Wojciech Bożejko*, Mariusz Uchroński*, Mieczysław Wodecki**

Równoległe szacowanie wartości funkcji celu w elastycznym problemie gniazdowym

1. Wstęp

W pracy rozpatrujemy pewne uogólnienie klasycznego silnie NP-trudnego problemu gniazdowego (*job shop*, [4]), tzw. elastyczny problem gniazdowy z równoległymi maszynami (zwany także ogólnym problemem gniazdowym). Algorytm dokładny rozwiązywania tego problemu został przedstawiony w pracy Pinedo [7]. Umożliwia on rozwiązanie przykładów o nie więcej niż 20 zadaniach i 10 maszynach. W literaturze opublikowano wiele algorytmów przybliżonych, głównie metaheurystyk. Hurink [5] oraz Mastrolilli i Gambardella [6] zaproponowali metodę poszukiwania z tabu (*tabu search*), Gao i in. [3], algorytmu genetycznego oraz poszukiwania ze zmiennym otoczeniem (tzw. VNS). Z kolei Bożejko, Uchroński i Wodecki [1] opracowali dwupoziomowy algorytm przeszukiwania z tabu wykorzystujący tzw. otoczenie golfowe. Bożejko w monografii [2] przedstawia metodę zrównoleglenia procesu wyznaczania tego otoczenia, co znacznie skraca obliczenia. W pracy proponujemy metodę szacowania wartości funkcji celu, która znacznie przyspiesza działanie algorytmów typu popraw bazujących na idei przeszukiwania otoczeń.

2. Sformułowanie problemu

Ogólny problem gniazdowy z równoległymi maszynami (w skrócie PJOBS) można sformułować następująco: dany jest zbiór zadań $J = \{1, 2, ..., n\}$, które należy wykonać na maszynach ze zbioru $M = \{1, 2, ..., m\}$. Niech $O = \{1, 2, ..., o\}$ będzie zbiorem wszystkich operacji. Zbiór ten można rozbić na ciągi operacji odpowiadające zadaniom, przy czym zdanie $j \in J$ jest ciągiem o_j operacji, które będą kolejno wykonywane na odpowiednich maszynach (tj. w ciągu technologicznym). Operacje te są indeksowane liczbami $(l_{j-1}+1, ..., l_{j-1}+o_j)$, gdzie l_j jest liczbą operacji pierwszych j zadań, przy czym $l_0 = 0$. Zbiór maszyn $M = \{1, 2, ..., m\}$ można z kolei rozbić na q podzbiorów maszyn tego samego typu (*gniazd*), przy czym i-ty typ M_i zawiera m_i maszyn, które

* Politechnika Wrocławska, Instytut Informatyki, Automatyki i Robotyki
** Uniwersytet Wrocławski, Instytut Informatyki

są indeksowane liczbami $(t_{i-1}+1, \ldots, t_{i-1} + m_i)$, gdzie t_i jest liczbą maszyn w pierwszych i typach, gdzie $t_0 = 0$. Operację $v \in O$ należy wykonać w gnieździe $\mu(v)$, tj. na jednej z maszyn ze zbioru $M^{\mu(v)}$ w czasie $p_{v,j}$ gdzie $j \in M^{\mu(v)}$. Niech $O^k = \{v \in O: \mu(v) = k\}$ będzie zbiorem operacji wykonywanych w k-tym gnieździe. Ciąg zbiorów operacji $Q = [Q^1, Q^2, \ldots, Q^m]$ będących rozbiciem zbioru O nazywamy *przydziałem operacji do maszyn*.

Jeżeli dokonano przydziału operacji do maszyn, wówczas wyznaczenie optymalnego terminu wykonywania operacji (w tym i kolejności wykonywania operacji na maszynach) sprowadza się do rozwiązania klasycznego problemu szeregowania, tzw. problemu gniazdowego.

Niech $Q = [Q^1, Q^2, \ldots, Q^m]$ będzie przydziałem operacji do maszyn, a $\pi_i(Q)$ permutacją elementów zbioru Q^i. Przez

$$\pi(Q) = (\pi_1(Q), \pi_2(Q), \ldots, \pi_m(Q)),$$

oznaczamy złożenie (konkatenację) tych permutacji. Dalej, niech Φ będzie zbiorem par $(Q, \pi(Q))$, których pierwszym elementem jest ciąg zbiorów (przydział operacji do maszyn), a drugim – konkatenacja permutacji elementów tych zbiorów. Dowolne rozwiązanie dopuszczalne problemu PJOBS jest parą $(Q, \pi(Q)) \in \Phi$, gdzie $\pi(Q)$ wyznacza kolejność wykonywania operacji na każdej z maszyn. Rozpatrywany w pracy problem polega na przydzieleniu operacji do maszyn oraz wyznaczeniu kolejności ich wykonywania (tj. wyznaczeniu elementu w Φ), aby czas wykonania wszystkich operacji $C_{\max}$ był minimalny. Obszernie problem ten jest opisany w monografii [2].

3. Reprezentacja grafowa rozwiązania

Dowolne rozwiązanie dopuszczalne $\Theta = (Q, \pi(Q)) \in \Phi$ problemu PJOBS można przedstawić w postaci grafu skierowanego z obciążonymi wierzchołkami (sieci) $G(\Theta) = (V, R \cup E(\Theta))$, gdzie V jest zbiorem wierzchołków, a $R \cup E(\Theta)$ zbiorem łuków, przy czym:

1. $V = O \cup \{s, c\}$, gdzie s i c są dodatkowymi (fikcyjnymi) operacjami reprezentującymi odpowiednio „start" i „zakończenie". Każdy wierzchołek $v \in V \setminus \{s, c\}$ ma dwie cechy:
   - $\lambda(v)$ – numer maszyny na której należy wykonać operację $v \in O$,
   - $p_{v,\lambda(v)}$ – wagę wierzchołka równą czasowi wykonywania operacji $v \in O$ na maszynie $\lambda(v)$.

   Wagi dodanych wierzchołków $p_s = p_c = 0$.
2. Zbiór R zawiera łuki łączące kolejne operacje tego samego zadania oraz łuki z wierzchołka s do pierwszej operacji każdego zadania i łuki od ostatniej operacji każdego zadania do wierzchołka c.
3. Z kolei łuki ze zbioru $E(\Theta)$ łączą operacje wykonywane na tej samej maszynie.

Łuki ze zbioru R wyznaczają kolejność wykonywania operacji w zadaniach (porządek technologiczny), a łuki ze zbioru $E(\Theta)$ kolejność wykonywania operacji na każdej z maszyn.

Uwaga 1. *Para $\Theta = (Q, \pi(Q)) \in \Phi$ jest rozwiązaniem dopuszczalnym dla problemu PJOBS wtedy i tylko wtedy, gdy graf $G(\Theta)$ nie zawiera cykli.*

Ciąg wierzchołków $(v_1, v_2, ..., v_k)$ grafu $G(\Theta)$ taki, że $(v_i, v_{i+1}) \in R \cup E(\Theta)$ dla $i = 1, 2, ..., k–1$, nazywamy *drogą* (lub *ścieżką*) z wierzchołka v_1 do v_k. Przez $C(v, u)$ oznaczmy najdłuższą drogę (zwaną *drogę krytyczną*) w grafie $G(\Theta)$ z wierzchołka v do u ($v, u \in V$), a przez $L(v, u)$ *długość* (sumę wag wierzchołków) tej drogi.

Łatwo zauważyć, że jeżeli $\Theta = (Q, \pi(Q))$ jest rozwiazaniem dopuszczalnym, to najkrótszy czas wykonywania wszystkich operacji $C_{\max}$ jest równy długości $L(s, c)$ drogi krytycznej $C(s, c)$ w grafie $G(\Theta)$. Rozwiązanie problemu gniazdowego z równoległymi maszynami sprowadza się więc do wyznaczenia takiego rozwiązania dopuszczalnego $\Theta = (Q, \pi(Q))$, dla którego odpowiadający mu graf $G(\Theta)$ ma najkrótszą drogę krytyczną, tj. minimalizuje $L(s, c)$.

Niech $C(s, c) = (s, v_1, v_2, \ldots, v_w, c)$, gdzie $v_i \in O$ $(1 \le i \le w)$ będzie drogą krytyczną w grafie $G(\Theta)$ z wierzchołka początkowego s do końcowego c. Drogę tę można podzielić na podciągi wierzchołków

$$B = [B^1, B^2, \ldots, B^r],$$

zwane ***blokami*** [2], przy czym:

1. blok jest podciągiem wierzchołków z drogi krytycznej zawierającym kolejne operacje,
2. blok zawiera operacje wykonywane na tej samej maszynie,
3. przekrój dwóch dowolnych bloków jest zbiorem pustym,
4. blok jest maksymalnym (ze względu na zawieranie) podzbiorem operacji z drogi krytycznej spełniającym ograniczenia 1–3.

Jeżeli B^k jest blokiem na maszynie M_i z gniazda t, to oznaczamy go następująco:

$$B^k = (\pi_i(a^k), \pi_i(a^k + 1), \ldots, \pi_i(b^k - 1), \pi_i(b^k)).$$

Operacje $\pi(a^k)$ i $\pi(b^k)$ są odpowiednio *pierwszą* i *ostatnią* w bloku B^k.

Z tzw. „tzw. „własności eliminacyjnych bloków" [2] wynika, że zmiana kolejności operacji w dowolnym bloku nie generuje rozwiązania o mniejszej wartości funkcji celu. Fakt ten będzie wykorzystywany przy generowaniu elementów otoczenia.

4. Problem pzydziału operacji do maszyn

Niech $\Theta = (Q, \pi(Q))$ będzie rozwiązaniem dopuszczalnym problemu PJOBS. Ciąg $Q = [Q^1, Q^2, ..., Q^m]$ jest przydziałem operacji do maszyn, a $\pi(Q) = (\pi_1(Q), \pi_2(Q), ..., \pi_m(Q))$ konkatenacją permutacji (w skrócie będziemy pisali $\pi = (\pi_1, \pi_2, ..., \pi_m)$).

Przez $t^i_j(k, l)$ oznaczamy ruch typu *transfer* (w skrócie *t-ruch*) polegający na przeniesieniu operacji znajdującej się na pozycji k w permutacji π_i (tj. operacji $\pi_i(k)$) na pozycję l w permutacji $\pi_j(k)$ (przesuwając wcześniej operacje znajdujące się na pozycjach k, k+1, itd. o jedną pozycję w prawo). Wykonanie ruchu $t^i_j(k, l)$ generuje z $\Theta = (Q, \pi) \in \Phi$ nowe rozwiązanie $\Theta' = (Q', \pi')$, tj. nowy przydział operacji do maszyn w pewnym gnieździe. Przez $\tau(\Theta)$ oznaczamy rozwiązanie wygenerowane z Θ przez wykonanie ruchu τ.

Dla ustalonego rozwiązania dopuszczalnego Θ, niech $T(\Theta)$ będzie zbiorem wszystkich *t-ruchów* generujących z Θ rozwiązania dopuszczalne. *Otoczeniem* Θ jest zbiór

$$N(\Theta) = \{\tau(\Theta) : \tau \in T(\Theta)\} \tag{1}$$

Proponowana przez nas metoda rozwiązania problemu PJOBS składa się z dwóch kroków. **Pierwszy** – wyznaczenie pewnego przydziału operacji do maszyn, oraz **drugi** – wyznaczenie kolejności wykonywania operacji, tj. rozwiązanie problemu gniazdowego.

Niech $\Theta = (Q, \pi)$ będzie rozwiązaniem dopuszczalnym problemu PJOBS. Nowy przydział operacji do maszyn Q' generujemy z Q następująco:

- wyznaczamy otoczenie $N(\Theta)$ zgodnie z (1),
- wybieramy z otoczenia rozwiązanie $\Theta' = (Q', \pi')$ o najmniejszej wartości funkcji celu, tj. nowy przydział operacji do maszyn Q'.

4.1. Wyznaczanie otoczenia

Wykonanie *t-ruchu* może generować rozwiązanie niedopuszczalne, tzn. odpowiadający temu rozwiązaniu graf zawiera cykl. W dalszej części przedstawimy twierdzenie umożliwiające w czasie stałym badanie dopuszczalności rozwiązań generowanych przez *t-ruchy*.

Niech $\Theta = (Q, \pi)$ będzie rozwiązaniem dopuszczalnym, przy czym $Q = [Q^1, Q^2, ..., Q^m]$ jest przydziałem operacji do maszyn, a $\pi = (\pi_1, \pi_2, ..., \pi_m)$ konkatenacją permutacji. Permutacja π_i wyznacza kolejność wykonywania operacji ze zbioru Q^i na maszynie M_i.

Rozpatrujemy dwie maszyny z tego samego gniazda M_i i M_j. Dla dowolnej operacji $\pi_i(k) \in Q^i$ definiujemy dwa parametry związane z drogami w grafie $G(\Theta)$:

$$\eta_j(k) = \begin{cases} 1 & \text{gdy } \forall v = 1,2,\ldots,\rho_j \text{ nie istnieje droga } C(\pi_j(v), \pi_i(k)), \\ 1 + \max\limits_{1 \le v \le \rho_j} \{\text{istnieje droga } C(\pi_j(v), \pi_i(k))\} & \text{w p. p.} \end{cases}$$

oraz

$$\rho_j(k) = \begin{cases} 1 & \text{gdy } \forall v = 1,2,\ldots,\rho_j \text{ nie istnieje droga } C(\pi_j(v), \pi_i(k)), \\ 1 + \min\limits_{\eta_j(k) \le v \le \rho_j} \{\text{istnieje droga } C(\pi_i(k), \pi_j(v))\} & \text{w p. p.} \end{cases}$$

Parametry te będą stosowane przy wyznaczaniu *t-ruchów* generujących otoczenie rozwiązania Θ.

Twierdzenie 1 [2]. Niech $\Theta = (Q, \pi)$ będzie rozwiązaniem dopuszczalnym dla problemu PJOBS oraz π_i, π_j permutacjami operacji wykonywanych na maszynach M_i, M_j. Jeżeli maszyny M_i, M_j należą do tego samego gniazda, to wykonanie *t-ruchu* $t_j^i(k,l)$ $(l = \eta_j(k), \eta_j(k)+1, \ldots, \rho_j(k))$ generuje rozwiązanie dopuszczalne.

Następne twierdzenie wyrażaja tzw. „własności eliminacyjne bloków”.

Twierdzenie 2 [2]. *Niech* $\Theta = (Q, \pi)$ *będzie rozwiązaniem dopuszczalnym dla problemu* PJOBS. *Jeżeli* B^u *jest blokiem na maszynie* M_i, *a* B^v *blokiem na* M_j *oraz obie maszyny należą do tego samego gniazda, to ruch typu transfer polegający na przeniesienie operacji z bloku wewnętrznego* B^u *do bloku wewnętrznego* B^v *nie generuje rozwiązania o mniejszej wartości funkcji celu.*

Wobec tego, aby przez wykonanie *t-ruchu* wygenerować ewentualnie lepsze rozwiązanie należy pierwszą lub ostatnią operację bloku przenieść przed pierwszą lub za ostatnią operację innego bloku.

Jeżeli $B = [B^1, B^2, \ldots, B^r]$ jest ciągiem bloków z drogi krytycznej w grafie $G(\Theta)$, wówczas przez $T^{acc}(\Theta) \subset T(\Theta)$ oznaczamy zbiór zwierający ruchy przestawiające pierwszą (lub ostatnią) operację każdego bloku na inną (z tego samego gniazda) maszynę. Jeżeli $\pi(v)$ jest pierwszą (lub ostatnią) operacją pewnego bloku oraz M_j jest maszyną z tego samego gniazda, wówczas zbiór $T^{acc}(\Theta)$ zawiera ruchy przestawiające $\pi(v)$ na następujące pozycje: $\eta_j(v), \eta_j(v+1), \ldots, \rho_j(v)$. Okazało się, że generowane przez te ruchy otoczenie jest duże i zawiera wiele „złych" ruchów. Ograniczyliśmy się więc do ruchów przestawiających pierwszą (lub ostatnią) operację $\pi(v)$ bloku **jedynie** na pozycję $\eta_j(v)$ lub $\rho_j(v)$. Ostatecznie więc

$$T^{subm}(\Theta) = \{t^i_j(v, w) \in T^{acc} : w \in \{a^k, b^k\}, w \in \{\eta_j(v), \rho_j(v)\}, k = 1, 2, \ldots, r\}.$$

Wówczas otoczeniem Θ jest zbiór rozwiązań dopuszczalnych

$$N(\Theta) = \{\tau(\Theta) : \tau \in T^{subm}(\Theta)\}$$

5. Szacowania wartości funkcji celu

Aby przyśpieszyć procedurę wyboru elementu z otoczenia, jako kryterium będziemy stosowali obliczane w czasie stałym dolne ograniczenia wartości funkcji celu.

Niech $\Theta = (Q, \pi)$ będzie rozwiązaniem dopuszczalnym $(Q = [Q^1, Q^2, \ldots, Q^m]$, a $B = [B^1, B^2, \ldots, B^r]$ ciągiem bloków drogi krytycznej w grafie $G(\Theta)$.

Rozpatrujemy dwie maszyny M_i oraz M_j należące do tego samego gniazda. Na maszynie M_i są wykonywane operacje ze zbioru Q^j w kolejności $\pi_i = (\pi_i(1), \pi_i(2), \ldots, \pi_i(\rho_i))$, a na maszynie M_j operacje ze zbioru Q^j w kolejności $\pi_j = (\pi_j(1), \pi_j(2), \ldots, \pi_j(\rho_j))$. Załóżmy, że blok

$$B^k = (\pi_i(a^k), \pi_i(a^k + 1), \ldots, \pi_i(b^k - 1), \pi_i(b^k)),$$

zawiera operacje wykonywane na maszynie M_i. Dla uproszczenia zapisu pomijamy indeks k oznaczający numer bloku. Wobec tego $\pi_i(a)$ jest pierwszą, a $\pi_i(b)$ ostatnią operacją bloku B^k.

Zgodnie ze strategią przeszukiwania otoczenia $N(\Theta)$ wybieramy taki ruch $\tau \in T^{subm}(\Theta)$, który wygeneruje graf $G(\tau(\Theta))$ – rozwiązanie dopuszczalne o możliwie najmniejszej wartości oszacowania długości drogi krytycznej (tj. wartości funkcji celu). Dla ruchów z $T^{subm}(\Theta)$

przestawiających pierwszą operację $\pi_i(a^2)$ bloku B^k na pozycję $\eta_j(a^k)$ lub $\rho_j(a^k)$ w permutacji π_j wprowadzamy oznaczenia

$$\Delta^{a^k}_{\mathrm{x}(a^k)} = \max\{L_1^{\mathrm{x}(a^k)}, L_2^{\mathrm{x}(a^k)}, L_3^{\mathrm{x}(a^k)}, L_4^{\mathrm{x}(a^k)}\}, \;\; \mathrm{x}(a^k) \in \{\eta_j(a^k), \rho_j(a^k)\},$$

gdzie:

$$L_1^{\mathrm{x}(a^k)} = L(s, \pi_i(a^k - 1)) - L(s, \pi_i(a^k)),$$

$$L_2^{\mathrm{x}(a^k)} = L(s, \pi_i(a^k + 1)) - L(s, \pi_i(a^k)) - p_{\pi_i(a^k+1)},$$

$$L_3^{\mathrm{x}(a^k)} = L(s, \pi_j(1)) + \sum_{h=2}^{w-1} p_{\pi_i(h)} + p_{\pi_i(a^k)} + L(\pi_j(w), c) +$$

$$-L(s, \pi_i(a^k)) - \sum_{h=a^k+1}^{b^k-1} p_{\pi_i(h)} - L(\pi_i(b^k), c),$$

$$L_4^{\mathrm{x}(a^k)} = \sum_{h=\mathrm{x}(a^k)+1}^{w-1} p_{\pi_j(h)} + L(\pi_j(w), c) - \sum_{h=a^k+1}^{b^k-1} p_{\pi_i(h)} - L(\pi_i(b^k), c).$$

Podobnie, dla ruchów z $T^{subm}(\Theta)$ przestawiających ostatnią operację $\pi_i(b^k)$ bloku B^k na pozycję $\eta_j(b^k)$ lub $\rho_j(b^k)$ w permutacji π_j wprowadzamy oznaczenia

$$\Delta^{b^k}_{\mathrm{y}(b^k)} = \max\{L_1^{\mathrm{y}(b^k)}, L_2^{\mathrm{y}(b^k)}, L_3^{\mathrm{y}(b^k)}, L_4^{\mathrm{y}(b^k)}\}, \;\; \mathrm{y}(b^k) \in \{\eta_j(b^k), \rho_j(b^k)\},$$

gdzie:

$$L_1^{\mathrm{y}(b^k)} = L(\pi_i(b^k - 1), c) - p_{\pi(b^k-1)} - L(\pi_i(b^k), c),$$

$$L_2^{\mathrm{y}(b^k)} = L(\pi_i(b^k + 1), c) - L(\pi_i(b^k), c),$$

$$L_3^{\mathrm{y}(b^k)} = L(s, \pi_j(1)) + \sum_{h=2}^{w-1} p_{\pi_j(h)} + p_{\pi_i(b^k)} + L(\pi_j(w), c) +$$

$$-L(s, \pi_i(a^k)) - \sum_{h=a^k+1}^{b^k-1} p_{\pi_i(h)} - L(\pi_i(b^k), c),$$

$$L_4^{y(b^k)} = L(s,\beta(1)) + \sum_{h=2}^{y(b^k)-1} p_{\pi_j(h)} - L(s,\pi_i(a^k)) - \sum_{h=a^k+1}^{b^k-1} p_{\pi_i(h)}.$$

Kolejne dwa twierdzenia pozwalające na szacowanie wartości funkcji celu dla rozwiązania wygenerowanego z Θ przez *t-ruchy* ze zbioru $T^{subm}(\Theta)$.

Twierdzenie 3 [1]. *Jeżeli rozwiązanie* $\Theta' = (Q', \pi')$ *zostało wygenerowane z* $\Theta = (Q, \pi)$ *przez wykonanie ruchu* $t_j^i(a^k, x(a^k)) \in T^{subm}$, $x(a^k) \in \{\eta_j(a^k), \rho_j(a^k)\}$ *to*

$$L'(s, c) \geq L(s, c) + \Delta^{a^k}_{x(a^k)}.$$

Twierdzenie 4 [1]. *Jeżeli rozwiązanie* $\Theta' = (Q', \pi')$ *zostało wygenerowane z* $\Theta = (Q, \pi)$ *przez wykonanie ruchu* $t_j^i(b^k, y(b^k)) \in T^{subm}$, $y(b^k) \in \{\eta_j(b^k), \rho_j(b^k)\}$ *to*

$$L'(s, c) \geq L(s, c) + \Delta^{b^k}_{y(b^k)}.$$

Wobec tego wyrażenie $\Delta^{a^k}_{x(a^k)}$, $x(a^k) \in \{\eta_j(a^k), \rho_j(a^k)\}$ lub $\Delta^{b^k}_{y(b^k)}$, $y(b^k) \in \{\eta_j(b^k), r_j(b^k)\}$ można wykorzystać do wyboru operacji (tj. elementu z otoczenia), która będzie przestawiana. Ostatecznie wybieramy operację $\pi(v) \in O$ taką, że

$$\Delta^v_{\chi(v)} = \min_{1 \leq k \leq r} \min\{\Delta^z_{\mu(z)} : z \in \{a^k, b^k\}, \mu(z) \in \{\eta_j(z), \rho_j(z)\}\}$$

Minimalna wartość $\Delta^v_{\chi(v)}$ odpowiada wówczas „najlepszemu" *t-ruchowi* polegającemu na przaestawieniu pierwszej lub ostatniej operacji z pewnego bloku na inną maszynę. Jeżeli $\Delta^v_{\chi(v)} > 0$, to w wygenerowanym grafie $G(\Theta')$ długość drogi krytycznej $L'(s, c) > L(s, c)$.

Reasumując, dla rozwiązania $\Theta = (Q, \pi)$ w grafie $G(\Theta)$ wyznaczamy drogę krytyczną $C(s, c)$ oraz obliczamy jej długość $L(s, c) = C_{\max}(\Theta)$. Następnie, wyznaczamy podział drogi na bloki $B = [B^1, B^2, \ldots, B^r]$ oraz zbiór ruchów $T^{subm}(\Theta)$. Obliczamy $\Delta^v_{\chi(v)}$ i wybieramy „najlepszy" *t-ruch* $t_j^i(v, \chi(v))$. Ruch ten generuje rozwiązanie z otoczenia $N(\Theta)$ o najmniejszej wartości dolnego ograniczenia funkcji celu.

6. Eksperymenty obliczeniowe

Przedstawiona metoda szacowania wartości funkcji celu została zastosowana w odpowiednio zmodyfikowanej wersji algorytmu poszukiwania z tabu zamieszczonego w pracy [1]. Obliczenia wykonano na znanych przykładach zamieszczonych w pracy Hurinka [5] i wykonano na komputerze HP xw4600 z procesorem Intel Core 2 Duo 3.16GHz (nVidia GeForce GTX480

GPU) działającym pod kontrolą systemu operacyjnego Linux Fedora 12. Otrzymane wyniki algorytmu przedstawiono w tabeli 1. Poszczególne kolumny oznaczają:

Flex – średnią liczbę maszyn równoległych na operację,
TS1 – algorytm *tabu search* z dokładnie wyznaczaną wartością funkcji celu,
TS2 – algorytm *tabu search* z „szacowaną" wartością funkcji celu,
t – czas działania funkcji wyznaczania (lub szacowania) wartości funkcji celu dla wszystkich rozwiązań generowanych przez *t-ruchy*,
$C_{\max}$ – wyznaczona wartość funkcji celu.
Adv – procentowy błąd wartości funcji celu algorytmu TS2 w stosunku do TS1.

Tabela 1
Wyniki eksperymentów obliczeniowych na przykładach Hurinka [5]

Problem	$n \times m$	Flex.	t [ms]		C_{max}		Adv [%]
			TS1	TS2	TS1	TS2	
abz5	10×10	2	0.1125	0.0139	1146	1181	3.05
abz6	10×10	2	0.1581	0.0198	933	952	2.04
abz7	20×15	2	0.4001	0.0201	637	641	0.63
abz8	20×15	2	0.3221	0.0158	635	640	0.79
abz9	20×15	2	0.3743	0.0169	647	656	1.39
średnio			**0.2734**	**0.0173**			**1.58**

Na podstawie otrzymanych wyników można stwierdzić, że zastosowanie szacowania wartości funkcji celu (o złożoności $O(1)$) zamiast wyznaczania jej dokładnej wartości powoduje znaczne, bo ponad 15-krotne, skrócenie czasu działania. Przy czym generalnie zysk ten rośnie, wraz ze wzrostem liczby zadań lub liczby maszyn (rozmiaru problemu). Szacowanie w algorytmie TS2 wartości funkcji celu prowadzi niestety do nieznaczego pogorszenia wyników (w stosunku do TS1) – średnnio o 1,58%.

7. Podsumowanie

W pracy przedstawiono metodę rozwiązywania silnie NP-trudnego problemu gniazdowego z równoległymi maszynami. Opisano główne element algorytmu przeszukiwania z tabu oraz sposoby szacowania wartości funkcji celu. Ich zastosowanie znacznie przyśpieszyło proces przeszukiwania otoczenia. Przeprowadzono eksperymenty obliczeniowe i w bardzo krótkim czasie otrzymano wyniki tylko nieznacznie różniące się od najlepszych obecnie znanych w literaturze.

Praca finansowana z projektów badawczych MNiSW nr N N514 232237.

Literatura

[1] Bożejko W., Uchroński M., Wodecki M., *The new golf neighborhood for the flexible job shop problem*. Proceedings of the ICCS 2010, Procedia Computer Science, 1, 2010, Elsevier, 289–296.
[2] Bożejko W., *A new class of parallel scheduling algorithms*. Oficyna Wydawnicza Politechniki Wrocławskiej, seria: monografie, 2010.
[3] Gao J., Sun L., Gen M., *A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems*. Computers & Operations Research, 35, 2008, 2892–2907.
[4] Garey M.R., Johnson D.S., Sethi R., *The complexity of the flowshop and jobshop scheduling*. Math. Oper. Res., 1/2, 1974, 117–128.
[5] Hurink E., Jurisch B., Thole M., *Tabu search for the job shop scheduling problem with multi-purpose machine*. Operations Research Spektrum, 15, 1994, 205–215.
[6] Mastrolilli M., Gambardella L.M., *Effective neighborhood functions for the flexible job shop problem*. Journal of Scheduling, 3/1, 2000, 3–20.
[7] Pinedo M., *Scheduling: theory, algorithms and systems*. Englewood Cliffs, NJ: Prentice-Hall, 2002.