ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1992

Seria: AUTOMATYKA z. 109 Nr kol. 1175

Tadeusz Sawik
Akademia Górniczo-Hutnicza

SZEREGOWANIE CZĘŚCI W ELASTYCZNEJ LINII PRODUKCYJNEJ BEZ MAGAZYNÓW

SCHEDULING OF PARTS IN A FLEXIBLE FLOW LINE WITH NO INTERMEDIATE BUFFERS

ORDONNANCEMENT DES PIECES DANS UNE LIGNE DE FABRICATION FLEXIBLE SANS LES TAMPONS DE STOCKAGE

Streszczenie: Przedstawiono nowy algorytm heurystyczny dla szeregowania części w elastycznej linii produkcyjnej bez magazynów. Linia składa się z szeregowo połączonych stadiów produkcyjnych zawierających zbiory jednakowych maszyn pracujących równolegle. W systemie wytwarza się różne typy części. Każda część jest poddawana obróbce na co najwyżej jednej maszynie w każdym stadium. Zaproponowana heurystyka w każdej iteracji wyznacza kompletny harmonogram dla jednej tylko części, tak aby zminimalizować całkowity czas przestoju maszyn tworzących marszrutę dla tej części.

Summary: A new algorithm is proposed for scheduling a flexible flow line with no intermediate buffers. The line is made up of a certain number of manufacturing stages, where each stage has one or more identical parallel machines. Each part is processed by at most one machine in each stage. The problem objective is to minimize the makespan of the schedule for a set of part types selected for processing. The algorithm proposed is a part-by-part heuristic that attempts to minimize total idle time of machines along the route of each part loaded into the line.

Resume: Nous presentons une nouvelle méthode heuristique, à une seule passe, pour ordonnancement des pièces dans une ligne de fabrication flexible sans les tampons des stockage.

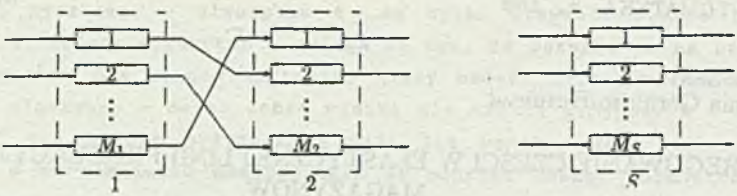
1. Wprowadzenie

W pracy przedstawiono nowy algorytm heurystyczny dla szeregowania części w elastycznej linii produkcyjnej (ELP) bez magazynów. ELP składa się z szeregowo połączonych stadiów produkcyjnych, które zawierają zbiory jednakowych maszyn pracujących równolegle.
W systemie wytwarza się wiele różnych typów części. Każda część jest poddawana obróbce na co najwyżej jednej maszynie w każdym stadium, chociaż pewne typy części mogą omijać niektóre stadia.

ELP jest kombinacją tradycyjnego systemu przepływowego, w którym każde stadium ma tylko jedną maszynę i jednostadialnego systemu jednakowych maszyn pracujących równolegle. Jednak w odróżnieniu od ELP, w tradycyjnym systemie przepływowym każda część musi przejść przez każdą maszynę, zaś pomiędzy maszynami mogą pojawiać się kolejki części o dowolnych długościach.

Wymogi praktyki produkcyjnej powodują, że przeważnie poszukuje się heurystycznych algorytmów harmonogramowania ELP, które prowadzą do dobrych rozwiązań przy stosunkowo niewielkich nakładach obliczeniowych, por. [1,2,5,6,7].

Przedstawiony w dalszym ciągu algorytm jest szczególnym przypadkiem algorytmu *RITM* (ang. Route Idle Time Minimization) szeregowania części w ELP z ograniczonymi buforami międzystadialnymi (zob. [3,4]). Algorytm ten jest heurystyką typu część-za-częścią, która w każdej iteracji wyznacza kompletny harmonogram dla jednej tylko części. Wybór tej części oraz harmonogram dla niej ustalane są w oparciu o uszeregowanie wyznaczone dla części wybranych we wcześniejszych iteracjach. Decyzje podejmowane są na podstawie procedury lokalnej optymalizacji, której celem jest minimalizacja przestojów maszyn tworzących marszrutę dla wybieranej części.



Rysunek 1. Elastyczna linia produkcyjna bez magazynów

Figure 1. Flexible flow line with no intermediate buffers

2. Model matematyczny zadania

Przedstawimy obecnie problem szeregowania części w elastycznej linii produkcyjnej oraz podamy jego model matematyczny.
Oznaczenia podstawowych parametrów i zmiennych występujących w modelu zadania oraz algorytmie szeregowania zamieszczono w Tablicy 1.

Elastyczna linia produkcyjna jest systemem $S \geq 2$ szeregowo połączonych stadiów produkcyjnych i, $(i = 1, \ldots, S)$, z których każde zawiera $M_i \geq 1$ jednakowych maszyn pracujących równolegle (Rys.1). Pomiędzy stadiami nie ma żadnych buforów międzyoperacyjnych, w których można by czasowo składować części oczekujące na kolejne stadia obróbki.

W systemie wytwarza się N różnych typów części. Każda część wymaga wykonania co najwyżej po jednej operacji kolejno w stadiach $1, 2, \ldots, S$ na dowolnej maszynie. Czas wykonywania części typu j $(j = 1, \ldots, N)$ w stadium i wynosi $p_{ij} \geq 0$, zaś czas transportu części ze stadium i do stadium $i+1$ jest równy q_i.

Część wykonana w stadium i $(i = 1, \ldots, S-1)$ przesyłana jest zgodnie ze swoją marszrutą technologiczną do następnego stadium, np. $i+1$, jeżeli w stadium tym jest wolna maszyna. W przeciwnym przypadku wykonana część pozostaje na maszynie w stadium i blokując ją do momentu zwolnienia maszyny w stadium $i+1$. W tym czasie zablokowana maszyna przejmuje rolę bufora i nie może rozpocząć wykonywania żadnej nowej części.

Znane są zapotrzebowania dla wszystkich typów części $(d_1, \ldots, d_N)$, gdzie d_j oznacza wymaganą liczbę części typu j $(j = 1, \ldots, N)$.
Należy wyznaczyć kolejność, w której poszczególne części będą wprowadzane do systemu oraz szczegółowy harmonogram obróbki wszystkich $P = \sum_{j=1}^{N} d_j$ części, tak aby całe zlecenie wykonać w najkrótszym czasie, tzn. aby zminimalizować długość uszeregowania $C_{max} = \max_{1 \leq i \leq S}(c_{iP})$, gdzie c_{iP} oznacza czas zakończenia wykonywania w stadium i ostatniej części P.

Harmonogram produkcji dla wszystkich P części jest wyznaczony przez podanie sekwencji wejściowej $[j_1, j_2, \ldots, j_P]$, w której części będą wprowadzane do systemu oraz wszystkich parametrów czasowych koniecznych do ustalenia szczegółowego harmonogramu wykonywania każdej pojedynczej części. W szczególności, dla każdej części j_k wprowadzanej do systemu w k-tej kolejności $(k = 1, \ldots, P)$ należy wyznaczyć momenty: rozpoczęcia wykonywania s_{ik}, zakończenia wykonywania c_{ik} oraz opuszczenia r_{ik} każdego stadium i, w którym ta część ma być poddawana obróbce. Wielkości te wyznaczane będą z następujących zależności:

$$s_{ik} = \max\{r_{i-1,k} + q_{i-1},\ y_{i,k-1}\},\ i = 2, \ldots, S \tag{1}$$

$$s_{1k} = y_{1,k-1}$$

$$c_{ik} = s_{ik} + p_{ij_k},\ i = 1, \ldots, S \tag{2}$$

$$r_{ik} = \max\{c_{ik},\ y_{i+1,k-1} - q_i\},\ i = 1, \ldots, S-1 \tag{3}$$

$$r_{Sk} = c_{Sk}$$

$$y_{ik} = \min_{1 \leq m \leq M_i}\{Y_{mik}\},\ i = 1, \ldots, S \tag{4}$$

Tablica 1

Oznaczenia

S	=	liczba stadiów produkcyjnych
M_i	=	liczba maszyn w stadium i, $(i = 1,\ldots,S)$
N	=	liczba typów części
d_j	=	zapotrzebowanie na części typu j, $(j = 1,\ldots,N)$
P	=	całkowita liczba części do wyprodukowania, $P = \sum_{j=1}^{N} d_j$
p_{ij}	=	czas wykonywania części typu j w stadium i
q_i	=	czas transportu części ze stadium i do stadium $i+1$
j_k	=	k-ta w kolejności część $(k = 1,\ldots,P)$ w sekwencji wejściowej $[j_1, j_2, \ldots, j_P]$
s_{ik}	=	moment rozpoczęcia wykonywania części j_k w stadium i, $(i = 1,\ldots,S)$
c_{ik}	=	moment zakończenia wykonywania części j_k w stadium i, $(i = 1,\ldots,S)$
r_{ik}	=	moment opuszczenia stadium i, $(i = 1,\ldots,S)$ przez część j_k
t_{ik}	=	czas przestoju maszyny w stadium i, związany z wykonywaniem części j_k (czas oczekiwania na rozpoczęcie wykonywania części j_k oraz czas blokowania maszyny przez wykonaną część j_k)
Y_{mik}	=	moment zwolnienia maszyny m, $(m = 1,\ldots,M_i)$ w stadium i po uszeregowaniu k pierwszych części
y_{ik}	=	najwcześniejszy moment zwolnienia maszyny w stadium i po uszeregowaniu k pierwszych części, $y_{ik} = \min_{1\le m\le M_i}\{Y_{mik}\}$
$m(i,k)$	=	maszyna w stadium i z najwcześniejszym momentem zwolnienia po uszeregowaniu $k-1$ pierwszych części, $m(i,k) = \arg\min_{1\le m\le M_i}\{Y_{mik-1}\}$
w_i	=	średnie obciążenie maszyny w stadium i, $(i = 1,\ldots,S)$, $w_i = \sum_{j=1}^{N} p_{ij} d_j / M_i$
w^*	=	średnie obciążenie maszyny w stadium i^* będącym wąskim gardłem, $w^* = w_{i^*} = \max_{1\le i\le S} w_i$

gdzie najwcześniejsze momenty zwolnienia maszyn Y_{mik} wyznacza się iteracyjnie według następującej zależności

$$Y_{mik} = \begin{cases} Y_{mik-1} & \text{jeżeli } m \neq m(i,k) \\ r_{ik} & \text{jeżeli } m = m(i,k) \end{cases} \tag{5}$$

Wyrażenie (5) wyprowadzono przy założeniu, że w każdym stadium i część j_k jest przydzielana do maszyny $m(i,k)$, z najwcześniejszym momentem zwolnienia.

Na koniec zauważmy, że czas przestoju t_{ik} maszyny w stadium i związany z wykonywaniem części j_k obejmuje następujące dwa składniki:
(i) $(s_{ik} - y_{i,k-1})$ - czas oczekiwania maszyny na rozpoczęcie wykonywania części j_k oraz
(ii) $(r_{ik} - c_{ik})$ - czas blokowania maszyny przez wykonaną część j_k.
Zatem, t_{ik} można wyrazić następująco:

$$t_{ik} = (s_{ik} - y_{i,k-1}) + (r_{ik} - c_{ik}) = r_{ik} - y_{i,k-1} - p_{ij_k},\ i = 1,\ldots,S,\ k = 1,\ldots,P \tag{6}$$

Przedstawiona w następnym punkcie heurystyka *RITM* daży do minimalizacji czasu C_{max} wykonywania wszystkich części poprzez minimializację całkowitego czasu przestoju maszyn $\sum_{i=1}^{S}\sum_{k=1}^{P} t_{ik}$.

3. Algorytm szeregowania części

Przedstawiony poniżej algorytm harmonogramowania jest jedno-przejściową heurystyką, w której kolejność wprowadzania każdej części do systemu wyznaczana jest tylko raz. W każdej iteracji k spośród wszystkich części oczekujących na obróbkę jest wybierana jedna do załadunku do systemu. Wybór ten dokonywany jest na podstawie szczegółowych harmonogramów obróbki każdego z oczekujących typów części. Wyznaczamy je uwzględniając ustalone już uszeregowanie $k-1$ części wcześniej wprowadzonych do systemu i wybierając najkorzystniejszą marszrutę przepływu przez system

$$[m(1,k), m(2,k), \ldots, m(S,k)] \tag{7}$$

Marszrutę taką tworzą maszyny $m(i,k)$ z kolejnych stadiów, które będą najwcześniej zwalniane po uszeregowaniu $k-1$ pierwszych części. Ostatecznie wybierana jest część takiego typu j, dla której łączny czas przestoju maszyn we wszystkich stadiach

$$t_j = \sum_{i=1}^{S} t_{ij} \tag{8}$$

będzie minimalny.

W algorytmie harmonogramowania typy części j, $(j = 1, \ldots, N)$ rozpatrywane są w kolejności nie rosnących wartości całkowitego czasu wykonywania p_j

$$p_j = \sum_{i=1}^{S} p_{ij} \tag{9}$$

W ten sposób, podobnie jak w heurystykach typu *LPTF* (ang. Longest Processing Time First), części o dłuższych całkowitych czasach wykonywania uzyskują dodatkowy priorytet na wejście do systemu w pierwszej kolejności.

Algorytm harmonogramowania przedstawiono poniżej (J oznacza zbiór takich typów części j, które wykonano w zadanych ilościach d_j).

Algorytm *RITM*

Krok 0. Początkowy

1. Uporządkuj typy części w kolejności nie rosnących wartości całkowitych czasów wykonywania p_j, (9), tzn. $p_1 \geq p_2 \geq \cdots \geq p_N$
2. Podstaw:
   $Y_{m0} = 0$; $m = 1, \ldots, M$, $i = 1, \ldots, S$; $m(i,1) = 1$, $i = 1, \ldots, S$,
   $J = \emptyset$; $k = 1$

Krok 1. Wybór kolejnej części do sekwencji wejściowej

1. Dla każdego typu części $j \notin J$ oczekujących na wejście do systemu wyznacz momenty rozpoczęcia s_{ij}, (1), i zakończenia c_{ij}, (2), obróbki w kolejnych stadiach na maszynach $m(i,k)$ z najwcześniejszymi czasami dostępności oraz momenty opuszczania r_{ij}, (3), stadiów i czasy przestoju maszyn t_{ij}, (6). Następnie wyznacz całkowity czas przestoju maszyn t_j, (8).
2. Do systemu wprowadź część takiego typu j_k, dla którego całkowity czas przestoju będzie najmniejszy, tzn.
$$j_k = \arg\min_{j \notin J}(t_j)$$

Krok 2. Wyznaczenie harmonogramu obróbki wybranej części

Dla wybranej części j_k wyznacz szczegółowy harmonogram obróbki (momenty s_{ik}, c_{ik}, r_{ik}) przydzielając ją w kolejnych stadiach do maszyn $m(i,k)$, które zostały najwcześniej zwolnione po uszeregowaniu $(k-1)$ pierwszych części.
Harmonogram obróbki części j_k dołącz do uszeregowania dla $k-1$ pierwszych części.

Krok 3. Sprawdzenie stanu wykonania zlecenia

Dla $j = j_k$ podstaw $d_j = d_j - 1$. Jeżeli $d_j = 0$, to podstaw $J = J \cup \{j\}$.
Jeżeli $J = \{1, \ldots, N\}$, to zakończ obliczenia.
Inaczej, dla każdej maszyny m, $(m = 1, \ldots, M_i,\ i = 1, \ldots, S)$ wyznacz najwcześniejszy moment dostępności Y_{mik}, (5) po uszeregowaniu k pierwszych części. Następnie dla każdego stadium i wyznacz maszynę $m(i, k+1) = \arg\min_{1 \le m \le M_i}(Y_{mik})$ z najwcześniejszym momentem dostępności y_{ik}, (4).
Podstaw $k = k + 1$ i wróć do Kroku 1. □

Złożoność obliczeniowa algorytmu *RITM* jest $O(P^2S)$, co wynika z następującego postępowania. W każdej iteracji jest wyznaczany harmonogram obróbki dla jednej spośród P części. Najpierw wyznaczamy jedną najkorzystniejszą marszrutę obejmującą ciąg najwcześniej zwalnianych maszyn w kolejnych stadiach. Wymaga to przeglądnięcia co najwyżej $m = \sum_{i=1}^{S} M_i$ maszyn. Następnie dla każdego z pozostałych do wykonania co najwyżej N typów części wyznaczamy harmonogram obróbki według tej marszruty. Wymaga to wyznaczenia wartości co najwyżej $O(NS)$ parametrów czasowych, gdyż wybrana marszruta obejmuje co najwyżej S maszyn. Procedurę tę powtarzamy w każdej iteracji, których liczba jest równa P, a ponieważ $N \le P$ ostatecznie otrzymujemy powyższą ocenę złożoności obliczeniowej algorytmu.

4. Przykłady liczbowe

Zastosowanie algorytmu zilustrujemy następującym przykładem liczbowym. Elastyczna linia produkcyjna zbudowana jest z $S = 3$ stadiów. Stadium 1 zawiera $M_1 = 2$ maszyny, stadium 2 $M_2 = 3$ maszyny oraz stadium 3 $M_3 = 2$ maszyny. Czasy transportu pomiędzy stadiami są równe $q_1 = 1$ i $q_2 = 1$.

Zadane zlecenie produkcyjne obejmuje $N = 4$ typy części, które należy wyprodukować w ilościach odpowiednio $d_1 = 8$, $d_2 = 4$, $d_3 = 2$, $d_4 = 3$ sztuk. Zatem całkowita liczba części do wykonania wynosi $P = 17$.
Czasy p_{ij} $(i = 1, 2, 3;\ j = 1, 2, 3, 4)$ wykonywania poszczególnych typów części w kolejnych stadiach są następujące:

$$p_{11} = 5,\ p_{21} = 3,\ p_{31} = 7,$$
$$p_{12} = 2,\ p_{22} = 4,\ p_{32} = 6,$$
$$p_{13} = 3,\ p_{23} = 6,\ p_{33} = 1,$$
$$p_{14} = 1,\ p_{24} = 4,\ p_{34} = 2.$$

Analiza powyższych danych wskazuje, że wąskim gardłem jest stadium $i^* = 3$, a średnie obciążenie każdej maszyny w tym stadium wynosi $w^* = 44$.

Harmonogram produkcji dla powyższego przykładu wyznaczono stosując algorytm *RITM*. Otrzymano harmonogram o długości $C_{max} = 52$ dla następującej sekwencji wejściowej części [4,4,2,1,3,2,3,1,1,1,1,2,1,1,1,2,4]. Minimalne czasy przestoju maszyn t_{j_k}, (8), związane z kolejnymi częściami j_k, $(k = 1, \ldots, 17)$ wprowadzanymi do systemu są następujące: 9,9,4,3,2,3,1,1,3,5,7,7,7,9,9,11,12.

Harmonogram produkcji przedstawiono w Tablicy 2. Podano w niej numery typów części przydzielonych w kolejnych okresach (o jednostkowej długości) do poszczególnych maszyn we wszystkich stadiach. Dla każdego przydziału części do maszyn w pierwszej kolumnie Tablicy 2 podano numer początkowego i końcowego okresu zastosowania tego przydziału.

Dla oceny efektywności algorytmu *RITM* i dokładności otrzymywanych wyników przeprowadzono serie eksperymentów obliczeniowych. Za pomocą algorytmu *RITM* wyznaczono harmonogramy dla 10 serii zadań testowych po 100 zadań w każdej serii, łącznie dla 1000 przykładów liczbowych.
Podstawowe dane wejściowe dla tych przykładów zamieszczono w Tablicy 3. Obejmują one następujące wielkości: liczba typów części N, liczba stadiów S oraz liczby maszyn M_i w kolejnych stadiach $i = 1, \ldots, S$. Ponadto przyjęto jednostkowe czasy transportu pomiędzy stadiami $q_i = 1$ $(i = 1, \ldots, S-1)$.
Pozostałe wielkości były generowane losowo: czasy wykonywania operacji p_{ij} $(i = 1, \ldots, S,\ j = 1, \ldots, N)$ z przedziału [0,200], zapotrzebowania na części d_j $(j = 1, \ldots, N)$ z przedziału [1,30].

Odległość od optimum dla otrzymywanych harmonogramów oceniono za pomocą wartości względnych błędów $\varepsilon = (C_{max} - w^*)/w^*$. Dla każdej serii 100 przykładów testowych wyznaczono średni błąd względny

Tablica 2

Harmonogram produkcji

Numery okresów od-do	Przydział części do maszyn						
	Stadium 1		Stadium 2			Stadium 3	
	Maszyny		Maszyny			Maszyny	
	1	2	1	2	3	1	2
1	4	4					
2	2	1					
3	2	1	4	4			
4	3	1	4	4			
5-6	3	1	4	4	2		
7	2	3			2		
8	2	3	1	3	2	4	4
9	1	3	1	3		4	4
10	1	1	1	3	2	2	
11	1	1	3	3	2	2	
12-14	1	1	3	3	2	2	1
15	1	1	3	1	2	2	1
16	1	1	3	1	1	3	1
17	1	1	3	1	1	2	1
18	1	1		1	1	2	1
19	2	1			1	2	3
20	2	1	1		1	2	1
21	1	1	1	1	1	2	1
22	1	1	1	1	2	2	1
23-27	1	1	1	1	2	2	1
28-31	2	1	1	1	2	1	1
32	2	4	1	1	2	1	1
33	2	4	1	1	1	1	1
34	2	4	1	1	1	1	2
35-37		4	2	1	1	1	2
38-39			2	1	1	1	2
40-42			2	4	1	1	1
43-45			2	4		1	1
46				4		1	1
47-49				4		1	2
50						1	2
51-52						4	2

$\bar{\epsilon}$ oraz odchylenie standardowe σ. Szybkość algorytmu oceniono poprzez średni czas obliczeń $\overline{CPU}$ w sekundach dla jednego zadania z każdej serii 100 zadań. Wyniki zamieszczono w trzech ostatnich kolumnach Tablicy 3. Potwierdzają one korzystne własności algorytmu *RITM*.
Obliczenia przeprowadzono na mikrokomputerze IBM PC-AT z procesorem o częstotliwości pracy 12 MHz i koprocesorem arytmetycznym, stosując program komputerowy napisany w języku Turbo Pascal 5.5.

Tablica 3

Dane wejściowe i wyniki dla przykładów testowych

Numer serii	N	P	S	Liczby maszyn M_i					$\bar{\epsilon}$ [%]	σ	$\overline{CPU}$ [s]
				M_1	M_2	M_3	M_4	M_5			
1	5	10	3	3	3	3	-	-	22.36	13.58	0.04
2	5	20	3	3	2	3	-	-	10.39	8.32	0.06
3	8	30	3	2	3	3	-	-	7.48	9.66	0.12
4	8	40	3	2	2	2	-	-	7.03	6.00	0.13
5	10	50	4	3	3	3	3	-	12.08	7.85	0.26
6	10	60	4	3	3	2	3	-	6.51	6.01	0.32
7	12	70	4	2	3	2	3	-	7.04	6.96	0.44
8	12	80	5	2	2	2	2	2	10.04	6.72	0.54
9	15	90	5	3	3	3	3	3	14.21	8.27	0.79
10	15	100	5	4	4	4	4	4	19.16	7.53	0.88

Oznaczenia: N - liczba typów części, P - całkowita liczba części, S - liczba stadiów, $\bar{\epsilon}$ - średnia wartość błędu względnego $\epsilon = (C_{max} - w^*)/w^*$ dla serii 100 zadań, σ - odchylenie standardowe błędu względnego, $\overline{CPU}$ - średni czas obliczeń

5. Podsumowanie

Przedstawiony algorytm szeregowania części w elastycznej linii produkcyjnej bez magazynów wyznacza dobre rozwiązania przy bardzo małych nakładach obliczeniowych. Algorytm konstruuje kompletny harmonogram obróbki dla każdej nowej części wybranej do załadunku do systemu, na podstawie uszeregowania wyznaczonego dla wcześniej wybranych części. Własności te sprawiają, że przedstawiona metoda harmonogramowania może być również zastosowana do bieżącego szeregowania części w trybie on-line.

Literatura

[1] Brah S.A., Hunsucker J.L.: Branch and bound algorithm for the flow shop with multiple processors. *European Journal of Operational Research*. Vol.51, 1991, s. 88-99.

[2] Sawik T.: Multilevel scheduling of multistage production with limited in-process inventory. *Journal of the Operational Research Society*. Vol.38, 1987, s. 651-664.

[3] Sawik T.: *Optymalizacja dyskretna w elastycznych systemach produkcyjnych*. WNT Warszawa, 1992.

[4] Sawik T.: A scheduling algorithm for flexible flow lines with limited intermediate buffers. *International Journal of Systems Automation: Research and Applications*, special issue on Flexible Manufacturing Systems. 1992, w druku.

[5] Sriskandarajah C. and Sethi S.P.: Scheduling algorithms for flexible flowshops: Worst and average case performance. *European Journal of Operational Research*. Vol.43, 1989, s. 143-160.

[6] Wittrock R.J.: Scheduling algorithms for flexible flow lines. *IBM Journal of Research and Development*. Vol.29, 1985, s. 401-412.

[7] Wittrock R.J.: An adaptable scheduling algorithm for flexible flow lines. *Operations Research*. Vol.36, 1988, s. 445-453.

Recenzent: Prof.dr inż. Henryk Kowalowski
Wpłynęło do Redakcji do 30.04.1992 r.

Abstract:

This paper presents a new heuristic algorithm for the scheduling of parts through a flexible manufacturing system, called the "Flexible Flow Line". A flexible flow line consists of several processing stages in series, where each stage has one or more identical parallel machines. The line produces several different part types. Each part must be processed by at most one machine in each stage, but some parts may skip some stages. Intermediate queues of parts waiting between the stages for their next operations in the system are not allowed.

The scheduling algorithm proposed is a single pass part-by-part heuristic in which the loading sequence and the corresponding complete schedule are determined once. During every iteration a part for loading into the system is chosen as well as its complete processing schedule is determined. The decisions in every iteration are made using a local optimization procedure aimed at minimizing total idle time along the route of the selected part . For this reason, the new heuristic is called "route idle time minimization" (*RITM*).

Given a cumulative partial schedule for the parts selected so far, first the best route along the line is found as a sequence of S machines (S is the number of all production stages in the system) with the earliest available times. For each part type waiting for entering the line a complete processing schedule is determined along the best route. To evaluate the processing schedule for each part type considered for loading, total duration of machine idle time along the route is calculated. Finally, the part type for which a schedule with the smallest total idle time is obtained is selected for loading and its complete processing schedule is added to the cumulative partial schedule obtained so far.

The algorithm requires P iterations, where P denotes total number of parts to be scheduled through the line. In every iteration a complete production schedule is determined for each of at most N part types waiting for loading into the system, which requires $O(NS)$ computations. Since $N \leq P$ the computational complexity of the algorithm *RITM* is $O(P^2S)$.

The efficiency of the algorithm was tested on several groups of random problems. The results obtained have indicated that the heuristic yields good solutions in very short CPU run time. Therefore, the approach presented can also be implemented for on-line use in a dynamic scheduling environment.