ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1986

Seria: AUTOMATYKA z. 81 Nr kol. 851

Franciszek MARECKI

Wydział Automatyki i Informatyki

Politechniki Śląskiej w Gliwicach

BALANSOWANIE LINII MONTAŻOWEJ Z OGRANICZENIAMI CZASOWYMI

Streszczenie. W pracy przedstawiono uogólniony model balansowania linii montażowej. W modelu tym uwzględniono dodatkowo ograniczenia czasowe. Dla optymalnego rozwiązania problemu podano algorytm programowania wieloetapowego.

1. WPROWADZENIE

Podstawowy problem balansowania linii montażowej, sformułowany przez M. E. Salvesona w [10] i [11], polega na wyznaczeniu minimalnej liczby podzbiorów operacji dla stanowisk pracy na linii. Zakłada się przy tym, że dany jest zbiór operacji montażowych, ich czasy oraz relacja kolejności między nimi. Ponadto wymuszony jest cykl linii. Cykl jest czasem, jaki ma do dyspozycji każdy monter dla wykonania wszystkich operacji na swoim stanowisku pracy. Podstawowy problem balansowania linii montażowej można rozwiązać za pomocą różnych algorytmów [12].

W rzeczywistym procesie montażu występuje wiele różnorodnych ograniczeń [7]. Balansowanie linii montażowej bez uwzględnienia tych ograniczeń czyni otrzymamy rezultat nieprzydatnyr z praktycznego punktu widzenia. Stąd też istotne znaczenie ma formułowanie modeli adekwatnych do rzeczywistości oraz poszukiwanie algorytmów rozwiązania tak sformułowanych problemów [2].

W niniejszej pracy zostanie sformułowany i rozwiązany problem balansowania linii montażowej z dodatkowymi ograniczeniami czasowymi. Ograniczenia tego typu występują pomiędzy parami operacji. Polegają one na określeniu przedziału czasu, w którym musi być wykonana operacja realizowana po pewnej operacji. Biorąc pod uwagę wszystkie pary operacji, można dla danej operacji wyznaczyć najwcześniejszy i najpóźniejszy termin jej rozpoczęcia. Terminy te są zależne od momentów zakończenia operacji poprzedzających daną operację. A zatem rozważane ograniczenia czasowe mają charakter względny.

W punkcie 2 zostanie sformułowany model matematyczny balansowania linii z ograniczeniami czasowymi. Algorytm rozwiązania tego problemu będzie przedstawiony w punkcie 3. Z kolei w punkcie 4 zostanie przeprowadzona dyskusja efektywności algorytmu.

2. SFORMUŁOWANIE PROBLEMU

Załóżmy, że dany jest zbiór operacji

$$\Omega = \{\omega_n\}, \quad n = 1,\dots,N \tag{1}$$

gdzie:

ω_n - n-ta operacja

N - liczba operacji

Relacja kolejności operacji dana jest macierzą:

$$\Gamma = [\gamma_{\vartheta,n}] \quad \begin{array}{l} \vartheta = 1,\dots,N \\ n = 1,\dots,N \end{array} \tag{2}$$

Elementy tej macierzy mają następujące znaczenia:

$$\gamma_{\vartheta,n} = \begin{cases} 1 : & \text{jeśli } \omega_\vartheta \text{ jest bezpośrednim poprzednikiem } \omega_n \\ 0 : & \text{w przypadku przeciwnym} \end{cases} \tag{2a}$$

Czasy operacji dane są wektorem:

$$\Theta = [\vartheta_n], \tag{3}$$

gdzie:

ϑ_n - czas operacji ω_n

Wymuszony cykl linii spełnia warunek:

$$\max_{1 \leqslant n \leqslant N} \vartheta_n \leqslant c < \sum_{n=1}^{n=N} \vartheta_n, \tag{4}$$

gdzie:

c - cykl linii

Załóżmy, że dane są minimalne czasy, jakie muszą upłynąć pomiędzy zakończeniem i rozpoczęciem odpowiednich operacji:

$$\Phi = [\varphi_{j,n}], \quad j = 1,\dots,N \tag{5}$$

gdzie:

$\varphi_{j,n}$ - minimalny czas, jaki musi upłynąć pomiędzy zakończeniem ω_j i rozpoczęciem ω_n.

Analogicznie przyjmiemy maksymalne czasy, jakie mogą upłynąć pomiędzy zakończeniem i rozpoczęciem odpowiednich operacji.

$$\Psi = [\psi_{j,n}], \tag{6}$$

gdzie:

$\psi_{j,n}$ - maksymalny czas, jaki może upłynąć pomiędzy zakończeniem ω_j i rozpoczęciem ω_n.

Jako kryterium balansowania linii montażowej przyjmiemy minimalizację liczby stanowisk pracy. Oznaczmy przez t_n moment zakończenia ω_n na linii. Funkcję celu zapiszemy w postaci:

$$Q = \sum_{k=1}^{k=K} q_k \longrightarrow \min \tag{7}$$

przy tym:

$$q_k = \begin{cases} 1 & : \text{ jeśli } \Omega_k = 0 \\ 0 & : \text{ w przypadku przeciwnym} \end{cases} \tag{7a}$$

Oznaczmy przez $[\cdot]^+$ najmniejszą liczbę całkowitą nie mniejszą niż wartość w nawiasie kwadratowym. Przez Ω_k oznaczamy podzbiór operacji przydzielonych do k - tej stacji montażowej.

Zauważmy, że numer stanowiska pracy, do którego przydzielono operację ω_n, można wyznaczyć z formuły:

$$m_n = \left[\frac{1}{c} t_n\right]^+, \tag{8}$$

gdzie:

m_n - numer stanowiska pracy, do którego przydzielono ω_n.

Dopuszczalny balans linii musi spełniać następujące ograniczenia:

- niepodzielności operacji

$$\left[\frac{1}{c} t_n\right]^+ = \left[\frac{1}{c}(t_n - \vartheta_n)\right]^+ \qquad n = 1,\dots,N \tag{9a}$$

- kolejności operacji

$$\forall_n \forall_{\vartheta} (\gamma_{\vartheta,n} = 1) \Longrightarrow (t_{\vartheta} \leq t_n - \vartheta_n) \tag{9b}$$

- czasowe

$$\forall_n \forall_j \; t_j + \varphi_{j,n} \leq t_n - \vartheta_n \leq t_j + \psi_{\vartheta,n} \tag{9c}$$

A zatem w sformułowanym problemie balansowania linii montażowej z ograniczeniami czasowymi występuje N niewiadomych, którymi są chwile zakończenia operacji na linii.

3. ALGORYTM

Do rozwiązania sformułowanego problemu przedstawiony zostanie algorytm programowania wieloetapowego. Algorytm ten jest oparty na idei wieloetapowych procesów decyzyjnych [1], [4], [9] oraz na metodzie podziału i ograniczeń [6], [3], [5]. Podstawowe znaczenie w algorytmie programowania wieloetapowego mają definicje: stanu procesu decyzyjnego, wartości stanu, procedur generowania stanów oraz reguł eliminacji stanów nieperspektywicznych.

Stan przedstawia sytuację po podjęciu decyzji, natomiast wartość stanu jest oceną tej sytuacji z punktu widzenia przyjętego kryterium. Ciąg stanów nazwiemy trajektorią. Każda trajektoria wychodzi z tego samego stanu początkowego. Stan początkowy (inicjujący obliczenia) przedstawia sytuację przed podjęciem jakiejkolwiek decyzji. Każdy stan końcowy przedstawia dopuszczalne rozwiązanie problemu. Rozwiązanie optymalne daje stan końcowy o najmniejszej wartości (dla kryterium minimalizacyjnego).

Generowanie stanów polega na wyznaczaniu kolejnych stanów trajektorii. W zależności od liczby podjętych decyzji stany dzielimy na etapy. Stan należący do η-tego etapu, $\eta = 0,1,\dots,N-1$ nazywamy stanem aktywnym, bowiem pozwala wygenerować dalsze stany. Stany N-tego etapu nie są aktywne. Stany generowane są z wybranego stanu aktywnego. Jako reguły wyboru stosuje się: FIFO, LIFO, LLB, itp. [5], przyjmując strategię czystą lub mieszaną. Załóżmy, że w generowaniu stanów wykorzystawane są jednokrokowe, zupełne reguły podziału. A zatem stan aktywny wybrany do dalszego generowania przestaje być aktywny. W trakcie obliczeń zapamiętywane są tylko stany aktywne oraz aktualnie najlepszy stan końcowy.

Dla zwiększenia efektywności algorytmu eliminowane są stany (nieperspektywiczne), które nie pozwalają osiągnąć rozwiązania optymalnego. Wyróżniamy przy tym reguły: wyczerpywania, sondowania i dominacji. Reguła wyczerpywania pozwala wyeliminować stan, z którego nie można uzyskać rozwiązania dopuszczalnego. Reguła sondowania eliminuje stan, który nie pozwala uzyskać rozwiązania lepszego niż aktualnie najlepsze. Reguła dominacji porównuje dwa stany aktywne. Z dwóch stanów aktywnych ten jest lepszy, którego stan lokalnie optymalny jest lepszy. Stan lokalnie optymalny jest najlepszym stanem końcowym, jaki można otrzymać z danego stanu aktywnego.

3.1. Określenia podstawowe

W dalszym ciągu l-ty stan η-tego etapu oznaczymy $P^{l,\eta}$. Stan lokalnie optymalny dla $P^{l,\eta}$ oznaczymy $P^{ol,\eta}$, natomiast stan globalnie optymalny P^{o}. Ponadto P^{a} oznaczać będzie aktualnie najlepszy stan końcowy P^{k}. Stan wybrany do generowania oznaczymy przez $P^{\lambda,\eta-1}$, a stan wygenerowyny przez P. Wartość stanu będzie oznaczona przez V z odpowiednimi indeksami.

Def. 1.: Stan procesu decyzyjnego jest wektorem

$$P^{l,\eta} = \left[p_n^{l,\eta}\right], \quad n = 1,\dots,N \quad l = 1,\dots,L \quad \eta = 0,\dots,N \tag{10}$$

gdzie:

η - numer etapu decyzyjnego

l - numer stanu w ramach η-tego etapu

L_η- liczba stanów η-tego etapu.

Elementy tego wektora mają następujące wartości:

$$p_n^{l,\eta} = \begin{cases} t_n & : \text{ jeśli podjęto decyzję o realizacji } \omega_n \\ 0 & : \text{ w przypadku przeciwnym} \end{cases} \tag{10a}$$

A zatem stan początkowy $P^{1,0}$ jest wektorem zerowym, a wszystkie współrzędne każdego stanu $p^{l,N}$ są dodatnie. Na podstawie stanu końcowego wyznaczamy wprost dopuszczalny balans linii, zgodnie z (8).

Def. 2.: Wartość stanu jest skalarem wyznaczanym z formuły

$$V^{l,\eta} = \sum_{k=1}^{k=K^{l,\eta}} q_k^{l,\eta}, \tag{11}$$

przy tym

$$q_k^{l,\eta} = \begin{cases} 1 : \text{ jeśli } \underset{n}{\exists} \left[\frac{1}{c} p_n^{l,\eta}\right]^+ = k \\ 0 : \text{ w przypadku przeciwnym} \end{cases} \tag{11a}$$

ponadto

$$T^{l,\eta} = \max_{1 \leqslant n \leqslant N} p_n^{l,\eta} \tag{12}$$

Optymalny stan końcowy wynika z warunku:

$$\left(\min_l V^{l,N} = V^{l^0,N}\right) \Longrightarrow \left(P^{l^0,N} = P^0\right) \tag{13a}$$

lub:

$$\left(\min_k V^k = V^{\varkappa}\right) \Longrightarrow \left(P^{\varkappa} = P^0\right) \tag{13b}$$

jako że optymalny stan końcowy ma najmniejszą wartość.

3.2. Generowanie stanów

W rozważaniach dotyczących generowania stanów pominiemy reguły i strategie wyboru, skupiając uwagę na procedurze generowania stanów. Załóżmy, że wybrany został $P^{\lambda,\eta-1}$. Stan ten należy uzupełnić o dopuszczalną operację ω_n tak, by otrzymać nowy stan dopuszczalny P na η-tym etapie. Stosując zupełną regułę podziału wykorzystujemy każdą operację ω_n, która pozwala wygenerować dopuszczalny stan P. Biorąc pod uwagę tylko jedną operację, otrzymujemy jednokrokową regułę podziału.

Procedura generowania stanów ma postać:

$$\forall_n \forall_\vartheta \forall_j \left(p_n^{\lambda,\eta-1} = 0\right) \wedge \left[\left(\gamma_{\vartheta,n} = 1\right) \Rightarrow \left(p_\vartheta^{\lambda,\eta-1} > 0\right)\right]$$

$$\wedge \left(t_n - \vartheta_n \leq T_{n,\psi}^{\lambda,\eta-1}\right) \Longrightarrow \left(P = P^{\lambda,\eta-1} + \Delta P\right) \tag{14}$$

Elementy wektora ΔP mają następujące wartości:

$$\Delta p_i = \begin{cases} t_n : & \text{dla } i = n \\ 0 : & \text{w przypadku przeciwnym} \end{cases} \tag{15}$$

Chwilę t_n wyznaczamy z formuły:

$$t_n = \begin{cases} \max\left(T^{\lambda,\eta-1}, T_{n,\varphi}^{\lambda,\eta-1}\right) + \vartheta_n : \ \text{jeśli} \\ \left[\frac{1}{c} \max\left(T^{\lambda,\eta-1}, T_{n,\varphi}^{\lambda,\eta-1}\right)\right]^+ = \\ = \left[\frac{1}{c}\left\{\max\left(T^{\lambda,\eta-1}, T_{n,\varphi}^{\lambda,\eta-1}\right) + \vartheta_n\right\}\right]^+ \\ c \cdot \left[\frac{1}{c} \max\left(T^{\lambda,\eta-1}, T_{n,\varphi}^{\lambda,\eta-1}\right)\right]^+ + \vartheta_n : \\ \text{w przypadku przeciwnym} \end{cases} \tag{16}$$

Przez $T_{n,\varphi}^{\lambda,\eta-1}$ oznaczono chwilę najwcześniejszego (a przez $T_{n,\psi}^{\lambda,\eta-1}$ chwilę najpóźniejszego) rozpoczęcia operacji ω_n, przy tym:

$$T_{n,\varphi}^{\lambda,\eta-1} = \max_{j \in J^{\lambda,\eta-1}} \left(p_j^{\lambda,\eta-1} + \varphi_{j,n}\right) \tag{17a}$$

$$T_{n,\psi}^{\lambda,\eta-1} = \min_{j \in J^{\lambda,\eta-1}} \left(p_j^{\lambda,\eta-1} + \psi_{j,n}\right) \tag{17b}$$

oraz:

$$\underset{j}{\forall}\ (p_j^{\lambda,\eta-1} > 0) \Rightarrow (j \in J^{\lambda,\eta-1}) \tag{17c}$$

W ten sposób wychodząc ze stanu $p^{1,0}$ i stosując procedurę (14) można wygenerować wszystkie trajektorie.

Równocześnie z generowaniem stanów wyznaczamy wartość stanu, stosując formułę:

$$V = \begin{cases} V^{\lambda,\eta-1}; & \text{jeśli } \left[\frac{1}{c}\ t_n\right]^+ = K^{\lambda,\eta-1} \\ V^{\lambda,\eta-1} + 1; & \text{w przypadku przeciwnym} \end{cases} \tag{18}$$

3.3. Eliminowanie stanów

W trakcie obliczeń eliminowane są stany nieperspektywiczne za pomocą reguł: wyczerpywania, sondowania i dominacji. Wygenerowany stan P jest testowany na perspektywiczność.

Reguła wyczerpywania ma postać twierdzenia.

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

$$\underset{n}{\exists}\ \underset{j}{\exists}\ (p_n = 0) \wedge (p_j + \varphi_{j,n} < t_n - \vartheta_n) \tag{19}$$

Dowód: Zgodnie z (14) operacja spełniająca warunek (19) nie pozwala wygenerować kolejnego stanu. Generowanie stanów na podstawie operacji spełniających warunek (14) jest niecelowe. Wynika to z faktu, że dla wyznaczenia rozwiązania dopuszczalnego trzeba wygenerować stan N-tego etapu, a zatem przydzielić do realizacji wszystkie operacje. Skoro w stanie P nie można przydzielić operacji ω_n, to tym bardziej nie można jej przydzielić w stanach następujących po P. A zatem ze stanu P nie można dojść do stanu końcowego, czyli w stanie P wyczerpane zostały możliwości generowania rozwiązań dopuszczalnych.

Reguła sondowania opiera się na oszacowaniu dolnego ograniczenia funkcji kryterialnej dla stanu P oraz na znajomości aktualnie najlepszego rozwiązania końcowego P^a.

Tw. 2.: Stan P jest nieperspektywiczny, jeżeli spełnia warunek:

$$V^a < \left[\frac{1}{c}\left(T + \sum_{j \in J} \vartheta_j\right)\right]^+ \tag{20}$$

Dla dowodu wykażemy, że prawa strona nierówności (20) jest dolnym ograniczeniem funkcji kryterialnej. W stanie P wszystkie operacje zostają zakończone w chwili T. Gdyby operacje nie należące do P (czyli $j \in J$) były wykonane od chwili T bez luzów czasowych, to montaż wszystkich operacji zakończyłby się po najkrótszym czasie. Stąd prawa strona nierówności (20) określla najmniejszą liczbę stanowisk pracy, jaką można otrzymać ze stanu P. Z kolei zauważmy, że liczba ta jest nie większa od jakiegokolwiek stanu końcowego P' otrzymanego z P. A zatem

$$V^a < \left[\frac{1}{c}\left(T + \sum_{j\in J} \vartheta_j\right)\right]^+ \leqslant V' \tag{21}$$

Z (21) widzimy, że każdy stan końcowy P' jest gorszy od P^a. A zatem stan P jest nieperspektywiczny.

Ze stanu P można również wyznaczyć (np. heurystycznie) jedną trajektorię, uzyskując stan końcowy P'. Jeżeli

$$\left[\frac{1}{c}\left(T + \sum_{j\in J} \vartheta_j\right)\right]^+ = V', \tag{22}$$

to wnioskujemy, że

$$P' = P^o \tag{23}$$

A zatem, skoro znany jest lokalnie optymalny stan P^o, to generowanie z P dalszych trajektorii nie jest potrzebne.

Sondowanie stanu P polega na sprawdzeniu w pierwszej kolejności warunku (20). Jeżeli warunek ten jest spełniony, to stan P zostaje wyeliminowany. W przeciwnym przypadku generujemy stan P' i sprawdzamy warunek (22). Jeżeli warunek ten nie jest spełniony, to stan P pozostaje perspektywiczny i należy z niego generować dalsze stany. Jednocześnie porównujemy P' z P^a, eliminując gorszy z nich. W przypadku gdy warunek (22) jest spełniony, nie generujemy dalszych stanów z P. Ponadto tak jak poprzednio, porównujemy P' z P^a, eliminując gorszy z nich.

Dla wyjaśnienia dominacji stanów wprowadzimy definicję.

<u>Def. 3.</u>: Stan $p^{1,\eta}$ dominuje nad stanem P, jeżeli jest spełniony warunek:

$$\overset{o}{V}{}^{1,\eta} \leqslant \overset{o}{V} \tag{24}$$

W przypadku równości w (24) wybieramy stan, który został wygenerowany wcześniej.

Reguła dominacji stanów jest oparta na twierdzeniu.

<u>Tw. 3.</u>: Stan $p^{1,}$ dominuje nad stanem P, jeżeli jest spełniony warunek:

$$\forall_n \left[(p_n{}^{1,\eta} = 0) \Longleftrightarrow (p_n = 0)\right] \wedge (T_{n,\varphi}^{1,\eta} \leqslant T_{n,\varphi}) \wedge (T_{n,\psi} \leqslant T_{n,\psi}^{1,\eta}) \wedge (V^{1,\eta} \leqslant V) \tag{25}$$

Dowód tego twierdzenia polega na wykazaniu, że optymalna trajektoria $P, \ldots, P^o$ może być zrealizowana od stanu $p^{1,\eta}$. Oznacza to, że operacje nie należące do P i $P^{1,\eta}$ mogą być wykonane od stanu $p^{1,\eta}$ w przedziałach czasu wyznaczonych z trajektorii optymalnej $P,\dots,P^{o}$. Możliwość taka wynika wprost z (25). Zauważmy ponadto, że otrzymana w ten sposób trajektoria $p^{1,\eta},\dots,P^{k}$ nie musi być trajektorią lokalnie optymalną $p^{1,\eta},\dots,\overset{o}{P}{}^{1,\eta}$. A zatem

$$\overset{o}{V}{}^{1,\eta} \leqslant V^{k} = \overset{o}{V}, \tag{26}$$

skąd wnioskujemy, że stan $p^{1,\eta}$ dominuje nad stanem P. Zmieniając kierunki nierówności w (25) otrzymujemy warunek dominacji stanu P nad stanem $p^{1,\eta}$.

4. EFEKTYWNOŚĆ ALGORYTMU

Problem balansowania linii montażowej jest NP-zupełny w sensie złożoności obliczeniowej. Stąd też do jego rozwiązania zastosowano algorytm przeglądowy. Jeżeli obliczenia zostają przerwane przed wygenerowaniem wszystkich rozwiązań dopuszczalnych, to otrzymujemy jedynie rozwiązanie suboptymalne P^{a}. Dla rozwiązania tego szacujemy błąd względny [5]:

$$\xi = \frac{V^{a}}{V^{o}} - 1 \leqslant \frac{V^{a}}{d} - 1, \tag{27}$$

gdzie:

d - najmniejsze dolne ograniczenie dla wygenerowanych stanów aktywnych.

Efektywność algorytmu może być określona przez czas dojścia do rozwiązania optymalnego lub, przy limitowanym czasie, błędem wyznaczonego rozwiązania suboptymalnego. W obydwu przypadkach efektywność ta zależy od liczby wygenerowanych oraz liczby wyeliminowanych stanów.

Generowanie stanów musi zapewnić wyznaczenie wszystkich rozwiązań dopuszczalnych, pomijając równocześnie rozwiązania nieperspektywiczne. Według procedury generowania stanów operacja ω_n zostaje zrealizowana w najwcześniejszym dopuszczalnym momencie t_n. Uwzględnienie realizacji operacji ω_n w późniejszym przedziale czasu dałoby stan nieperspektywiczny. Można to łatwo wykazać za pomocą reguły dominacji stanów. A zatem reguła dominacji stanów pozwala sprawdzić, czy są generowane stany nieperspektywiczne.

Liczba generowanych stanów jest zależna od stosowanych reguł i strategii wyboru. Dobre reguły i strategie wyboru pozwalają szybko osiągnąć rozwiązanie zbliżone do optymalnego. Tym samym liczba eliminowanych gorszych trajektorii rośnie. Określenie zbioru reguł i strategii ich wyboru wymaga rutyny i znajomości rozwiązywanego problemu. Z tych względów preferowane są mieszane strategie wyboru.

Eliminacja stanu powoduje pominięcie wiązki trajektorii wychodzącej z tego stanu. A zatem maleje liczba generowanych i zapamiętywanych stanów.

Jednakże, jeżeli warunek eliminacji stanu nie jest spełniony, to czas obliczeń rośnie. Z powyższych względów reguły eliminacji stanów powinny dawać duże prawdopodobieństwo wyeliminowania stanu. Ponadto czas potrzebny na sprawdzenie warunku eliminacji winien być krótki. Są to przeciwstawne wymagania, dlatego efektywność algorytmu często ocenia się na podstawie testów komputerowych. Należy jednak podkreślić, że rezultaty testów komputerowych są zależne od danych liczbowych oraz od konstrukcji programu komputerowego.

Poza wymienionymi wyżej elementami decydującymi o efektywności algorytmu można wprowadzić pewne uporządkowane struktury danych [13]. Uporządkowanie to jest uzależnione od stosowanych reguł wyboru i reguł eliminacji stanów. Stosując reguły wyboru FIFO lub LIFO stany aktywne można uporządkować wg kolejności ich wygenerowania. Reguła LLB określa wybór stanu o najmniejszym dolnym ograniczeniu funkcji kryterialnej. Czas poszukiwania takiego stanu skraca się, jeżeli stany są uporządkowane wg wartości odpowiednich dolnych ograniczeń. Dla reguły DF/LLB zbiór stanów aktywnych należy podzielić na podzbiory stanów η-tego etapu. Stosowanie mieszanych strategii wyboru komplikuje problem najlepszego uporządkowania stanów aktywnych. Najczęściej zapamiętywane są uporządkowania stanów odpowiadające każdej regule wyboru.

Uporządkowanie stanów jest również celowe ze względu na procedury eliminacji stanów. Reguły sondowania nie wymagają informacji o stanach aktywnych (parametry wygenerowanego stanu P są porównywane jedynie ze stanem aktualnie najlepszym P^a). W regule dominacji natomiast porównuje się stan P ze stanami aktywnymi $p^{l,\eta}$. Z tego względu dokonujemy przeglądu stanów aktywnych.

Analiza warunku dominacji stanów (25) prowadzi do wniosku, że wprowadzenie porządku leksykograficznego w zbiorze stanów aktywnych skraca czas obliczeń. Podstawowe znaczenie ma fakt, że dla wykrycia dominacji nie muszą być sprawdzone wszystkie stany aktywne. Po pierwsze dominacja może zachodzić tylko między stanami tego samego etapu. Z tego względu zbiór stanów aktywnych należy podzielić na podzbiory stanów η-tego etapu. Ponadto liczba stanów η-tego etapu może być duża: $L_\eta \leqslant \binom{N}{\eta}$. Z tego względu wprowadzimy porządek leksykograficzny stanów η-tego podzbioru. Gdy takie uporządkowanie istnieje, to sprawdzenie dominacji stanu P nie wymaga przeglądu wszystkich stanów aktywnych η-tego etapu. W pierwszym kroku można sprawdzić stan o numerze: $l^1 = \frac{1}{2} L_\eta$. Jeżeli stany $p^{l^1,\eta}$ i P nie spełniają warunku dominacji, to w dalszym ciągu sprawdzamy wykluczające się podzbiory stanów o numerach od 1 do l^1-1 oraz od stanu l^1+1 do L_η. Wybór odpowiedniego podzbioru zależy od tego, czy P poprzedza lub następuje po $p^{l^1,\eta}$. W ten sposób, dokonując podziału połówkowego, pomijamy niektóre stany. Rezultatem sprawdzenia może być dominacja stanu P (lub $p^{l,\eta}$) lub wpisanie tego stanu do η-tego podzbioru stanów aktywnych. Wpisanie to może nastąpić na pierwszą, ostatnią lub pośrednią pozycję - zależnie od relacji porządku leksykograficznego.

Dla wyjaśnienia porządku leksykograficznego wprowadzimy odpowiednią definicję. Załóżmy, że dane są dwa K-wymiarowe wektory X i Y.

Def. 4.: Wektor X poprzedza leksykograficzne Y, jeżeli jest spełniony warunek:

$$\underset{1 \leq k < \varkappa \leq K}{\forall} (x_1 < y_1) \vee (x_k = y_k) \wedge (x_\varkappa < y_\varkappa) \tag{28}$$

Relację poprzedzania zapiszemy w postaci:

$$X \prec Y \tag{29}$$

Wektory X i Y są alternatywne, jeżeli spełniają warunek:

$$(X \not\prec Y) \wedge (Y \not\prec X) \tag{30}$$

Aby wykorzystać wprowadzoną definicję porządku leksykograficznego do uporządkowania stanów aktywnych η-tego etapu, przyjmiemy:

- dla $k = 1, \ldots, \eta$

$$(x_k = y_k) \Longleftrightarrow (\vartheta_k^{1,\eta} = \vartheta_k) \tag{31a}$$

$$(x_k < y_k) \Longleftrightarrow (\vartheta_k^{1,\eta} < \vartheta_k) \tag{31b}$$

gdzie:

$\vartheta_k^{1,\eta}$ - numer k-tej dodatniej współrzędnej w stanie $P^{1,\eta}$

- dla $k = \eta + 1, \ldots, N$

$$(x_k = y_k) \Longleftrightarrow (T^{1,\eta}_{n_{k-\eta}, \varphi} \leq T_{n_{k-\eta}, \varphi}) \tag{32a}$$

$$(x_k < y_k) \Longleftrightarrow (T^{1,\eta}_{n_{k-\eta}, \varphi} > T_{n_{k-\eta}, \varphi}) \tag{32b}$$

- dla $k = N + 1, \ldots, 2N - \eta$

$$(x_k = y_k) \Longleftrightarrow (T_{n_{k-N}, \psi} \leq T^{1,\eta}_{n_{k-N}, \psi}) \tag{33a}$$

$$(x_k < y_k) \Longleftrightarrow (T_{n_{k-N}, \psi} > T^{1,\eta}_{n_{k-N}, \psi}) \tag{33b}$$

Jak widać, do określenia porządku leksykograficznego wykorzystuje się wszystkie warunki (z wyjątkiem warunku wartości stanów) występujące w regule dominacji stanów. Przyjmiemy, że stany $p^{1,\eta}$ i P są alternatywne, jeżeli odpowiednie wektory X i Y są alternatywne. Dominacji podlegają tylko stany alternatywne, a zdominowany zostaje stan alternatywny o mniejszej wartości. W zbiorze stanów aktywnych nie ma stanów alternatywnych.

5. ZAKOŃCZENIE

W pracy przedstawiono model matematyczny dla problemu balansowania linii montażowej z uwzględnieniem ograniczeń czasowych. Ograniczenia te wyznaczają względny przedział czasu, w którym musi być wykonana każda operacja. Przedział ten nie jest z góry dany, lecz wynika z terminów zakończenia operacji, które na linii są wykonane wcześniej niż dana operacja.

W procesie montażu ograniczenia czasowe wynikają z operacji klejenia, malowania, itp. Na liniach technologicznych ograniczenia są często związane z temperaturą, jaką musi posiadać obrabiany obiekt w trakcie określonej operacji.

Sformułowany problem rozwiązano za pomocą algorytmu programowania wieloetapowego. Efektywność tego algorytmu zależy w dużej mierze od konstrukcji programu komputerowego. Niektóre elementy algorytmu (podział i leksykograficzne uporządkowanie stanów aktywnych) zwiększają jego efektywność. Wpływ reguł i strategii wyboru oraz reguł eliminacji stanów na efektywność algorytmu można analizować na podstawie testów komputerowych.

Za pomocą przedstawionego modelu można rozwiązywać problemy balansowania linii z ograniczeniami wykluczania operacji [8] lub tzw. strefami (pozytywnymi i negatywnymi) [2]. Istota zagadnienia polega na tym, że jeżeli dwie operacje ω_ϑ i ω_n nie mogą być wykonane na jednym stanowisku pracy, to wystarczy przyjąć $\varphi_{\vartheta,n} \geqslant c$. Jeżeli z kolei operacje te mają tworzyć sekwencję, to zakładamy $\varphi_{\vartheta,n} = \Psi_{\vartheta,n} = 0$. A zatem model balansowania linii montażowej z ograniczeniami czasowymi stanowi uogólnienie wymienionych wyżej przypadków.

LITERATURA

[1] Bellman R.: Adaptacyjne procesy sterowania. PWN, Warszawa 1965, ss. 80-92.

[2] Buxey G. M.: Assembly Line Balancing with Multiple Stations, Management Science, V.20, No. 6, 1974, pp. 1010-1021.

[3] Garfinkel R. S., Nemhauser G. L.: Integer Programming, John Wiley and Sons, New York - London - Sydney - Toronto, 1972.

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

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

[6] Korbut A. A., Finkelsztejn J. J.: Programowanie dyskretne. PWN, Warszawa 1974, ss. 164-176.

[7] Marecki F.: Modelowanie symulacyjne linii montażowej samochodu małolitrażowego. INFORMATYKA, No 7-8, 1975, ss. 25-28.

[8] Marecki F.: Balansowanie linii montażowej z ograniczeniami wykluczenia operacji. ZN Pol. Śl., seria: Automatyka, z. 63, Gliwice 1982, ss. 81-89.

[9] Marecki F.: Control of Discrete Processes, 5-th International Conference on "Control Systems and Computer Science", Politechnical Institute of Bucharest, Bucharest 1983.

[10] Salveson M.E.: The Assembly Line Balancing Problem, Transactions of the ASME, V. 77, No 6, 1955, pp. 939-947.

[11] Salveson M. E.: The Assembly Line Balancing Problem, The Journal of Industrial-Engineering, V. 6, No. 3, 1955, pp. 18-25.

[12] Szkurba W. W., Bieleckij S. A.: Czislienyje mietody w rieszenii zadaczi bałansirowanija sborocznoj linii. Kibiernietika, Kijew 1977, ss. 96-108.

[13] Wirth W.: Algorytmy + Struktury Danych = Programy, WNT, Warszawa 1980.

[14] Piasecki S.: Operatywne kierowanie pracą linii montażowych. Prace IMS PAN, Warszawa 1981, ss. 3-20.

Recenzent: prof. dr hab. inż. Stanisław Piasecki

Wpłynęło do Redakcji 25.04.1984 r.

БАЛАНСИРОВАНИЕ МОНТАЖНОЙ ЛИНИИ С ВРЕМЕННЫМИ ОГРАНИЧЕНИЯМИ

Р е з ю м е

В работе представлена обобщенная модель балансировки монтажной линии. Модель учитывает временные ограничения. С целью оптимального решения проблемы применён алгоритм многоэтапного программирования.

ASSEMBLY LINE BALANCING IN THE PRESENCE OF TIME CONSTRAINTS

S u m m a r y

A generalised model of an assembly line balancing is presented. The model takes additionally time constraints into account. Multistage programming algorithm is presented to find optimal solutions.