ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1984
Seria: AUTOMATYKA z. 75 Nr kol. 811

Franciszek Marecki
Politechnika Śląska

MINIMALIZACJA STRAT PRODUKCYJNYCH PROCESU WYKRAWANIA BLACH KAROSERYJNYCH

Streszczenie. W pracy sformułowano i rozwiązano problem minimalizacji strat produkcyjnych procesu wykrawania blach karoseryjnych. Wyróżniono straty wynikające z przestojów maszyn oraz opóźnienia realizacji zadań. Problem rozwiązano algorytmem programowania wieloetapowego.

1. Wprowadzenie

W pracy rozważony zostanie problem harmonogramowania realizacji zadań na równoległych maszynach. Jako kryterium optymalizacji przyjęto minimalizację strat produkcyjnych. Wyróżniono przy tym straty wynikające z przestojó maszyn oraz opóźnienia realizacji zadań.

Jako system o strukturze równoległej rozpatrywana jest krajalnia blach karoseryjnych [8]. W systemie tym maszynami są, tzw. nożyce gilotynowe, a obiektami przepływającymi przez maszyny, tzw. pakiety blach. Zadanie polega na wykrojeniu, z każdej blachy należącej do pakietu, określonych formatów na, tzw. wytłoczki [9]. Koncepcja harmonogramowania procesu wykrawania blach karoseryjnych została przedstawiona w [10]. Przez harmonogramowanie rozumiane jest wyznaczenie maszyny i przedziału czasu realizacji dla każdego obiektu [12].

Proces wykrawania blach karoseryjnych był analizowany w kilku pracach [22], [16], [21]. Rozważono przy tym modele z różnymi ograniczeniami. Z teoretycznego punktu widzenia istotne znaczenie ma problem harmonogramowania zadań na równoległych maszynach [15], [14], [3]. Do rozwiązania tego problemu wykorzystywano algorytmy oparte na idei wieloetapowych procesów decyzyjnych.

W niniejszej pracy zostanie rozważony problem minimalizacji strat produkcyjnych realizacji niezależnych zadań procesu wykrawania blach karoseryjnych. Do rozwiązania problemu wykorzystamy metodę programowania wieloetapowego [11]. Idea tej metody wywodzi się z wieloetapowych procesów decyzyjnych [1] i programowania dynamicznego [4], [5]. Ponadto wykorzystywane są pewne elementy metody podziału i ograniczeń [7], [2], [13] w szczególności wyróżnienie parametrów metody [6].

Ogólny problem harmonogramowania zadań na równoległych maszynach jest NP - zupełny w sensie złożoności obliczeniowej [17], [18]. Z tego względu do jego rozwiązania stosowane są algorytmy przeglądowe lub heurystyczne. Efektywność tych algorytmów zależy od komputerowych struktur danych [19],

[20] oraz języka programowania [3].

2. Sformułowanie problemu

Załóżmy, że dany jest zbiór maszyn /nożyc gilotynowych/:

$$\mathcal{A} = \{A_m\} \qquad m = 1,\ldots,M \qquad /1/$$

gdzie: A_m - m-ta maszyna, M - liczba maszyn.

Każda maszyna może pracować wyłącznie w przedziale czasu, ograniczonym chwilami danymi wektorami

$$R = [r_m] \qquad m = 1,\ldots,M \qquad /2/$$

gdzie: r_m - najwcześniejsza chwila rozpoczęcia pracy agregatu A_m;

oraz

$$Z = [z_m] \qquad /3/$$

gdzie: z_m - najpóźniejsza chwila zakończenia pracy agregatu A_m.

Załóżmy, że dany jest zbiór obiektów /pakietów blach/:

$$\Omega = \{\omega_n\} \qquad n = 1,\ldots,N \qquad /4/$$

gdzie: ω_n - n-ty obiekt, N - liczba obiektów

Dane są chwile dostępności obiektów

$$\Phi = [\varphi_n] \qquad n = 1,\ldots,N \qquad /5/$$

gdzie: φ_n - chwila najwcześniejszego rozpoczęcia obsługi obiektu ω_n /realizacji n-tego zadania/.

Analogicznie przyjmiemy chwile najpóźniejszego zakończenia zadań

$$\Psi = [\psi_n] \qquad n = 1,\ldots,N \qquad /6/$$

gdzie: ψ_n - chwila najpóźniejszego zakończenia obsługi obiektu ω_n. Jeżeli obiekt ω_n jest obsłużony po terminie ψ_n, to w procesie wystę ją straty.

Załóżmy, że alternatywne przyporządkowanie obiektów do agregatów da jest macierzą

$$U = [u_{m,n}] \qquad m = 1,\ldots,M \quad n = 1,\ldots,N \qquad /7/$$

Elementy tej macierzy mają następujące znaczenie

$$u_{m,n} = \begin{cases} 1 : \text{jeśli obiekt } \omega_n \text{ może być obsłużony w agregacie } A_m \\ 0 : \text{w przypadku przeciwnym} \end{cases} \qquad /7a/$$

Przyporządkowanie /7/ wynika z wymiarów blach i parametrów nożyc gilotynowych.

Czasy realizacji zadań dane są macierzą

$$\Theta = [\vartheta_{m,n}] \qquad /8/$$
$$m = 1,\ldots,M$$
$$n = 1,\ldots,N$$

gdzie: $\vartheta_{m,n}$ - ozas obsługi obiektu ω_n w agregacie A_m, /gdy $u_{m,n} = 1$/.

Zakładamy, że przy zmianie obiektu, agregat jest przezbrajany. Czasy przezbrojeń agregatów dane są macierzami:

$$T^m = [\tau^m_{\nu,n}] \qquad /9/$$
$$\nu = 0,\ldots,N$$
$$n = 1,\ldots,N$$
$$m = 1,\ldots,M$$

gdzie: $\tau^m_{\nu,n}$ - ozas przezbrojenia agregatu A_m, po obiekcie ω_ν i przed obiektem ω_n.

Przez ω_o^m oznaczamy obiekt, który był obsłużony w agregacie A_m jako ostatni w poprzednim okresie harmonogramowania.

Przy powyższych założeniach należy wyznaczyć harmonogram realizacji zadań o postaci:

$$H = \langle\langle m_1, \varrho_{m_1,1}, t_{m_1,1}\rangle, \ldots, \langle m_n, \varrho_{m_n,n}, t_{m_n,n}\rangle, \ldots, \langle m_N, \varrho_{m_N,N}, t_{m_N,N}\rangle\rangle \qquad /10/$$

gdzie: $\langle\cdot\rangle$ - zbiór uporządkowany liniowo;

m_n - numer maszyny, w której obsłużono obiekt ω_n;

$\varrho_{m_n,n}$ - chwila rozpoczęcia obsługi obiektu ω_n w maszynie A_m;

$t_{m_n,n}$ - chwila zakończenia obsługi obiektu ω_n w agregacie A_m.

Harmonogram dopuszczalny musi spełniać następujące ograniczenia procesu:

- sekwencyjnej realizacji zadań w agregatach

$$\forall_\mu \forall_\nu \forall_n \left(m_\nu = \mu\right) \wedge \left(m_n = \mu\right) \Rightarrow \left[\left(t_{\mu,\nu} \leq \varrho_{\mu,n}\right) \vee \left(t_{\mu,n} \leq \varrho_{\mu,\nu}\right)\right] \qquad /11a/$$

- niepodzielności zadań

$$\forall_m \forall_n \left(u_{m,n} = 1\right) \Rightarrow \left(t_{m,n} = \varrho_{m,n} + \vartheta_{m,n}\right) \qquad /11b/$$

- czas realizacji zadań

$$\forall_m \forall_n \left(u_{m,n} = 1\right) \Rightarrow \left[\rho_{m,n} \geqslant \max\left(\varphi_n, r_m\right)\right] \wedge \left(t_{m,n} \leqslant z_m\right) \qquad /11c/$$

Spośród harmonogramów dopuszczalnych należy wyznaczyć harmonogram optymalny. Jako kryterium optymalizacji przyjmiemy minimalizację strat produkcyjnych. Wyróżnimy straty wynikające z przestoju agregatów oraz straty z opóźnienia realizacji zadań.

Załóżmy, że jednostkowe straty /na jednostkę czasu/ przestoju agregatów dane są wektorem:

$$S^1 = \left[s_m^1\right] \qquad m = 1,\ldots,M \qquad /12/$$

gdzie: s_m^1 - strata przestoju agregatu A_m w jednostce czasu.

Analogicznie zapiszemy jednostkowe straty opóźnienia zadań

$$S^2 = \left[s_n^2\right] \qquad n = 1,\ldots,N \qquad /13/$$

gdzie: s_n^2 - strata opóźnienia n-tego zadania o jednostkę czasu.

Oznaczmy przez μ_i, $i = 1,\ldots,M_i$ numer obiektu, który był obsłużony w agregacie A_μ jako i-ty z kolei. Zatem straty przestoju agregatu A_μ zapiszemy w postaci

$$q_\mu = s_\mu^1 \sum_{i=1}^{i=N_\mu} \left(\rho_{\mu_i} - t_{\mu_{i-1}}\right) \qquad /14/$$

gdzie: $t_{\mu_o} = r_\mu$.

Natomiast dla wszystkich agregatów otrzymamy

$$Q^1 = \sum_{\mu=1}^{\mu=M} s_\mu^1 \sum_{i=1}^{i=N_\mu} \left(\rho_{\mu_i} - t_{\mu_{i-1}}\right) \qquad /15/$$

Oznaczmy z kolei przez d_n opóźnienie realizacji n-tego zadania, czyli

$$d_n = \begin{cases} t_{m,n} - \varphi_n & \text{jeśli } t_{m,n} > \varphi_n \\ 0 : \text{w przypadku przeciwnym} \end{cases} \qquad /16/$$

A zatem straty wynikające z opóźnienia zadań zapiszemy w postaci

$$Q^2 = \sum_{n=1}^{n=N} d_n \cdot s_n^2 \qquad /17/$$

Ostatecznie kryterium optymalizacji harmonogramu /10/ zapiszemy jako

$$Q = Q^1 + Q^2 \rightarrow \min \qquad /18/$$

3. Algorytm programowania wieloetapowego

Podstawowymi elementami konstrukcyjnymi algorytmu programowania wieloetapowego są: stan, wartość stanu, generowanie stanów oraz eliminowanie stanów nieperspektywicznych. Proces przydzielania N obiektów do obsługi w równoległych maszynach A_m, jest traktowany jako N-etapowy proces decyzyjny.

Stan interpretuje sytuację, jaka powstaje po podjęciu decyzji o przydzieleniu obiektu do obsługi. Stan początkowy jest dany /sytuacja przed przydzielaniem obiektów/. Stany końcowe /N-tego etapu/ dają dopuszczalne harmonogramy realizacji zadań. Ciąg stanów nazwiemy trajektorią. Wiązka trajektorii wychodząca ze stanu początkowego przedstawia kompletne drzewo decyzyjne. Stan jest zapisywany tak, by dana była historia realizacji zadań. Pozwala to zapamiętywać tylko ostatni stan każdej trajektorii /nie kompletnej/. Ponadto zapis taki umożliwia rozwiązywanie problemów dla procesów, które nie spełniają własności Markowa.

Każdy stan opisuje harmonogram przydzielonych zadań. Do oceny tego harmonogramu służy wartość stanu. Wartość stanu koresponduje z przyjętym kryterium optymalizacji.

Aby wyznaczyć harmonogram dopuszczalny, należy wygenerować stan końcowy. Stan końcowy o najmniejszej wartości daje harmonogram optymalny.

Generowanie stanów jest oparte na koncepcji stanów aktywnych. Stany aktywne pozwalają wygenerować dalsze stany. Stan aktywny jest ostatnim stanem trajektorii /nie końcowym/. Stany końcowe nie są aktywne.

W N-etapowym procesie decyzyjnym wyróżniamy etapy η, $\eta = 0,\ldots,N$. Stan η-tego etapu zawiera informację o realizacji η zadań. Stany aktywne są pogrupowane w N zbiorach, dla $\eta = 0,\ldots,N-1$. Generowanie stanów polega na wyborze stanu ze zbioru η-1-go etapu i wyznaczeniu nowych stanów /bezpośrednich następników/, które wchodzą do zbioru η-tego etapu. Na podstawie wybranych stanów N-1-go etapu generowane są stany końcowe. A zatem obliczenia kończą się, jeżeli wszystkie zbiory stanów aktywnych są puste.

W generowaniu stanów istotne znaczenie dla efektywności algorytmu mają: reguły wyboru i podziału oraz procedury generowania. Jako reguły wyboru stanu aktywnego stosuje się: FIFO /pierwszy wygenerowany - pierwszy wybrany/, LIFO /ostatni wygenerowany - pierwszy wybrany/, LLB /wybór stanu o najmniejszym dolnym ograniczeniu wartości/, lub inne. Z wybranego stanu aktywnego można generować następniki lub tylko bezpośrednie następniki. Do tego celu służą odpowiednio wielokrokowe lub jednokrokowe procedury podziału. Ponadto z wybranego stanu aktywnego można wygenerować wszystkie /procedurą zupełną/ lub tylko niektóre /procedurą częściową/ następniki. Następniki te są generowane w zależności od określonego porządku /leksykograficznego, chronologicznego, itp./. Procedura generowania stanów

jest zdaniem logicznym pozwalającym przejść od stanu aktywnego do jego następnika. Ponadto określa ona formuły dla wyznaczenia elementów nowego stanu.

W trakcie obliczeń eliminowane są stany, które nie prowadzą do rozwiązania optymalnego. Do eliminacji stanów służą reguły wyczerpywania, dominacji i sondowania. Stan zostaje wyeliminowany regułą wyczerpywania, jeżeli nie pozwala wygenerować dopuszczalnego stanu końcowego. Dominacja stanów jest oparta na stanach lokalnie optymalnych. Stan lokalnie optymalny jest najlepszym stanem końcowym /o najmniejszej wartości/, który można otrzymać z danego stanu. Z dwóch stanów aktywnych lepszy jest ten, którego stan lokalnie optymalny jest lepszy. Sondowanie polega na porównaniu stanu aktywnego z danym stanem końcowym aktualnie najlepszym. Jeżeli dolne szacowanie wartości stanu lokalnie optymalnego jest większe niż wartość aktualnie najlepszego stanu końcowego - to dany stan aktywny jest nieperspektywiczny.

Efektywność algorytmu jest zależna od przyjętej definicji stanu, reguły wyboru podziału, oraz eliminacji stanów. Jeżeli stosowana jest częściowa procedura podziału, to można dowolnie zmniejszyć wymagany w algorytmie obszar pamięci komputerowej. W procedurze podziału częściowego ze stanu aktywnego generowane są tylko niektóre stany. A zatem wybrany stan aktywny pozostaje aktywnym po generowaniu. Aby przy ponownym wyborze tego stanu generowane były jego kolejne bezpośrednie następniki - wprowadza się pojęcie wskaźnika stanu. Wskaźnik stanu określa parametry stanu aktywnego dla generowania jego kolejnych bezpośrednich następników.

W algorytmie programowania wieloetapowego istotne znaczenie ma czas obliczeń. Czas ten jest zależny od liczby wygenerowanych i liczby wyeliminowanych stanów. W trakcie generowania można pominąć strategie /ciągi decyzji/, które prowadzą do stanów identycznych. W eliminacji stanów istotne znaczenie ma czas sprawdzenia i prawdopodobieństwo spełnienia reguły. Wielkości te w dużym stopniu zależą od konstrukcji programu komputerowego i danych liczbowych. Dla reguły dominacji można uporządkować stany aktywne, co skraca czas poszukiwania stanu zdominowanego.

4. Rozwiązanie problemu

Do rozwiązania sformułowanego problemu wykorzystany zostanie algorytm programowania wieloetapowego. Zdefiniujemy stan procesu decyzyjnego, algorytmiczność stanu oraz podamy procedury generowania i eliminacji stanów.

4.1. Podstawowe definicje i określenia

Załóżmy, że η , $(\eta = 0,\ldots,N)$ jest numerem etapu decyzyjnego, a $l, (l = 1,\ldots,L\eta$, gdzie: $L\eta$ - liczba stanów η -tego etapu) numerem stanu aktywnego w ramach η -tego etapu. Zbiór stanów aktywnych η -tego etapu

oznaczymy przez $\mathcal{L}_\eta$.

Def. 1.: Stan jest macierzą

$$P^{1,\eta} = \left[p^{1,\eta}_{n,j} \right] \qquad /19/$$

$$n = 1,\ldots,N$$
$$j = 1,2$$

Elementy tej macierzy określamy następująco

$$p^{1,\eta}_{n,1} = \begin{cases} m : \text{jeśli obiekt } \omega_n \text{ przydzielono do maszyny } A_m \\ 0 : \text{w przypadku przeciwnym} \end{cases} \qquad /20a/$$

$$p^{1,\eta}_{n,2} = \begin{cases} t_{m,n} : \text{jeśli } p^{1,\eta}_{n,i} = m \\ 0 : \quad \text{w przypadku przeciwnym} \end{cases} \qquad /20b/$$

Tak więc, stan początkowy $P^{1,0}$ jest macierzą zerową, natomiast każdy stan końcowy $P^{1,N}$ ma wszystkie elementy dodatnie. Ze stanu końcowego $P^{1,N}$ można wprost określić dopuszczalny harmonogram /10/, bowiem:

$$m_n = p^{1,N}_{n,1} \qquad /21a/$$

$$t_{m,n} = p^{1,N}_{n,2} \qquad /21b/$$

oraz

$$\rho_{m,n} = t_{m,n} - \vartheta_{m,n} \qquad /21c/$$

Z każdym stanem $P^{1,\eta}$ jest związana jego wartość, którą oznaczymy przez $V^{1,\eta}$.

Def. 2.: Wartość stanu jest wektorem

$$V^{1,\eta} = \left[v_i^{1,\eta} \right] \qquad /22/$$

$$i = 1,2$$

Elementy tego wektora wyznaczamy z rekurencyjnych formuł

$$v_i^{1,\eta} = v_i^{\lambda,\eta-1} + \Delta v_i \qquad /23a/$$

gdzie: $P^{\lambda,\eta-1}$ wybrany stan aktywny; $P^{1,\eta}$ - bezpośredni następnik stanu $P^{\lambda,\eta-1}$, Δv_i - przyrost i-tej współrzędnej wartości stanu, przy przejściu od stanu $P^{\lambda,\eta-1}$ do stanu $P^{1,\eta}$.

Przyrost Δv_i wyznaczamy po określeniu pewnych pomocniczych wielkości.

Załóżmy, że stan $P^{1,\eta}$ został wygenerowany przez przydzielenie obiektu ω_n do maszyny A_m. Chwilę $T_m^{\lambda,\eta-1}$ zwolnienia maszyny A_m w stanie $P^{\lambda,\eta-1}$ określamy jako:

$$T_m^{\lambda,\eta-1} = \begin{cases} \max\limits_{i \in \beta_m^{\lambda,\eta-1}} p_{i,2}^{\lambda,\eta-1} & : \text{jeśli } \left|\beta_m^{\lambda,\eta-1}\right| > 0 \\ r_m & : \text{w przypadku przeciwnym} \end{cases} \qquad /24/$$

przy tym

$$\forall_i \left(p_{i,1}^{\lambda,\eta-1} = m\right) \Rightarrow \left(i \in \beta_m^{\lambda,\eta-1}\right) \qquad /25/$$

A zatem dla stanu $P^{\lambda,\eta}$, zamiast /24/ można podać formułę rekurencyjną

$$T_m^{\lambda,\eta} = \max\left(\varphi_n,\ T_m^{\lambda,\eta-1} + \tau_{k_m,n}^{m}\right) + \vartheta_{m,n} \qquad /26/$$

przy tym

$$k_m = \begin{cases} i_m & : \text{jeśli } \left(p_{i_m,1}^{\lambda,\eta-1} = m\right) \wedge \left(T_m^{\lambda,\eta-1} = p_{i_m,2}^{\lambda,\eta-1}\right) \\ 0 & : \text{w przypadku przeciwnym} \end{cases} \qquad /27/$$

W ten sposób można wyznaczać chwile zwolnienia maszyn dla każdego stanu. Przy przejściu od stanu $P^{\lambda,\eta-1}$ do stanu $P^{\lambda,\eta}$, /przydzielenie ω_n do A_m/ modyfikacji ulega jedynie chwila zwolnienia maszyny A_m. Chwila zwolnienia pozostałych maszyn A_μ, $\mu = 1,\ldots,m-1,m+1,\ldots,M$, nie ulegają zmianie.

Jeśli znane są chwile zwolnienia maszyn $T_m^{\lambda,\eta-1}$ i $T_m^{\lambda,\eta}$, to można wyznaczyć współrzędne Δv_i przyrostu wartości stanu. Tak więc:

$$\Delta v_1 = \left[T_m^{\lambda,\eta} - \left(T_m^{\lambda,\eta-1} + \vartheta_{m,n}\right)\right] \cdot s_m^1 \qquad /28/$$

oraz

$$\Delta v_2 = \begin{cases} \left(T_m^{\lambda,\eta} - \varphi_n\right) \cdot s_n^2 & : \text{jeśli } T_m^{\lambda,\eta} > \varphi_n \\ 0 & : \text{w przypadku przeciwnym} \end{cases} \qquad /29/$$

ponieważ

$$T_m^{\lambda,\eta} = t_{m,n}$$

dla obiektu ω_n przydzielonego w stanie $P^{\lambda,\eta-1}$ do maszyny A_m.
A zatem współrzędne v_1 korespondują z kryterium Q^1, natomiast v_2 - z Q^2. Posługiwanie się formułami rekurencyjnymi skraca czas obliczeń.

Optymalny stan końcowy P^o wyznaczamy z warunku

$$\left[\min_{1 \le l \le L_N} \left(v_1^{l,N} + v_2^{l,N}\right) = v_1^{l^o,N} + v_2^{l^o,N}\right] \Rightarrow \left(P^{l^o,N} = P^o\right) \qquad /30/$$

Ze stanu P^o otrzymujemy wprost optymalny harmonogram realizacji zadań. W obliczeniach zbiór stanów końcowych jest jednoelementowy, gdyż zapamiętywany jest tylko aktualnie najlepszy stan końcowy P^n. Wygenerowany stan $P^{\lambda,N}$ może stać się aktualnie najlepszym lub zostać wyeliminowanym - jako gorszy niż P^n.

4.2. Generowanie stanów

Generując stany przechodzimy od stanu początkowego $P^{1,0}$ do stanów końcowych $P^{1,N}$. W tym celu stosujemy reguły wyboru stanu aktywnego, reguły podziału oraz procedurę generowania.

Do rozwiązania sformułowanego problemu wykorzystamy regułę LLB. Dla stanu $P^{\lambda,\eta-1}$ należy wyznaczyć dolne ograniczenie $b^{\lambda,\eta-1}$. Załóżmy, że ze stanu $P^{\lambda,\eta-1}$ wygenerowana została kompletna wiązka trajektorii $P^{\lambda,\eta-1},\ldots,P^{1,N}$. Niech trajektoria $P^{\lambda,\eta-1},\ldots,P^{\lambda^\circ,N}$ będzie lokalnie optymalna, tzn.

$$\min_{l}\left(v_1^{1,N}+v_2^{1,N}\right)=v_1^{\lambda^\circ,N}+v_2^{\lambda^\circ,N} \qquad /31/$$

A zatem stan $P^{\lambda^\circ,N}$ jest lokalnie optymalny.
Dolne ograniczenie $b^{\lambda,\eta-1}$ określimy jako:

$$b^{\lambda,\eta-1}\leq v_1^{\lambda^\circ,N}+v_2^{\lambda^\circ,N} \qquad /32/$$

przyjmując

$$b^{\lambda,\eta-1}=v_1^{\lambda,\eta-1}+v_2^{\lambda,\eta-1} \qquad /33/$$

Widzimy więc, że równość w /32/ ma miejsce tylko w przypadku, gdy na trajektorii lokalnie optymalnej $P^{\lambda,\eta-1},\ldots,P^{\lambda^\circ,N}$ nie występują straty /15/ i /17/. Straty /15/ między innymi są wynikiem przezbrojenia agregatów, a zatem w /32/ raczej występuje nierówność ostra. Chociaż ograniczenie $b^{\lambda,\eta-1}$ nie jest zbyt dokładne, to jednak łatwo można się nim posługiwać przy wyborze stanu aktywnego.

W generowaniu stanów wykorzystamy regułę jednokrokowego podziału częściowego z porządkiem leksykograficznym. Załóżmy, że liczba stanów aktywnych w każdym zbiorze $\mathcal{L}_\eta$, $\eta=1,\ldots,N-1$, jest ograniczona do L_η. Wówczas z wybranego stanu $P^{\lambda,\eta-1}$ można wygenerować najwyżej $L_\eta-|\mathcal{L}_\eta|$ stanów. A zatem stan $P^{\lambda,\eta-1}$ pozostaje dalej aktywny, dlatego powraca do zbioru $\mathcal{L}_{\eta-1}$. Aby przy powtórnym wyborze stanu $P^{\lambda,\eta-1}$ można było generować jego kolejno bezpośrednio następniki wprowadzimy pojęcie indeksu stanu

$$I^{\lambda,\eta-1}=\langle m^{\lambda,\eta-1},n^{\lambda,\eta-1}\rangle \qquad /34/$$

gdzie: $I^{\lambda,\eta-1}$ - indeks stanu $P^{\lambda,\eta-1}$; $m^{\lambda,\eta-1}$ - numer ostatniej maszyny wykorzystanej do generowania stanów z $P^{\lambda,\eta-1}$; $n^{\lambda,\eta-1}$ - numer ostatniego obiektu wykorzystanego do generowania stanów z $P^{\lambda,\eta-1}$. Tak więc wybierając powtórnie $P^{\lambda,\eta-1}$ stany generujemy dla

$$m^{\lambda,\eta-1}\leq m\leq M \qquad /35a/$$

$$n>n^{\lambda,\eta-1}\quad\text{dla}\quad m=m^{\lambda,\eta-1} \qquad /35b/$$

$$1 \leq n \leq N \qquad \text{dla} \quad m^{\lambda, \eta-1} < m \leq M \qquad /35c/$$

Wskaźnik stanu jest licznikiem. Jeżeli n = N, to następuje przeniesienie "jedynki" na starszą pozycję, $(m + 1)$. Ze stanu $P^{\lambda, \eta-1}$ wygenerowano wszystkie stany, jeżeli m = M i n = N, wówczas stan ten jest wyeliminowany z dalszych obliczeń.

Najmniejszy indeks $I^{\lambda, \eta-1} = \langle \mu^{\lambda, \eta-1}, \nu^{\lambda, \eta-1} \rangle$ stanu $P^{\lambda, \eta-1}$ określamy jako:

$$\mu^{\lambda, \eta-1} = \max_{1 \leq i \leq N} p_{i,1}^{\lambda, \eta-1} \qquad /36a/$$

$$\nu^{\lambda, \eta-1} = 0 \qquad /36b/$$

Przyjęcie warunku porządku leksykograficznego /36a/ zmniejsza liczbę generowanych stanów. Wyeliminowane zostają stany identyczne. Można to zilustrować na prostym przykładzie. Załóżmy, że dane są dwie maszyny A_1 i A_2, oraz dwa obiekty ω_1 i ω_2. Decyzję o przydzieleniu obiektu ω_1 do maszyny A_1 oznaczymy przez (ω_1, A_1). A zatem, np. strategie /ciągi decyzji/: $\sigma_1 = \langle (\omega_1, A_1), (\omega_2, A_2) \rangle$ i $\sigma_2 = \langle (\omega_2, A_2), (\omega_1, A_1) \rangle$ są równoważne, tzn. dają identyczny stan końcowy $\{(\omega_1, A_1), (\omega_2, A_2)\}$. Jeżeli natomiast wprowadzimy warunek porządku leksykograficznego, to strategia σ_2 nie zostanie zrealizowana. Wynika to z faktu, że decyzja (ω_2, A_2) nie może następować po decyzji (ω_1, A_1).

W ogólnym przypadku strategia jest ciągiem decyzji, a stan można interpretować jako zbiór /nieuporządkowany/ tych decyzji. A zatem ten sam stan można wygenerować realizując różne strategie. Generowanie stanów identycznych zmniejsza efektywność algorytmu. Dlatego ważne znaczenie ma eliminowanie strategii prowadzących do stanów identycznych. Reguła podziału z warunkiem /36a/ spełnia ten postulat.

Ogólna procedura generowania stanów ma postać

$$\forall_m \forall_n \left(\mu^{\lambda, \eta-1} \leq m \leq M\right) \wedge \left\{\left[\left(m = \mu^{\lambda, \eta-1}\right) \Rightarrow \left(n > \nu^{\lambda, \eta-1}\right)\right] \vee \left[\left(m > \mu^{\lambda, \eta-1}\right) \Rightarrow \left(1 \leq n \leq N\right)\right]\right\} \wedge \left(u_{m,n} = 1\right) \wedge \left(p_{n,1}^{\lambda, \eta-1} = 0\right) \wedge \left(t_{m,n} \leq \tau_m\right) \Rightarrow \left(P = P^{\lambda, \eta-1} + \Delta P\right) \qquad /37/$$

Elementy macierzy ΔP określamy jako:

$$\Delta p_{i,1} = \begin{cases} m : & \text{dla} \quad i = n \\ 0 : & \text{w przypadku przeciwnym} \end{cases} \qquad /38a/$$

$$\Delta p_{i,2} = \begin{cases} t_{m,n} : & \text{dla} \quad \Delta p_{i,1} = m \\ 0 : & \text{w przypadku przeciwnym} \end{cases} \qquad /38b/$$

Chwilę $t_{m,n}$ wyznaczamy z formuły

$$t_{m,n} = \varphi_{m,n} + \vartheta_{m,n} \qquad /39a/$$

przy tym

$$\varphi_{m,n} = \max\left(\varphi_n,\ T_m^{\lambda,\eta-1} + \tau_{k_m,n}^{m}\right) \qquad /39b/$$

gdzie: $T_m^{\lambda,\eta-1}$ wyznaczamy z /24/ lub wartość ta jest w obliczeniach zapamiętywana. Wartość k_m wyznaczamy z /27/. Za pomocą procedury /37/ można wygenerować $L_\eta = |\mathcal{L}_\eta|$ kolejnych stanów η-tego etapu.

4.3. Eliminowanie stanów nieperspektywicznych

Do eliminacji stanów nieperspektywicznych wykorzystywane są reguły : wyczerpywania, dominacji oraz sondowania.

Reguła wyczerpywania ma postać twierdzenia.

Tw. 1.: Stan P jest nieperspektywiczny, jeżeli spełnia warunek

$$\exists_n \forall_m \left(p_{n,1} = 0\right) \wedge \left[\left(u_{m,n} = 1\right) \Rightarrow \left(t_{m,n} > z_m\right)\right] \qquad /40/$$

Dowód tego twierdzenia jest łatwy. W stanie końcowym $P^{1,N}$ muszą być zrealizowane wszystkie zadania, tzn. wszystkie elementy macierzy $p^{1,N}$ muszą być dodatnie. Jeżeli spełniony jest warunek /40/, to obiekt ω_n nie może być obsłużony, bo nie spełniona jest ograniczenie czasu realizacji zadań. A zatem nie można wygenerować stanu końcowego, za pomocą procedury /37/. Generowanie innych stanów /wiązki trajektorii ze stanu $P^{\lambda,\eta-1}$/, jest nieperspektywiczne, ponieważ nie można osiągnąć stanu końcowego.

Reguła dominacji stanów jest regułą względną, pozwalającą porównać dwa stany $P^{l_1,\eta}$, /aktywny/ i $P^{l_2,\eta}$, /wygenerowany/. Reguła ta jest oparta na następującym twierdzeniu.

Tw. 2.: Stan $P^{l_1,\eta}$ dominuje nad stanem $P^{l_2,\eta}$, jeżeli jest spełniony warunek

$$\forall_{1\le m\le M}\left(\beta_m^{l_1,\eta} = \beta_m^{l_2,\eta}\right) \wedge \left(k_m^{l_1,\eta} = k_m^{l_2,\eta}\right) \wedge \left(T_m^{l_1,\eta} \le T_m^{l_2,\eta}\right) \wedge$$

$$\wedge\left(v_2^{l_1,\eta} \le v_2^{l_2,\eta}\right) \qquad /41/$$

gdzie: $\beta_m^{l,\eta}$ - zbiór obiektów przydzielonych do maszyny A_m; $k_m^{l,\eta}$ - numer ostatniego obiektu przydzielonego do maszyny A_m.

Dowód tego twierdzenia polega na porównaniu lokalnie optymalnych trajektorii $P^{l_1,\eta},\ldots,P^{l_1,N}$ i $P^{l_2,\eta},\ldots,P^{l_2,N}$. Dla trajektorii tych możemy napisać

$$v_1^{l_1,\eta} + v_2^{l_1,\eta} + \Delta v_1^{l_1,\eta} + \Delta v_2^{l_1,\eta} = v_1^{l_1^o,N} + v_2^{l_1^o,N} \qquad /42a/$$

$$v_1^{l_2,\eta} + v_2^{l_2,\eta} + \Delta v_1^{l_2,\eta} + \Delta v_2^{l_2,\eta} = v_1^{l_2^o,N} + v_2^{l_2^o,N} \qquad /42b/$$

Jeżeli

$$\bigwedge_{1\le m\le M} \left(\beta_m^{l_1,\eta} = \beta_m^{l_2,\eta}\right) \wedge \left(T_m^{l_1,\eta} \le T_m^{l_2,\eta}\right) \qquad /43/$$

to

$$v_1^{l_1,\eta} \le v_2^{l_2,\eta} \qquad /44/$$

Uwzględniając /41/ i /44/ w /42/ otrzymamy

$$\left(v_1^{l_1^o,N} + v_2^{l_2^o,N}\right) - \left(v_1^{l_2^o,N} + v_2^{l_2^o,N}\right) \le \left(\Delta v_1^{l_1,\eta} + \Delta v_2^{l_1,\eta}\right) + - \left(\Delta v_1^{l_2,\eta} + \Delta v_2^{l_2,\eta}\right) \qquad /45/$$

Jeżeli spełniony jest warunek /41/, to harmonogram lokalnie optymalny wynikający z trajektorii $P^{l_2,\eta},\ldots,P^{l_2,N}$, może być również zrealizowany od stanu $P^{l_1,\eta}$. Wówczas jednak otrzymamy trajektorię $P^{l_1,\eta},\ldots,P^{\lambda,N}$, dla której możemy napisać

$$v_1^{l_1,\eta} + v_2^{l_1,\eta} + \Delta v_1 + \Delta v_2 = v_1^{\lambda,N} + v_2^{\lambda,N} \qquad /46/$$

Zauważmy, że $P^{\lambda,N}$ nie jest stanem lokalnie optymalnym, zatem z /42a/ i /46/ otrzymamy

$$\Delta v_1 + \Delta v_2 \ge \Delta v_1^{l_1,\eta} + \Delta v_2^{l_2,\eta} \qquad /47/$$

Ponadto

$$\Delta v_1 = \Delta v_1^{l_2,\eta} \qquad /48a/$$

oraz

$$\Delta v_2 = \Delta v_2^{l_2,\eta} \qquad /48b/$$

A zatem z /47/ i /48/ dochodzimy do

$$\Delta v_1^{l_1,\eta} + \Delta v_2^{l_1,\eta} \le \Delta v_1^{l_2,\eta} + \Delta v_2^{l_2,\eta} \qquad /49/$$

Uwzględniając /49/ w /45/ wnioskujemy, że

$$v_1^{l_1^o,N} + v_2^{l_1^o,N} \le v_1^{l_2^o,N} + v_2^{l_2^o,N} \qquad /50/$$

A zatem zgodnie z warunkiem /30/ stan $P^{l_1,\eta}$ dominuje nad stanem $P^{l_2,\eta}$.

Jeżeli stany $P^{l_1,\eta}$ i $P^{l_2,\eta}$ są równoważne, /równość w /59//, to wybieramy stan aktywny $P^{l_1,\eta}$, ponieważ został wygenerowany wcześniej.

Reguła sondowania jest oparta na twierdzeniu.

Tw. 3.: Stan P jest nieperspektywiczny, jeżeli spełnia warunek

$$v_1 + v_2 > v_1^a + v_2^a \quad . \qquad /51/$$

Dowód wynika z przyjętego dolnego ograniczenia /33/. Dla każdego stanu końcowego $P^{1,N}$ otrzymanego z P możemy napisać

$$v_1^{1,N} + v_2^{1,N} \geq v_1 + v_2 \qquad /52/$$

Z /51/ i /52/ wynika, że

$$v_1^{1,N} + v_2^{1,N} > v_1^a + v_2^a \qquad /53/$$

Ponieważ ze stanu P nie można uzyskać stanu końcowego $P^{1,N}$ lepszego niż aktualnie najlepszy stan końcowy P^a, zatem stan P jest nieperspektywiczny.

Jeżeli ze stanu P wyznaczymy pewien stan końcowy $P^{\lambda,N}$ /np. generując heurystycznie jedną trajektorię/, to, gdyż

$$v_1^{\lambda,N} + v_2^{\lambda,N} = v_1 + v_2 \qquad /54/$$

nie trzeba generować wiązki trajektorii. Stan $P^{\lambda,N}$ staje się aktualnie najlepszy - lub pomijamy go, gdy spełnia nierówność /53/.

4.4. Efektywność algorytmu

Sformułowany problem harmonogramowania zadań na równoległych maszynach należy do klasy NP-zupełnych w sensie złożoności obliczeniowej. Dlatego dla rozwiązania tego problemu podano algorytm przeglądowy.

Efektywność algorytmu może być rozpatrywana w sensie czasu obliczeń lub zajętości pamięci komputerowej. Parametry te są wzajemnie zależne i na ogół polepszenie jednego powoduje pogorszenie drugiego. Jednakże w przedstawionym algorytmie programowania wieloetapowego zajętość pamięci można ograniczyć.

Zajętość pamięci komputerowej jest zależna w głównej mierze od liczby stanów, które muszą być zapamiętywane w trakcie obliczeń. Wielkość ta może być sterowana poprzez reguły wyboru i podziału. Dla reguły FIFO zapamiętywane są stany aktywne pojedynczego etapu /lub w trakcie generowania niektóre stany dwóch sąsiednich etapów/. Stosując reguły LIFO lub LLB zapamiętywane są niektóre stany aktywne różnych etapów.

Zajętość pamięci komputerowej można ograniczyć, wprowadzając dopuszczalne liczby L_η stanów w zbiorze $\mathcal{L}_\eta$, $\eta = 0,\ldots,N-1$. W skrajnym przypadku przyjmując $L_\eta = 1$, w trakcie obliczeń zapamiętywana jest tylko jedna trajektoria stanów. To podejście wymaga jedynie wprowadzenia reguły podziału częściowego oraz indeksu stanu.

Czas obliczeń komputerowych /dla wyznaczenia rozwiązania optymalnego/ jest zależny od sposobu przeglądu kompletnej wiązki trajektorii. A zatem

na czas obliczeń mają wpływ reguły: wyboru, podziału i eliminacji stanów nieperspektywicznych. Ilościowy wpływ tych reguł wymaga na ogół przeprowadzenia testów komputerowych dla konkretnych danych liczbowych.

Dla reguł eliminacji stanów nieperspektywicznych można podać pewne uwagi o ich efektywności. O efektywnościoi reguły decydują: czas testowania oraz prawdopodobieństwo wyeliminowania stanu. Reguła efektywna ma krótki czas testowania i duże prawdopodobieństwo wyeliminowania stanu. Jednakże na ogół parametry te są przeciwstawne, tzn. reguła o dłuższym czasie obliczeń daje większe prawdopodobieństwo wyeliminowania stanu.

W niektórych przypadkach prawdopodobieństwo wyeliminowania stanu jest dla danej reguły, zależne od etapu decyzyjnego. Przykładem może być reguła wyczerpywania stanu /40/. Również prawdopodobieństwo wyeliminowania stanu za pomocą reguły sondowania rośnie wraz ze wzrostem numeru etapu decyzyjnego. Wynika to z faktu, że oszacowanie dolnego ograniczenia jest bardziej dokładne dla późniejszych etapów obliczeniowych. Prawdopodobieństwo wyeliminowania stanu za pomocą reguły dominacji jest w niektórych przypadkach zerowe. Stąd nie stosujemy reguły dominacji stanów dla $\eta = 1$ oraz w stosunku do stanów,które pochodzą z tego samego bezpośredniego poprzednika /np. dla reguły LIFO nie ma dominacji stanów/.

Czas obliczeń komputerowych jest zależny od zbioru stosowanych reguł eliminacji. Obszerne analizy w tym zakresie zostały przeprowadzone w pracach [22], [21], [14], i [3]. Eliminacja stanu skraca czas obliczeń /bo nie jest generowana wiązka trajektorii/, a negatywny wynik testu powoduje wydłużenie obliczeń. W większości przypadków najlepsze rezultaty daje stosowanie pełnego zbioru reguł eliminacji stanów.

Efektywność algorytmu programowania wieloetapowego zależy od języka, w którym napisany jest program komputerowy. Dla przykładu w pracach [21] i [3] przeprowadzono obliczenia /na komputerze ODRA 1305/ dla tych samych zestawów danych,stosując odpowiednio języki FORTRAN oraz PASCAL. Programy w PASCAL-u były od 5 do 20 razy szybsze. Wynika to przede wszystkim z możliwości lepszej organizacji struktur danych.

Organizacja struktur danych ma istotne znaczenie przede wszystkim dla dominacji stanów. Reguła dominacji /41/ wymaga porównania wygenerowanego stanu P z danymi stanami aktywnymi $P^{1,\eta}$. Istotne znaczenie ma czas przeszukiwania zbioru stanów aktywnych. Czas ten można skrócić przez odpowiednie uporządkowanie stanów aktywnych. W tym celu wyróżniamy zbiory stanów aktywnych $\mathcal{L}_\eta$ dla etapów decyzyjnych. Z kolei w zbiorach $\mathcal{L}_\eta$ wyróżniamy podzbiory $\mathcal{L}_\eta^i$ zawierające stany o tych samych zbiorach $\beta_m^{1,\eta}$. Z kolei wyróżniamy podzbiory $\mathcal{L}_\eta^{i,j}$ do których należą stany spełniające dodatkowo warunek $k_m^{1,\eta} = k_m^{2,\eta}$. Na ostatnim poziomie klasyfikacji stany można uporządkować w zależności od chwili $T_m^{1,\eta}$. Czas potrzebny na odszukanie stanu aktywnego, który podlega dominacji /lub stwierdzenie, że takiego stanu nie ma/ jest krótszy w uporządkowanym zbiorze stanów aktywnych, niż w zbiorze nie uporządkowanym.

5. Zakończenie

W pracy sformułowano i rozwiązano problem minimalizacji niezależnych zadań procesu wykrawania blach karoseryjnych. Rozważany uproszczony model przedstawiał przepływ obiektów /pakietów blach/ przez równoległe maszyny /nożyce gilotynowe/. Rozwinięcie tego modelu do przypadków charakterystycznych dla procesu wykrawania blach karoseryjnych nie wymaga zmiany metody rozwiązania. Przypadki te obejmują: zadania zależne, dodatkowe zasoby /środki transportowe i operatorzy/ oraz ograniczenia procesu. Zależność zadań wynika z karty rozkroju technologicznego, gdy z jednego arkusza blachy wytwarza się różne przygotówki. Środki transportowe wnoszą istotne ograniczenia procesu przepływu blach od magazynu blach do magazynu przygotówek. Dla obsługi pakietu blach potrzebna jest określona liczba operatorów. Zbiór operatorów jest stały dla danego okresu harmonogramowania. A zatem operatorzy stanowią dodatkowy zasób odnawialny, który w zależności od potrzeb jest wykorzystywany do realizacji zadań. Spośród dodatkowych ograniczeń procesu istotne znaczenie ma dopuszczalny czas pomiędzy wykonaniem różnych przygotówek z tego samego pakietu blachy /z uwagi na utratę własności tłocznych blachy/.

Sformułowany w pracy problem rozwiązano algorytmem programowania wieloetapowego. Podstawowe elementy tego algorytmu zilustrowano na przykładzie uproszczonego problemu wykrawania blach karoseryjnych. Za pomocą tego algorytmu można z powodzeniem /i lepszym uzasadnieniem/ rozwiązać wymienione wyżej przypadki charakterystyczne.

Podstawowe znaczenie w algorytmie programowania wieloetapowego ma stan procesu decyzyjnego. Dla problemów harmonogramowania stan procesu decyzyjnego ma naturalną interpretację w postaci stanu procesu technologicznego.

Efektywność algorytmu programowania wieloetapowego zależy od liczby wygenerowanych oraz liczby wyeliminowanych stanów. W systemie o strukturze równoległej różne strategie /ciągi decyzji/ mogą dać identyczny stan /nieuporządkowany zbiór decyzji/. Stany identyczne nie są generowane po wprowadzeniu do procedur odpowiedniego porządku /np. leksykograficznego/. Konstrukcja reguł eliminacji stanów nieperspektywicznych wymaga sformułowania odpowiednich twierdzeń.

W przedstawionym algorytmie programowania wieloetapowego ograniczenia pamięci komputerowej nie ma znaczenia. Stosując regułę podziału częściowego i indeks stanu aktywnego, wystarczy zapamiętywać tylko jedną trajektorię. W przypadku ograniczonego czasu obliczeń, zamiast stanu optymalnego P^o można uzyskać jedynie stan aktualnie najlepszy P^a. Błąd względny rozwiązania wynikającego ze stanu P^a obliczamy jako:

$$\varepsilon = \frac{v_1^a + v_2^a}{v_1^o + v_2^o} - 1 \qquad /55/$$

a ponieważ $v_1 + v_2 \leq v_1^o + v_2^o$ /56/

gdzie: $v_1 + v_2$ - najmniejsze dolne ograniczenie stanów aktywnych

zatem

$$\varepsilon \leq \frac{v_1^a + v_2^a}{v_1 + v_2} - 1 \qquad /57/$$

Tak więc, algorytm programowania wieloetapowego daje rozwiązanie dokładne P^o /dla nie ograniczonego czasu obliczeń/ lub przybliżone P^a /dla ograniczonego czasu obliczeń/, z oszacowaniem błędu.

LITERATURA

[1] Bellman R.: Adaptacyjne procesy sterowania, PWN, Warszawa 1965.

[2] Garfinkel R.S., Nemhauser G.L.: Programowanie całkowitoliczbowe, PWN Warszawa 1978

[3] Hajewski M.: Poszukiwanie efektywnego algorytmu harmonogramowania zadań w systemach o strukturze równoległej. Praca dyplomowa magisterska /niepublikowana/, Instytut Automatyki, Pol.Śl., Gliwice 1983.

[4] Held M., Karp R.M.: A Dynamic Programming Approach to Sequencing Problem, SIAM, Journal Appl. Math., 10, No. 1, 1962, pp. 196-210.

[5] Held M., Karp R.M.: The Construction of Discrete Dynamic Programming Algorithms, IBM Systems Journal, 4, No.2, 1965, pp. 136-147.

[6] Kohler H.W., Steiglitz K.: Przeglądowe i iteracyjne metody obliczeniowe, Teoria szeregowania zadań /red. Coffman E.G. jr/, WNT, Warszawa 1980, ss. 241-301.

[7] Korbut A.A., Finkelsztejn J.J.: Programowanie dyskretne, PWN, Warszawa 1974.

[8] Kowalowski H., i inni: Harmonogramowanie produkcji krajami. Raport z pracy naukowo-badawczej - etap III.1, Instytut Automatyki, Gliwice 1

[9] Kowalowski H., i inni: Automatyzacja dyskretnych procesów przemysłowych, WNT /w druku/.

[10] Marecki F., Szendzielorz K.: Koncepcja harmonogramowania pracy krajami w FSM Tychy, Zeszyty Naukowe Pol.Śl., Seria: Automatyka, z.44, Gliwice 1978, ss. 79-86.

[11] Marecki F.: Control of Discrete Processes, International Conference on "Control Systems and Computer Science", Politechnical Institute of Bucharest, Bucharest 1983.

[12] Niderliński A.: Harmonogramowanie produkcji a wielopoziomowe wielowymiarowe dyskretne układy regulacji nadążnej, ZN Pol.Śl., seria: Automatyka, z. 55, Gliwice 1980, ss.63-70.

[13] Potrzebowski H.: Podział i ograniczenie jako metoda dekompozycji problemu wyznaczania harmonogramu na równoległych maszynach, Archiwum Automatyki i Telemechaniki, t. XXIV, z.2, 1979, ss. 301-313.

[14] Prejsnar J.: Analiza algorytmów harmonogramowania dyskretnych procesów przemysłowych w systemach o strukturze równoległej, Praca dyplomowa magisterska /niepublikowana/, Instytut Automatyki, Pol.Śl., Gliwice 1983.

[15] Szendzielorz K.: Harmonogramowanie zadań na maszynach równoległych, III Konferencja nt. "Metody i środki projektowania automatycznego", Instytut Podstaw Budowy Maszyn, Pol.Warszawska, Warszawa 1981.

[16] Szendzielorz K.: Kalendarnoje płanirowanije riezanija karoserijnych listow, ARS'81, Ostrawa 1981.

[17] Węglarz J.: Złożoność obliczeniowa problemów szeregowania zadań w celu minimalizacji maksymalnego opóźnienia, Zeszyty Naukowe Pol. Śl., seria: Automatyka z. 46, Gliwice 1978, ss. 160-167.

[18] Węglarz J.: Złożoność obliczeniowa problemów szeregowania zadań na równoległych maszynach z uwzględnieniem dodatkowych zasobów, ZN Pol. Śl., seria: Automatyka, z.56, Gliwice 1980,

[19] Węgrzyn S.: Podstawy informatyki, PWN, Warszawa 1982.

[20] Wirth N.: Algorytmy + struktury danych = programy, PWN, Warszawa 1981.

[21] Wołany D. : Harmonogramowanie procesu wykrawania blach karoseryjnych, Praca dyplomowa magisterska /niepublikowana/, Instytut Automatyki, Pol. Śl., Gliwice 1982.

[22] Zemełka P.: Analiza sterowania procesem wykrawania blach karoseryjnych, Praca dyplomowa magisterska /niepublikowana/, Instytut Automatyki, Pol. Śl., Gliwice 1980.

Recenzent:Prof.dr hab.inż.Stanisław Piasecki

Wpłynęło do Redakcji do 30.03.1984r.

МИНИМАЛИЗАЦИЯ ПРОИЗВОДСТВЕННЫХ ПОТЕРЬ ПРОЦЕССА РЕЗКИ КУЗОВНЫХ ЛИСТОВ

Р е з ю м е

В работе дана формулировка и решение задачи минимализации производственных потерь процесса резки кузовных листов. Выделены потери от холостых ходов машин а также от несвоевременной реализации заданий. Поставленную задачу решено алгоритмом многоэтапного программирования.

MINIMIZATION OF PRODUCTION DETERIORATION IN THE PROCESS OF BODY SHEETS CUTTING

S u m m a r y

A problem of minimization of production deterioration in the process of motor - car body sheets cutting is formulated and solved. Deteriorations resulted from breakdowns and delays in tasks realization are considered.