CYKLICZNY SYSTEM PRODUKCYJNY Z DWUMASZYNOWYMI GNIAZDAMI

Wojciech BOŻEJKO, Piotr NADYBSKI, Mariusz UCHROŃSKI, Mieczysław WODECKI

Streszczenie: W pracy rozpatrujemy elastyczny system produkcji cyklicznej, w którym operacje są wykonywane przez maszyny równoległe pogrupowane w dwumaszynowe gniazda. Harmonogramowanie operacji w tym systemie produkcji wymaga jednoczesnego podjęcia decyzji na dwóch poziomach: (i) przydziału operacji do maszyn, (ii) wyznaczenia kolejności wykonywania operacji na każdej maszynie. Ze względu na NP-trudność tego problemu, przedstawiamy konstrukcje algorytmów heurystycznych. Porównujemy czasy wykonywania oraz wartości rozwiązań wyznaczonych przez algorytmy: przeszukiwania z tabu oraz symulowanego wyżarzania.

Słowa kluczowe: szeregowanie zadań, cykliczny problem gniazdowy, przezbrojenia maszyn, algorytm metaheurystyczny.

1. Wprowadzenie

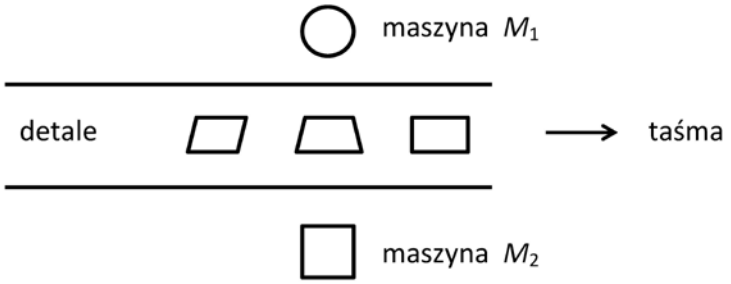
W mało zróżnicowanej produkcji masowej są stosowane cykliczne systemy wytwarzania. Umożliwia to znaczne uproszczenie czynności logistycznych związanych z dostarczaniem surowców do produkcji oraz odbiorcom wyrobów gotowych. W stałych odstępach czasu (zwanym *czasem cyklu*) jest wytwarzana pewna mieszanka produktów oznaczana w literaturze skrótem *MPS* (ang. *Minimal Part Set*). Każdy produkt jest wytwarzany cyklicznie, tj. wielokrotnie po upływie czasu cyklu. Zakładamy ponadto, że całe *MPS*-y są powielane cyklicznie, tj. wykonywanie produktów w ramach kolejnego *MPS*-a rozpoczyna się dopiero po zakończeniu wytwarzania wszystkich produktów z *MPS*-a poprzedniego cyklu. Tak więc kolejne *MPS*-y (tj. produkty z tych *MPS*-ów) są wykonywane bezpośrednio jeden po drugim w sposób cykliczny. Innymi słowy, nie ma mieszania produktów z *MPS*-ów wykonywanych w ramach różnych cykli.

Podstawy cyklicznego szeregowania zadań zostały szeroko opisane w rozprawie doktorskiej przez Kampmeyera [13], a różne szczegółowe modele są przedstawione w pracach Caggiano i Jakson [7], Brukera i Kampmeyera [5, 6], Lrevnera i in. [14] oraz Bożejkę i in. [3], [4]. W rozpatrywanym w pracy problemie dany jest zbiór zadań oraz zbiór maszyn. Maszyny tego samego typu, tj. o tych samych własnościach funkcjonalnych tworzą *gniazdo*. Każde gniazdo zawiera dwie maszyny (tzw. *maszyny równoległe*). Zadanie należy wykonać na jednej z maszyn każdego gniazda, z jednakowym dla wszystkich zadań, ustalonym *porządkiem technologicznym*, tj. kolejnością "przechodzenia" pomiędzy gniazdami. Dane są czasy wykonywania zadań na maszynach oraz czasy przezbrojenia maszyn pomiędzy kolejno wykonywanymi zadaniami. Należy przydzielić zadania do odpowiednich maszyn, w każdym z gniazd oraz ustalić kolejność ich wykonywania tak, aby zminimalizować czas cyklu. Spełnione muszą być przy tym następujące ograniczenia:

1. Każde zadanie może być wykonywane jednocześnie tylko na jednej, odpowiedniego typu, maszynie.

2. Żadna maszyna nie może wykonywać jednocześnie więcej niż jedno zadanie.
3. Wykonywanie zadania nie może być przerwane.
4. Zachowany musi być porządek technologiczny wykonywania zadań.
5. S każdym gnieździe zadania są wykonywane w takiej samej kolejności.
6. Pomiędzy kolejno wykonywanymi zadaniami należy dokonać przezbrojenia maszyny.
7. Każde zadanie jest kolejno wykonywane po upływie czasu cyklu.

Przykład ilustrujący pracę pojedyńczego gniazda takiego systemu jest symbolicznie przedstawiony na rysunku 1. Na taśmie przemieszczane są produkty (zadania). Pewną operację każdego produktu należy wykonać na jednej z dwóch maszyn (gniazdo - maszyny M_1 i M_2) obsługiwanych przez jednego pracownika.



Rys. 1. System przepływowy z dwoma równoległymi maszynami

Rozwiązanie rozpatrywanego problemu polega na:

A. przydzieleniu zadań do maszyn w poszczególnych gniazdach,

B. wyznaczeniu momentów rozpoczęcia wykonywania zadań na maszynach tak,

aby spełnione były ograniczenia (1)-(7) oraz czas cyklu (czas po którym dowolne zadanie jest wykonywane w kolejnym *MPS*-ie) był minimalny.

Z ograniczenia (5) wynika, że każde rozwiązanie może być reprezentowane przez odpowiednią permutację zadań (kolejność "przepływu" przez gniazdo). Dla ustalonego w każdym gnieździe przydziału zadań do maszyn, problem sprowadza się do wyznaczenia permutacji zadań minimalizującej czas cyklu (Bożejko i in. [2]).

Pierwsze prace dotyczące problemów cyklicznego szeregowania zadań ukazały się w latach 70-tych XX wieku (Crowston i in. [9]). Wykaz ważniejszych prac z szeregowania cyklicznego oraz omówienie wyników badań prowadzonych na przestrzeni ostatnich lat przedstawili między innymi: Hanen i Munier [11], Crama i in. [8], Levner i in. [14], a także Jeong i Tae [12], Sawik [18, 19], oraz Bożejko i in. [2, 3, 4].

W następnym rozdziale krótko przedstawiamy gniazdowy problem przepływowy z przezbrojeniami maszyn, a w dalszej części pracy - cykliczna wersję tego problemu.

2. Gniazdowy problem przepływowy

Rozpatrywany w pracy gniazdowy problem przepływowy z przezbrojeniami maszyn można sformułować następująco.

Dany jest zbiór zadań $J = \{1,2,\ldots,n\}$, które należy wykonać na maszynach ze zbioru $M = \{1,2,\ldots,q\}$. Istnieje rozbicie zbioru maszyn na typy (gniazda), tj. podzbiory maszyn

o tych samych własnościach funkcjonalnych. Każde zadanie "przechodzi" przez wszystkie gniazda w takiej samej kolejności (porządek technologiczny). Zadanie jest ciągiem operacji, z których każdą należy wykonać, w ustalonym czasie, na dokładnie jednej maszynie z każdego gniazda. Problem polega przydzieleniu zadań do maszyn oraz na wyznaczeniu kolejności ich wykonywania na maszynach, aby zoptymalizować pewne kryterium. Muszą być przy tym spełnione ograniczenia od (1) do (6). Problem ten będziemy w skrócie oznaczali przez NFS. Jeżeli każde gniazdo zawiera tylko jedną maszynę wówczas jest to klasyczny w teorii szeregowania problemem przepływowy (ang. *flow shop*) rozpatrywany na przykład w pracach [16], [10] dodatkowo z przezbrojeniami maszyn.

Zbiór maszyn $\mathsf{M} = \{1,2,\ldots,q\}$ można rozbić na m podzbiorów maszyn tego samego typu (*gniazd*), przy czym i-ty $(i = 1,2,\ldots,m)$ typ M^i zawiera dwie maszyny. Niech $\mathsf{O} = \{1,2,\ldots,o\}$ będzie zbiorem wszystkich operacji. Zbiór ten można rozbić na ciągi odpowiadające zadaniom, przy czym zadanie $j \in \mathsf{J}$ jest ciągiem m operacji, które będą kolejno wykonywane na odpowiednich maszynach (tj. w kolejnych gniazdach w ciągu technologicznym). Operację $v \in \mathsf{M}$ należy wykonać w gnieździe $\mu(v)$, tj. na jednej z maszyn zbioru $\mathsf{M}^{\mu(v)}$ w czasie $p_{v,j}$, gdzie $j \in \mathsf{M}^{\mu(v)}$.

Niech

$$\mathsf{O}^k = \{v \in \mathsf{O} : \mu(v) = k\},$$

będzie zbiorem operacji wykonywanych w k-tym $(k = 1,2,\ldots,m)$ gnieździe. Ciąg rozłącznych zbiorów operacji

$$\mathsf{Q} = [\mathsf{Q}^1, \mathsf{Q}^2, \ldots, \mathsf{Q}^q],$$

nazywamy *przydziałem operacjido maszyn* (Q^k zawiera operacje wykonywane na k-tej maszynie). Zbiór wszystkich takich przydziałów będziemy oznaczali przez A.

Niech π będzie pewną kolejnością (permutacją) wykonywania zadań ze zbioru J oraz P zbiorem wszystkich takich permutacji. Dowolne rozwiązanie rozpatrywanego w tym rozdziale problemu będzie reprezentowane przez parę (Q, π), gdzie Q jest przydziałem operacji do maszyn $(\mathsf{Q} \in \mathsf{A})$, a π - permutacją zadań $(\pi \in \mathsf{P})$. Przez AP oznaczamy zbiór wszystkich takich par (rozwiązań).

Jeżeli $(\mathsf{Q}, \pi) \in \mathsf{AP}$ jest pewnym rozwiązaniem gniazdowego problemu przepływowego, to kolejność wykonywania zadań na k-tej maszynie (tj. zadań ze zbioru Q^k) jest taka sama, jak kolejność ich występowania w permurtacji π.

Przykład 1.

Dany jest zbiór zadań $\mathsf{J} = \{J_1, J_2, J_3\}$, które należy wykonać na maszynach ze zbioru $\mathsf{M} = \{M_1, M_2, M_3, M_4, M_5, M_6\}$. Zbiór maszyn dzieli się na trzy dwumaszynowe gniazda:

$$\mathsf{M}^1 = \{M_1, M_2\},\ \mathsf{M}^2 = \{M_3, M_4\},\ \mathsf{M}^3 = \{M_5, M_6\}.$$

W tym przypadku, liczba zadań $n = 3$, maszyn $q = 6$, gniazd $m = 3$. Zakładamy, że zgodnie z wymogami technologicznymi, każde zadanie należy wykonać na jednej z maszyn gniazda M^1, następnie gniazda M^2 i na koniec gniazda M^3. Wobec tego, każde zadanie składa się z trzech operacji. Jeżeli $\mathsf{O} = \{O_1, O_2, O_3, O_4, O_5, O_6, O_7, O_8, O_9\}$ jest zbiorem

wszystkich operacji, to zadanie $J_1 = [O_1, O_2, O_3]$, $J_2 = [O_4, O_5, O_6]$ i $J_3 = [O_7, O_8, O_9]$. Ponieważ $\mu(O_1) = \mu(O_4) = \mu(O_7) = 1$ (operacje te należy wykonać na jednej z maszyn 1-szego gniazda M^1), więc zbiór $\mathsf{O}^1 = \{O_1, O_4, O_7\}$. Dalej, $\mu(O_2) = \mu(O_5) = \mu(O_8) = 2$ (gniazdo M^2, $\mathsf{O}^2 = \{O_2, O_5, O_8\}$) oraz $\mu(O_3) = \mu(O_6) = \mu(O_9) = 3$ (gniazdo M^3, $\mathsf{O}^3 = \{O_3, O_6, O_9\}$). Czasy wykonywania operacji na poszczególnych maszynach są zamieszczone w tabeli 1. Przyjmujemy, że wszystkie czasy przezbrojeń maszyn są równe 1.

Tab. 1. Czasy wykonywania operacji na maszynach.

	Gniazdo M^1		Gniazdo M^2		Gniazdo M^3	
	M_1	M_2	M_3	M_4	M_5	M_6
zadanie J_1	1	2	2	3	3	2
zadanie J_2	1	3	3	1	5	2
zadanie J_3	2	3	3	3	4	2

Rozpatrujemy pewien przydział operacji do maszyn (w poszczególnych gniazdach) $\mathsf{Q} = [\mathsf{Q}^1, \mathsf{Q}^2, \mathsf{Q}^3, \mathsf{Q}^4, \mathsf{Q}^5, \mathsf{Q}^6]$, gdzie:

1. $\mathsf{Q}^1 = \{O_1, O_4, O_7\}$ i $\mathsf{Q}^2 = \varnothing$ jest przydziałem operacji w gnieździe M^1 (wszystkie operacje będą wykonywane na maszynie M_1),
2. $\mathsf{Q}^3 = \{O_2\}$ i $\mathsf{Q}^4 = \{O_5, O_8\}$ jest przydziałem operacji w gnieździe M^2 (O_2 będzie wykonywana na maszynie M_3, a O_5, O_8 na maszynie M_4),
3. $\mathsf{Q}^5 = \{O_6\}$ i $\mathsf{Q}^6 = \{O_3, O_9\}$ jest przydziałem operacji w gnieździe M^3 ($\{O_6\}$ będzie wykonywana na maszynie M_5, a $\{O_3, O_9\}$ na M_6).

Załóżmy, że w każdym gnieździe zadania są wykonywane w tej samej kolejności $\pi = (1,2,3)$. Łatwo sprawdzić, że para (Q, π) jest rozwiązaniem rozpatrywanego przykładu problemu NFS.

3. Cykliczny gniazdowy problem przepływowy

Zasadniczym tematem pracy jest cykliczna wersja (w skrócie oznaczana przez CNFS), opisanego w poprzednim rozdziale gniazdowego problemu przepływowego z przezbrojeniami maszyn. W rozpatrywanym systemie produkcyjnym, ustalony zbiór zadań (*MPS*) wykonywany jest wielokrotnie. Zakładamy, że w każdym z *MPS*-ów, przydziały zadań do maszyn oraz kolejności ich wykonywania są takie same. Wobec tego, każdy cykliczny harmonogram może być reprezentowany przez harmonogram wykonywania operacji w pierwszym *MPS*-ie, tj. parę $(Q, \pi) \in \mathsf{AP}$ - rozwiązanie gniazdowego problemu przepływowego (rozdz. 2). Należy jedynie przesunąć terminy rozpoczęcia zadań w kolejnych *MPS*-ach o odpowiednią wielkość (czas cyklu), aby spełnione było ograniczenie (*7*) dotyczące cykliczności procesu.

Niech $(\mathsf{Q}, \pi) \in \mathsf{AP}$ będzie rozwiązaniem gniazdowego problemu przepływowego, tj. problemu NFS. Rozpatrujemy cykliczną wersję tego problemu. W każdym z *MPS*-ów

operacje będą wykonywane zgodnie z rozwiązaniem (Q,π). Przez $[\,S^l\,]_{n\times q}$ oznaczamy macierz terminów rozpoczęcia wykonywania zadań l - tego *MPS*-a dla tego rozwiązania ($S^l_{i,j}$ jest terminem rozpoczęcia wykonywania zadania i na maszynie j). Z założenia cykliczności systemu wynika, że istnieje stała $T(\mathrm{Q},\pi)$ taka, że

$$S^{l+1}_{\pi(i),j} = S^l_{\pi(i),j} + T(\mathrm{Q},\pi),\ i=1,...,n,\ j=1,...,q,\ l=1,2,..., \tag{1}$$

gdzie wartości $S^1_{\pi(i),j}$ wyznaczamy na podstawie rozwiązania (Q,π) problemu NFS

Wielkość $T(\mathrm{Q},\pi)$ będziemy nazywali *czasem cyklu* dla rozwiązania (Q,π).

Dla ustalonego rozwiązania $(\mathrm{Q},\pi)\in \mathrm{AP}$ czas wykonywania zadań w *k*-tym gnieździe ($k=1,2,\ldots,m$) w ramach pierwszego MPS-a (wraz z przezbrojeniem każdej maszyny po wykonaniu przez nią ostatniego zadania)

$$T_k(\mathrm{Q},\pi) = \sum_{i=1}^{2}\sum_{j=1}^{n} p_{\pi_i(j),i} + \sum_{i=1}^{2}\sum_{j=1}^{n} s^k_{\pi_i(j),\pi_i(j+1)} + \sum_{i=1}^{2} s^k_{\pi_i(m_i),\pi_i(1)}, \tag{2}$$

gdzie $\pi=(\pi_1,\pi_2,\ldots,\pi_q)$, a π_i jest kolejnością wykonywania zadań na i-tej maszynie. Pierwszy składnik wyrażenia (2) jest sumą czasów wykonywania wszystkich zadań przez maszyny tego gniazda, drugi - sumą przezbrojeń poszczególnych maszyn pomiędzy kolejno wykonywanymi zadaniami. Z kolei ostatni składnik jest sumą przezbrojeń maszyn pomiędzy ostatninm zadaniem wykonywanym przez maszynę (w pewnym *MPS*-ie), a pierwszym zadaniem w następnym *MPS*-ie.

Wielkość $T_k(\mathrm{Q},\pi)$ nazywamy *k -tym pikiem*, a

$$T_{\max}(\mathrm{Q},\pi) = \max\{T_i(\mathrm{Q},\pi) : i=1,2,\ldots,m\}, \tag{3}$$

pikiem dominującym rozwiązania (Q,π).

Własność 1. *Dla rozwiązania* $(\mathrm{Q},\pi)\in \mathrm{AP}$ *minimalną wartością czasu cyklu jest równa wartości pika dominującego* $T_{\max}(\mathrm{Q},\pi)$.

Dowód. Należy udowodnić, że:

1. $T_{\max}(\mathrm{Q},\pi)$ jest czasem cyklu, tj. przyjmując w (1), $T(Q,\pi)=T_{\max}(\mathrm{Q},\pi)$ czasy rozpoczęcia zadań spełniają ograniczenia (*1*)-(*7*),
2. nie istnieje liczba $T'<T_{\max}(\mathrm{Q},\pi)$ spełniająca ograniczenia dotyczące czasu cyklu.

Dowody obu punktów są proste, więc je pomijamy.

Rozpatrywany w tym rozdziale cykliczny problem gniazdowy sprowadza się więc do wyznaczenia przydziału operacji do maszyn i ustaleniu kolejności ich wykonywania, aby zminimalizować wartość pika dominującego, tj. wyznaczenia *rozwiązania optymalnego* $(\mathrm{Q}^*,\pi^*)\in \mathrm{AP}$ takiego, że

$$T(\mathrm{Q}^*,\pi^*) = \min\{T_{\max}(\mathrm{Q},\pi) : (\mathrm{Q},\pi)\in \mathrm{AP}\}. \tag{4}$$

Własność ta będzie podstawą algorytmu rozwiązania problemu CNFS

Przykład 2.

Dla danych oraz rozwiązania z Przykładu 1, czasy pracy poszczególnych gniazd (2) wynoszą odpowiednio:

$$T_1(\mathrm{Q},\pi)=7,\quad T_2(\mathrm{Q},\pi)=8,\quad T_3(\mathrm{Q},\pi)=11,$$

a wartość pika dominującego (3)

$$T_{\max}(\pi,\mathrm{Q}) = max\{T_1(\mathrm{Q},\pi), T_2(\mathrm{Q},\pi), T_3(\mathrm{Q},\pi)\} = 11.$$

Minimalna wartość czasu cyklu, dla tego rozwiązania, jest więc równa 11.

Dla ustalonego przydziału operacji do maszyn w gniazdach $Q \in \mathrm{A}$, niech

$$T_{\min}(Q) = \min\{T(Q,\pi) : \pi \in \mathrm{P}\}. \tag{5}$$

będzie *minimalnym czasem cyklu* dla Q tzn., że żadna zmiana kolejności wykonywania zadań na dowolnej maszynie nie zmniejszy wartości $T_{\min}(Q)$. Wyznaczenie minimalnego czasu cyklu $T_{\min}(Q)$ sprowadza się do rozwiązania NP-trudnego cyklicznego problemu przepływowego z przezbrojeniami maszyn (zobacz np. [3]).

Jeżeli $\pi^{\circ} \in \mathrm{P}$ jest permutacją dla której w (5) osiągnięto minimum, to istnieje gniazdo k którego czas pracy jest pikiem dominującym (dla rozwiązania (Q,π°)), tj. $T_{\min}(Q) = T_k(Q,\pi^{\circ})$. Z kolei, gdy $T_k(\pi)$ jest pikiem dominującym rozwiązania $(\mathrm{Q}),\pi \in \mathrm{AP}$, to warunkiem koniecznym zmniejszenia bieżącej wartości czasu cyklu jest taka zmiana rozwiązania (przydziału lub kolejności wykonywania zadań), która spowoduje zmniejszenie czasu pracy k -tego gniazda, tj. sumy (2). Należy jednak pamiętać, że zmiana ta może spowodować powstanie większego pika dominującego (tj. może się zwiększyć czas pracy innego gniazda, a więc i wartość czasu cyklu).

4. Algorytm wyznaczania czasu cyklu

Rozpatrujemy przykład cyklicznego problemu gniazdowego z przezbrojeniami maszyn. Niech Q $(\mathrm{Q} \in \mathrm{A})$ będzie pewnym ustalonym przydziałem operacji do maszyn. W tym przypadku (dla ustalonego Q) wyznaczenie minimalnego czsu cyklu $T_{\min}(\mathrm{Q})$ sprowadza się do rozwiązania cyklicznego problemu przepływowego, opisanego w pracy [3]. Generując przydziały i wyznaczając minimalne czasy cyklu można rozwiązać problem CNFS. Bazując na powyższej idei, schemat algorytmu przedstawia się następująco:

ALGORYTM (**ACC**)

$T^* := \infty$;

repeat

Krok 1: Wyznaczyć przydział operacji do maszyn;

Krok 2: Wyznaczyć (zgodnie z (5)) minimalny czas cyklu $T_{\min}(\mathrm{Q})$

if $T_{\min}(\mathrm{Q}) < T^*$ **then** $T^* := T_{\min}(\Theta)$;

until *Warunek_konca.*

Wyznaczenie optymalnej wartości czasu cyklu, dla ustalonego przydziału operacji do maszyn, (Krok 2 algorytmu) jest problemem NP-trudnym. Sprowadza się bowiem do rozwiązania cyklicznego problemu przepływowego z przezbrojeniami maszyn. Dokładny opis tego problemu oraz metody przybliżone jego rozwiazywania przedstawiono, między innymi, w pracach [3] i [4]. Zamieszczone tam algorytmy metaheurystyczne (przeszukiwania z tabu oraz simulowanego wyżarzania) będziemy stosowali w Kroku 2 algorytmu **ACC**. W dalszej części pracy skupiamy się na problemie wyznaczania przydziału operacji (Krok 1).

5. Wyznaczanie przydziału zadań do maszyn

Do wyznaczania przydziałów zadań do maszyn został zastosowany algorytm poszukiwania zstępującego DS (ang. *descending search*). Elementarny krok tego algorytmu polega na przeszukaniu otoczenia rozwiązania bazowego w celu znalezienia elementu o najmniejszej wartości funkcji celu, który to element przyjmujemy za rozwiązanie bazowe w następnej iteracji algorytmu. Proces poszukiwań jest kontynuowany dopóki wartość funkcji celu maleje. Generowanie otoczenia przydziałów operacji do maszyn polega na zmianie maszyny, na której ma być wykonana operacja. Jeśli zmiana przydziału poprawia wartość funkcji celu to jest ona zapamiętywana.

Niech $(\mathsf{Q}, p) \in \mathsf{AP}$ $(\mathsf{Q} = [\mathsf{Q}^1, \mathsf{Q}^2, \ldots, \mathsf{Q}^q])$ będzie tzw. rozwiązaniem bazowym (startowym) algorytmu (ustalona permutacja zadań $p \in \mathsf{P}$ nie zmienia się w czasie działania algorytmu). Przez $\mathsf{N}_k^1(\mathsf{Q})$ oznaczamy zbiór przydziałow, które w k-tym gnieździe różnią się od Q przydziałem dokładnie jednej operacją na jednej maszynie, tj.

$$\mathsf{N}_k^1(\mathsf{Q}) = \{\Theta \in \mathsf{A} : \exists l, t \in \mathsf{M}^k \; i$$

$$\exists v \in \mathsf{O}^k, \Theta^l = \mathsf{Q}^l \setminus \{v\}, \Theta^t = \mathsf{Q}^t \cup \{v\}, \Theta^i = \mathsf{Q}^i, i \neq l, t, i = 1,2,\ldots,q\}. \qquad (6)$$

Przypominamy, że M^k jest zbiorem maszyn k-tego gniazda, a O^k zbiorem operacji wykonywanych w tym gnieździe. Poniżej przedstawiamy algorytm wyznaczania suboptymalnego przydziału zadań do maszyn w k-tym gnieździe, w którym zastosowano otoczenie (6).

Algorytm PZM $(\mathsf{Q}, p; k)$

$T^\circ \leftarrow T^* \leftarrow T(\mathsf{Q}, p)$; $T \leftarrow \infty$; $\mathsf{Q}^* \leftarrow \mathsf{Q}$;

while $T^\circ < T$ **do**
begin
  $T \leftarrow T^\circ$;
  wygenerować otoczenie $\mathsf{N}_k^1(\mathsf{Q})$;
  wyznaczyć przydział $\mathsf{Q}^\circ \in \mathsf{N}_k^1(\mathsf{Q})$ taki, że
  $T^\circ = T(\mathsf{Q}^\circ, p) = \min\{T(\Theta, p) : \Theta \in \mathsf{N}_k^1(\mathsf{Q})\}$;
  if $T^\circ < T^*$ **then** $T^* \leftarrow T^\circ$ **and** $\mathsf{Q}^* \leftarrow \mathsf{Q}^\circ$;
  $Q \leftarrow Q^\circ$
end{*while*}

Jeżeli Q^* jest przydziałem zadań do maszyn wyznaczonym przez algorytm **PZM** $(Q, p; k)$, to przestawienie w k-tym gnieździe pojedynczej operacji na inną maszynę (tego samego gniazda) nie generuje przydziału o mniejszej wartości czasu cyklu

6. Gniazdo dwumaszynowe

W tym rozdziale rozpatrywać będziemy gniazdo zawierające tylko dwie maszyny. Przedstawimy algorytm suboptymalny (ze względu na czas pracy (2)) wyznaczania przydziału operacji do maszyn. Główne idee można uogólnić na większą liczbę maszyn.

Dla uproszczenia zapisu przyjmujemy następujące oznaczenia.

Dany jest zbiór zadań $\mathsf{J} = \{1,2,\ldots,n\}$, kolejność ich wykonywania $\pi \in \mathsf{P}$ oraz gniazdo składające się z dwóch maszyn $\mathsf{M} = \{1,2\}$. Każde zadanie należy wykonać na dokładnie jednej maszynie z M. Dane są także czasy $p_{ij}, i \in \mathsf{J}, j \in \mathsf{M}$ wykonywania zadań na maszynach oraz czasy przezbrojeń maszyn $s_{i,j}^{k}, k \in \mathsf{M}, i \neq j, i, j \in \mathsf{J}$, tj. jeżeli zadania i, j są wykonywane na tej samej k-tej maszynie oraz zadanie j jest wykonywane bezpośrednio po i, to przed rozpoczęciem wykonywania j należy przezbroić maszynę. Problem minimalizacji czasu pracy takiego gniazda (kryterium (2)) będziemy w skrócie oznaczali przez $\mathbf{GC}_{\max}$. Należy on do klasy problemów NP-trudnych. Dlatego, w Kroku 1 algorytmu **ACN** do wyznaczania przydziału operacji do maszyn będziemy stosowali algorytm przybliżony $\mathbf{PZM}(\mathsf{Q}, p; k)$. Poniżej przedstawiamy dokładny jego opis dla przypadku pojedynczego gniazda z dwoma maszynami.

Para zbiorów zadań $\mathsf{Q} = (\mathsf{Z}_1, \mathsf{Z}_2)$ jest przydziałem zadań do maszyn, jeżeli

$$\mathsf{Z}_1 \cup \mathsf{Z}_2 = \mathsf{J}, \quad \mathsf{Z}_1 \cap \mathsf{Z}_2 = \varnothing.$$

Przyjmujemy, że zadania ze zbioru Z_1 będą wykonywane na maszynie 1-szej, a z Z_2 na maszynie 2-giej.

Rozwiązaniu $(\mathsf{Q}, \pi) \in \mathsf{AP}$ (zobacz rozdział 2) przypisujemy *graf rozwiązania* $H(\mathsf{Q}, \pi) = (\mathsf{S}, \mathsf{T})$ z obciążonymi wierzchołkami oraz łukami. Zbiór wierzchołków $\mathsf{S} = \{1,2,\ldots,n,n+1,n+1,\ldots,2n\}$, gdzie n jest liczbą zadań. Waga wierzchołka i $(i = 1,2,\ldots,n)$ jest równa $p_{i,1}$, a wierzchołka j $(j = n+1, n+2,\ldots,2n)$ wynosi $p_{j-n,2}$. Zbiór łuków $\mathsf{T} = \mathsf{X} \cup \mathsf{Y}$. Łuki ze zbioru X reprezentują przezbrojenia maszyny, a ze zbioru Y - kolejność wykonywania zadań (permutację π). Jeżeli zadanie j jest wykonywane bezpośrednio po i oraz są one wykonywane:

- na tej samej maszynie, to łuk $(i, j) \in \mathsf{X}$, a jego waga jest równa czasowi przezbrojenia maszyny pomiędzy i oraz j,
- na różnych maszynach, to $(i, j) \in \mathsf{Y}$, a waga tego łuku wynosi 0.

6.1. Zmiana przydziału operacji do maszyn

Niech $\mathsf{Q} = (\mathsf{Z}_1, \mathsf{Z}_2)$ będzie przydziałem zadań do maszyn, a π kolejnością ich wykonywania w dwumaszynowym gnieździe. Przydział ten może być jednoznacznie reprezentowany przez ciąg binarny $q = (q_1, q_2, \ldots, q_n)$, gdzie

$$q_i = \begin{cases} 1, & \text{if } i \in \mathsf{Z}_1, \\ 0, & \text{if } i \in \mathsf{Z}_2. \end{cases}$$

Obecnie zbadamy, jak zmieni się czas pracy gniazda $T(\mathsf{Q}, \pi)$, jeżeli zmienimy przydział pewnego zadania, tj. zmienimy maszynę, na której ma być ono wykonywane. Niech $l \in \mathsf{J}$ będzie pewnym zadaniem. Osobno rozpatrujemy dwa przypadki (w zależności od maszyny na której zadanie jest wykonywane).

Przypadek 1.

Zakładamy, że zadanie l jest wykonywane na pierwszej maszynie, tj. $q_l = 1$. Niech α^- będzie zadaniem bezpośrednio poprzedzającym zadanie l-te na pierwszej maszynie. Jeżeli l jest pierwszym wykonywanym zadaniem, to przyjmujemy $\alpha^- = 0$ (w tym przypadku czas przezbrojenia maszyny $s^1_{0,l} = 0$). Podobnie, definiujemy zadanie α^+, które jest wykonywane bezpośrednio za l-tym na pierwszej maszynie (w szczególności może to być zadanie z drugiego *MPS*-a). Dalej, przez β^- i β^+ oznaczamy zadania wykonywane bezpośrednio przed i za l-tym zadaniem, gdyby było one wykonywane na drugiej maszynie.

Zmieniamy przydział l-tego zadania z pierwszej na drugą maszynę. W tym przypadku nowy przydział $\mathsf{Q}' = (\mathsf{Z}_{1'}, \mathsf{Z}_{2'},)$ gdzie $\mathsf{Z}_{1'} = \mathsf{Z}_1 \setminus \{l\}$ oraz $\mathsf{Z}_{2'} = \mathsf{Z}_2 \cup \{l\}$. Ciąg binarny dla tego rozwiązania $q' = (q'_1, q'_2, \ldots, q'_n)$, gdzie $q'_i = q_i$, $i = 1,2,\ldots,l-1,l+1,\ldots,n$ oraz $q'_l = 0$.

Jeżeli zmieniamy przydział zadania l z pierwszej na druga maszynę (tj. zmieniamy wartość zmiennej q_i z 1 na 0), to graf nowego rozwiązania $H(Q', \pi)$ otrzymujemy z grafu $H(Q, \pi)$ usuwając łuki:

(a) (α^-, l) oraz (l, α^+), na pierwszej maszynie,

(b) (β^-, β^+), na maszynie drugiej,

a następnie wstawiając nowe łuki:

(a') (α^-, α^+), na pierwszej maszynie,

(a') (β^-, l) oraz (l, β^+), na maszynie drugiej.

Wyrażenie

$$\Delta^1_k = s^1_{\alpha^-,\alpha^+} + s^2_{\beta^-,l} + s^2_{l,\beta^+} - s^1_{\alpha^-,l} - s^1_{l,\alpha^+} - s^2_{\beta^-,\beta^+} - p_{l,1} + p_{l,2} \tag{7}$$

jest sumą czasów przezbrojeń maszyn reprezentowanych przez łuki dodane do grafu rozwiązania $H(\mathsf{Q}, \pi)$, od której jest odjęta suma łuków usuniętych. Dwa ostatnie składniki są czasmi wykonywania zadania l odpowiednio na pierwszej i drugiej maszynie. Niech Q będzie pewnym przydziałem zadań do maszyn w dwumaszynowym gnieździe, w którym zadanie $l \in \mathsf{J}$ jest wykonywane na pierwszej maszynie. Jeżeli zmienimy przydział z Q na Q' przydzielając zadaniu l drugą maszynę, wówczas czas pracy gniazda (wartość wyrażenia (2))

$$T(\mathsf{Q}', \pi) = T(\mathsf{Q}, \pi) + \Delta^1_l. \tag{8}$$

Dla dowodu wystarczy, korzystając z wyrażenia (7) oraz z definicji (2), obliczyć czas pracy gniazda dla rozwiązania (Q', π), tj. wartość $T(\mathsf{Q}', \pi)$.

Przypadek 2.

Rozpatrujemy przypadek przydziału $\mathsf{Q} = (\mathsf{Z}_1, \mathsf{Z}_2)$, w którym zadanie $l \in \mathsf{J}$ jest wykonywane na drugiej maszynie, tj. $l \in \mathsf{Z}_2$, a więc $q_l = 0$. Podobnie, jak w Przypadku 1, definiujemy zadania γ^- oraz γ^+ wykonywane bezpośrednio przed i za l - tym zadaniem na

drugiej maszynie. Podobnie, δ^- i δ^+ odpowiednio zadanie występujące bezpośrednio przed i za l-tym zadaniem, gdyby było one wykonywane na pierwszej maszynie. Tak jak w przypadku (7) wprowadzamy oznaczenie

$$\Delta_l^2 = s^2_{\gamma^-,\gamma^+} + s^2_{\delta^-,l} + s^2_{l,\delta^+} - s^2_{\gamma^-,l} - s^2_{k,\gamma^+} - s^2_{\delta^-,\delta^+} - p_{l,2} + p_{l,1}. \quad (9)$$

Niech Q będzie pewnym przydziałem zadań do maszyn w dwumaszynowym gnieździe, w którym zadanie $l \in \mathsf{J}$ jest wykonywane na drugiej maszynie. Jeżeli zmienimy przydział z Q na Q' przydzielając zadaniu l pierwszą maszynę, wówczas czas pracy gniazda (wartość wyrażenia (2))

$$T(\mathsf{Q}',\pi) = T(\mathsf{Q},\pi) + \Delta_l^2. \quad (10)$$

Dowód jest podobny, jak w Przypadku 1.

Wartość wyrażenia (7) lub (9) może być policzona w czasie $O(n)$.

Jeżeli w algorytmie wynosi **PZM** otoczenie jest generowane przez zmianę przydziału polegającą na przeniesieniu pojedynczego zadania na inną maszynę (w dwumaszynowym gnieździe), wówczas wartość wyrażenia (7) lub (9) odpowiada zmianie wartości czasu pracy gniazda (wartości pika). Wobec tego z otoczenia wybieramy przydział, który minimalizuje wartość pika, tj. wartość wyrażenia (8) lub (10).

7. Eksperymenty obliczeniowe

Eksperymenty obliczeniowe wykonano na dwóch wersjach algorytmu **ACC** rozwiązywania cyklicznego gniazdowego problemu przepływowego z przezbrojeniami maszyn. W pierwszej ($\mathbf{A}_{TS}$), w kroku 2 zastosowano algorytm przeszukiwania z tabu, a w drugiej ($\mathbf{A}_{SW}$) - symulowanego wyżarzania. W kroku 1 algorytmu **ACC**, przydział operacji do maszyn jest wyznaczany przez algorytm **PZM**$(\mathsf{Q}, p; k)$ (dla k=1). Algorytmy zaimplementowano w języku C++.

Dane testowe generowano na bazie przykładów z pracy Ruiza i Stutzle [17], gdzie jest rozpatrywany problemu przepływowy z przezbrojeniami maszyn. Przedstawiono tam 60 przykładów podzielonych na sześć grup (po dziesięć w każdej grupie) o liczbie zadań $n = 20$ i 50 oraz maszyn $m = 5{,}10$ i 20. Eksperymenty obliczeniowe zostały przeprowadzone na klastrze Bem we Wrocławskim Centrum Sieciowo–Superkomputerowym (grant nr 96) pracującym pod kontrolą 64–bitowego systemu operacyjnego Scientific Linux 6.7 (Carbon) wyposażonym w procesory Intel Xeon E5-2670 (2.30GHz). Oba algorytmy $\mathbf{A}_{TS}$ i $\mathbf{A}_{SW}$ zostały uruchomione dla liczby iteracji $it = 200$. Otrzymane wyniki porównano z wynikami algorytmu konstrukcyjnego NEH [15]. Dla każdego przykładu wyznaczono procentowy błąd względny (poprawę rozwiązania)

$$PRD = \frac{F_{NEH} - F_{alg}}{F_{NEH}} \cdot 100\% \quad (11)$$

gdzie, F_{NEH} jest wartością funkcji celu dla rozwiązania referencyjnego wyznaczonego przez NEH, a F_{alg} jest wartością rozwiązania uzyskanego przez badany algorytm. Średnie tych błędów przedstawiono w Tabeli 2. Na podstawie zamieszczonych tam wyników możemy stwierdzić, że zastosowanie w Kroku 2 algorytmu **ACC** metody TS daje rozwiązania o lepszej jakości (w sensie wartości funkcji celu) niż w przypadku użycia symulowanego wyżarzania. Chociaż, średnie poprawy (rozwiazania algortytmu, w którym

stosowano NEH) są podobne i wynosza odpowiednio -21.7% i 19.0%. Podobnie, czasy obliczeń (w milisekundach) obu wersji algorytmu algorytmu **ACC** różnią się niewiele.Dla najwiekszych przykładów czas obliczeń nie przekracza 2 sek.

Tab. 2. Średni błąd względny *PRD* oraz czas obliczeń (*LS*) algorytmów $\mathbf{A}_{TS}$ oraz $\mathbf{A}_{SA}$.

$n \times m$	$\mathbf{A}_{TS}$	LS_{TS}	$\mathbf{A}_{SA}$	LS_{SA}
20×5	-27.5	12.124	-25.6	11.82
20×10	-22.8	18.219	-20.6	22.46
20×20	-20.3	31.964	-19.3	42.51
50×5	-23.3	620.39	-18.9	403.42
50×10	-19.7	1029.08	-16.1	811.72
50×20	-16.4	1841.19	-13.5	1771.86
średnia	-21.7		-19.0	

8. Podsumowanie

W pracy rozpatrujemy cykliczny problem gniazdowy z przezbrojeniami maszyn. Przedstawiamy dwu etapową metodę jego rozwiązania. W etapie pierwszym przydzielamy operacje do maszyn a w drugim, wyznaczamy kolejności wykonywania operacji na maszynach. Przedstawiamy algorytmy przybliżone rozwiązywania rozpatrywanego zagadnienia. Do wyznaczania przydziału operacji stosujemy algorytm typu popraw. Natomiast, do ustalanie kolejności wykonywania operacji korzystamy z przeszukiwania z tabu lub symulowanego wyżarzania. Porównujemy także czasy wykonywania oraz wartości wyznaczonych rozwiązań.

Literatura

1. Błażewicz J., Ecker K., Pesch E., Schmidt G., Węglarz J.: Handbook on Scheduling: From Theory to Applications, Springer, 2007.
2. Bożejko W., Gniewkowski Ł., Pempera J., Wodecki M., Cyclic hybrid flow-shop scheduling problem with machine setups, Procedia Computer Science, 2014, 29, 2127-2136.
3. Bożejko W., Uchroński M., Wodecki M., Block approach to the cyclic flow shop scheduling, Computers & Industrial Engineering, 2015, 81, 158-166.
4. Bożejko W., Uchroński M., Wodecki M., Parallel metaheuristics for the cyclic flow shop scheduling problem, Computers & Industrial Engineering, 2016, 95, 156–163.
5. Brucker P., Kampmeyer T., Tabu search algorithm for cyclic machine scheduling problems, Journal of Scheduling, 2005, 8, 303–322.
6. Brucker P., Kampmeyer T., Cyclic flow shop scheduling problems with blocking, Ann. Oper. Res., 2008, 159, 161–181.
7. Caggiano K., Jackson P.L.: Finding minimum flow time cyclic schedules for non-identical, multistage jobs. IIE Transactions, 40, 2008, 45-65.
8. Crama Y, Kats V., Van de Klundert, Levner E., Cyclic scheduling in robotic flowshop, Annals of Operations Research, 96, 2000, 97-124.
9. Crowston N.B., Wagner M., Williams J.F., Economic lot size detrmination in multi-stage assembly systems, Management Science, 19(5), 1973, 517–527.
10. Grabowski J., Wodecki M.: A very fast tabu search algorithm for the permutation flow

shop problem ith makespan criterion, Computers and Operations Research, 2004, 31, 1891–1909.
11. Hanen C., Munier A., Cyclic scheduling on parallel processors: An overview. In P. Chretienne P., Coffman E.G., Lenstra J.K., Liu Z., editors, Theory and its Applications, John Wiley&Sons, 1995.
12. Jeong-Won Seo, Tae-Eog Lee, Steady-state analysis and scheduling of cyclic job shop with overtaking, International Journal of Flexibe Manufacturing Systems, 14(4), 2002, 291–318.
13. Kampmeyer T., Cyclic Scheduling Problems, Ph.D. Dissertation, Universitat Osnabruck, 2006.
14. Levner E., Kats V, Lopez A.P., Cheng T.C.E.: Complexity of cyclic scheduling problems: A state-of-the-art survey. Computers and Industrial Engineering, 59, 2010, 352-361.
15. Nawaz, M., Enscore, E. E., Jr, Ham, I., A heuristic algorithm for the m– machine, n–job flow–shop sequencing problem. OMEGA International Journal of Management Science, 11, 1983, 91–95.
16. Nowicki E., Smutnicki C.: A Fast tabu serach algorithm for permutation flow shop problem,European Journal of Operational Research, 91, 1996, 160–175.
17. Ruiz, R., Stützle, T., An Iterated Greedy heuristic for the sequence dependent setup times flowshop problem with makespan and weighted tardiness objectives. European Journal of Operational Research, 187(3), 2008, 1143–1159.
18. Sawik T., Batch versus cyclic scheduling of flexible flow shop by mixed-integer programming, International Journal of Production Research, 50(18), 2012, 5017–5034.
19. Sawik T., A mixed integer programming for cyclic scheduling of flexible flow lines, Bulletin of the Polish Academy of Sciences, Technical Sciences, 62(1), 2014, 121–128.

Dr hab. Wojciech BOŻEJKO, prof. nadzw.
Dr inż. Mariusz UCHROŃSKI
Katedra Automatyki, Mechatroniki i Systemów Sterowania
Wydział Elektroniki, Politechnika Wrocławska
Wyb. Wyspiańskiego 27, 50-370 Wrocław
e-mail: wojciech.bozejko@pwr.wroc.pl

Mgr inż. Piotr NADYBSKI
Państwowa Wyższa Szkoła Zawodowa
im. Witelona w Legnicy
ul. Sejmowa 5a,
e-mail: nadybskip@pwsz.legnica.edu.pl

Dr hab. Mieczysław WODECKI, prof. nadzw.
Instytut Informatyki Uniwersytetu Wrocławskiego
ul. Joliot-Curie 50-383 Wrocław
e-mail: mieczyslaw.wodecki@ii.uni.wroc.pl