ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1982

Seria: AUTOMATYKA z. 63 Nr kol. 735

Franciszek MARECKI

Instytut Automatyki
Politechniki Śląskiej

BALANSOWANIE SZEREGOWEJ LINII MONTAŻOWEJ Z OGRANICZENIAMI WYKLUCZANIA OPERACJI

Streszczenie. W pracy przedstawiono model matematyczny i algorytm balansowania szeregowej linii montażowej z ograniczeniami wykluczania operacji. Ograniczenia wykluczania wynikają z konieczności wykonywania operacji po różnych stronach linii i (lub) limitowanego obszaru dla kontenerów z detalami na stanowisku pracy. Postawiony problem został rozwiązany za pomocą algorytmu programowania wieloetapowego.

1. WPROWADZENIE

Pod pojęciem balansowania linii montażowej rozumiemy rozdział operacji na stanowiska pracy. Przy formułowaniu problemu zwykle zakłada się, że dane są: zbiór operacji, ograniczenia kolejnościowe wykonywania oraz czasy operacji. Ponadto zadany jest cykl linii. Jako kryterium optymalizacji balansowania przyjmuje się minimalizację liczby stanowisk pracy. Jednakże rezultatem rozwiązania problemu winna być nie tylko minimalna liczba stanowisk pracy, ale również podzbiory operacji dla każdego stanowiska. Przegląd metod rozwiązania tak sformułowanego problemu zamieszczono w [1]. Pewne uogólnienia tego problemu przedstawiono w [2] i [3].

W dalszym ciągu będzie rozpatrywana linia montażowa z szeregowymi stanowiskami pracy. Jednocześnie przyjmiemy, że transporter główny nie umożliwia przejść monterów z lewej na prawą stronę linii (i odwrotnie). A zatem stanowisko pracy musi być zlokalizowane po jednej stronie linii. Drugie ograniczenie przestrzenne jest związane z gabarytem detali. Założymy, że niektóre podzbiory operacji nie mogą być wykonywane na jednym stanowisku pracy, z uwagi na zbyt duże gabaryty detali. Konsekwencją tych założeń jest wykluczenie niektórych operacji na pewnych stacjach.

Problem balansowania linii montażowej, z uwzględnieniem ograniczeń przestrzennych, zostanie rozwiązany metodą programowania wieloetapowego. W punkcie 2 sformułowany zostanie problem w postaci matematycznej. Następnie w punkcie 3 przedstawimy algorytm, a w punkcie 4 uwagi końcowe i wnioski.

2. SFORMUŁOWANIE PROBLEMU

Załóżmy, że dany jest zbiór operacji:

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

gdzie:

ω_n - n-ta operacja,

N - liczba operacji.

Czasy wykonywania operacji dane są wektorem:

$$\Theta = \left[\vartheta_n\right], \tag{2}$$

gdzie:

ϑ_n - czas wykonywania operacji ω_n.

Ograniczenia kolejności wykonywania operacji przedstawia macierz:

$$\Gamma = \left[\gamma_{\nu,n}\right]_{(\nu=\overline{1,N})} \tag{3}$$

Elementy tej macierzy mają następujące znaczenie:

$$\gamma_{\nu,n} = \begin{cases} 1: \text{ jeśli } \omega_\nu \text{ jest bezpośrednim poprzednikiem operacji } \omega_n, \\ 0: \text{ w przeciwnym przypadku} \end{cases} \tag{4}$$

Niechaj będzie dany cykl linii, spełniający warunek:

$$\min_{1\leq n\leq N} \vartheta_n \leq C < \sum_{n=1}^{n=N} \vartheta_n \tag{5}$$

gdzie:

C - cykl linii.

Lokalizacja operacji w poprzek linii zapisana jest wektorem:

$$B = \left[b_n\right] \tag{6}$$

Elementy tego wektora mają następujące znaczenia:

$$b_n = \begin{cases} +1: \text{ jeśli operacja } \omega_n \text{ \underline{musi być} wykonana po prawej stronie linii;} \\ 0: \text{ jeśli operacja } \omega_n \text{ \underline{może być} wykonana po prawej lub lewej stronie linii;} \\ -1: \text{ jeśli operacja } \omega_n \text{ \underline{musi być} wykonana po lewej stronie linii.} \end{cases} \tag{7}$$

Przed wprowadzeniem warunku, jaki winny spełniać operacje przydzielone na jedno stanowisko pracy, określimy dwa wektory:

$$W = [w_n], \tag{8}$$

gdzie:

w_n - współczynnik gabarytu detali dla operacji ω_n,

oraz

$$D = [d_k]_{(k=\overline{1,K})} \tag{9}$$

gdzie:

d_k - dopuszczalna pojemność k-tego stanowiska pracy,

K - liczba stacji linii montażowej.

Niechaj d_k oznacza zbiór operacji przydzielonych na k-te stanowisko pracy. Operacje te winny spełniać warunek:

$$\sum_{\omega_n \in \sigma_k} \omega_n \leq d_k \tag{10}$$

Wektory: (6), (8) i (9) przedstawiają elementy ograniczeń wykluczania operacji na jednym stanowisku pracy. Warto ponadto podkreślić, że rozpatrujemy linię szeregową składającą się z K stacji. Stanowiska pracy mogą być zlokalizowane po lewej lub prawej (wyłącznie) stronie transportera, na poszczególnych stacjach. Tym samym liczba szeregowych stanowisk pracy nie może przekroczyć liczby stacji (K) na linii.

Oznaczmy przez t_n moment zakończenia wykonywania operacji ω_n na linii. Moment ten jest liczony od chwili wejścia chassis montowanego obiektu na linię. Numer stanowiska, na którym została wykonana operacja ω_n, wyznaczamy z formuły:

$$m = \left[\frac{1}{C} t_n\right]^+ \tag{11}$$

gdzie:

m - numer stanowiska pracy,

$[\cdot]^+$ - najmniejsza liczba całkowita nie mniejsza od wartości w nawiasie kwadratowym.

Kryterium optymalizacji balansowania linii można zapisać w postaci:

$$Q' = \max_{1 \leq n \leq N} \left[\frac{1}{C} t_n\right]^+ \rightarrow \min \tag{12a}$$

gdzie:

Q - minimalna liczba stanowisk pracy.

Rezultat (12a) można osiągnąć prościej, stosując kryterium:

$$Q = \max_{1 \leq n \leq N} t_n \longrightarrow \min \tag{12b}$$

gdzie:

Q - minimalny czas wykonania wszystkich operacji na linii.

Ponadto kryterium (12b) pozwala wybrać najlepsze rozwiązanie spośród tych, które dają minimalną liczbę stanowisk pracy.

Zakładamy, że operacje są nie podzielne, tzn. każda operacja musi być wykonana na jednym stanowisku pracy. Ograniczenie to można zapisać w postaci

$$\left[\frac{1}{C}(t_n - \vartheta_n)\right]^+ = \left[\frac{1}{C}\, t_n\right]^+. \tag{13}$$

Ograniczenia kolejnościowe zapisujemy jako

$$\forall_{\nu} \forall_{n} (\gamma_{\nu,n} = 1) \Rightarrow (t_{\nu} < t_n) \tag{14}$$

Ograniczenia lokalizacji mają postać

$$\forall_{n} \forall_{i} \left\{ \left[\frac{1}{C}\, t_n\right]^+ = \left[\frac{1}{C}\, t_i\right]^+ \right\} \Rightarrow (b_n \cdot b_i \geq 0). \tag{15}$$

Ograniczenia gabarytowe są przedstawione następująco

$$\forall_{n} \left\{ \left[\frac{1}{C}\, t_n\right]^+ = k \right\} \Rightarrow \left(\sum_{\omega_n \in \sigma_k} w_n \leq d_k \right). \tag{16}$$

Tak więc w funkcji celu (12) oraz ograniczeniach: (12), (14), (15) i (16) niewiadomymi są chwile t_n $(n = \overline{1,N})$. Po wyznaczeniu tych chwil można z (11) określić podzbiory operacji na stanowiska pracy.

3. ALGORYTM

Algorytm rozwiązania sformułowanego wyżej problemu będzie oparty na programowaniu wieloetapowym. Kolejne etapy oznaczymy indeksem η, $(\eta = \overline{0,N})$. Podstawowymi elementami tego algorytmu są: stan procesu decyzyjnego, wartość stanu, procedura generowania stanów oraz procedura eliminowania stanów niepersspektywicznych. Decyzja jest interpretowana jako przydział operacji do realizacji. Ciąg decyzji nazwiemy <u>strategią</u>. Podejmowanie decy-

zji rozpoczniemy w sytuacji, gdy nie przydzielono jeszcze żadnej operacji do realizacji. Stan procesu decyzyjnego interpretuje zatem sytuację na linii po podjęciu decyzji. Ciąg stanów nazwiemy trajektorią. Generowanie stanów będzie prowadzone etapami, tzn. wszystkie stany etapu η-tego otrzymujemy na podstawie tylko stanów poprzedniego η-1-go etapu. Sposób generowania stanów uzasadnia nazwę - programowanie wieloetapowe. Z każdym stanem zwiążemy jego wartość, którą w rozpatrywanym przypadku będzie chwila zakończenia wykonywania operacji należących do tego stanu. W ten sposób stan ostatniego N-tego etapu, który ma najmniejszą wartość, daje optymalne rozwiązanie problemu.

3.1. Stan procesu decyzyjnego i wartość stanu

Stany procesu decyzyjnego będą numerowane indeksem λ (lub l) w ramach etapu η (lub η-1).

Def. 1: Stan jest wektorem:

$$p^{\lambda,\eta} = \left[p_n^{\lambda,\eta}\right]_{\substack{(n=\overline{1,N})\\(\lambda=\overline{1,L_\eta})\\(\eta=\overline{0,N})}} \tag{17}$$

gdzie:

L_η - liczba stanów η-tego etapu.

Elementy tego wektora określamy następująco:

$$p_n^{\lambda,\eta} = \begin{cases} t_n & \text{- jeżeli operacja } \omega_n \text{ należy do stanu;} \\ 0 & \text{- w przeciwnym przypadku.} \end{cases} \tag{18}$$

A więc stan początkowy $p^{1,0}$ jest wektorem zerowym, natomiast każdy stan końcowy $p^{\lambda,N}$ ma wszystkie współrzędne dodatnie.

Z każdym stanem $p^{\lambda,\eta}$ zwiążemy jego wartość, którą oznaczymy $v^{\lambda,\eta}$

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

$$v^{\lambda,\eta} = \max_{1\leq n\leq N} p_n^{\lambda,\eta} \tag{19}$$

Każdy stan $p^{\lambda,N}$ daje dopuszczalne rozwiązanie problemu, a jego wartość jest czasem wykonania wszystkich operacji. A zatem kryterium (12) można napisać w postaci:

$$Q = \max_{1\leq n\leq N} p_n^{\lambda,N} \to \min \tag{20}$$

Wynika stąd, że optymalny stan ostatniego etapu wyznaczamy jako:

$$\left(\min_{1 \le \lambda \le L_N} V^{\lambda,N} = V^{\lambda^o,N} \right) \Rightarrow \left(P^{\lambda^o,N} = P^o \right), \tag{21}$$

gdzie:

P^o - optymalny stan N-tego etapu.

Z optymalnego stanu P^o, zgodnie z (11), określamy przydział operacji na stanowiska pracy.

3.2. Procedura generowania stanów

Procedura generowania stanów polega na uzupełnieniu stanu $P^{l,\eta-1}$ o dopuszczalną operację ω_n, co w konsekwencji daje nowy stan $P^{\lambda_n,\eta}$. Operacja ω_n jest dopuszczalna, jeśli wcześniej wykonano jej wszystkie poprzedniki lub gdy nie ma poprzedników. Ogólna procedura generowania stanów ma następującą postać:

$$\forall_n \forall_\vartheta \left\{ \left(p_n^{l,\eta-1} = 0 \right) \wedge \left\{ \left(\sum_{i=1}^{i=N} \gamma_{i,n} = 0 \right) \vee \left[\left(\gamma_{\vartheta,n} = 1 \right) \Rightarrow \left(p_\vartheta^{l,\eta-1} > 0 \right) \right] \right\} \right\} \wedge$$

$$\wedge \left\{ \left[\frac{1}{C} t_n \right]^+ \le K \right\} \Rightarrow \left(P^{\lambda_n,\eta} = P^{l,\eta-1} + \Delta P \right). \tag{22}$$

Elementy wektora ΔP mają następujące wartości:

$$\Delta p_i = \begin{cases} t_n & \text{gdy } i = n \\ 0 & \text{w przeciwnym przypadku} \end{cases} \tag{23}$$

Moment t_n wyznaczamy z formuły:

$$t_n = \begin{cases} v^{l,\eta-1} + \vartheta_n & \text{jeśli: } \left\{ \left[\frac{1}{C} (v^{l,\eta-1} + \vartheta_n \right]^+ = \left[\frac{1}{C} v^{l,\eta-1} \right]^+ \right\} \wedge \\ & \wedge (b_i \cdot b_n \ge 0) \wedge \left(\sum_i w_i + w_n \le d_k \right); \\ \left[\frac{1}{C} v^{l,\eta-1} \right]^+ \cdot C + \vartheta_n & \text{: w przeciwnym przypadku} \end{cases} \tag{24}$$

przy tym:

$$k = \left[\frac{1}{C} p_i^{l,\eta-1} \right]^+ = \left[\frac{1}{C} v^{l,\eta-1} \right]^+ \tag{25}$$

Tak więc operacja ω_n jest przydzielona na k-te stanowisko pracy (stację), jeżeli pozwala na to "luz" czasu i ograniczenia wykluczenia. W przeciwnym przypadku operację tę przydzielamy na k+1-sze stanowisko pracy -

pod warunkiem, że $k+i \leq K$. Jeżeli warunek ten nie jest spełniony, to nie można wygenerować stanu.

3.3. Procedura eliminowania stanów

W trakcie obliczeń niektóre stany można wyeliminować, gdyż nie prowadzą one do rozwiązania optymalnego. Eliminacja stanów zmniejsza zapotrzebowania na pamięć komputerową i może skrócić czas obliczeń. Po wyeliminowaniu stanu w dalszych obliczeniach pomijana jest wiązka trajektorii wychodzących z wyeliminowanego stanu. Z tego względu skraca się czas obliczeń. Jednakże, w przypadku gdy po testowaniu stan nie zostanie wyeliminowany - czas obliczeń zwiększa się. Stąd oszacowanie czasu obliczeń wymaga przeprowadzenia testów komputerowych (balansowanie linii jest problemem NP - zupełnym).

Załóżmy, że wychodząc ze stanu $P^{\lambda_1,\eta}$ można po optymalnej trajektorii otrzymać stan końcowy $P^{\lambda_1^o,N}$. Analogicznie z $P^{\lambda_2,\eta}$ otrzymamy $P^{\lambda_2^o,N}$.

Def. 3: Stan $P^{\lambda_1,\eta}$ dominuje nad stanem $P^{\lambda_2,\eta}$ jeżeli:

$$(V^{\lambda_1^o,N} \leq V^{\lambda_2^o,N}) \Leftrightarrow (P^{\lambda_1,\eta} \vdash P^{\lambda_2,\eta}), \tag{26}$$

gdzie:

$\vdash$ symbol dominacji stanów.

Ponieważ o dominacji stanów należy rozstrzygnąć na etapie η-tym, zatem wprowadzimy następującą regułę:

Tw. Stan $P^{\lambda_1,\eta}$ dominuje nad stanem $P^{\lambda_2,\eta}$ jeżeli:

$$\forall_n \left[(p_n^{\lambda_1,\eta} > 0) \Leftrightarrow (p_n^{\lambda_2,\eta} > 0)\right] \wedge (\sigma_k^{\lambda_1,\eta} = \sigma_k^{\lambda_2,\eta}) \wedge (V^{\lambda_1,\eta} \leq V^{\lambda_2,\eta}) \Rightarrow$$

$$\Rightarrow (P^{\lambda_1,\eta} \vdash P^{\lambda_2,\eta}), \tag{27}$$

przy tym:

$$\forall_i \left\{ \left\lceil \frac{1}{C} p_i^{\lambda,\eta} \right\rceil^+ = k \right\} \Rightarrow (\omega_i \in \sigma_k^{\lambda,\eta}). \tag{28}$$

Dowód tego twierdzenia opiera się na spostrzeżeniu, że trajektoria optymalna od stanu $P^{\lambda_2,\eta}$ do $P^{\lambda_2^o,N}$ jest realizowana od stanu $P^{\lambda_1,\eta}$. Ponieważ dodatkowo stan $P^{\lambda_1,\eta}$ ma mniejszą wartość, zatem $V^{\lambda_1^o,N} \leq V^{\lambda_2^o,N}$. Zatem ze stanu $P^{\lambda_2,\eta}$ nie można uzyskać rozwiązania lepszego, aniżeli ze stanu $P^{\lambda_1,\eta}$. Stąd stan $P^{\lambda_2,\eta}$ eliminujemy z dalszych obliczeń.

4. UWAGI KOŃCOWE I WNIOSKI

Przedstawiony w pracy model balansowania szeregowej linii montażowej różni się istotnie od opisów przedstawionych w literaturze przedmiotu. Wprowadzone ograniczenia wykluczania operacji na stanowisku pracy czynią model bardziej przydatny do wykorzystania w praktyce.

Ograniczenia wykluczania komplikują organizację linii montażowej, przez co rośnie liczba potrzebnych stacji (i nie wykorzystany czas ich pracy). Z tego względu może wystąpić konieczność (z uwagi na K) wprowadzenia równoległych stanowisk pracy na stacjach linii.

Przedstawiony algorytm programowania wielostapowego jest szczególnym przypadkiem algorytmu podziału i ograniczeń (bez powrotów). Stosowanie tego algorytmu jest uzasadnione w przypadkach dużej liczby ograniczeń.

Eliminacja stanów nieperspektywicznych może być prowadzona nie tylko poprzez dominację. Każdy stan może być sondowany, czy można z niego uzyskać rozwiązanie dopuszczalne. W tym celu należy sprawdzić, czy K-k stacji wystarcza dla pozostałych operacji. Inną możliwość daje uzyskane wcześniej rozwiązanie heurystyczne (można je wyznaczyć, generując na podstawie (22) jedną trajektorię). Jeżeli wartość wygenerowanego stanu jest większa niż rozwiązanie heurystycznego, to stan taki pomijamy w dalszych obliczeniach.

Jeżeli w obliczeniach występuje przekroczenie dopuszczalnego zakresu pamięci komputerowej, to uzyskane rozwiązanie jest heurystyczne. Celem otrzymania rozwiązania optymalnego można od pewnego etapu η prowadzić obliczenia lokalne, startując od każdego stanu $P^{\lambda,\eta}$. Oszacowanie czasu obliczeń wymaga każdorazowo przeprowadzenia testów komputerowych. Obliczenia takie przeprowadzone w pracy [4] dla modeli o praktycznym znaczeniu wskazują, że czas ten jest rzędu minut (programy napisane w FORTRANIE na EMC MIŃSK-32).

LITERATURA

[1] Шкурба В.В., Белецкий С.А.: Численные методы в решении задачи балансирования сборочной линии. АН УССР, Кибернетика, I, 1977.

[2] Marecki F.: Static Control of Tasks Alloctation on Assembly Line, Intern. Konf. Control Systems and Computer Science, Bukareszt 1981.

[3] Marecki F.: Assembly line balancing problem, Konf. Systems Science VII, Wrocław 1981.

[4] Przybysz T.: Balansowanie szeregowej linii montażowej, praca dyplomowa (nie publikowana) Instytut Automatyki, Politechniki Śląskiej, Gliwice 1981.

Recenzent: Prof. dr hab. inż. Antoni NIEDERLIŃSKI

Wpłynęło do Redakcji 15.05.1982 r.

БАЛЛАНСИРОВАНИЕ ПОСЛЕДОВАТЕЛЬНОЙ МОНТАЖНОЙ ЛИНИИ
С ОГРАНИЧЕНИЯМИ ИСКЛЮЧЕНИЯ ОПЕРАЦИЙ

Р е з ю м е

В работе представлено математическую модель и алгоритм балансирования последовательной монтажной линии с ограничениями исключения операций. Ограничения эти возникают по поводу использования операций по обеим сторонам поточной линии и (или) лимитированного места для контейнеров с деталями на рабочем месте. Задачу балансирования решено методом многошагового программирования.

SERIAL ASSEMBLY LINE BALANCING WITH OPERATION
EXCLUSION CONTRAINTS

S u m m a r y

In the paper we present a mathematical model and an algorithm of balancing of a serial assembly line with operation exclusion contraints. The operation exclusion constraints are the consequence of performing the operations on both sides of assembly line and (or) of area limitation for spare parts containers at stations. The problem is solved with the aid of a multi-stage programming algorithm.