SYSTEM WSPOMAGANIA HARMONOGRAMOWANIA PRZEDSIĘWZIĘĆ BUDOWLANYCH

Wojciech BOŻEJKO, Zdzisław HEJDUCKI, Mariusz UCHROŃSKI, Mieczysław WODECKI

Streszczenie: W pracy przedstawiamy system wspomagający harmonogramowanie przedsięwzięć, szczególnie z zakresu robót budowlanych. Aplikacja obejmuje moduły: harmonogramowania prac budowlanych, harmonogramowania transportu belek przy montażu konstrukcji mostowych oraz moduł harmonogramowania w warunkach niepewności. W modułach optymalizacyjnych zastosowano algorytm poszukiwania z zabronieniami. Przy modelowaniu niepewności danych wykorzystano elementy teorii zbiorów rozmytych.

Słowa kluczowe: szeregowanie zadań, zarządzanie przedsięwzięciami.

1. Budowa nowych modeli - transport w systemie JIT

Jednym z etapów przedsięwzięcia budowlanego polegającego na budowie mostu lub wiaduktu z zastosowaniem elementów nośnych wykonanych z kompozytów jest transport dźwigarów z miejsca prefabrykacji na plac budowy [1, 2]. Na podstawie przygotowanego wcześniej harmonogramu prowadzenia prac budowlanych zostają ustalone terminy oraz kolejność dostawy dźwigarów (zgodnie z porządkiem technologicznym). Zakładamy przy tym, że prace montażowe są wykonywane w systemie JIT. Nie ma więc możliwości składowania dźwigarów na placu budowy (ewentualnie jedynie na pojeździe). Wobec tego dźwigary są bezpośrednio z pojazdów umieszczane na podporach. Problem optymalnego transportu dźwigarów (w skrócie problem **TD**) polega na ustaleniu takich terminów dostawy dźwigarów, aby zminimalizować przyjęte kryterium (np. kosztów niedotrzymania terminów dostawy, liczby niezbędnych pojazdów, itp.). Muszą być przy tym spełnione następujące ograniczenia:

a) dźwigary należy dostarczać zgodnie z kolejnością montażu (porządkiem technologicznym),
b) jednocześnie pojazd może przewozić tylko jeden dźwigar (lub partię dźwigarów),
c) po załadunku, dźwigar może być zdjęty jedynie bezpośrednio przed montażem.

W dalszej części rozpatrujemy różne warianty realizacji problemu transportu dźwigarów. Przedstawimy odpowiadające im modele optymalizacyjne oraz algorytmy ich rozwiązywania.

1.1. Problem transportu z najwcześniejszymi i najpóźniejszymi terminami dostaw

Przyjmujemy, że na podstawie czasów wykonywania prac oraz wymogów technologii został wyznaczony szczegółowy harmonogram (diagram Gantta) prowadzenia robót budowlano-montażowych. Zawiera on między innymi terminy rozpoczęcia oraz zakończenia montażu dźwigarów. Na tej podstawie są wyznaczane najwcześniejsze oraz najpóźniejsze możliwe terminy dostawy poszczególnych dźwigarów (uwzględniające także

porządek technologiczny). Zaleca się, aby najpóźniejszy termin dostawy uwzględniał czas rozładunku dzięki czemu, rozpoczęcie montażu dźwigara może nastąpić zgodnie z przyjętym harmonogramem. Zaleca się ponadto, wprowadzenie niewielkiego buforu czasowego, uwzględniając w ten sposób mogące się ewentualnie pojawić niewielkie komplikacje. Najwcześniejsze terminy dostawy wynikają z przyjętego założenia, że dźwigary nie mogą być magazynowane na placu budowy. W związku z tym, przybyły wcześniej pojazd nie może być przed tym terminem rozładowany. Najwcześniejszy i najpóźniejszy termin dostawy tworzą tzw. okno czasowe. Jest to przedział czasowy, w którym dostawa nie generuje dodatkowych kosztów. Poniżej przedstawiamy model matematyczny omawianego zagadnienia.

Niech $\mathcal{B} = \{B_1, B_2, \ldots, B_m\}$ będzie zbiorem dźwigarów przewidzianych do montażu, które należy dostarczyć z miejsca prefabrykacji na plac budowy. Dla dowolnego dźwigara $B_i \in \mathcal{B}$ wprowadzamy oznaczenia:

z_i	-	czas załadunku dźwigara,
t_i	-	czas transportu dźwigara na plac budowy,
r_i	-	czas rozładunku dźwigara na placu budowy,
p_i	-	czas powrotu (czas przejazdu z placu budowy do miejsca prefabrykacji, po dostarczeniu dźwigara B_i),
e_i	-	żądany najwcześniejszy termin przywozu na plac budowy,
d_i	-	żądany najpóźniejszy termin przywozu na plac budowy,
v_i	-	współczynnik funkcji kary za zbyt wczesne przybycie na plac budowy,
w_i	-	współczynnik funkcji kary za zbyt późne przybycie na plac budowy.

Zakładamy ponadto, że powyższe parametry przyjmują wartości naturalne (ewentualnie zero).

Załóżmy, że na podstawie harmonogramu prowadzenia prac budowlano-montażowych wyznaczono kolejność montażu dźwigarów (w takiej kolejności należy je dostarczać na plac budowy). Dla ustalenia uwagi przyjmujemy, że jest to kolejność $(B_1, B_2, \ldots, B_m)$. Zakładamy ponadto, że załadunek pierwszego dźwigara rozpoczyna się w chwili 0.

Przez S_i oznaczamy termin dostarczenia dźwigara B_i na plac budowy. Terminy $S_1, S_2, \ldots, S_m$ muszą spełniać następujące ograniczenia:

(*i*) $S_1 \geq z_1 + t_1$,

(*ii*) $S_{i+1} \geq S_i,\ i = 1,2,\ldots,m-1$,

(*iii*) $\forall\ B_i, B_j \in \mathcal{B}, S_i - r_i - t_i - z_i \geq S_j - r_j - t_j - z_j$ lub $S_j - r_j - t_j - z_j \geq S_i - r_i - t_i - z_i$,

(*iv*) $S_{i+1} \geq S_i + r_i + p_i + z_{i+1} + t_{i+1},\ i = 1,2,\ldots,m-1$.

W sformułowaniu tym ograniczenie (*ii*) jest matematycznym zapisem ograniczenia (a), (*iii*) odpowiada ograniczeniu (b), a (*iv*) ograniczeniu (c). Natomiast z ograniczenia (*i*) wynika, że załadunek pierwszego dźwigara B_1 rozpoczyna się w chwili 0.

Gdyby transport i montaż odbywały się w sposób ciągły, wówczas termin dostarczenia

dźwigara B_i

$$S_i = \sum_{j=1}^{n} \left(z_j + t_j + \eta_j + p_j\right) + z_i + t_i. \tag{1}$$

Problem transportu dźwigarów sprowadza się więc do wyznaczenia terminów dostaw $S_1, S_2, \ldots, S_m$ spełniających ograniczenia (*i*)--(*iv*), które optymalizują pewne ustalone kryterium optymalizacyjne.

1.2. Transport dokładnie na czas jednym pojazdem

Zakładamy w tym rozdziale, że wszystkie dźwigary są transportowane na plac budowy przez jeden pojazd. Harmonogram transportu musi ponadto spełniać ograniczenia (a)--(c).

Niech $(S_1, S_2, \ldots, S_m)$ będzie ciągiem dopuszczalnych terminów dostarczania dźwigarów (tj. spełniającym ograniczenia (*i*)-(*iv*)). Wówczas

$$E_i = \max\{0, e_i - S_i\} \tag{2}$$

jest *przyśpieszeniem*, a

$$T_i = \max\{0, S_i - d_i\} \tag{3}$$

spóźnieniem dostawy dźwigara $B_i \in \mathcal{B}$ na plac budowy (gdzie e_i oraz d_i są odpowiednio najwcześniejszym i najpóźniejszym żądanym terminem przywozu). Wielkości $u_i \cdot E_i$ oraz $w_i \cdot T_i$ są odpowiednio *karą* za przyśpieszenie lub spóźnienie dostawy.
Problem transportu dźwigarów dokładnie na czas jednym pojazdem (w skrócie oznaczany przez **TBJP**) polega na wyznaczeniu dopuszczalnego ciągu terminów dostaw $S = (S_1, S_2, \ldots, S_m)$, dla którego funkcja celu (kryterium optymalizacyjne)

$$F(S) = \sum_{i=1}^{m} \left(u_i E_i + w_i T_i\right) \tag{4}$$

osiąga wartość minimalną.
Można pokazać, że w szczególnym przypadku (tj. przy pewnych dodatkowych założeniach) problem **TBJP** jest równoważny silnie *NP-trudnemu* jednomaszynowemu problemowi szeregowania zadań na jednej maszynie z przezbrojeniami (oznaczanemu w literaturze przez $1|e_i, d_i|\sum (u_i E_i + w_i T_i)$, Bożejko i Wodecki 2009, Wodecki 2009). Wobec tego problem **TBJP** należy także do klasy problemów silnie *NP-trudnych*. Dlatego też do jego rozwiązywania będziemy stosowali algorytmy przybliżone o wielomianowej złożoności obliczeniowej.

1.3. Transport dokładnie na czas wieloma pojazdami

W tym rozdziale rozpatrujemy problem transportu dźwigarów, w którym mamy do dyspozycji wiele pojazdów. Niech $\mathbb{l}$ będzie liczbą pojazdów. Każdy z nich może przewozić dowolny dźwigar. W tym przypadku problem optymalnego transportu dźwigarów polega na ustaleniu takich terminów dostawy poszczególnych dźwigarów, aby zminimalizować przyjęte kryterium. Oprócz ograniczeń (*a*)--(*c*) należy dodatkowo przyjąć, że

d) każdy dźwigar może być transportowany tylko przez jeden pojazd.

Wprowadzamy zmienne decyzyjne

$$x_{i,j} = \begin{cases} 1, & gdy\ \mathrm{i}-ty\ dźwigar\ jest\ transportowany\ przez\ \mathrm{j}-ty\ pojazd; \\ 0, & w\ przeciwnym\ przypadku. \end{cases}$$

dla $i = 1,2,\ldots,n,\ j = 1,2,\ldots,l.$

Wówczas, odpowiednikiem ograniczenia (d) w modelu matematycznym (*i*)--(*iv*) są zależności:

(*v*) $\sum_{i=1}^{l} x_{i,j} = 1,$ dla każdego $j = 1,2,\ldots,l.$

Problem transportu dźwigarów dokładnie na czas przy użyciu *l* pojazdów (w skrócie oznaczany przez **TDWP**) polega na wyznaczeniu ciągu terminów dostaw $S = (S_1, S_2, \ldots, S_m)$ oraz zmiennych $x_{i,j}$ $(i = 1,2,\ldots,n,\ j = 1,2,\ldots,l)$ spełniających ograniczenia (*i*)--(*v*), dla którego funkcja celu osiąga wartość optymalną.
Podobnie jak w punkcie 2 będziemy minimalizowali sumę kar za nieterminowość dostaw dźwigarów, czyli funkcję

$$F(S) = \sum_{i=1}^{m} (u_i E_i + w_i T_i) \tag{5}$$

Można pokazać, że problem **TDWP** (podobnie jak **TDJP**) należy do klasy problemów silnie *NP-trudnych*. Stąd do jego rozwiązania będą stosowane algorytmy aproksymacyjne.

Rozpatrywane w tym rozdziale funkcje celu są kryteriami nieregularnymi. Obecnie, są one traktowane jako podstawowe dla systemów w strategii JIT, w której zachodzi potrzeba precyzyjnego dostarczania elementów. Powoduje to jednak, że należą one do klasy najtrudniejszych z rozpatrywanych w literaturze problemów optymalizacyjnych. Stosowanie w algorytmach ich rozwiązywania ogólnie znanych własności jest mało efektywne. Skuteczne są jedynie metody dedykowane -- silne wykorzystujące specyficzne własności problemu.

2. System wspamagania harmonogramowania przedsięwzięć budowlanych

Aplikacja wspomagająca harmonogramowanie przedsięwzięć budowlanych dostarcza graficznego interfejsu użytkownika, który został stworzony przy użyciu zintegrowanego środowiska programistycznego „Qt Creator". Funkcjonalności związane z harmonogramowaniem przedsięwzięć budowlanych zostały zaimplementowane w języku C++. Aplikacja została stworzona dla platformy Windows. Zastosowanie przenośnych bibliotek Qt dla języka C++ do tworzenia graficznego interfejsu użytkownika daje możliwość przeniesienia aplikacji na inne platformy. Główne moduły aplikacji to:

- moduł harmonogramowania prac budowlanych,
- moduł harmonogramowania transportu belek,
- moduł harmonogramowania w warunkach niepewności.

Całość aplikacji wykonana została w czterech fazach, obejmujących:

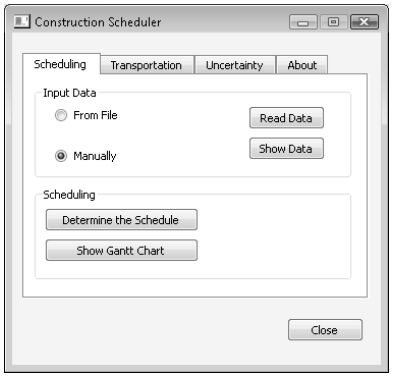
1. Projekt systemu,
2. Implementację,

3. Testowanie i symulacje wszystkich modułów, oraz
4. Wykonanie dokumentacji.

2.1. Projekt systemu. Moduł harmonogramowania prac budowlanych

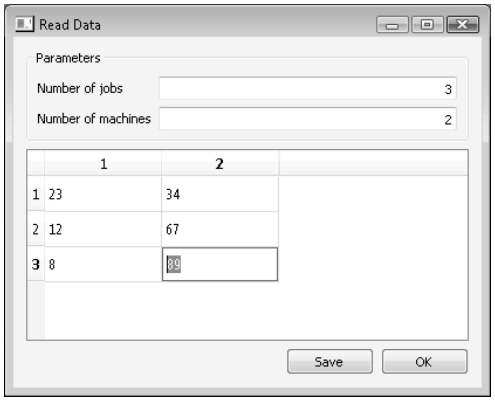
Moduł harmonogramowania prac budowlanych dostarcza następujących funkcjonalności:

- pobieranie danych wejściowych z pliku tekstowego,
- ręczne wprowadzanie danych,
- wyznaczanie harmonogramu dla przedsięwzięcia budowlanego,
- graficzna reprezentacja harmonogramu w formie wykresu Gantt-a,
- zapis do pliku terminów rozpoczęcia/zakończenia wykonywania poszczególnych prac budowlanych.



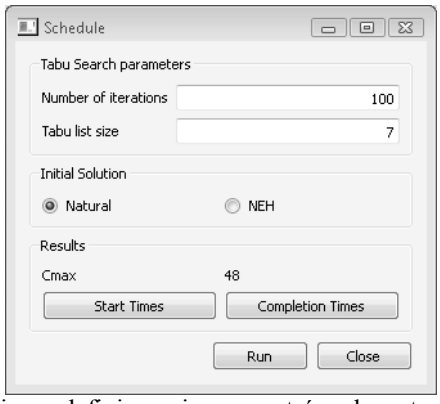
Rys. 1. Okno główne

Aby wczytać dane wejściowe z pliku należy w zakładce "Scheduling" (rys. 1) wybrać opcję "From File",a po wciśnięciu przycisku "Read Data" wskazać lokalizację pliku wejściowego na dysku.



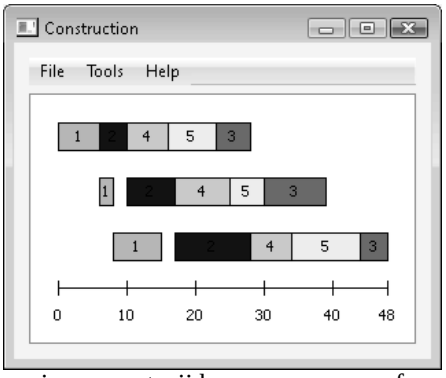
Rys. 2. Okno ręcznego wprowadzania danych wejściowych

Ręczne wprowadzanie danych wejściowych sprowadza się do wpisania czasów wykonania poszczególnych robót budowlanych w komórkach Tabeli tak jak zostało to przedstawione na rys. 2. wprowadzone dane można zapisać do pliku tekstowego po wciśnięciu przycisku „Save".



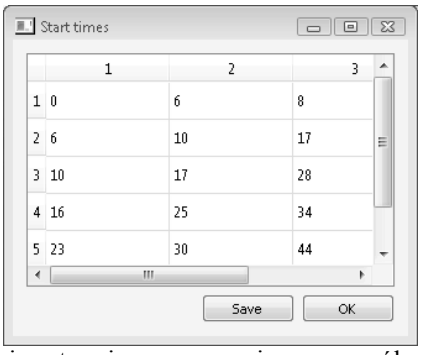
Rys. 3. Okno umożliwiające zdefiniowanie parametrów algorytmu poszukiwań z zabronieniami

Po wprowadzeniu danych wejściowych aplikacja pozwala na wyznaczenie harmonogramu robót budowlanych za pomocą algorytmu poszukiwań z zabronieniami. Okno przedstawione na Rysunku 3 pozwala na zdefiniowanie parametrów algorytmu wyznaczającego harmonogram -- liczba iteracji algorytmu, długość listy zabronień oraz rozwiązanie startowe (permutacja naturalna lub permutacja uzyskana przez algorytm konstrukcyjny NEH). Uruchomienie algorytmu wyznaczającego harmonogram prac budowlanych następuje po wciśnięciu przycisku „Run". Po zakończeniu działania algorytmu w oknie zostanie wyświetlona wartość funkcji celu dla rozwiązania uzyskanego przez algorytm *tabu search* (na podstawie opisu zaczerpniętego z pracy [2]).



Rys. 4. Okno graficznej reprezentacji harmonogramu w formie wykresu Gantt-a

Harmonogram w postaci wykresu Gantt-a można uzyskać po wciśnięciu przycisku ,,Show Gantt Chart". Okno, w którym wyświetlany jest wykres Gantt-a zostało przedstawiony na Rysunku 4. Pasek narzędzi okna, w którym wyświetlany jest wykres Gantt-a dostarcza funkcjonalności pozwalających na zapis wykresu Gantt-a w formacie PNG, zmianę rozmiaru wykresu oraz zamknięcie okna.



Rys. 5. Okno wyświetlające terminy rozpoczecia poszczególnych robót budowlanych

Wyniki działania algorytmu wyznaczającego harmonogram dla przedsięwzięcia budowlanego mogą również zostać przedstawione w formie czasów rozpoczęcia/zakończenia wykonywania poszczególnych robót budowlanych (rys. 5).

Harmonogram prowadzenia prac budowlanych wyznaczany jest przez przybliżony algorytm poszukiwań lokalnych - tabu search [3]. Algorytm ten w każdej iteracji przeszukuje sąsiedztwo rozwiązania bazowego, wybiera z sąsiedztwa najlepsze rozwiązania wg. określonego kryterium, a następnie proces poszukiwania jest powtarzany. W celu zapobieżenia cyklicznego powtarzania się rozwiązań pamiętana jest historia poszukiwań w postaci listy zabronień. Kod źródłowy algorytmu poszukiwania z zabronieniami został przedstawiony na Rysunku 6. Jako kryterium optymalizacji został przyjęty termin zakończenia wykonywania wszystkich robót budowlanych.

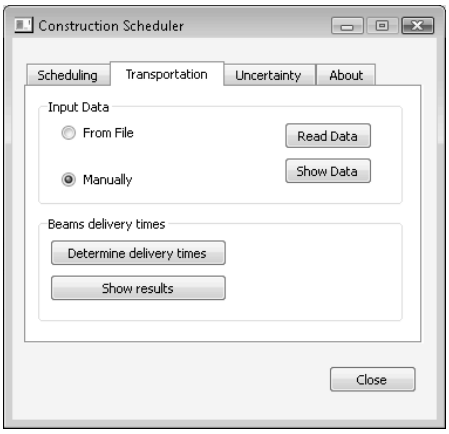
3. Implementacja. Moduł harmonogramowania transportu belek

Aplikacja została stworzona w języku C++ na platformie Microsoft Windows. W szczególności, poza modułem harmonogramowania robót budowlanych, opisanym wcześniej, wykonano moduły: harmonogramowania transportu belek oraz moduł harmonogramowania w warunkach niepewności. Moduł harmonogramowania transportu belek pozwala na wyznaczenie terminów dostawy belek na plac budowy. Terminy te wyznaczane są na podstawie harmonogramu prac budowlanych wyznaczonego przez moduł harmonogramowania prac budowlanych. Dodatkowe parametry wymagane do wyznaczenia terminów dostawy belek na plac budowy mogą być wprowadzone ręcznie lub wczytane z pliku tekstowego. Okno modułu harmonogramowania transportu belek zostało przedstawione na rys. 7.

```
int TabuSearch::tabu(int *pi, const int &CmaxInit)
{
  const int n = fs_.getn();
  int *piTS = new int[n+1];
  int *piIt = new int[n+1];
  int CmaxBest = CmaxInit;
  Copy(n, pi, piTS);
  Copy(n, pi, piIt);
  for(int i=1;i<=it_;i++)
  {
    Move move(0, 0);
    generateNeighborhood(piTS, piIt, move);
    int CmaxIt = fs_.cmax(piIt);
    cout << i << " -> " << CmaxBest << " " << CmaxIt << " " << e

    if(CmaxIt < CmaxBest)
    {
      CmaxBest = CmaxIt;
      Copy(n, piIt, pi);
    }
    t_.AddToList(Move(piTS[move.geta()], piTS[move.getb()]) );
    Copy(n, piIt, piTS);
  }
  delete[] piTS;
  delete[] piIt;
  return CmaxBest;
}
```

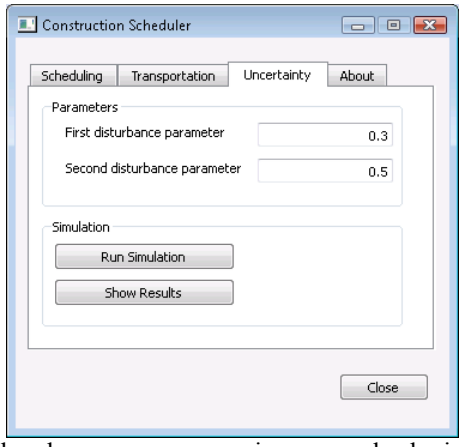
Rys. 6. Kod źródłowy algorytmu tabu search



Rys. 7. Okno harmonogramowania transportu belek

4. Testowanie i symulacje wszystkich modułów. Moduł harmonogramowania w warunkach niepewności

Moduły zostały przetestowane na przykładach testowych wybranych z literatury branżowej. W szczególności, moduł harmonogramowania w warunkach niepewności pozwala na wyznaczenie harmonogramów dla sytuacji, w których nie można precyzyjnie określić czasów wykonania poszczególnych robót budowlanych. Czasy wykonania robót budowlanych są modelowane za pomocą liczb rozmytych w trzypunktowej reprezentacji. Na Rysunku 8 zostało przedstawione okno modułu harmonogramowania w warunkach niepewności. Pozwala ono na wprowadzenie parametrów związanych z trzypunktową reprezentacją liczb rozmytych. W celu przeprowadzenia obliczeń w warunkach niepewności należy wcisnąć przycisk „Run simulation". Po przeprowadzeniu obliczeń dostęp do wyników można uzyskać prze wciśnięcie przycisku „Show results".



Rys. 8. Okno harmonogramowania w warunkach niepewności

5. Eksperymenty obliczeniowe

Eksperymenty obliczeniowe zostały przeprowadzone na komputerze z procesorem Intel Core 2 Duo 3.0 GHz pracującym pod kontrolą systemu operacyjnego Windows Vista Business dla instancji testowych o różnych rozmiarach: $10x7$, $10x11$, $10x16$, $20x7$, $20x11$, $20x16$, $30x7$, $30x11$, $30x16$. Jako rozwiązania referencyjne zostały przyjęte rozwiązania uzyskane przy użyciu algorytmu konstrukcyjnego NEH [4].

Wartośc procentowego błędu względnego *PRD* (*Percentage Relative Deviation*) została wyznaczona według następującej formuły:

$$PRD\,[\%] = \frac{C_{maxNEH} - C_{maxTS}}{C_{maxNEH}} \cdot 100\%, \tag{6}$$

gdzie: C_{maxNEH} – wartość funkcji celu uzyskana przy użyciu algorytmu konstrukcyjnego NEH,

C_{maxTS} - wartość funkcji celu uzyskana przy użyciu algorytmu poszukiwania z zabronieniami.

Tab. 1. Procentowy błąd względny zaproponowanego algorytmu dla różnych długości listy zabronień

problem	nxm	PRD [%]			
		$\|T\| = 7$	$\|T\| = 8$	$\|T\| = 9$	$\|T\| = 10$
TB 01-10	$10x7$	-3.15047	-3.31387	-3.31387	-3.31387
TB 11-20	$10x11$	-1.6111	-1.751	-1.73709	-1.5939
TB 21-30	$10x16$	-1.49879	-1.4522	-1.48746	-1.54089
TB 31-40	$20x7$	-2.53011	-2.56118	-2.56118	-2.72837
TB 41-50	$20x11$	-3.61138	-3.57945	-3.91385	-3.71792
TB 51-60	$20x16$	-4.64157	-4.624	-4.85642	-4.84387
TB 61-70	$30x7$	-2.55808	-2.52155	-2.65694	-2.73077
TB 71-80	$30x11$	-4.03188	-4.08068	-4.59128	-4.42457
TB 81-90	$30x16$	-5.21084	-5.0071	-5.3435	-5.67687
Średnia		-3.20491	-3.21011	-3.38462	-3.39678

Tab. 2. Procentowy błąd względny dla różnej liczby iteracji

problem	nxm	PRD [%]			
		$\|IT\| = 500$	$\|IT\| = 1000$	$\|IT\| = 2000$	$\|IT\| = 5000$
TB 01-10	$10x7$	-3.31387	-3.31387	-3.31387	-3.31387
TB 11-20	$10x11$	-1.53736	-1.56549	-1.5939	-1.5939
TB 21-30	$10x16$	-1.49879	-1.54089	-1.54089	-1.54089
TB 31-40	$20x7$	-2.69253	-2.71731	-2.71731	-2.72837
TB 41-50	$20x11$	-3.59034	-3.59034	-3.71792	-3.71792
TB 51-60	$20x16$	-4.66131	-4.7331	-4.82046	-4.84387
TB 61-70	$30x7$	-2.64038	-2.65694	-2.65694	-2.73077
TB 71-80	$30x11$	-4.37996	-4.41716	-4.41716	-4.42457
TB 81-90	$30x16$	-5.24632	-5.30845	-5.51101	-5.67687
Średnia		-3.28454	-3.31595	-3.3655	-3.39678

Wyniki eksperymentów obliczeniowych zostały zamieszczone w Tabeli 1 oraz w Tabeli 2. Tabela 1 zawiera wartości procentowego błędu względnego dla różnej długości listy zabronień oraz ustalonej liczby iteracji - $|IT| = 5000$. Uzyskane wyniki średniej wartości błędu względnego pokazują, że jakość uzyskanych rozwiązań (w sensie wartości funkcji celu) rośnie wraz ze wzrostem długości listy zabronień. Warto zauważyć, że dla instancji testowych o małych rozmiarach ($10x7$) zwiększanie długości listy zabronień powyżej wartości $|T| = 8$ nie prowadzi do poprawy jakości uzyskanych rozwiązań. Natomiast dla instancji testowych o dużych rozmiarach ($30x7$) zwiększenie długości listy zabronień skutkuje uzyskaniem lepszych rozwiązań. Tabela 2 zawiera wartości procentowego błędu względnego dla różnej liczby iteracji oraz ustalonej długości listy zabronień $|T| = 10$. Z uzyskanych wyników można zauważyć, że dla problemów o małych rozmiarach ($10x7$, $10x11$, $10x16$, $20x7$) zwiększanie liczby iteracji nie prowadzi do poprawy uzyskanych rozwiązań. Natomiast dla problemów o większych rozmiarach ($20x16$, $30x7$, $30x11$, $30x16$) wzrost liczby iteracji powoduje poprawę uzyskanych rozwiązań.

6. Podsumowanie

Aplikacja wspomagająca harmonogramowanie przedsięwzięć budowlanych dostarcza prostego w obsłudze graficznego interfejsu użytkownika, z wieloma opisami stanowiącymi samokomentującą się dokumentację programu w postaci opisów i podpowiedzi. Dane niezbędne do wyznaczenia harmonogramu mogą zostać wprowadzone ręcznie lub z pliku tekstowego co pozwala na szybkie uzyskanie wyników dla problemów o różnych rozmiarach. Wyznaczony przez aplikację harmonogram wykonania robót budowlanych może zostać przedstawiony w formie graficznej - wykres Gantt-a lub w postaci tekstowej - terminy rozpoczęcia/zakończenia wykonywania poszczególnych robót budowlanych. Uzyskane wyniki mogą zostać zapisane na dysk w formie pliku PNG (wykres Gantta) lub pliku tekstowego (terminy rozpoczęcia/zakończenia). Zastosowanie szybkiego algorytmu przybliżonego do harmonogramowania przedsięwzięć budowlanych pozwala na uzyskanie rozwiązań o wysokiej jakości (w sensie wartości funkcji celu) w krótkim czasie.

Literatura

1. Bożejko W., Wodecki M., Rogalska M., Hejducki Z.: Scheduling of a construction projects with a hybrid evolutionary algorithm's application. in: Evolutionary Algorithms (Eisuke Kita, ed.), InTech Publishing, 2011, 295-308.
2. Bożejko W.: A new class of parallel scheduling algorithms. Oficyna Wydawnicza Politechniki Wrocławskiej, 2010, 1–280.
3. Glover F.: Tabu Search - Part I. INFORMS Journal on Computing, Vol. 1, No. 3, 1989, 190-206.
4. Nawaz, M., Enscore Jr, E. and Ham, I.: A heuristic algorithm for the m-machine n-job flow-shop sequencing problem. Omega. The International Journal of Management Science. Vol 11. 91-95.
5. Wodecki M.: Metody agregacji w problemach optymalizacji dyskretnej. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław, 2009, 1-299.

Dr hab. Wojciech BOŻEJKO
Mgr inż. Mariusz UCHROŃSKI
Instytut Informatyki, Automatyki i Robotyki
Politechnika Wrocławska
50-370 Wrocław, Wyb. Wyspiańskiego 27
tel.: (0-71) 320 2961
e-mail: wojciech.bozejko@pwr.wroc.pl
mariusz.uchronski@pwr.wroc.pl

Dr hab. inż. Zdzisław HEJDUCKI, prof. PWr
Instytut Budownictwa
Politechnika Wrocławska
50-370 Wrocław, Wyb. Wyspiańskiego 27
e-mail: zdzislaw.hejducki@pwr.wroc.pl

Dr hab. Mieczysław WODECKI
Instytut Informatyki
Uniwersytet Wrocławski
51-360 Wrocław, Joliot-Curie 15
tel.: (0-71) 375 7815
e-mail: mwd@ii.uni.wroc.pl