ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1996
Seria: AUTOMATYKA z.118 Nr kol. 1338

Józef GRABOWSKI, Jarosław PEMPERA
Politechnika Wrocławska

ALGORYTMY SZEREGOWANIA ZADAŃ W PEWNYM DYSKRETNYM PROCESIE PRODUKCYJNYM

Streszczenie. Praca zawiera sformułowania oraz algorytmy rozwiązania pewnego zagadnienia kolejnościowego z ograniczeniami typu "bez czekania" i "bez magazynowania" występującego w pewnym procesie produkcji wyrobów z betonu komórkowego.

AN ALGORITHM FOR SCHEDULING IN PRODUCTION OF SOME CONCRETE WERES

Summary. This paper deals with a tabu search algorithm for the flow-shop problem with the "no wait" and "no store" requirements in process. This problem arises in production of some concrete weres.

1. Wstęp

W nowoczesnych zakładach produkcyjnych nastawionych na dużą rentowność swojej działalności duży nacisk kładzie się na maksymalizację wydajności procesu technologicznego. Cel ten można osiągnąć poprzez optymalne lub suboptymalne szeregowanie zadań w procesie. W niniejszej pracy przedstawiony jest pewien problem związany z taśmowym procesem produkcyjnym, w którym istotne znaczenie mają ograniczenia typu "bez czekania" (no wait) oraz "bez magazynowania" (no store).

2. Opis procesu technologicznego

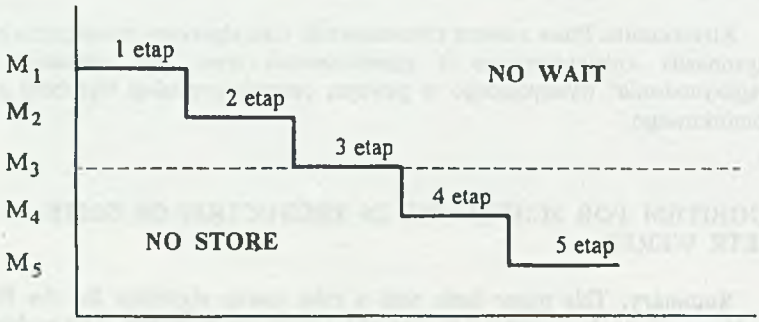
Głównym produktem Zakładów Produkcji Betonów są wyroby z betonu komórkowego.

Wyroby produkowane są w kilku rozmiarach oraz w kilku gatunkach betonu. Rodzaj betonu wpływa na parametry eksploatacyjne wyrobów, tzn. wytrzymałość, izolację termiczną, lekkość konstrukcji budowlanej oraz cenę wyrobu. Rozmiary wyrobów uzależnione są od przeznaczenia, tzn. wyroby o dużej szerokości przeznaczone są do budowy ścian nośnych. Cienkie bloczki (płytki) przeznaczone są do budowy ścian działowych bądź komór

ocieplających konstrukcję. Technologia wykonania wyrobów oraz ich parametry zapewniają dobrą izolację termiczną oraz wymianę powietrza poprzez ściany.

Proces technologiczny produkcji wyrobów składa się z pięciu podstawowych etapów:

1. Mieszanie składników i wylewanie masy do formy.
2. Dojrzewanie (twardnienie) masy betonowej.
3. Krojenie masy betonowej na bloki.
4. Autoklawizacja wyrobów.
5. Transport wyrobów do magazynu.



Rys. 1.

W pierwszym etapie produkcyjnym następuje wymieszanie wszystkich składników betonu, tj. kruszywa (piasek, żwir), cementu, wapnia oraz substancji chemicznych opartych na aluminium, które powodują pulchnienie masy. Proces mieszania składników ma określony czas dla danego gatunku betonu i nie należy go przekraczać, ponieważ po tym czasie zaczynają się procesy wiążące beton. Dalsze mieszanie może spowodować zniszczenie powstałych wiązań i pogorszenie własności betonu. Ponadto następuje wyrastanie betonu w mieszarce, które może spowodować uszkodzenie mieszarki. Z tego względu przejście z etapu 1 do 2 traktujemy jako "no wait". Po wylaniu masy betonowej do formy następuje drugi etap technologiczny tj. twardnienie masy betonowej. Proces ten jest przyspieszony przez umieszczenie masy w odpowiednich piecach. Czas dojrzewania masy betonowej jest określony dla danego gatunku betonu. Czasu dojrzewania nie wolno ani skracać ani wydłużać, ponieważ w pierwszym przypadku masa może być zbyt miękka nie nadając się do krojenia, zaś w drugim przypadku masa może stać się zbyt twarda i uniemożliwić krojenie. Z tego względu przejście od etapu 2 do 3 przyjmujemy jako przejście typu "no wait".

Trzecim etapem jest krojenie masy betonowej na bloczki. Czas krojenia uzależniony jest od rozkroju. Czwartym etapem jest proces autoklawizacji (tj. proces przyspieszonego twardnienia wyrobów). W ostatnim etapie produkcyjnym następuje transport wyrobów do magazynu. Między etapami 3 i 4 oraz 4 i 5 nie istnieje możliwość magazynowania wyrobów. Stąd też przejście między etapami możemy przyjąć jako "no store".

3. Model matematyczny zagadnienia

Załóżmy, że zadanie produkcyjne jest dane w postaci zbioru zadań

$J = \{J_1, J_2, \ldots, J_n\}$ – reprezentuje porcje betonu o określonym rozkroju i gatunku,

które mają być wykonane przy użyciu zbioru maszyn

$M = \{M_1, M_2, M_3, M_4, M_5\}$ – reprezentują urządzenia realizujące etapy produkcyjne

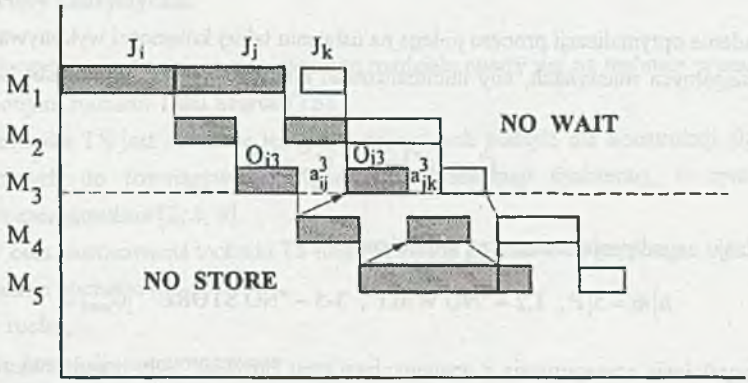
Każde zadanie $J_j \in J$ składa się z operacji

$$J_j = \langle O_{j1}, O_{j2}, O_{j3}, O_{j4}, O_{j5} \rangle .$$

Operacje te mają być wykonane w zadanym porządku technologicznym, tzn. każda operacja O_{jk} ma być wykonana po zakończeniu O_{jk-2}, a przed wykonaniem operacji O_{jk+1}.

Wszystkie zadania mają być wykonane kolejno na (urządzeniach) maszynach M_1, M_2, M_3, M_4, M_5, co oznacza, że operacja O_{jk} ma być wykonana na maszynie M_k.

3. Pewne własności zagadnienia



Rys. 2.

Na rysunku 2 przedstawiono wykres Gantta dla dwóch przykładowych zadań. Proces technologiczny można podzielić na dwie części typu "NO WAIT" i typu "NO STORE".

Własności części typu "NO WAIT"

Niech $\pi \in \Pi$ będzie dowolną permutacją określającą kolejność wykonywania w części "no wait" zadań oraz niech zadania J_i oraz J_j będą kolejnymi zadaniami w π. Dalej niech wartość a_{ij}^l będzie odległością między operacjami O_{il} oraz O_{jl} na M_l-tej maszynie.

Odległość a_{ij}^l jest stała i równa

$$a_{ij}^l = \max\left(0, a_{ij}^{l-1} + p_{i,l-1} - p_{j,l}\right), \quad l \geq 2, \qquad (l = 3 - \text{dla części typu "NO WAIT"}),$$

$$a_{ij}^1 \triangleq 0$$

dla wszystkich permutacji β, w których zadania J_i oraz J_j są wykonywane obok siebie w takiej samej kolejności jak w π.

Wprowadzimy jeszcze następujące oznaczenia:

S_{jk} – termin rozpoczęcia wykonywania zadania J_j na maszynie M_k,
C_{jk} – termin zakończenia wykonywania zadania J_j na maszynie M_k,
p_{jk} – czas trwania wykonywania zadania J_j na maszynie M_k.

Na proces wykonania zadań nałożone są następujące wymagania:

1) $S_{jk} = S_{jk+1} - p_{jk}$ dla $k = 1, 2$, wymaganie "NO WAIT",
2) $S_{j+1,k} \geq S_{j,k+1}$ dla $k = 3, 4$, wymaganie "NO STORE".

Zadanie optymalizacji procesu polega na ustaleniu takiej kolejności wykonywania zadań na poszczególnych maszynach, aby minimalizować maksymalny czas wykonania wszystkich zadań

$$C_{\max}\left(\pi^*\right) = \min_{\pi} C_{\max}(\pi).$$

Tego rodzaju zagadnienie oznaczamy symbolem

$$n|m = 5|P,\ 1,2 - \text{"NO WAIT"},\ 3\text{-}5 - \text{"NO STORE"}\ |C_{\max}.$$

Zagadnienia szeregowania z wymaganiami typu "no wait" były przedstawione w wielu pracach, np. [1] [3] [8]. W szczególności dla $m = 2$, w pracy [1] zaprezentowano algorytm wielomianowy.

Własności części typu "NO STORE"

W zagadnieniach z wymaganiem: "NO STORE" definiuje się pojęcie antybloku [9].

Antyblokiem nazywamy fragment drogi krytycznej zaczynający i kończący się łukami reprezentującymi operację, natomiast pozostałe łuki reprezentują wymaganie "NO STORE" i łączą węzły reprezentujące początki operacji (rys. 3).

Własności:

1) jeśli β jest dowolną permutacją otrzymaną z π poprzez zamianę zadań wewnątrz dowolnego bloku lub antybloku w π, to $C_{\max}(\beta) \geq C_{\max}(\pi)$,
2) jeśli β jest dowolną permutacją otrzymaną z π przez zamianę miejscami dwóch dowolnych zadań pomiędzy dwoma antyblokami A_i i A_j w π, to $C_{\max}(\beta) \geq C_{\max}(\pi)$,
3) jeżeli β jest dowolną permutacją otrzymaną z π poprzez zamianę miejscami dwóch dowolnych zadań pomiędzy dwoma blokami B_i i B_j, takimi że $\mu(B_i) = \mu(B_j)$, to

$$C_{\max}(\beta) \geq C_{\max}(\pi),$$

$\mu(B)$ – określa maszynę, na której wykonywane są operacje z bloku B.

Zagadnienia szeregowania typu "no store" były przedmiotem badań w wielu pracach [3, 9, 10]. W szczególności w pracach [9, 10] przedstawiono możliwości wykorzystania podejścia blokowego stosowanego w klasycznych zagadnieniach szeregowania [5, 6].

4. Algorytmy heurystyczne

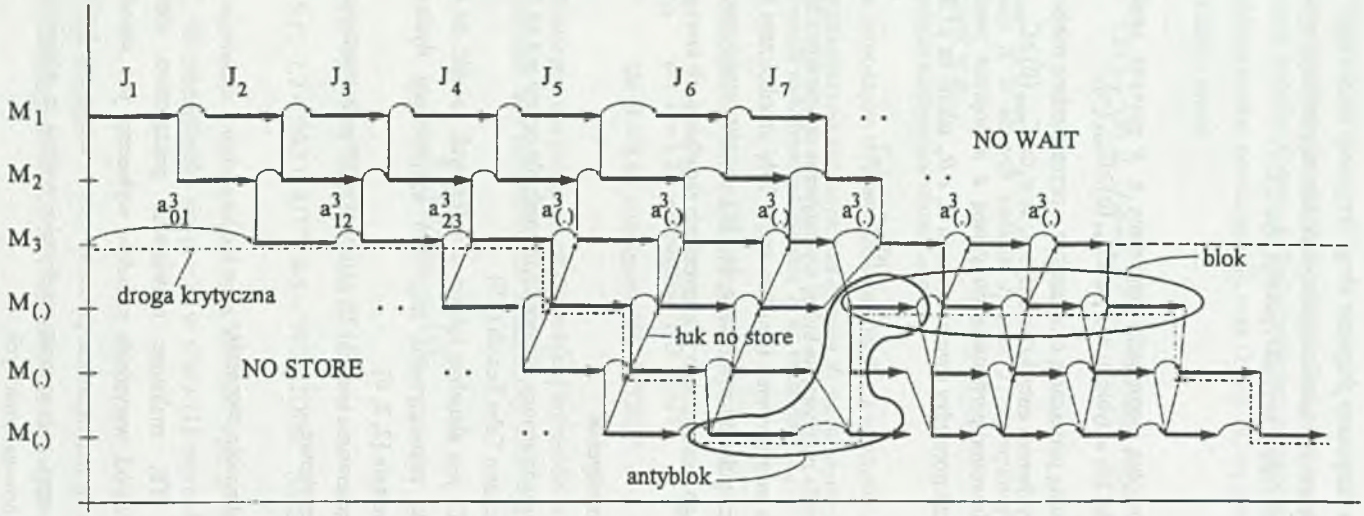
Algorytm przedstawiony w niniejszym rozdziale oparty jest na technice przeszukiwania z zabronionymi ruchami Tabu Search (TS).

Technika TS jest aktualnie jednym z najlepszych podejść dla konstrukcji algorytmów heurystycznych do rozwiązywania zagadnień optymalizacji dyskretnej, w tym również zagadnień szeregowania [2, 3, 6].

W celu zastosowania techniki TS należy określić podstawowe pojęcia:
– rozwiązanie początkowe,
– pojęcie ruchu,
– strategię przeszukiwania otoczenia,
– funkcję aspiracji.

Algorytm TS, analizując rozwiązanie początkowe, określa zbiór rozwiązań "sąsiedzkich". Spośród wszystkich sąsiadów wybierany jest sąsiad "najlepszy". Wybrane rozwiązanie staje się rozwiązaniem początkowym w następnym kroku. W celu zapobieżenia powstaniu cyklu oraz zapewnienia możliwości wyjścia z minimum lokalnego, tworzy się cykliczną listę ruchów zabronionych.

Droga krytyczna dla pewnego rozwiązania



Rys. 3.

4.1. Ruchy i sąsiedzi

Niech $v=(a,b)$ będzie (ruchem) parą pozycji $a,b\in\{1,2,\ldots,n\}$ w permutacji π uszeregowanych zadań. Permutację β otrzymujemy z π przez przesunięcie zadania $\pi(a)$ z pozycji "a" na pozycję "b", tzn.

$$\pi_v=(\pi(1),\ldots,\pi(a-1),\pi(a+1),\ldots,\pi(b),\pi(a),\ldots,\pi(n)),\quad a<b$$

lub

$$\pi_v=(\pi(1),\ldots,\pi(b-1),\pi(a),\pi(b),\ldots,\pi(a-1),\pi(a+1),\ldots,\pi(n)),\quad a>b.$$

Zdefiniowany w ten sposób zbiór możliwych przesunięć (ruchów), ze względu na własności bloków i antybloków ([5, 9]) można zredukować do przesunięcia zadania przed lub za blok oraz antyblok dla fragmentu drogi krytycznej zawierającej ten blok oraz antyblok (rys. 3). Natomiast we fragmencie drogi krytycznej związanym z wymaganiami "no wait" przesunięcia będą realizowane poprzez zamianę sąsiednich zadań.

Niech $T=(T_1,T_2,\ldots,T_w)$ będzie listą "reprezentującą" ruchy zabronione, w której dowolny element $T_j=(\pi(g),\pi(h))$ będzie parą indeksów zadań $J_{\pi(g)}$ oraz $J_{\pi(h)}$ na pozycji g oraz h permutacji π. Ruch $v=(a,b)$ jest zabroniony (nie może być wykonany) w permutacji π, jeżeli istnieje w liście T element T_i taki, że

$$T_i=(\pi(a),\pi(j)),\quad j=b,\ldots,a-1,\quad a<b,\quad i=1,2,\ldots,w$$

lub

$$T_i=(\pi(j),\pi(a)),\quad j=a+1,\ldots,b,\quad a>b,\quad i=1,2,\ldots,w.$$

Początkowa postać listy T zawiera wartości zerowe, tzn. $T_j=(0,0)$, $j=1,2,\ldots,w$. W trakcie przebiegu algorytmu "dołączenie" ruchu $v=(a,b)$ do listy T polega na przesunięciu wszystkich ruchów z listy o jedną pozycję w lewo (element pierwszy listy jest usuwany), a na miejscu ostatnim jest podstawiony element

$$T_w:=(\pi(a),\pi(a+1))\ \text{dla}\ a<b\ ,\ \text{lub}\ T_w:=(\pi(a-1),\pi(a))\ \text{dla}\ a>b.$$

4.2. Strategia przeszukiwań i funkcja aspiracji

Wyznaczenie wartości funkcji celu $C_{\max}(\pi)$ dla uszeregowania π stanowi stosunkowo czasochłonną operację algorytmu. Stąd też wybór $v=(a,b)$ ruchu (przesunięcia zadań) w π będzie dokonywany zgodnie z pewną wielkością $\Delta(v)$ szacującą maksymalne zmniejszenie wartości $C_{\max}(\pi)$, zgodnie z zależnością:

$$C_{\max}(\pi_v)\geq C_{\max}(\pi)+\Delta(v),$$

gdzie $C_{\max}(\pi_v)$ będzie wartością funkcji celu nowej permutacji otrzymanej z π po wykonaniu ruchu v. Z zależności tej wynika, że w pierwszej kolejności należy wybrać przesunięcie v o minimalnej wartości $\Delta(v)$, co pozwoli na odrzucenie podczas obliczeń wszystkich ruchów v, dla których $C^*_\pi < C_{\max}(\pi) + \Delta(v)$, gdzie C^*_π jest najmniejszą wartością funkcji celu dla wygenerowanych permutacji β z π. Wielkości $\Delta(v)$ zostały zdefiniowane w pracach [4, 9].

4.3. Algorytm

Krok 0. $\pi = \pi^*$, $C^* := C_{\max}(\pi)$, iter := 0, ret := 0, $T := \{\}$

Krok 1. iter := iter + 1; ret := ret +1.

Spośród wszystkich ruchów niezabronionych oraz korzystnych ruchów zabronionych v, takich że $C_{\max}(\pi_v) < C^*$, wyznacz zbiór $Z = \left\{ v \middle| C_{\max}(\pi_v) = \min_v C_{\max}(\pi_v) \right\}$.

Podczas wyznaczania zbioru Z skorzystaj z własności szacujących $\Delta(v)$.

Jeżeli $Z = \emptyset$, to dodaj do T ruch (0,0), idź do kroku 1.

Krok 2. Jeżeli $C_{\max}(\pi_v) < C^*$ to $T^* := T$, $Z^* := Z - v$, $C^* := C_{\max}(\pi_v)$, $\pi^* := \pi_v$, ret := 0

Podaj do listy T ruch v

$$\pi := \pi_v$$

Krok 3. Jeżeli iter $\geq$ max iter, to Stop.

Jeżeli ret < max ret, to idź do kroku 1.

Krok 4. Jeżeli $Z^* = \emptyset$, to Stop.

Ze zbioru Z^* usuń pierwszy element (ruch) v_1, czyli $Z^* := Z^* - v_1$; oraz podstaw $v := v_1$.

$T := T^*$ ret := 0

Idź do kroku 2.

LITERATURA

1. Gilmore P.C., Gomory R.E.: Sequencing a One State variable Machine: A Solvable Case of the Salesman problem. OR, 12 , 1964, pp. 655-679.
2. Glover F.: Tabu Search. Part I. ORSA Journal of Computing, 1, 1989, pp. 190-206.
3. Glover F.: Tabu Search. Part II. ORSA Journal of Computing, 2, 1990, pp. 4-32.
4. Grabowski J.: Algorytm rozwiązania zagadnienia komiwojażera przy użyciu grafów dysjunktywnych. Matematyka Stosowana, XII, 1978, s. 81-98.
5. Grabowski J., Skubalska E., Smutnicki C.: On flow shop with release and due dates to minimalize maximum lateness. Journal of Operation Research Society, 34, 1983, pp. 615-620.

6. Grabowski J., Nowicki E., Zdrzałka S.: A block approach for single-machine scheduling with release dates and due dates. European Journal of Operational Research, 26, 1986, pp. 278-285.

7. Nowicki E. and Smutnicki C.: A fast tabu search algorithm for the permutation flow shop problem. European J. Oper. Res., 1966, (w druku).

8. Rajedran C.: A no wait flow shop scheduling heuristic to minimalize makespan. Journal Operational Reserch Society, vol. 45, No. 4, 1994, pp. 472-478.

9. Smutnicki C.: Pewne własności zagadnień kolejnościowych z ograniczeniami składowania. Zeszyty Naukowe Akademii Górniczo-Hutniczej, Automatyka, z. 4, Kraków 1983, s. 223-232.

10. Smutnicki C.: Metoda blokowa w zagadnieniach kolejnościowych taśmowych z ograniczeniami składowania. Zeszyty Naukowe Politechniki Śląskiej, seria Automatyka, z. 84, Gliwice 1986, s. 223-232.

Recenzent: Prof.dr hab.inż. Ryszard Gessing

Wpłynęło do Redakcji do 30.06.1996 r.

Abstract

This deals with the permutation flow-shop problem which arises in production of some concrete weres. The problem can be formulated as follows. There is the set of jobs $J_1,\ldots,J_n$, each of n jobs has to be processed on 5 machines $M_1,\ldots,M_5$ in that order. Between the pairs of machines (M_1,M_2), (M_2,M_3) and (M_3,M_4), (M_4,M_5) there are "no wait" and "no store" requirements, (respectively). The objective is sequencing jobs such that project duration be minimized. Some properties are presented. The tabu search algorithm of solving this problem is given.