze elementy składowe tego algorytmu.

3.1. Sąsiedztwo

Sąsiedztwo permutacji $\pi \in \Pi$ wykorzystywane w algorytmie TMN bazuje na ruchach typu wstaw. Niech $z^{\pi,j} = (z_1^{\pi,j}, \ldots, z_m^{\pi,j})$ oznacza zestaw pozycji czynności operacji j w π, gdzie $\pi_l(z_l^{\pi,j}) = (j,l)$ dla $l \in M_j$; dla prostoty oznaczeń, niewykonywanie na maszynie $l \notin M_j$ czynności operacji j notuje się przez położenie $z_l^{\pi,j} = 0$. Niech

$$Z^{\pi,j} = \{z = (z_1, \ldots, z_m) : 1 \leq z_l \leq |\pi_l|,\ l \in M_j,\ z_l = 0,\ l \notin M_l\} \tag{4}$$

oznacza zbiór wszystkich zestawów pozycji, jakie mogą zajmować czynności operacji j w permutacji π. Zachodzi $z^{\pi,j} \in Z^{\pi,j}$ oraz $|Z^{\pi,j}| = \prod_{l \in M_j} |\pi_l|$. Ruch $v = (j, z)$ typu wstaw polega na przesunięciu wszystkich czynności operacji j na pozycje określone przez zestaw $z \in Z^{\pi,j} \setminus \{z^{\pi,j}\}$; otrzymaną permutację oznaczamy przez $\pi_{(v)}$. Dokładniej, odbywa się to w trzech etapach: (i) pobranie czynności (j, l) z pozycji $w = z_l^{\pi,j}$, (ii) przesunięcie czynności $\pi_l(w+1), \ldots, \pi_l(z_l)$ o jedną pozycję w lewo, jeżeli $w < z_l$ lub czynności $\pi_l(z_l), \ldots, \pi_l(w-1)$ o jedną pozycję w prawo, jeżeli $w > z_l$ oraz (iii) wstawienie czynności (j, l) na zwolnioną pozycję z_l; $l \in M_j$. Zbiór wszystkich ruchów typu wstaw dla operacji j oznaczamy przez $V(\pi, j) = \{v = (j, z) : z \in Z^{\pi,j} \setminus \{z^{\pi,j}\}\}$. Jest oczywiste, że jeżeli żadna czynność operacji j nie leży na ścieżce krytycznej w grafie $G(\pi)$, to dla każdego ruchu $v = (j, z) \in V(\pi, j)$ zachodzi $C_{\max}(\pi_{(v)}) \geq C_{\max}(\pi)$. Oznacza to, że należy się ograniczyć do przesuwania czynności tylko tych operacji j, które należą do zbioru $\Omega \subset O$ zawierającego wszystkie operacje, których przynajmniej jedna z czynności leży na ścieżce krytycznej w grafie $G(\pi)$. Ostatecznie zbiór ruchów ma postać $V(\pi) = \bigcup_{j \in \Omega} V(\pi, j)$; $|V(\pi)| = \sum_{j \in \Omega} (\prod_{l \in M_j} |\pi_l| - 1)$. Zbiór ten generuje sąsiedztwo $\mathcal{N}(\pi) = \{\pi_{(v)} : v \in V(\pi)\}$.

Ze względu jednak na zbyt dużą liczność zbioru ruchów $V(\pi, j)$ ograniczamy się do pewnego jego podzbioru $VB(\pi, j) \subset V(\pi, j)$. Niech γ jest permutacją powstałą po usunięciu czynności operacji j z π; graf $G(\gamma)$ otrzymujemy z $G(\pi)$ przez usunięcie łuków $(\pi_l(z_l^{\pi,j} - 1), \pi_l(z_l^{\pi,j}))$, $(\pi_l(z_l^{\pi,j}), \pi_l(z_l^{\pi,j} + 1))$ i dodanie luku $(\pi_l(z_l^{\pi,j} - 1), \pi_l(z_l^{\pi,j} + 1))$ (jeżeli $z_l^{\pi,j} \in \{1, |\pi_l|\}$, to tylko jeden łuk jest usuwamy i żaden nie jest dodawany), $l \in M_j$. Dla każdego zestawu pozycji $z \in Z^{\pi,j}$ definiujemy wartość $h(z) = h_1(z) + h_2(z)$, $h_1(z) = \max\{A, \max_{l \in M_j} R_l(z_l - 1)\}$, $h_2(z) = \max\{B, \max_{l \in M_j} Q_l(z_l)\}$, gdzie

$$A = r^{\gamma}((j^-, l_{j^-}^+)), \quad R_l(z_l - 1) = r^{\gamma}(\gamma_l(z_l - 1)) - \tau_{j,l}, \tag{5}$$

$$B = \tau_{j,l_j^+} + p_{j,l_j^+} + q^{\gamma}((j^+, l_{j^+}^-)), \quad Q_l(z_l) = \tau_{j,l} + p_{j,l} + q^{\gamma}(\gamma_l(z_l)) \tag{6}$$

oraz j^- (j^+) jest poprzednikiem (następnikiem) technologicznym operacji j; jeżeli operacja j nie ma poprzednika (następnika) technologicznego, to $j^- = 0$ ($j^+ = 0$); $l_0^- = l_0^+ = 0$, $\gamma_l(0) = \gamma_l(|\pi_l|) = (0, 0)$, $l \in M_j$, $r^{\gamma}((0, 0)) = q^{\gamma}((0, 0)) = 0$. Łatwo zauważyć, że $h(z)$ jest dolnym ograniczeniem wartości najdłuższej drogi w grafie $G(\pi_{(v)})$, $v = (j, z)$, zawierającej co najmniej jeden wierzchołek odpowiadający czynności operacji j. Oznacza to, że $\max\{C_{\max}(\gamma), h(z)\}$ jest dolnym ograniczeniem wartości $C_{\max}(\pi_{(v)})$, $v = (j, z)$.

Z definicji $r^\gamma(k)$, $q^\gamma(k)$, $k \in \mathcal{O}$ wynika, że $R_l(1) < R_l(2) < \ldots < R_l(|\gamma_l|)$ oraz $Q_l(1) > Q_l(2) > \ldots > Q_l(|\gamma_l|)$, $l \in M_j$. Niech a_l oznacza najmniejszą liczbę taką, że $1 \leq a_l \leq |\gamma_l|$, $R_l(a_l) > A$, jeżeli $R_l(|\gamma_l|) > A$ lub $a_l = |\gamma_l|+1$ – w przeciwnym wypadku. Niech b_l oznacza najmniejszą liczbę taką, że $1 \leq b_l \leq |\gamma_l|$, $Q_l(b_l) \leq B$, jeżeli $Q_l(|\gamma_l|) \leq B$ lub $b_l = |\gamma_l| + 1$ – w przeciwnym wypadku. Zestaw pozycji $z^B = (z_1^B, \ldots, z_m^B) \in Z^{\pi,j}$ nazywamy zestawem *bazowym*, jeżeli

$$a_l \leq z_l^B \leq c_l \stackrel{\text{def}}{=} \max\{a_l, b_l\}, \quad l \in M_j \quad \text{oraz} \quad \max_{l \in M_j} \hat{R}_l(z_l^B - 1) < \min_{l \in M_j} \hat{R}_l(z_l^B); \tag{7}$$

$\hat{R}_l(x) = 0$ dla $x < a_l$, $\hat{R}_l(x) = R_l(x)$ dla $a_l \leq x < c_l$ oraz $\hat{R}_l(x) = \infty$ dla $x = c_l$. Zbiór wszystkich takich zestawów oznaczamy przez $ZB^{\pi,j}$. Jest oczywiste, że dla każdego zestawu pozycji $z \in Z^{\pi,j}$ istnieje zestaw bazowy $z^B \in ZB^{\pi,j}$ taki, że $h_1(z^B) = h_1(z)$ oraz $h_2(z^B) \leq h_2(z)$, czyli $h(z^B) \leq h(z)$. Dlatego też proponujemy zbiór ruchów $V(\pi, j)$ ograniczyć do $VB(\pi, j) = \{v = (j, z) : z \in ZB^{\pi,j} \setminus \{z^{\pi,j}\}\}$. Liczność tego zbioru jest nie większa niż $\sum_{l \in M_j} |\pi_l|$, co oznacza istotne zmniejszenie w porównaniu z licznością $V(\pi, j)$.

W proponowanym algorytmie TMN dla każdej operacji $j \in \Omega$ wyznaczamy ruch $v_j^R = \arg\min\{C_{\max}(\pi_{(v)}) : v \in VB(\pi, j)\}$ zwany reprezentantem operacji j. Ostatecznie rozpatrywane sąsiedztwo permutacji π ma postać $\mathcal{N}^{TMN}(\pi) = \{\pi_{(v_j^R)} : j \in \Omega\}$.

3.2. Akceleratory wyboru reprezentanta

Podstawowa idea akceleratorów wyboru reprezentanta v_j^R polega na takim wykorzystaniu własności problemu, aby nie wyznaczając wartości $C_{\max}(\pi_{(v)})$ dla *wszystkich* ruchów $v \in VB(\pi, j)$, określić go dokładnie. W tym celu proponujemy wykorzystać jednocześnie dwa następujące sposoby akceleracji.

Istotą pierwszego sposobu jest całkowita rezygnacja z wyznaczenia reprezentanta v_j^R, jeżeli wartość $C_{\max}(\gamma)$ jest nie mniejsza niż wartość funkcji celu dla reprezentanta innej operacji, który nie jest tabu.

Drugi sposób polega na tym, że przed obliczeniem dla danej pozycji bazowej $z \in ZB^{\pi,j}$ wartości funkcji celu $C_{\max}(\pi_{(v)})$, $v = (j, z)$, wyznaczamy wartość dolnego ograniczenia $LB = h(z)$. Jeżeli wartość LB jest nie mniejsza niż obliczona już wartość funkcji celu: (*i*) dla innego zestawu bazowego operacji j lub (*ii*) dla reprezentanta innej operacji, który nie jest tabu, to taki zestaw bazowy jest opuszczany bez wyliczania $C_{\max}(\pi_v)$. Badania testowe pokazały, iż LB ma bardzo dobre własności, mianowicie dla zdecydowanej

większości instancji dolne ograniczenie LB jest równe wartości funkcji celu. Poza tym złożoność obliczeniowa wyznaczenia LB dla operacji j wynosi $O(|M_j|)$, podczas gdy dokładne obliczenie wartości funkcji celu ma złożoność $O((\sum_{i \in O} |M_i|)^2)$.

3.3. Postać listy tabu

Elementem listy tabu T, zgodnie z ideą zawartą w [7], są atrybuty permutacji i ruchu. Dokładniej, jest to łuk określający kolejność pomiędzy pewną parą czynności. Po wykonaniu ruchu $v = (j, z)$ dla permutacji π na listę T zapisywany jest zbiór $\mathcal{AV}(\pi, v) = \bigcup_{l \in M_j} \mathcal{AV}_l(\pi, v)$ zawierający od 1 do $|M_j|$ łuków, gdzie:

$$\mathcal{AV}_l(\pi, v) = \begin{cases} \{((j,l), \pi_l(z_l^{\pi,j} + 1))\}, & z_l^{\pi,j} < z_l, \\ \{(\pi_l(z_l^{\pi,j} - 1), (j,l))\}, & z_l^{\pi,j} > z_l, \\ \emptyset, & z_l^{\pi,j} = z_l. \end{cases} \tag{8}$$

Ruch $v = (j, z)$ uważamy za tabu, jeżeli przesunięcie każdej czynności wchodzącej w skład operacji j jest tabu. Przesunięcie czynności (j, l) w prawo jest tabu wtedy i tylko wtedy, gdy na liście T znajduje się przynajmniej jeden łuk (j_1, j_2) taki, że $j_1 = \pi_l(i)$ dla pewnego $z_l^{\pi,j} < i \leq z_l$ oraz $j_2 = (j, l)$. Symetrycznie przesunięcie w lewo jest tabu, gdy na liście T znajduje się przynajmniej jedna para operacji (j_1, j_2) taka, że $j_1 = (j, l)$ oraz $j_2 = \pi_l(i)$ dla pewnego $z_l \leq i < z_l^{\pi,j}$. W algorytmie zastosowaliśmy efektywne określanie statusu ruchu zgodnie z ideą z [7]; obliczenia wstępne mają złożoność $O(max\{|M| \cdot |O|, |T|\})$, zaś sprawdzenie, czy przesunięcie pojedynczej czynności jest tabu, ma złożoność $O(1)$.

3.4. Dodatkowe techniki poprawiające jakość algorytmu TMN

Algorytm TMN został wzbogacony o detektor cyklu oraz o metodę skoku powrotnego zapewniającą dywersyfikacje poszukiwań [7]. Idea tej metody polega na tym, że jeżeli po upływie ustalonej liczby kroków TMN nie znajduje lepszego rozwiązania lub zostaje wykryty cykl, to algorytm powraca do ostatniego najlepszego rozwiązania, lista tabu jest zerowana, zbiór reprezentantów pomniejszony i proces poszukiwań jest kontynuowany.

3.5. Rozwiązanie początkowe

Do generacji rozwiązania początkowego posłużyliśmy się algorytmem konstrukcyjnym IN wykorzystującym tzw. technikę wstawień, który jest uogólnieniem odpowiedniego

algorytmu z [7]. Zrezygnowaliśmy z bardzo szybkich algorytmów konstrukcyjnych przedstawionych w [6] i stosujących reguły priorytetowe. Algorytm IN generował uszeregowania o 10,2% lepsze niż najlepszy zaproponowany tam algorytm i o 3,6% lepsze niż najlepsze spośród uszeregowań uzyskanych wszystkimi proponowanymi tam algorytmami.

4. Analiza eksperymentalna

Przykłady testowe zostały wygenerowane na bazie 120 przykładów z pracy [1]. Oryginalne dane zostały zredukowane do jednego sposobu wykonywania, po czym odpowiednio rozbudowane o parametry $\tau_{j,l}$ oraz $p_{j,l}$, $l \in M_j$, $j \in O$.

Przykłady testowe podzielone są na 3 klasy (Rdata, Mdata, Vdata), po 40 instancji. Klasy różnią się między sobą średnią oraz maksymalną liczbą czynności jednej operacji; wielkości te wynoszą odpowiednio: 2 i 3 dla Rdata, $\frac{1}{3}|M|$ i $\frac{3}{5}|M|$ dla Mdata oraz $\frac{1}{2}|M|$ i $\frac{4}{5}|M|$ dla Vdata. Instancje w każdej z klas tworzą grupy charakteryzujące się jednakową liczbą maszyn $|M|$, zadań $|J|$ oraz operacji $|O|$; dokładne dane przedstawione są w tablicy 1. Przykłady są bardzo mocno zróżnicowane pod względem rozmiaru; liczba czynności zmienia się od 77 do 1722.

Tablica 1

Parametry grup przykładów testowych

Grupa przykładów	$\|M\|$	$\|J\|$	$\|O\|$	Grupa przykładów	$\|M\|$	$\|J\|$	$\|O\|$
01 – 05	5	10	50	21 – 25	10	15	150
06 – 10	5	15	75	26 – 30	10	20	200
11 – 15	5	20	100	31 – 35	10	30	300
16 – 20	10	10	100	36 – 40	15	15	225

Algorytm TMN oceniamy na podstawie jakości wygenerowanych rozwiązań w odniesieniu do rozwiązań startowych dostarczonych przez algorytm konstrukcyjny IN. Dla każdego przykładu obliczamy względną procentową poprawę

$$\rho = 100\% \cdot (C^{IN} - C^{TMN})//C^{IN},$$

gdzie C^{IN} jest wartością funkcji celu dla rozwiązania początkowego wygenerowanego przez algorytm IN, zaś C^{TMN} – dla rozwiązania otrzymanego algorytmem TMN. Algorytm TMN był uruchamiany z następującymi parametrami: długość listy tabu - $6|M|$, maksymalna liczba iteracji - 1 000, liczba iteracji bez poprawy rozwiązania, której osiągnięcie powoduje skok powrotny - 250. Wyniki przedstawiono w tablicy 2.

Tablica 2

Wyniki testów algorytmu TMN dla poszczególnych grup przykładów

Grupa przykładów	Średnia względna poprawa ρ		
	Rdata	Mdata	Vdata
01 – 05	7,6	10,8	7,1
06 – 10	7,8	10,2	8,4
11 – 15	7,4	6,2	6,0
16 – 20	15,4	11,1	8,7
21 – 25	15,8	11,2	8,2
26 – 30	13,7	8,5	8,7
31 – 35	12,4	8,3	4,6
36 – 40	14,7	8,6	6,2
01 – 40	11,8	9,4	7,2

Z analizy tej tablicy wynika, że algorytm TMN poprawia rozwiązania dostarczone przez algorytm konstrukcyjny IN średnio od 7,2% do 11,8%; im mniejsza jest liczba czynności w operacji, tym poprawa jest większa. Czas obliczeń na komputerze Duron (900MHz) waha się od kilku sekund do kilkunastu minut, z wyjątkiem grupy przykładów 36-40 klasy Vdata, dla których czas obliczeń był rzędu 60 minut. Tak duży czas obliczeń (dla tej ostatniej grupy przykładów) w odniesieniu do czasu pracy algorytmu z pracy [7] dla operacji wielomaszynowych z równoczesnym wykorzystaniem wszystkich maszyn wynika przede wszystkim ze wzrostu liczby węzłów w grafie $G(\pi)$ (klasycznie wierzchołek reprezentował jedną operację, a tutaj reprezentuje jedną czynność). Dodatkowo, w sytuacji klasycznej graf ten dla permutacji dopuszczalnej nie zawierał cykli, zaś tutaj zawiera cykle o niedodatniej długości.

Reasumując, wstępne wyniki badań są bardzo zachęcające. Jednakże algorytm TMN wymaga jeszcze dalszych badań, w szczególności przyszłe prace powinny dotyczyć przyśpieszenia jego pracy. Wydaje się, że efekt ten jest możliwy do uzyskania przez opracowanie specjalistycznych metod wyznaczania wartości najdłuższej ścieżki w grafie $G(\pi)$ posiadającym cykle o niedodatniej długości.

LITERATURA

1. Brucker P., Neyer J.: Tabu-Search for Multi-Mode Job-Shop Problem, OR Spectrum, 20, 1998, pp. 21-28.
2. Drozdowski M.: Scheduling multiprocessor task - An Overview, European Journal Operational Research, 94, 1996, pp. 215-230.
3. Glover F., Laguna M.: Tabu Search. Kluwer Academic Publishers, Massachusetts USA, 1997.
4. Grabowski J.: Sformułowanie i rozwiązanie zagadnienia kolejnościowego z równoległym wykorzystaniem maszyn, Archiwum Automatyki i Telemechaniki, 1978, pp. 91-113.
5. Kramer A.: Scheduling Multiprocessor Tasks on Dedicated Processors, PhD-Thesis, Fachbereich Mathematik/Informatik, Universitat Osnabruck, 1995.
6. Makuchowski M., Nowicki E.: Operacje wielomaszynowe z nierównoczesnym użyciem maszyn, Komputerowo zintegrowane zarządzanie, Tom II, WNT, Warszawa 2002, pp. 114-123.
7. Nowicki E.: Metoda tabu w problemach szeregowania zadań produkcyjnych. Oficyna Wydawnicza Politechniki Wrocławskiej, Ser. Monografie 27, Wrocław 1999.
8. Nowicki E., Smutnicki C.: A decision support system for the resource constrained project scheduling problem, European Journal of Operational Research 79, 1994, pp. 183-195.
9. Sawik T.: Planowanie i sterowanie produkcji w elastycznych systemach montażowych, WNT, Warszawa 1996.

Recenzent: Dr hab. inż. Eugeniusz Toczyłowski, Prof. Pol. Warsz.

Abstract

The paper deals with the criterion of the makespan minimisation for the job shop problem with multimachine operations and non-simultaneously used machines. Proccesses which involve some machines set for operation performing are modeled as a multimachine operations. The range where machine performs an operation (so-called an activity) is

defined for each machine. An easily implemented approximation algorithm is proposed. Due to exploiting some structural properties of the problem combined with a local search technique controlled by a tabu search strategy the algorithm is able to achieve good results for instances up to 1700 activites, 300 operations and 15 machines. Computational results for test data arising from benchmark instances enlarged by activity times are presented.