Harmonogramowanie robót budowlanych z zastosowaniem algorytmu Tabu Search z rozmytymi czasami wykonania zadań

Dr inż. Magdalena Rogalska, Politechnika Lubelska, dr inż. Wojciech Bożejko, dr hab. inż. Zdzisław Hejducki, Politechnika Wrocławska, dr Mieczysław Wodecki, Uniwersytet Wrocławski

1. Wprowadzenie

Praca ta stanowi kontynuację studiów autorów [1, 2, 3, 4, 21] nad zagadnieniami związanymi z uwzględnieniem niepewności w harmonogramowaniu robót budowlanych. Jednym ze sposobów reprezentowania niepewności jest zastosowanie elementów teorii zbiorów rozmytych, umożliwiających oszacowanie czasów wykonania prac oraz cyklu realizacji kompleksu robót. Do rozwiązania zagadnienia szeregowania prac, w ramach eksperymentu obliczeniowego, zastosowano metaheurystykę przeszukiwania z zabronieniami (ang. tabu search, w skrócie TS) z modyfikacją uwzględniającą niepewne czasy robót jako liczby rozmyte w trzypunktowej reprezentacji. Zagadnienie obliczeniowe dotyczy problemu optymalizacji kombinatorycznej, o dużej złożoności obliczeniowej, należące do klasy problemów *NP-trudnych*.

Realizacja obiektów budowlanych związana jest często z pojawiającymi się w toku robót przerwami technologicznymi, organizacyjnymi i innymi. Czynniki zewnętrzne i zakłócenia wewnętrzne występujące w procesie budowy, przewidywalne i nieprzewidywalne, są przyczyną dezaktualizacji harmonogramów i występowania odchyleń od ustalonych w umowach terminów realizacji zakontraktowanych robót. Dlatego poszukuje się sposobów określania danych, niezbędnych w procesie decyzyjnym, do wyznaczania realnych terminów wykonania zadań inwestycyjnych. Konsekwencją błędów są w praktyce wysokie kary umowne (straty finansowe przedsiębiorstw) i osłabienie wiarygodności firm budowlanych, a nawet wykluczenie z procesów przetargowych. W artykule przedstawiono problem harmonogramowania robót budowlanych z rozmytymi czasami wykonania prac z zastosowaniem algorytmu metaheurystycznego, opartego na metodzie przeszukiwania z zabronieniami, do rozwiązania zagadnienia szeregowania zadań.

Podjęto próbę oszacowania średniego błędu względnego rozwiązań wyznaczonych przez algorytm dla zaburzonych danych, w stosunku do najlepszych rozwiązań, jako miary ryzyka przedsięwzięcia budowlanego. Może on na przykład służyć jako parametr umożliwiający oszacowanie wielkości ryzyka i kontyngencji czasu realizacji przedsięwzięcia budowlanego, wielkości buforów czasu (wg. metodyki CCS/BM), jako wspomaganie procesów decyzyjnych. Przed przystąpieniem do robót bardzo trudno jest określić jednoznacznie czasy wykonywania poszczególnych prac i wielkość zakłóceń mogących pojawić się w trakcie wykonania zadania inwestycyjnego. Zjawiska te można uwzględnić np. stosując liczby rozmyte. Pierwsze prace dotyczące harmonogramowania z rozmytymi danymi dotyczyły metody PERT [17]. Autorzy prac [5, 12, 13, 17] jako jedni z pierwszych uwzględnili rozmyte parametry w klasycznych modelach szeregowania zadań jedno- i wielomaszynowych. Dla rozwiązywania ogólnego problemu sieciowego z ograniczeniami zasobowymi, w pracach [9, 10] zaprezentowano heurystyki uwzględniające rozmyte parametry czasowe. Należy również zaznaczyć szeroko stosowane podejście do określania niepewnych parametrów za pomocą rozkładów zmiennych losowych [16, 19], co wymaga zebrania danych statystycznych.

2. Podstawowe założenia

Prezentowane zagadnienie sprowadza się do klasycznego problemu szeregowania zadań – permutacyjnego problemu przepływowego [7, 8, 15] z kryterium C_{max} – odpowiednika typowego problemu harmonogramowania robót budowlanych. W literaturze jest on oznaczany przez $F||C_{max}$. W pracy przedstawiono ideę algorytmu opartego na metodzie przeszukiwania z zabronieniami z rozmytymi czasami wykonywania robót, w trzypunktowej reprezentacji [16, 17].

Do modelowania robót budowlanych przyjęto potokową metodę organizacji robót zapewniającą ciągłość wykonywania prac. Warunek ten spełnia metoda sprzężeń czasowych MSC I [11, 18]. Zaleca się ją sto-

sować w przypadku, kiedy podstawowym założeniem jest zapewnienie ciągłości wykonywania procesów (P_1,P_2, ... ,P_n) na działkach roboczych. Zwykle wiąże się to z zapewnieniem ciągłości technologicznej lub organizacyjnej, np. przy realizacji robót ciągłych – betonowaniu konstrukcji. Zdarza się, że niektóre firmy wykonawcze, szczególnie wysoko wyspecjalizowane, stawiają jako warunek kontraktu zapewnienie ciągłości ich prac. Dane do obliczeń przyjmujemy według opracowanej struktury podziału prac (WBS) wyznaczane w sposób normatywny lub oszacowane jako czasy wykonania robót ($T_{1,1}$,$T_{1,2}$, ... ,$T_{n,m}$) na działkach roboczych (S_1,$S_{2, \dots}$,S_n). Jako efekt obliczeń otrzymamy całkowity czas wykonania przedsięwzięcia *TT*. Zakłada się warunek ciągłości polegający na zastosowaniu zerowych sprzężeń czasowych eliminujących przerwy czasowe ($LT_{1,1}$,$LT_{1,2}$, ... ,$LT_{n,m}$) pomiędzy procesami w toku a kolejnymi – czyli zagwarantowana zostanie ciągłość procesów (P_1,P_2, ... ,P_n).
Na rysunku 1 przedstawiono powiązania priorytetowe, dla których przerwy czasowe *LT* są równe zeru.

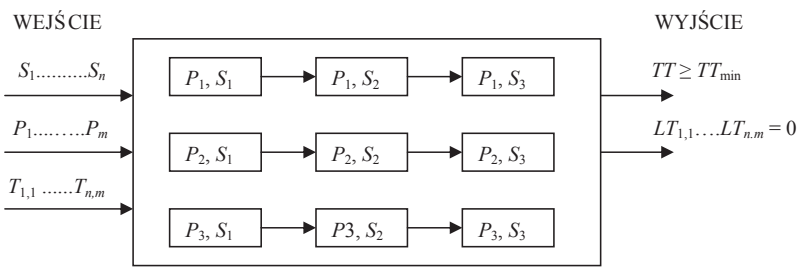
Zagadnienie harmonogramowania rozpatrywanego przedsięwzięcia budowlanego można sprowadzić do standardowego problemu szeregowania zadań – permutacyjnego problemu przepływowego (*flow shop problem* [8]), w którym zadania należy kolejno wykonać na maszynach z pewnego zbioru. Adekwatne pojęcia z teorii szeregowania zadań przyjęto w metodyce harmonogramowania przedsięwzięć budowlanych z wykorzystaniem algorytmów metaheurystycznych [20].

3. Permutacyjny problem przepływowy

Korzystając z oznaczeń przyjętych w teorii szeregowania zadań, optymalizacja harmonogramów budowlanych wymaga znalezienia właściwych modeli obliczeniowych uwzględniających przyjęte ograniczenia. Są one podstawą opracowanych algorytmów i programów komputerowych.
W problemie przepływowym [8, 15], dany jest zbiór *n* zadań $J=\{1,2, \dots ,n\}$, które należy wykonać kolejno, bez przerywania, na każdej z *m* maszyn ze zbioru $M=\{1,2, \dots ,m\}$. Wykonywanie zadania na maszynie *k* (dla $k=2, \dots ,m$) może się rozpocząć dopiero po zakończeniu wykonywania tego zadania na maszynie *k*-1 oraz kolejność wykonywania zadań na wszystkich maszynach musi być taka sama. Każda maszyna w dowolnej chwili może, co najwyżej, wykonywać jedno zadanie. Niech $p_{i,k}$ będzie czasem wykonywania zadania *i* na maszynie *k*.
Należy wyznaczyć kolejność wykonywania zadań (taką samą dla każdej maszyny), która minimalizuje
$C_{max}=\max\{C_{i,m}: i=1,2, \dots ,n\}$,
gdzie $C_{i,m}$ jest czasem zakończenia wykonywania zadania *i* na ostatniej, *m*-tej maszynie.
Jeżeli liczba maszyn wynosi trzy lub więcej, wówczas problem ten jest silnie *NP-trudny* [6]. W ostatnich latach opublikowano wiele algorytmów metaheurystycznych rozwiązywania problemu przepływowego. Najlepsze z nich są oparte na metodzie poszukiwania z zabronieniami [8, 15]. Wyznaczane przez nie rozwiązania, dla reprezentatywnej grupy przykładów, tylko nieznacznie różnią się od najlepszych znanych obecnie w literaturze.
Łatwo zauważyć, że rozpatrywany problem przepływowy sprowadza się do wyznaczenia permutacji optymalnej (o minimalnym terminie zakończenia wykonywania zadań) w zbiorze wszystkich permutacji zadań $J=\{1,2, \dots ,n\}$.

4. Metoda przeszukiwania z tabu

Zamieszczone w literaturze wyniki porównawcze wskazują, że do wyznaczania bardzo dobrych rozwiązań przybliżonych, dla większości klasycznych *NP-trudnych* problemów szeregowania zadań (np. *flow shop*, *job shop*), najlepsze są algorytmy oparte na metodzie poszukiwań z zabronieniami. Generalnie, metoda ta polega na iteracyjnym polepszaniu bieżącego rozwiązania poprzez lokalne przeszukiwanie. Rozpoczyna się od pewnego rozwiązania początkowego (startowego) x^0. W każdej iteracji, dla bieżącego rozwiązania x^i, wyznacza się jego otoczenie $N(x^i)$ – podzbiór zbioru rozwiązań dopuszczalnych. Otoczenie jest generowane przez ruchy, tj. ustalone przekształcenia rozwiązania x^i. Następnie, z otoczenia jest wyznaczany najlepszy element x^{i+1}, który przyjmuje się za bieżące rozwiązanie w następnej iteracji. Dopuszcza się przy tym możliwość zwiększania wartości funkcji celu (przy wyznaczaniu nowego bieżącego rozwiązania), aby w ten sposób zwiększyć szansę na osiągnięcie minimum globalnego. Takie ruchy „w górę" należy jednak w pewien sposób kontrolować, ponieważ w przeciwnym razie, po osiągnięciu minimum lokalnego, nastąpiłby szybki do niego powrót. Aby zapobiec generowaniu, w nowych iteracjach rozwiązań niedawno rozpatrywanych (powstawaniu cykli), zapamiętuje się je (ich atrybuty) na liście rozwiązań zakazanych, tzw. liście tabu.



Rys. 1. *Powiązania priorytetowe w metodzie sprzężeń czasowych MSC I, [11, 18]*

Niech π będzie dowolną permutacją startową, *TL* listą tabu, a π^* najlepszym do tej pory znalezionym rozwiązaniem (za rozwiązanie startowe oraz π^* można przyjąć dowolną permutację, np. wyznaczoną przez algorytm konstrukcyjny NEH [14]).

Schemat algorytmu przeszukiwania z tabu (*TSF*)
repeat
Wyznaczyć otoczenie $N(\pi)$ permutacji π;
Usunąć z $N(\pi)$ permutacje zakazane przez listę ***TL***;
Znaleźć permutację $\delta \in N(\pi)$ taką, że:

$F(\delta) = min\ \{F(\beta) : \beta \in N(\pi)\}$;
if $F(\delta) < F(\pi)$ **then** $\pi^* \leftarrow \delta$;
Umieść atrybuty δ na liście ***TL***;
$\pi \leftarrow \delta$;
until ***Warunek_Zakończenia***.

Warunkiem zakończenia jest zazwyczaj ustalona liczba iteracji lub czas obliczeń.
Zasadnicze znaczenie w implementacji metody TS ma sposób generowania i przeglądania otoczenia oraz organizacja listy ruchów zakazanych.

Ruch i otoczenie
W każdej iteracji algorytmu TS jest wyznaczane otoczenie – podzbiór zbioru rozwiązań dopuszczalnych. Jest on generowany przez ruchy – przekształcenia określone na zbiorze wszystkich permutacji. W wielu dobrych algorytmach rozwiązywania problemów szeregowania zdań, otoczenie jest generowane przez ruchy typu wstaw (ang. *insert*).
Niech *k* i *l* ($k \neq l$) będzie parą pozycji w permutacji:

$\pi = (\pi(1), \pi(2), ..., \pi(k-1), \pi(k), \pi(k+1), ..., \pi(l-1), \pi(l), \pi(l+1), ... \pi(n))$.

Ruch typu wstaw (*i-ruch*) i_l^k, polega na przestawieniu zadania $\pi(k)$ z pozycji *k* na pozycję *l*. Generuje on nową permutację $i_l^k(\pi) = \pi_l^k$. Jeżeli $k < l$, to nowa permutacja

$\pi_l^k = (\pi(1), \pi(2), ..., \pi(k-1), \pi(k+1), ..., \pi(l-1), \pi(l), \boxed{\pi(k)}, \pi(l+1), ... \pi(n))$.

Podobnie jest, gdy $k > l$. Przez $R(\pi)$ oznaczamy zbiór wszystkich takich ruchów.
Otoczeniem permutacji π, generowanym przez ruchy $R(\pi)$, jest zbiór:
$N(\pi) = \{i_l^k(\pi) : i_l^k \in R(\pi)\}$.
Liczba elementów tego otoczenie jest równa $n(n-1)$.

Lista ruchów zakazanych
Aby zapobiec powstawaniu cykli (powrotowi do tej samej permutacji, po niewielkiej liczbie iteracji algorytmu), pewne atrybuty każdego ruchu zapamiętuje się na liście ruchów zakazanych. Obsługiwana jest ona na zasadzie kolejki *FIFO*. Wykonując ruch $i_j^r \in R(\pi)$ (tj. generując z π permutację π_j^r) na listę zapisujemy atrybuty tego ruchu, trójkę: $(\pi(r), j, F(\pi_j^r))$. Załóżmy, że rozpatrujemy ruch $i_l^k \in R(\beta)$ generujący permutację β_l^k. Jeżeli na liście jest trójka (r, j, Ψ) taka, że $\beta(k) = r$, $l = j$ oraz $F(\beta_l^k) \geq \Psi$ to ruch taki eliminujemy (usuwamy) ze zbioru $R(\beta)$.
Złożoność obliczeniowa algorytmu opartego na metodzie przeszukiwania z tabu zależy od sposobu realizacji jego elementów, tj. metody wyznaczania otoczenia, organizacji oraz długości listy, sposobu wyliczania wartości funkcji celu oraz warunku zakończenia.

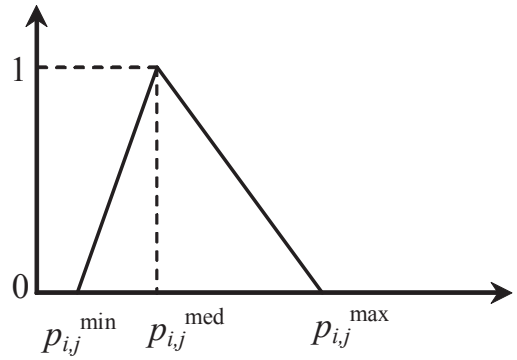
4. Algorytm przeszukiwania z tabu (TSF) z rozmytymi czasami wykonywania zadań

Niepewne czasy wykonywania zadań na maszynach będą reprezentowane przez trzypunktowe liczby rozmyte

$$\tilde{p}_{i,j} = (p_{i,j}^{\min}, p_{i,j}^{\text{med}}, p_{i,j}^{\max})$$

gdzie $p_{i,j}^{\min} \leq p_{i,j}^{\text{med}} \leq p_{i,j}^{\max}$ dla $i \in J, j \in M$.

Funkcja przynależności takiej liczby jest przedstawiona na rysunku 2.



Rys. 2. *Funkcja przynależności trzypunktowej liczby rozmytej* $\tilde{p}_{i,j} = (p_{i,j}^{min}, p_{i,j}^{med}, p_{i,j}^{max})$

Wynikiem działań arytmetycznych na liczbach rozmytych jest liczba rozmyta [3]. Jeżeli więc czas wykonywania zadania $\pi(i)$ na maszynie *j* jest określony przez liczbę rozmytą

$$\tilde{p}_{\pi(i),j} = (p_{\pi(i),j}^{\min}, p_{\pi(i),j}^{\text{med}}, p_{\pi(i),j}^{\max}),$$

to wówczas termin jego zakończenia jest liczbą rozmytą postaci:

$$\tilde{C}_{\pi(i),j} = \left(C_{\pi(i),j}^{\min}, C_{\pi(i),j}^{\text{med}}, C_{\pi(i),j}^{\max}\right),$$

gdzie $C_{\pi(i),j}^{\min}, C_{\pi(i),j}^{\text{med}}$ oraz $C_{\pi(i),j}^{\max}$ można wyznaczyć z następujących zależności rekurencyjnych:

$$C_{\pi(i),j}^{\min} = \max\{C_{\pi(i-1),j}^{\min}, C_{\pi(i),j-1}^{\min}\} + p_{\pi(i),j}^{\min},$$

$$C_{\pi(i),j}^{\text{med}} = \max\{C_{\pi(i-1),j}^{\text{med}}, C_{\pi(i),j-1}^{\text{med}}\} + p_{\pi(i),j}^{\text{med}},$$

$$C_{\pi(i),j}^{\max} = \max\{C_{\pi(i-1),j}^{\max}, C_{\pi(i),j-1}^{\max}\} + p_{\pi(i),j}^{\max},$$

z warunkami początkowymi

$$C^{\min}_{\pi(0),j} = C^{\text{med}}_{\pi(0),j} = C^{\max}_{\pi(0),j} \quad j = 1,2,\ldots,m$$
$$\text{oraz} \quad C^{\min}_{\pi(i),0} = C^{\text{med}}_{\pi(i),0} = C^{\max}_{\pi(i),o} \quad i = 1,2,\ldots,n.$$

Czas wykonywania wszystkich zadań (w kolejności π) jest także liczbą rozmytą

$$\tilde{C}_{\max}(\pi) = (C^{\min}_{\pi(n),m}, C^{\text{med}}_{\pi(n),m}, C^{\max}_{\pi(n),m}).$$

Dla porównywania rozmytych wartości funkcji celu została zastosowana słaba reguła porównania [10]. W przypadku trzypunktowej liczby rozmytej $\tilde{C}_{\max}(\pi)$ funkcja defuzyfikacyjna jest wówczas postaci:

$$I(\tilde{C}_{\max}(\pi)) = \frac{1}{4}\left(C^{\min}_{\pi(n),m} + C^{\text{med}}_{\pi(n),m} + C^{\text{med}}_{\pi(n),m} + C^{\max}_{\pi(n),m}\right).$$

Uwzględniając założenia dotyczące czasów wykonywania zadań (liczby rozmyte w trzypunktowej reprezentacji), dokonano odpowiednich modyfikacji, przedstawionego w poprzednim rozdziale, algorytmie ***TSF***. Tę wersję algorytmu będziemy oznaczali przez ***TSF^{Roz}***. Przeprowadzono eksperymenty obliczeniowe dotyczące porównania działania algorytmu deterministycznego ***TSF*** i z danymi rozmytymi ***TSF^{Roz}***.

Oba algorytmy były testowane na losowo generowanych przykładach. Dane zwane dalej *deterministycznymi* generowano w następujący sposób. Dla ustalonej liczby zadań *n* oraz maszyn *m* losowano z przedziału [30, 100] (zgodnie z rozkładem jednostajnym) czasy wykonywania zadań na maszynach $p_{i,j}$, i=1,2, ..., n, j= 1, 2, ..., m. Dla każdego n i m losowano w ten sposób 10 przykładów.

Niepewne czasy wykonywania zadań $\tilde{p}_{i,j}$ =1,2, ..., n, j= 1, 2, ..., m są reprezentowane przez trzypunktową liczbę rozmytą

$$(p^{\min}_{i,j}, p^{\text{med}}_{i,j}, p^{\max}_{i,j}),$$

gdzie

$$p^{\text{med}}_{i,j} = p_{i,j}, \quad p^{\min}_{i,j} = \left\lceil p_{i,j} - p_{i,j}/6 \right\rceil$$
$$\text{oraz} \quad p^{\max}_{i,j} = \left\lceil p_{i,j} + p_{i,j}/3 \right\rceil.$$

Jeżeli $\tilde{p}_{i,j} = (p^{\min}_{i,j}, p^{\text{med}}_{i,j}, p^{\max}_{i,j})$ (i=1,2,...,n, j=1,2,...,m) jest rozmytym czasem wykonywania zadania, to czas zaburzony generowano losowo zgodnie z rozkładem jednostajnym na przedziale $[p^{\min}_{i,j}, p^{\max}_{i,j}]$.

Dla każdego przykładu rozwiązanego przez algorytm $A \in \{TSF, TSF^{Roz}\}$ wyznaczono procentowy błąd względny (względne odchylenie)

$$\delta = \frac{F^A - F^*}{F^*} \cdot 100\%,$$

gdzie F^A jest wartością rozwiązania wyznaczonego przez algorytm A, a F^* wartością wyznaczoną przez najlepszy obecnie algorytm zamieszczony w pracy [8]. Średnie wartości tych błędów δ_{aprd} oraz δ^{Roz}_{aprd} (z 10 przykładów danych zaburzonych, dla każdego rozmiaru) zamieszczono w tabeli 1.

Tabela 1. *Średnie błędy względne algorytmów (w %)*

Liczba zadań n	Algorytm detrministyczny TSF (średni błąd względny δ_{aprd})	Algorytm rozmyty TSF^{Roz} (średni błąd względny δ^{Roz}_{aprd})
25	6,6	3,1
50	9,1	2,9
100	13,9	5,4
150	23,2	8,7
200	21,8	9,3
250	25,6	10,5
500	28,3	12,7
średnio	**18,4**	**7,5**

Na podstawie zamieszczonych wyników możemy stwierdzić, że rozwiązania (tj. permutacje, czyli kolejności wykonywania zadań) wyznaczone przez algorytm rozmyty ***TSF^{Roz}*** są znacznie bardziej stabilne. Niepewność szacunków czasów wykonywania zadań, dla rozwiązania uzyskanego przez algorytm rozmyty, ma znacznie mniejszy wpływ na ostateczny czas wykonania wszystkich zadań niż w przypadku rozwiązania wyznaczonego przez algorytm deterministyczny. Średnie błędy względne wynoszą odpowiednio 18,4% oraz 7,5%. Zatem, można przypuszczać, że gdy kolejność zadań jest wyznaczona przez algorytm z rozmytymi parametrami, znacznie mniej zmienia się wartość funkcji celu przy zaburzeniu czasów wykonywania zadań.

6. Studium przypadku

Na podstawie wstępnego eksperymentu obliczeniowego z użyciem algorytmu tabu z rozmytymi czasami wykonania zadań, przeprowadzono obliczenia optymalizacyjne dla danych z praktyki budowlanej.

Zadanie inwestycyjne dotyczy realizacji kompleksu dwunastu budynków mieszkalnych, tworzących fragment nowego osiedla mieszkaniowego. Są one zbliżone pod względem technologicznym. Podstawą szacowania czasów wykonania robót są przedmiary robót oraz np. KNR (Katalog Nakładów Rzeczowych), na bazie których można określić pracochłonność robót. Po ustaleniu i uzgodnieniu niezbędnych zasobów ludzkich (grup roboczych), oblicza się możliwy czas trwania robót, przekształcając nakłady rzeczowe z liczby tzw. roboczogodzin [r-h], w dni robocze [dni-rob]. Tak ustalone dane tworzą tzw. Strukturę Podziału Prac (SPP), określając porządek technologiczny robót. Dane liczbowe do obliczeń optymalizacyjnych dotyczą kompleksu 12 obiektów budowlanych, na których planuje się realizację 9 procesów budowlanych. Przedsięwzięcie reprezentuje macierz czasów trwania procesów budowlanych T_{9x12}, gdzie $T_{i,j}$ jest czasem wykonania roboty i na obiekcie j. Poniżej przedstawiono przykładową macierz czasów trwania robót na obiektach budowlanych.

$$T = \begin{vmatrix} 7 & 8 & 7 & 7 & 7 & 8 & 7 & 7 & 6 & 7 & 5 & 4 \\ 8 & 11 & 8 & 9 & 9 & 11 & 8 & 9 & 8 & 9 & 8 & 8 \\ 8 & 11 & 10 & 9 & 9 & 11 & 10 & 9 & 11 & 9 & 9 & 9 \\ 7 & 8 & 7 & 7 & 8 & 8 & 7 & 7 & 8 & 8 & 8 & 7 \\ 6 & 7 & 7 & 7 & 7 & 7 & 7 & 7 & 7 & 7 & 8 & 15 \\ 11 & 14 & 11 & 13 & 13 & 14 & 11 & 13 & 14 & 13 & 14 & 8 \\ 9 & 14 & 9 & 11 & 10 & 13 & 9 & 11 & 8 & 10 & 11 & 9 \\ 4 & 8 & 6 & 7 & 5 & 7 & 7 & 8 & 9 & 9 & 9 & 5 \\ 6 & 9 & 5 & 9 & 7 & 5 & 8 & 9 & 8 & 7 & 7 & 7 \end{vmatrix}$$

Elementy macierzy czasów trwania robót opracowano na podstawie przedmiarów, z uwzględnieniem dostępnych zasobów, dla kompleksu budynków mieszkalnych. Właściwe oszacowanie cyklu realizacji inwestycji jest niezbędne do sformułowania warunków kontraktu na roboty budowlane dla Inwestora, Wykonawców oraz Podwykonawców.

Do wyznaczenia cyklu realizacji przedsięwzięcia budowlanego (*TT*), zastosowano metodę sprzężeń czasowych MSC I [11, 18]. W wyniku przeprowadzonych obliczeń wyznaczono cykl realizacji wynoszący *TT*=250 jednostek, z uwzględnieniem warunku ciągłości procesów. Po przeprowadzeniu obliczeń z zastosowaniem algorytmu *TSF* uzyskano *TT*=233 jednostki. Do obliczeń optymalizacyjnych przyjęto „rozmyte" czasy wykonania robót. Dla danych zaburzonych, średni błąd względny wynosi zaledwie 4,2%. Może on służyć na przykład jako parametr umożliwiający oszacowanie wielkości ryzyka i kontyngencji czasu realizacji przedsięwzięcia budowlanego, wielkości buforów czasu (np. wg. metodyki CCS/BM), jako wspomaganie procesów decyzyjnych.

7. Podsumowanie

W artykule przedstawiliśmy wyniki eksperymentów obliczeniowych dotyczących harmonogramowania prac budowlanych z uwzględnieniem warunku ciągłości robót (permutacyjny problem przepływowy). W praktyce bardzo trudno jest określić, przed przystąpieniem do robót, rzeczywiste czasy wykonywania poszczególnych robót. Modelujemy je więc za pomocą liczb rozmytych. Do rozwiązywania zadania optymalizacyjnego zastosowano algorytm oparty na metodzie przeszukiwania z zabronieniami z rozmytymi danymi. Wyznaczone przez ten algorytm rozwiązania są stabilniejsze (mniej wrażliwe na zmiany parametrów rozpatrywanego zagadnienia) od rozwiązań algorytmu z parametrami określonymi przez liczby całkowite. Na podstawie przeprowadzonej analizy w p. 4, stwierdzono, że zmiana czasów wykonywania zadań, dla rozwiązania wyznaczonego przez algorytm rozmyty, ma znacznie mniejszy wpływ na ostateczny czas wykonania wszystkich zadań niż w przypadku rozwiązania wyznaczonego przez algorytm deterministyczny. Średnie błędy względne wynoszą odpowiednio 18,4% oraz 7,5%. Po zaburzeniu czasów wykonywania robót budowlanych, znacznie mniej zmienia się wartość funkcji celu, gdy kolejność ich wykonywania jest wyznaczona przez algorytm z rozmytymi parametrami zadań. Uzyskane dane liczbowe mogą zatem wspomagać oszacowanie wielkości buforów czasu (np. wg metodyki CCS/BM). Niniejsza praca stanowi kontynuację studiów autorów [1, 2, 3, 4, 11, 21] nad zastosowaniem algorytmów metaheurystycznych w harmonogramowaniu robót budowlanych (ze szczególnym uwzględnieniem liczb rozmytych w problemach optymalizacji dyskretnej).

BIBLIOGRAFIA

[1] Bożejko W., Wodecki M., Algorytmy tabu search z rozmytymi danymi, Komputerowo Zintegrowane Zarządzanie, WNT, Warszawa, 2001, 95–103

[2] Bożejko W., Wodecki M., Modelowanie niepewności w pewnym problemie szeregowania zadań, Zeszyty Naukowe Politechniki Śląskiej, AUTOMATYKA, t. 5, z. 1/2, 2001, 103–112

[3] Bożejko W., Wodecki M., Stabilność metaheurystyk dla wybranych problemów szeregowania zadań, Komputerowo Zintegrowane Zarządzanie, WNT, Warszawa, 2001, 85–94

[4] Bożejko W., Hejducki Z., Wodecki M., Harmonogramowanie kompleksu robót budowlanych w warunkach niepewności. Komputerowo Zintegrowane Zarządzanie, PTZP Opole 2008, TI ,109–118

[5] Dubois D., Prade H., Operations on fuzzy numbers, International Journal System Science, vol. 9, 1978, 613–626

[6] Garey M. R., Johnson D. S., Seti R., The complexity of flowshop and jobshop scheduling, Mathematics of Operations Research, 1, 1976, 117–129

[7] Grabowski J., Pempera J., New block properties for the permutation flow-shop problem with application in TS, Journal of Operational Research Society, 52, 2001, 210–220

[8] Grabowski J., Wodecki M., A new local search algorithm for the flow shop problem, Computers & Operations Research, 31, 2004, 1891–1909

[9] Hapke M., Programowanie sieciowe w warunkach niepewności, Zeszyty Naukowe Politechniki Śląskiej, AUTOMATYKA, z. 117, 1996, 47–56

[10] Hapke M., Jaszkiewicz A., Słowiński R., Rozmyte podejście do wielokryterialnego programowania w warunkach niepewności, Zeszyty Naukowe Politechniki Śląskiej, AUTOMATYKA, z. 123, 1998, 155–167

[11] Hejducki Z., Rogalska M., Metody sprzężeń czasowych TCM, Przegląd Budowlany, 2, 2005, 38–45

[12] Iscibuchi H., Yamamoto N., Misaki S. and Tanaka H., Local Search Algorithms for Flow Shop Scheduling with Fuzzy Due-Dates, International Journal of Production Economics, 33, 1994, 53–66

[13] Ishii H.: Fuzzy combinatorial optimization, Japanese Journal of Fuzzy Theory and Systems, Vol. 4, no. 1, 1992

[14] Navaz M., Enscore E.E., Ham I., A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem, OMEGA, 11/1, 1983, 91–95

[15] Nowicki E., Smutnicki C., A fast tabu search algorithm for the permutation flow-shop problem, European Journal of Operational Research, 91, 1996, 160–175

[16] Pinedo M., Scheduling: Theory, Algorithms, and Systems, Prentice Hall, 1995

[17] Prade H., Using fuzzy set theory in a scheduling problem, Fuzzy Sets and Systems, 2, 1979, 153–165

[18] Mrozowicz J., Metody organizacji procesów budowlanych uwzględniające sprzężenia czasowe DWE, Wrocław 1997

[19] Righter R., „Stochastic Scheduling", in Stochastic Orders, M. Shaked and Shandhkumar (eds.), Academic Press, San Diego, 1994

[20] Podolski M., Analiza nowych zastosowań teorii szeregowania zadań w organizacji robot budowlanych. Praca doktorska. Wrocław 2008

[21] Rogalska M., Bożejko W., Hejducki Z., Time/cost optimization using hybrid evolutionary algorithm in construction project scheduling, Automation in Construction, Elsevier, 2008, V18/1, 24–31