ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1982

Seria: AUTOMATYKA z. 64 Nr kol. 736

Franciszek MARECKI
Elżbieta ZIELIŃSKA-KRÓL

Instytut Automatyki
Politechniki Śląskiej

MODEL MATEMATYCZNY I ALGORYTM HARMONOGRAMOWANIA PROCESU REGENERACJI WALCÓW

Streszczenie. W pracy przedstawiono model matematyczny procesu regeneracji walców dla potrzeb walcowni kęsów. Dla wyznaczenia optymalnego harmonogramu procesu przyjęto kryterium maksymalizacji minimalnego wyprzedzania realizacji zadań. Do rozwiązania zadania wykorzystano programowanie wieloetapowe.

1. WPROWADZENIE

W procesie walcowania kęsów (przedstawionym w pracach [1], [2], [3] i [4]) określa się harmonogram walcowania partii wyrobów różnego asortymentu. Cechą charakterystyczną walcowni ciągłej kęsów (WCK) jest konieczność wymiany walców zużytych w procesie walcowania. Walce należy wymieniać w takich chwilach, by zminimalizować sumaryczny przestój WCK. Zakłada się przy tym, że walec zastępczy (tzw. dubler) jest dostępny w momencie wymiany. Innymi słowy, harmonogram walcowania wymusza terminy zakończenia regeneracji odpowiednich walców.

W procesie walcowania biorą na ogół udział dwa złożenia walców każdego typu (podstawowe i dubler). Walec zużyty jest poddawany regeneracji. W czasie gdy pracuje walec podstawowy, jego dubler jest regenerowany. Najwcześniejsze terminy oddania walców do regeneracji oraz najpóźniejsze terminy zakończenia regeneracji wynikają z harmonogramu procesu walcowania.

Proces regeneracji walców składa się z dwóch faz: napawania zużytej powierzchni oraz toczenia do odpowiednich wymiarów. Do tego celu wykorzystywany jest równoległy układ napawarek połączony szeregowo z równoległym układem tokarek. Dla każdego typu walca określone są alternatywne marszruty technologiczne przechodzące przez odpowiednią napawarkę i tokarkę. Z uwagi na stopień zużycia i typ walca określone są czasy jego obróbki na każdym agregacie (napawarce i tokarce).

Celem spełnienia wymagań narzuconych przez harmonogram nadrzędny (walcowania) należy wyznaczyć harmonogram podrzędny (regeneracji). Harmonogram regeneracji winien określać dla każdego walca przedział czasu, w którym walec jest obrabiany na wyznaczonej napawarce i tokarce. Zakłada się przy

tym, że harmonogram nadrzędny nie może być zmieniony. W praktyce, poszukując dopuszczalnego harmonogramu procesu regeneracji walców, staramy się uzyskać pewien zapas czasu dla każdego zregenerowanego walca (tzn. staramy się zakończyć regenerację przed terminem). Z tego względu można sformułować minimaksowe kryterium optymalnego harmonogramowania regeneracji walców. Kryterium to polega na maksymalizacji minimalnego zapasu czasu dla zregenerowanych walców.

W niniejszym referacie przedstawiony jest model matematyczny i algorytm harmonogramowania tak sformułowanego problemu regeneracji walców.

2. MODEL MATEMATYCZNY

Załóżmy, że dany jest system składający się z dwóch podsystemów połączonych szeregowo. Każdy podsystem stanowi układ równolegle pracujących agregatów.

Zbiór wszystkich agregatów zapiszemy w postaci:

$$\mathcal{A} = \left[A_{m,j}\right]_{\substack{(m=\overline{1,M_j}) \\ (j=\overline{1,2})}} \tag{1}$$

gdzie:

$A_{m,j}$ - m-ty agregat j-tego podsystemu,

M_j - liczba agregatów j-tego podsystemu.

Dany jest zbiór zadań:

$$\Omega = \left\{\omega_n\right\}_{(n=\overline{1,N})} \tag{2}$$

gdzie:

ω_n - n-te zadanie,

N - liczba zadań.

Zadaniem w omawianym problemie jest proces regeneracji walca.

Załóżmy, że dane są alternatywne marszruty technologiczne zadań poprzez agregaty. Każde zadanie poddawane jest operacjom najpierw na jednym agregacie pierwszego podsystemu, a następnie przechodzi przez jeden agregat drugiego podsystemu. Możliwość realizacji zadania ω_n na agregacie $A_{m,j}$ zapiszemy w macierzy:

$$U_j = \left[u_{j,m,n}\right] \tag{3}$$

Elementy tej macierzy definiowane są następująco:

$$u_{j,m,n} = \begin{cases} 1; \text{ jeśli } \omega_n \text{ może być realizowane na } A_{m,j} \\ 0; \text{ w przeciwnym przypadku} \end{cases}$$

Niechaj czasy realizacji zadań na poszczególnych agregatach będą dane macierzą:

$$\Theta = [\vartheta_{j,m,n}] \tag{4}$$

gdzie:

$\vartheta_{j,m,n}$ - czas realizacji zadania ω_n na agregacie $A_{m,j}$.

Załóżmy, że dane są terminy dostępności agregatów w punkcie startowym rozpatrywanego okresu harmonogramowania:

$$R_j = [r_{j,m}] \tag{5}$$

gdzie:

$r_{j,m}$ - termin dostępności agregatu $A_{m,j}$.

Ponadto założymy, że dane są terminy dostępności zadań, zapisane wektorem:

$$\Phi = [\varphi_n] \tag{6}$$

gdzie:

φ_n - termin dostępności zadania ω_n.

Analogicznie zapiszemy terminy najpóźniejszego zakończenia realizacji zadań:

$$\Psi = [\psi_n] \tag{7}$$

gdzie:

ψ_n - termin najpóźniejszego zakończenia zadania ω_n.

Termin φ_n nie może być wyprzedzony, a termin ψ_n nie może być przekroczony.

Rozpatrzmy harmonogram pracy walcowni. Walcownia nie pracuje w czasie wymiany walców. Przeanalizujemy okres harmonogramowania od t_o do t^o. Zadaniami będzie regeneracja tych walców, które mają być zamontowane na WCK w okresie harmonogramowania (t_o-t^o). Aby walec zamontować, należy wcześ-

niej wymontować walec zużyty. Tak więc w chwili t_0 niektóre zadania mogą oczekiwać na regenerację. Zadaniami są również te walce, które winny być wymontowane i zamontowane w okresie harmonogramowania. Natomiast walce, które są wymontowane przed upływem t^0, lecz mają być zamontowane po t^0, nie stanowią zadań dla rozpatrywanego okresu harmonogramowania procesu regeneracji.

Podstawowe znaczenie ma następujące spostrzeżenie: zadanie ω_ϑ musi poprzedzać wszystkie zadania ω_n, dla których zachodzi warunek:

$$\psi_\vartheta < \varphi_n \tag{8}$$

Wynika to z faktu, że zgodnie z podstawowym harmonogramem walcowni najpierw należy zamontować walec ω_ϑ, następnie walcować do chwili φ_n, w której trzeba wymontować walec ω_n. Stąd zadania ω_n nie są dostępne do zrealizowania zadania ω_ϑ.

Analizując terminy φ_n oraz ψ_ϑ w podstawowym harmonogramie walcowni można określić macierz poprzedników i następników:

$$\Gamma = \left[\gamma_{\vartheta,n}\right]_{\substack{(\vartheta=\overline{1,N})\\(n=1,N)}} \tag{9}$$

Elementy tej macierzy definiujemy następująco:

$$\gamma_{\vartheta,n} = \begin{cases} 1; & \text{jeśli } \omega_\vartheta \text{ jest poprzednikiem } \omega_n; \\ 0; & \text{w przeciwnym przypadku} \end{cases}$$

Tak więc w procesie regeneracji walców występują ograniczenia kolejnościowe.

W trakcie realizacji zadań (regeneracji walców) agregaty są obsługiwane przez operatorów. Z tego względu w praktyce unika się przestojów agregatów. W modelu matematycznym procesu regeneracji załóżmy, że agregaty nie mogą mieć przestojów.

Oznaczmy przez t_n moment zakończenia realizacji zadania ω_n. Wtedy kryterium optymalnego harmonogramowania zadań zapiszemy w postaci:

$$Q = \min_{1 \leqslant n \leqslant N} (\psi_n - t_n) \longrightarrow \max \tag{10}$$

Przy powyższych założeniach problem harmonogramowania może nie mieć rozwiązania (z uwagi na graniczne ψ_n). Jeżeli istnieje rozwiązanie problemu, to kryterium (10) pozwala wyznaczyć harmonogram maksymalizujący minimalne wyprzedzenie realizacji zadań.

3. ALGORYTM

Do rozwiązania problemu opracowano algorytm oparty na programowaniu wieloetapowym. Algorytm ten pozwala uzyskać rozwiązanie optymalne lub najlepsze ze zbioru wyznaczonych rozwiązań dopuszczalnych. Rozwiązywany problem należy do klasy problemów NP-zupełnych. Dlatego stosowanie algorytmu o wykładniczej złożoności obliczeniowej lub algorytmu heurystycznego jest uzasadnione.

Podstawowymi elementami algorytmu są: stan procesu decyzyjnego, wartość stanu, procedura generowania stanów oraz procedura eliminowania stanów nieperspektywicznych.

3.1. Stan procesu decyzyjnego i wartość stanu

Oznaczmy przez φ ($\varphi = \overline{1,2N}$) numer etapu decyzyjnego oraz przez λ ($\lambda = 1, L\varphi$) numer stanu w ramach etapu ($L\varphi$ jest liczbą stanów w φ-tym etapie).

Definicja 1. Stan procesu decyzyjnego jest macierzą:

$$p^{\lambda,\varphi} = \left[p^{\lambda,\varphi}_{n,i}\right]_{\substack{(n=\overline{1,N})\\(i=\overline{1,4})}} \tag{11}$$

Elementy macierzy (11) określamy następująco:

$$p^{\lambda,\varphi}_{n,1} = \begin{cases} \mu & \text{- jeśli zadanie } \omega_n \text{ zostało przydzielone do } A_{\mu,1} \\ 0 & \text{- w przeciwnym przypadku} \end{cases} \tag{12a}$$

$$p^{\lambda,\varphi}_{n,2} = \begin{cases} t_{\mu,n} & \text{- jeśli } p^{\lambda,\varphi}_{n,1} = \mu \\ 0 & \text{- w przeciwnym przypadku} \end{cases} \tag{12b}$$

$$p^{\lambda,\varphi}_{n,3} = \begin{cases} m & \text{- jeśli zadanie } \omega_n \text{ zostało przydzielone do } A_{m,2} \\ 0 & \text{- w przeciwnym przypadku} \end{cases} \tag{12c}$$

$$p^{\lambda,\varphi}_{n,4} = \begin{cases} t_{m,n} & \text{- jeśli } p^{\lambda,\varphi}_{n,3} = m \\ 0 & \text{- w przeciwnym przypadku} \end{cases} \tag{12d}$$

gdzie:

$t_{\mu,n}$ - moment zakończenia realizacji zadania ω_n na agregacie $A_{\mu,1}$,

$t_{m,n}$ - moment zakończenia realizacji zadania ω_n na agregacie $A_{m,2}$.

Zatem stan początkowy $P^{1,0}$ jest macierzą zerową a stany końcowe $P^{\lambda,2N}$ mają wszystkie elementy dodatnie. Z każdego stanu $P^{\lambda,\eta}$ można wyznaczyć harmonogram realizacji zadań przydzielonych do stanu.

Z każdym stanem $P^{\lambda,\eta}$ jest związana jego wartość, którą oznaczymy przez $V^{\lambda,\eta}$. Wartość stanu interpetuje przyjęte kryterium (10).

Definicja 2: Wartość stanu jest liczbą wyznaczoną następująco:

$$V^{\lambda,\eta} = \max_{n \in \sigma^{\lambda,\eta}} (\psi_n - p^{\lambda,\eta}_{n,4}) \tag{13}$$

przy tym:

$$\bigvee_n (p^{\lambda,\eta}_{n,4} > 0) \Rightarrow (n \in \sigma^{\lambda,\eta}) \tag{14}$$

Dla stanu $P^{1,0}$ przyjmujemy wartość zerową. Wartość każdego stanu końcowego $P^{\lambda,2N}$ określa najmniejsze wyprzedzenie czasowe (dla jednego z zadań). Zgodnie z kryterium (10) optymalny stan końcowy wyznaczamy z warunku:

$$\left(\max_{1 \leq \lambda \leq L_{2N}} V^{\lambda,2N} = V^{\lambda^o,2N} \right) \Rightarrow (P^{\lambda^o,2N} = P^o) \tag{15}$$

gdzie:

P^o - stan optymalny.

Ze stanu optymalnego odczytujemy wprost optymalny harmonogram realizacji zadań.

3.2. Generowanie stanów

Procedura generowania stanów polega na uzupełnieniu stanu $P^{1,\eta-1}$ o zadanie ω_n tak, by otrzymać dopuszczalny stan $P^{\lambda,\eta}$. Należy zwrócić uwagę na fakt, że przydzielając zadanie ω_n na $A_{\mu,1}$ wszystkie zadania ω_ν, dla których $\eta_{\nu,n} = 1$, muszą być zrealizowane w drugim podsystemie. Ponadto przydzielając zadanie ω_n na $A_{m,2}$ nie sprawdzamy ograniczeń kolejnościowych, ponieważ poprzedniki ω_n musiały być wykonane, skoro ω_n zrealizowano w pierwszym podsystemie.

Oznaczmy przez $T_j^{l,\eta-1}$ - czas dostępności j-tego podsystemu w stanie $(l,\eta-1)$

$$T_j^{l,\eta-1} = \min_{1 \le m \le M_j} T_{j,m}^{l,\eta-1} \tag{16}$$

gdzie:

$$T_{j,m}^{l,\eta-1} = \begin{cases} r_{j,m} = \left| \sigma_m^{l,\eta-1} \right| = 0 \\ \max_{i \in \sigma_m^{l,\eta-1}} p_i^{l,\eta-1} \quad : \text{- w przeciwnym przypadku} \end{cases} \tag{16a}$$

oraz:

$$\bigvee_i \left(p_{i,2j-1}^{l,\eta-1} = m \right) \Rightarrow \left(i \in \sigma_m^{l,\eta-1} \right) \tag{16b}$$

omawianym problemie można wyróżnić 4 procedury generowania stanów:

Procecura I - startowa (zadania znajdują się wyłącznie przed pierwszym odsystemem), a więc:

$$\sum_{i=1}^{N} p_{i,1}^{l,\eta-1} = 0 \tag{17}$$

ąd mamy:

$$\bigvee_n \bigvee_\mu \left(\sum_{i=1}^{N} \vartheta_{i,n} = 0 \right) \wedge \left(u_{1,\mu,n} = 1 \right) \wedge \left(\varphi_n \le r_{1,\mu} \right) \wedge \left(t_n^* \le \Phi_n \right) \Rightarrow$$

$$\Rightarrow \left(P^{\lambda,\eta} = P^{l,\eta-1} + \Delta P \right) \tag{18}$$

lementy macierzy ΔP mają postać:

$$\Delta P_{i,1} = \begin{cases} \mu: \quad i = n \\ 0: \quad \text{w pozostałych przypadkach} \end{cases} \tag{18a}$$

$$\Delta p_{i,2} = \begin{cases} t_{1,\mu,n} : i = n \\ 0 \quad \text{w przeciwnym przypadku} \end{cases} \tag{18b}$$

$$\left.\begin{aligned} \Delta p_{i,3} &= 0 \\ \Delta p_{i,4} &= 0 \end{aligned}\right\} \tag{18c}$$

Termin $t_{1,\mu,n}$ obliczamy na podstawie zależności:

$$t_{1,\mu,n} = r_{1,\mu} + \vartheta_{1,\mu,n} \tag{18d}$$

natomiast t_n^* (optymistyczny termin zakończenia realizacji zadania w systemie) obliczamy następująco:

$$t_n^* = \min_{m \in \beta_n^{1,\eta-1}} (p_{n,2}^{1,\eta-1} + \vartheta_{2,m,n}) \tag{18e}$$

przy tym:

$$\bigvee_m (u_{2,m,n} = 1) \wedge (p_{n,2}^{1,\eta-1} \geqslant T_{2,m}^{1,\eta-1}) \Rightarrow (m \in \beta_n^{1,\eta-1}) \tag{18f}$$

Rzeczywisty termin zakończenia realizacji zadania ω_n nie może być wcześniejszy od t_n^*.

Procedura II - końcowa (wszystkie zadania zostały zrealizowane w pierwszym podsystemie), a więc:

$$\bigvee_{1 \leqslant i \leqslant N} (p_{i,1}^{1,\eta-1} > 0) \tag{19}$$

Stąd mamy:

$$\bigvee_n \bigvee_m (u_{2,m,n} = 1) \wedge (p_{n,2}^{1,\eta-1} \leqslant T_{2,m}^{1,\eta-1}) \wedge (p_{n,2}^{1,\eta-1} + \vartheta_{2,m,n} \leqslant \psi_n) \Rightarrow$$

$$\Rightarrow (P^{\lambda,\eta} = P^{1,\eta-1} + \Delta P) \tag{20}$$

Elementy macierzy ΔP mają postać:

$$\left.\begin{aligned} \Delta p_{i,1} &= 0 \\ \Delta p_{i,2} &= 0 \end{aligned}\right\} \tag{20a}$$

$$\Delta p_{i,3} = \begin{cases} m : i = n \\ 0 : \text{w przeciwnym przypadku} \end{cases} \tag{20b}$$

$$\Delta p_{i,4} = \begin{cases} t_{2,m,n} : i = n \\ 0 : \text{w przeciwnym przypadku} \end{cases} \tag{20c}$$

Termin $t_{2,m,n}$ obliczamy z zależności:

$$t_{2,m,n} = T_{2,m}^{1,\eta-1} + \vartheta_{2,m,n} \tag{20d}$$

Procedura III - zadania przydzielamy na agregaty I podsystemu, ponieważ podsystem ten zwalnia się wcześniej niż II podsystem, a więc:

$$T_1^{1,\eta-1} \leqslant T_2^{1,\eta-1} \tag{21}$$

Stąd mamy:

$$\bigvee_n \bigvee_\nu \bigvee_\mu (p_{n,i}^{1,\eta-1} = 0) \wedge \left\{ \left(\sum_{i=1}^{N} \vartheta_{i,n} = 0\right) \vee \left[(\vartheta_{i,n} = 1) \Rightarrow (p_{\nu,3}^{1,\eta-1} > 0)\right] \wedge \right.$$

$$\left. \wedge (T_{i,\mu}^{1,\eta-1} \leqslant T_2^{1,\eta-1}) \wedge (u_{i,\mu,n} = 1) \wedge (\varphi_n \leqslant T_{i,\mu}^{1,\eta-1}) \wedge (\varepsilon_n^* \leqslant \psi_n) \right\} \Rightarrow$$

$$\Rightarrow (P^{\lambda,\eta} = P^{I,\eta-1} + \Delta P) \tag{22}$$

Elementy macierzy ΔP mają postać:

$$\Delta p_{i,1} = \begin{cases} \mu : i = n \\ 0 : \text{w przeciwnym przypadku} \end{cases} \tag{22a}$$

$$\Delta p_{i,2} = \begin{cases} t_{1,\mu,n} & i = n \\ 0 : \text{w przeciwnym przypadku} \end{cases} \tag{22b}$$

$$\left.\begin{aligned} \Delta p_{i,3} &= 0 \\ \Delta p_{i,4} &= 0 \end{aligned}\right\} \tag{22c}$$

Termin $t_{1,\mu,n}$ wyznaczamy ze wzoru:

$$t_{1,\mu,n} = T^{1,\eta-1}_{1,\mu} + \vartheta_{1,\mu,n} \tag{22d}$$

Natomiast t^{*}_{n} obliczamy wg (18e) i (18f).

Procedura IV - zadanie przydzielamy na agregaty II podsystemu, ponieważ podsystem ten zwalnia się wcześniej niż I podsystem, a więc:

$$T^{1,\eta-1}_{1} > T^{1,\eta-1}_{2} \tag{23}$$

Stąd mamy:

$$\bigvee_{n}\bigvee_{m}(p^{1,\eta-1}_{n,1} > 0) \wedge (p^{1,\eta-1}_{n,3} = 0) \wedge (T^{1,\eta-1}_{2,m} < T^{1,\eta-1}_{1}) \wedge (u_{2,m,n} = 1) \wedge$$

$$\wedge (p^{1,\eta-1}_{n,2} \leq T^{1,\eta-1}_{2,m}) \wedge (T^{1,\eta-1}_{2,m} + \vartheta_{2,m,n} \leq \phi_{n}) \Rightarrow$$

$$\Rightarrow (P^{\lambda,\eta} = P^{1,\eta-1} + \Delta P) \tag{24}$$

Elementy macierzy ΔP mają postać:

$$\left.\begin{aligned} \Delta p_{i,1} &= 0 \\ \Delta p_{i,2} &= 0 \end{aligned}\right\} \tag{24a}$$

$$\Delta p_{i,3} = \begin{cases} m : & i = n \\ 0 : & \text{w przeciwnym przypadku} \end{cases} \tag{24b}$$

$$\Delta P_{i,4} = \begin{cases} t_{2,m,n} : i = n \\ 0 : \text{w przeciwnym przypadku} \end{cases} \tag{24c}$$

Termin $t_{2,m,n}$ obliczamy wg (20d).

3.3. Eliminacja stanów

Celem eliminacji stanów jest pominięcie tych stanów, które nie prowadzą do dopuszczalnego rozwiązania problemu zgodnie z przyjętymi ograniczeniami. Wyeliminowanie stanu prowadzi do zmniejszenia zajętości pamięci maszyny cyfrowej, ponieważ nie generuje się wiązki trajektorii wychodzących z tego stanu.

Jeżeli czas potrzebny na wykonanie procedur eliminacji jest mniejszy od czasu potrzebnego na generację stanów, które mogą być wyeliminowane, to procedura eliminacji przyspiesza realizację algorytmu.

Dla eliminacji stanów nieperspektywicznych określimy macierze:

$$B_j^{\lambda,\eta} = \left[b_{j,i,m}^{\lambda,\eta} \right]_{\substack{j=1,2 \\ i \in \delta_j^{\lambda,\eta}}} \tag{25}$$

gdzie:

$\delta_j^{\lambda,\eta}$ - zbiór numerów zadań nie wykonanych w j-tym podsystemie w stanie $P^{\lambda,\eta}$.

Elementy tych macierzy określamy następująco:

1°

$$b_{1,i,\mu}^{\lambda,\eta} = \begin{cases} 1 : (u_{1,\mu,i} = 1) \wedge (\varphi_i \leq T_{1,\mu}^{\lambda,\eta}) \wedge (t_{1,i}^{*} \leq \Phi_i) \\ 0 : \text{w przeciwnym przypadku} \end{cases} \tag{26}$$

termin $t_{1,i}^{*}$ wyznacza wzór:

$$t_{1,i}^{*} = \min_{m \in \beta_i^{\lambda,\eta}} (T_{2,m}^{\lambda,\eta} + \vartheta_{2,m,i}) \tag{26a}$$

oraz:

$$\bigvee_m (u_{2,m,i} = 1) \wedge (T_{1,\mu}^{\lambda,\eta} + \vartheta_{1,\mu,i} \leq T_{2,m}^{\lambda,\eta}) \Rightarrow (m \in \beta_i^{\lambda,\eta}) \tag{26b}$$

2^{o}

$$b_{2,m,i}^{\vartheta,\eta} = \begin{cases} 1 : (u_{2,m,i} = 1) \wedge (p_{2,i}^{\vartheta,\eta} \leq T_{2,m}^{\vartheta,\eta}) \wedge (t_{2,i}^{*} \leq \psi_i) \\ 0 : \text{w przeciwnym przypadku} \end{cases} \tag{27}$$

gdzie:

$$t_{2,i}^{*} = t_i^{*} \tag{27a}$$

Wiersze macierzy B_j odpowiadają niezrealizowanym zadaniom, natomiast kolumny - agregatom podsystemów. Jeżeli w macierzy B_j istnieje wiersz złożony z samych zer, wiadomo, że nie da się zrealizować zadanie odpowiadającego numerowi tego wiersza. Wystarczy w tej sytuacji poprzestać na sprawdzeniu macierzy B_1. Jeżeli w wierszach macierzy B_1 istnieją elementy niezerowe, wówczas sprawdzamy macierz B_2. Stan "nie wypada", jeśli we wszystkich wierszach macierzy B_1 i B_2 istnieje co najmniej jedna jedynka.

WNIOSKI KOŃCOWE

Przedstawiony problem harmonogramowania procesu regeneracji walców został rozwiązany metodą programowania wieloetapowego, które jest skrajnym przypadkiem metody podziału i ograniczeń bez procedury powrotu. Problem ten rozpatrywano przy przyjęciu pewnych założeń (np. dot. przestojów agregatów, buforów przed podsystemami itp.).

Perspektywy dalszych prac są związane z eliminowaniem tych ograniczeń, co niewątpliwie spowoduje zmiany w przedstawionym algorytmie. Podobnie ma się problem przyjętego kryterium, które maksymalizuje minimalne wyprzedzenie realizacji zadań. Biorąc pod uwagę, że terminy ψ_n z harmonogramu walcowni są ustalone w oparciu o statystyczną ocenę żywotności walców, można przyjąć kryterium maksymalizacji ważonego wyprzedzenia zadań po regeneracji.

Dopuszczenie przestojów agregatów spowoduje również zmianę algorytmu.

Reasumując, przedstawiony problem harmonogramowania procesu regeneracji walców wymaga jeszcze wielu dalszych prac, którym winny towarzyszyć testy komputerowe.

LITERATURA

[1] Praca zbiorowa: System automatycznej kontroli i sterowania jakością wyrobów dla celów automatycznego sterowania, cz. III i IV Instytut Automatyki, Gliwice 1975, 1976.

[2] MARECKI F., KRÓL E.: Harmonogramowanie procesu walcowania na WCK, ZN Pol. Śl. s. Automatyka, z. 44, Gliwice 1978.

[3] KRÓL E.: Sterowanie operatywne procesem walcowania ciągłego kęsów, Prace VIII KKA, Szczecin 1980.

[4] WITASZEK E.: Sterowanie operatywne procesem walcowania kęsów. Praca dyplomowa niepublikowana, IA Pol. Śl., Gliwice 1981.

Recenzent: Prof. dr hab. inż. Zdzisław TRYBALSKI

Wpłynęło do Redakcji 15.05.1982 r.

МАТЕМАТИЧЕСКАЯ МОДЕЛЬ И АЛГОРИТМ СОСТАВЛЕНИЯ ГРАФИКОВ ПРОЦЕССА РЕГЕНЕРАЦИИ ВАЛКОВ ПРОКАТНОГО СТАНА

Р е з ю м е

В работе дана математическая модель процесса регенерации валков прокатного стана для нужд прокатного цеха. С целью оптимализации процесса принято критерий максимализации минимального времени опережения выполнения работ. Для решения задачи использовано многошаговое программирование.

MATHEMATICAL MODEL AND ALGORITHM OF SCHEDULING OF THE ROLLERS REGENERATION PROCESS

S u m m a r y

We present a mathematical model of the rollers regeneration process for the continuous rolling department. The criterion of optimal scheduling is the maximization of the minimum earliness of task performance. To solve the problem, we used the multistage programming method.