ALGORYTMY MEMETYCZNE DLA PEWNEGO PROBLEMU POTOKOWEGO W BUDOWNICTWIE

Wojciech BOŻEJKO, Zdzisław HEJDUCKI, Paweł RAJBA, Mieczysław WODECKI

Streszczenie: System pracy potokowej w budownictwie dotyczy realizacji kompleksu obiektów składających się z wielu jednakowych prac wykonywanych kolejno przez wyspecjalizowane brygady. Rozpatrywany jest problem harmonogramowania projektu z niepewnymi czasami wykonywania prac reprezentowanymi przez liczby rozmyte lub rozkłady zmiennych losowych. Przedstawiamy algorytmy memetyczne (hybrydowe algorytmy genetyczne) oraz eksperymenty obliczeniowe, których celem było zbadanie stabilności wyznaczanych rozwiązań.

Słowa kluczowe: problem potokowy, niepewne dane, harmonogramowanie, algorytm memetyczny.

1. Wprowadzenie

System pracy potokowej w budownictwie [11, 7] jest odpowiednikiem produkcji taśmowej (przepływowej) w przemyśle. Dotyczy realizacji kompleksu obiektów składających się z wielu jednakowych prac wykonywanych przez wyspecjalizowane brygady. Obiektom odpowiadają zadania, brygadą - maszyny, a pracą wykonywanym przez brygady - operacje. Kolejności wykonywania prac na obiekcie odpowiada porządek technologiczny. Planowanie przebiegu robót budowlanych w systemach potokowych jest uzasadnione w przypadku realizacji obiektów, na których można wydzielić: działki robocze, sektory, odcinki niezależne technologicznie, tj. zadania o dużej pracochłonności robót, np. kompleksy przemysłowe, zespoły budynków mieszkalnych, odcinki dróg, sieci wodociągowe i kanalizacyjne, itp. Pojawia się wówczas problem synchronizacji w czasie i przestrzeni wielu robót budowlanych możliwych do prowadzenia równolegle i jednocześnie. Zachodzi wówczas możliwość znalezienia najlepszego harmonogramu robót uwzględniającego ustalone kryteria optymalizacji np.: typu czas/koszt/zasoby. Są to bardzo ważne zagadnienia praktyki budowlanej, mające znaczący i bezpośredni wpływ na ostateczne koszty realizacji. Prowadzi to jednak do złożonych dyskretno-ciągłych problemów optymalizacyjnych z niepewnymi parametrami oraz nieregularnymi funkcjami celu. Przenosząc problemy harmonogramowania przedsięwzięć budowlanych w dziedzinę klasycznej teorii szeregowania zadań napotyka się na wiele trudności związanych z doborem właściwego modelu oraz odpowiedniego algorytmu. Są to zazwyczaj zupełnie nowe, *silnie NP-trudne* problemy optymalizacji kombinatorycznej.

Ze względu na stosowanie nowych technik i technologii, unikalność, warunki atmosferyczne i geologiczne, często nie ma możliwości jednoznacznego określenia wartości pewnych parametrów. W takich przypadkach mamy do czynienia z procesem podejmowania decyzji w warunkach niepewności. Niepewność danych przekłada się bezpośrednio na wielkość ryzyka. Ponadto, w trakcie realizacji procesu może się okazać, że niektóre parametry różnią się od wstępnie przyjętych ("typowych"), co przy braku

stabilności rozwiązania prowadzi do zupełnie nieprzydatnych w praktyce rozwiązań. Niepowodzenia wynikające z bezpośredniego stosowania klasycznych algorytmów deterministycznych wskazują na konieczność uwzględnienia niepewności już na etapie budowy modelu oraz konstrukcji samego algorytmu.

Problemy podejmowania decyzji w warunkach niepewności rozwiązuje się stosując metody probabilistyki lub teorii zbiorów rozmytych. W pierwszym przypadku [3, 17] istotna jest znajomość rozkładów zmiennych losowych. Niektóre procesy z natury mają charakter losowy. Zależą od pogody, natężenia ruchu, liczby wypadków, warunków geologicznych, awaryjności sprzętu, itp. Jeżeli ponadto posiadają pewną "historię", więc na bazie istniejących danych statystycznych można określić ich rozkłady.

W wielu jednak zagadnieniach niepewność danych nie ma charakteru losowego, lecz wynika z powodu unikalności procesu, błędu pomiaru, itp. W tym przypadku naturalnym sposobem reprezentowania niepewności są liczby rozmyte [9,10]. Dlatego dużym problemem jest właściwy dobór funkcji przynależności oraz metoda dyfuzyfikacji. Mają one decydujący wpływ na jakość wyznaczanych rozwiązań.

Systemy pracy potokowej w budownictwie są przedmiotem zainteresowania w wielu ośrodków naukowych. Wynika to z potrzeby optymalnego zarządzania budowami wielkich obiektów inżynieryjnych, m. in. odcinków dróg, mostów, kompleksów budynków, obiektów przemysłowych, itp. Istnieje ogromna potrzeba doskonalenia tradycyjnych metod harmonogramowania (uwzględniającego np. nowe technologie, niepewność danych) i efektywnego wykorzystania nowoczesnych metod obliczeniowych.

W pracy przedstawiamy idee algorytm memetycznego rozwiązywania problem harmonogramowania przedsięwzięć budowlanych realizowanych w systemie potokowym z niepewnymi czasami wykonywania prac. Porównujemy stabilność rozwiązań w przypadku, gdy niepewne dane są reprezentowane przez zmienne losowe o rozkładzie normalnym lub liczb rozmytych w trzypunktowej reprezentacji. Niektóre wyniki prowadzonych badań zostały przedstawione w pracy Bożejko, Hejducki, Rajba, Wodecki [2] oraz Rogalska, Bożejko, Hejducki, Wodecki [16].

2. Systemy potokowe w budownictwie

Rozpatrujemy przedsięwzięcie budowlane (w skrócie **PB**) polegające na wykonaniu n obiektów ze zbioru

$$O = \{O^1, O^2 \ldots, O^n\},$$

przez m brygad ze zbioru

$$B = \{B_1, B_2 \ldots, B_m\}.$$

Każdy obiekt $O^i \in O$ jest ciągiem m prac

$$O^i = [P_{i,1}, P_{i,2}, \ldots, P_{i,m}],$$

przy czym praca P_j^i $(i = 1,2,\ldots,n,\ j = 1,2,\ldots,m)$ jest wykonywana przez brygadę B_j w czasie $p_{i,j}$. Prace na obiekcie $O^i \in O$ należy wykonać w zadanym porządku technologicznym, tzn. dowolna praca $P_{i,j}$ ma być wykonywana po zakończeniu $P_{i,j-1}$, a przed rozpoczęciem $P_{i,j+1}$ $(2 \le j \le m-1)$. Muszą być przy tym spełnione następujące ograniczenia:

(i) każda praca (na obiekcie) może być wykonywana tylko przez jedną, określoną przez ciąg technologiczny brygadę,

(ii) żadna brygada nie może wykonywać jednocześnie więcej niż jedną pracę,
(iii) na każdym obiekcie musi być zachowany porządek technologiczny,
(iv) wykonywanie żadnej pracy nie może być przerwane przed jej zakończeniem.

Niech π będzie pewną permutacją obiektów (elementów zbioru O). Permutacja ta wyznacza kolejność wykonywania poszczególnych prac na obiektach, tj. brygada $B_j \in B$ wykonuje prace $P_{\pi(i),j}$ na obiekcie $\pi(i) \in O$, dopiero po wykonaniu prac $P_{\pi(1),j}, P_{\pi(2),j}, \ldots,$ $P_{\pi(i-1),j}$ kolejno na obiektach $\pi(1), \ldots \pi(i-1)$, a przed wykonaniem $P_{\pi(i+1),j}, P_{\pi(i+2),j}, \ldots,$ $P_{\pi(n),j}$ na obiektach $\pi(i+1), \ldots, \pi(n)$. Oznaczmy przez Φ zbiór wszystkich możliwych permutacji obiektów. Moc tego zbioru wynosi $n!$.

Jeżeli prace na obiektach są wykonywane w kolejności $\pi \in \Phi$ oraz $p_{\pi(i),j}$ jest czasem wykonywania pracy $P_{\pi(i),j}$ $(i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, m)$, to moment zakończenia tej pracy $C_{\pi(i),j}$ możemy wyznaczyć z następujących zależności rekurencyjnej:

$$C_{\pi(i),j} = \begin{cases} \sum_{k=1}^{i} p_{\pi(i),j}, & j = 1, \\ C_{\pi(i),j-1} + p_{\pi(i),j}, & i = 1,\ j > 1, \\ \max\{C_{\pi(i),j-1}, C_{\pi(i-1),j}\} + p_{\pi(i),j}, & i > 1,\ j > 1, \end{cases} \tag{1}$$

a moment rozpoczęcia jej wykonywania

$$S_{\pi(i),j} = C_{\pi(i),j} - C_{\pi(i),j}. \tag{2}$$

Można łatwo sprawdzić, że określone przez (1) i (2) momenty rozpoczęcia i zakończenia prac na obiektach spełniają ograniczenia (i)-(iv), więc są rozwiązaniem dopuszczalnym problemu **BP**. Harmonogramowanie przedsięwzięcia budowlanego **PB** sprowadza się do wyznaczenia permutacji optymalnej $\pi^* \in \Phi$, dla której

$$C_{\max}(\pi^*) = \min\{C_{\max}(\pi) :\ \pi \in \Phi\},$$

gdzie $C_{\max}(\pi) = C_{\pi(n),m}$ jest terminem zakończenia wszystkich prac wykonywanych w kolejności π. Opisany wyżej model przedsięwzięcia budowlanego jest znany w teorii szeregowania jako problem przepływowy (ang. *flow sho*). Dla $m \geq 3$ należy on do klasy problemów *silnie NP-trudnych* ([Garey 4]). W ostatnich latach opublikowano wiele algorytmów metaheurystycznych jego rozwiązywania, tzn. symulowanego wyżarzania: Ogbu i Smith [13], Bożejko Wodecki [1] (algorytm równoległy), przeszukiwania z tabu: Nowicki i Smutnicki [12], Grabowski i Wodecki [6] oraz algorytmu genetycznego, Reeves [15].

3. Algorytm memetyczny

Stosowany przez nas algorytm memetyczny jest hybrydą powstałą przez połączenie algorytmu genetycznego oraz przeszukiwania z tabu (stosowanego do wyznaczania minimów lokalnych). Po wygenerowaniu nowej populacji, każdy osobnik jest punktem startowym dla algorytmu przeszukiwania z tabu (tj. wyznaczania minimum lokalnego). Następnie, w miejsce punktu startowego wstawiamy, do bieżącej populacji, wyznaczone minimum lokalne. Reasumując, jest to algorytm genetyczny, w którym każda populacja

zawiera osobniki będące minimami lokalnymi, tj. przestrzeń rozwiązań dopuszczalnych tego algorytmu składa się jedynie z minimów lokalnych. Poniżej przedstawiamy poszczególne elementy algorytmu memetycznego.

3.1 Algorytm genetyczny

Określenie „algorytmy genetyczne" obejmuje grupę metod obliczeniowych, których wspólną cechą jest korzystanie, przy rozwiązywaniu danego problemu, z mechanizmu opartego na zjawisku naturalnej ewolucji gatunków [8]. W klasycznej postaci, są one bezpośrednią adaptacją tego zjawiska stąd w ich opisie używa się pojęć z genetyki.

Działanie algorytmu genetycznego rozpoczyna się od utworzenia populacji (podzbioru zbioru rozwiązań) początkowej P_0, której liczebność jest zazwyczaj stała przez cały czas działania algorytmu. Niech k będzie kolejnym numerem iteracji algorytmu. Nowa k+1 generacja (tj. zbiór P_{k+1}) jest tworzona w następujący sposób. Z bieżącej populacji P_k wybierana jest pewna ilość najlepszych osobników (rodziców, podzbiór P'_k) - operacja selekcji. Z nich, poprzez mechanizm krzyżowania, generuje się nowych osobników (potomstwo, zbiór P''_k). Na części z nich jest dokonywana operacja mutacji, której celem jest większe zróżnicowanie populacji. Tak wygenerowane potomstwo zastąpi najgorszych osobników w bieżącej populacji (wymiana), tworząc w ten sposób nową generacje. Algorytm zazwyczaj kończy działanie po wygenerowaniu z góry ustalonej liczby osobników.

W konstrukcji algorytmu genetycznego zastosowaliśmy "naturalną", tj. w postaci permutacji, reprezentację kolejności wykonywania zadań (rozwiązań dopuszczalnych). W tym przypadku nie można bezpośrednio stosować klasycznych operatorów krzyżowania i mutacji, bowiem w wyniku ich działania można otrzymać potomków nie będących permutacjami, a więc rozwiązaniami dopuszczalnymi. Miarą przystosowania osobników (permutacji) będzie wartość funkcji celu, czyli termin zakończenia wykonywania wszystkich prac. Ponadto, stosujemy tzw. elitarną strategię ewolucji polegającą na tym, że kolejne pokolenie zawsze zawiera (pomijając proces selekcji naturalnej i reprodukcji) stałą liczbę najlepiej przystosowanych osobników z poprzedniej populacji (rodziców). Aby ich zachować we wszystkich populacjach odrzuca się pewną liczbę najgorzej przystosowanych osobników potomnych z populacji P''_k powstałej w wyniku procesu reprodukcji. Dzięki użyciu tej metody gwarantuje się przetrwanie najlepiej przystosowanych osobników w całej historii gatunku.

Klasyczny algorytm genetyczny (AG)

$k \leftarrow 0$;
$\boldsymbol{P_k} \leftarrow$ Losowa Populacja;
repeat
  Selekcja($\boldsymbol{P_k}$,$\boldsymbol{P'_k}$); {*Wybór rodziców*}
  Krzyżowanie($\boldsymbol{P'_k}$,$\boldsymbol{P''_k}$); {*Generowanie potomstwa*}
  Mutacja($\boldsymbol{P''}_k$);
  $\boldsymbol{P_{k+1}} \leftarrow \boldsymbol{P''_k}$; {*Nowa populacja*}
  $k \leftarrow k + 1$;
until Warunek_Końca;

Algorytm kończy działanie (Warunek_Końca) zazwyczaj po wygenerowaniu z góry określonej liczby iteracji. Złożoność obliczeniowa algorytmu zależy przede wszystkim, od liczby iteracji, liczebności populacji oraz sposobu krzyżowania osobników. Realizacja algorytmu wymaga ustalenia wartości parametrów liczbowych oraz określenia sposobu realizacji poszczególnych operacji. W konstrukcjach algorytmów, poszczególne elementy, zrealizowano w następujący sposób.

Selekcja

Miarą przystosowania osobników (permutacji) będzie wartość funkcji celu, czyli koszt permutacji. Ponadto, stosujemy tzw. elitarną strategię ewolucji polegającą na tym, że kolejne pokolenie zawsze zawiera (pomijając proces selekcji naturalnej i reprodukcji) stałą liczbę (50%) najlepiej przystosowanych osobników z poprzedniej populacji (rodziców). Aby ich zachować we wszystkich populacjach odrzuca się pewną liczbę najgorzej przystosowanych osobników potomnych z populacji P_k'' powstałej w wyniku procesu reprodukcji.

Krzyżowanie

W literaturze opisano wiele metod krzyżowania permutacji (rodziców) generujących potomków będących także permutacjami. Operator krzyżowania **PMX** (*Partial – Mapped Crossover*) został opisany w pracy Goldberga [5]. Polega on na wylosowaniu dwóch liczb naturalnych (zgodnie z rozkładem jednostajnym) z przedziału [*1..n*], które wyznaczają podział permutacji na trzy części. Środkowe części są zamieniane tzn. środkowa część pierwszego rodzica trafia do drugiego potomka, środkowa część drugiego rodzica trafia do pierwszego potomka. Pozostałe części są kopiowane do odpowiednich potomków, przy czym, jeśli kopiowana liczba burzy strukturę permutacji, to zamiast niej jest wstawiana wartość wyznaczona przez określone przyporządkowanie.

Mutacja

Operacja mutacji jest wykonywana zazwyczaj na niewielkiej liczbie osobników w celu ich większego zróżnicowania. W algorytmie zastosowaliśmy operator wymiany (ang. *swap*) polegający na losowym wyznaczeniu dwóch elementów w permutacji i zmianie ich miejscami.

Warunek końca

Parametr - ustalona liczba iteracji algorytmu.

Zastosowanie algorytmu genetycznego do rozwiązywania problemu przepływowego zostało między innymi opisane w pracy Bożejki i Wodeckiego [1].

3.2 Algorytm przeszukiwania z tabu

Obecnie najlepsze znane w literaturze efektywne algorytmy rozwiązywania permutacyjnego problemu przepływowego [12, 6] są oparte na metodzie przeszukiwania z tabu (ang. *tabu search*). Generalnie, polega ona na iteracyjnym polepszaniu bieżącego rozwiązania poprzez lokalne przeszukiwanie. Rozpoczyna się od pewnego rozwiązania początkowego (startowego π). Następnie, generuje się jego otoczenie (sąsiedztwo V_π) oraz wyznacza najlepsze rozwiązanie z tego otoczenia, które przyjmuje się za rozwiązanie startowe w następnej iteracji. Dopuszcza się możliwość zwiększania wartości funkcji celu

(przy wyznaczaniu nowego rozwiązania startowego), 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 nastąpił by 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 *LTS*.

Poniżej zamieszczamy schemat algorytmu, w którym F jest funkcją celu, π - punktem startowym, a π^* - wyznaczonym rozwiązaniem (minimum lokalnym).

Algorytm przeszukiwania z tabu

repeat
Wyznaczyć otoczenie V_π permutacji π;
Usunąć z V_π permutacje zakazane przez listę
LTS, uwzględniając kryterium aspiracji;
Znaleźć permutację $\delta \in V_\pi$ taką, że: $F(\delta) = \min\{F(\beta): \beta \in V_\pi\}$;
if $F(\delta)<F(\pi^*)$ **then** $\pi^* \leftarrow \delta$;
Umieść atrybuty δ na liście *LTS*;
$\pi \leftarrow \delta$;
until Warunek_Końca.

Złożoność obliczeniowa algorytmu opartego na metodzie tabu search w dużej mierze zależy od wyboru rodzaju poszczególnych jego elementów, tj. metody wyznaczania sąsiedztwa, rodzaju elementów przechowywanych w liście tabu jak również rodzaju oraz długości tej listy, sposobu wyliczania wartości funkcji celu oraz warunku zakończenia.

Realizacja algorytmu wymaga ustalenia wartości parametrów liczbowych oraz określenia sposobu realizacji poszczególnych operacji. W konstrukcjach algorytmów, poszczególne elementy, zrealizowano ustalonym sposobem.

Otoczenie

Otoczenie V_π jest generowane przez ruchy typu wstaw. Polega on na przestawieniu elementu na inną pozycje w permutacji (dokładnie jest to przedstawione w pracy Grafowskiego i Wodeckiego [6]. Moc takiego otoczenia wynosi $n(n\text{-}1)$.

Lista ruchów zakazanych

Zawierająca atrybuty pewnej liczby ostatnio rozpatrywanych rozwiązań (startowych). Jest skojarzona z operatorami modyfikacji rozstrzygającymi, w jaki sposób dodajemy nowe oraz usuwamy istniejące atrybuty rozwiązań. Jeżeli z otoczenia V_π wybieramy permutację β powstałą z π przez przestawienie elementu $\pi(i)$ na pozycję j, to na listę zapisujemy trójkę $(\pi(i), j, C_{\max}(\beta))$. Z kolei, gdy rozpatrujemy permutację δ powstałą z π przez przestawienie elementu $\pi(k)$ na pozycję l, a na liście jest trójka (a, b, F) taka, że $\pi(k) = a, l = b$ oraz $C_{\max}(\delta) \geq F$, wówczas permutację δ pomijamy.

Kryterium aspiracji

Są to warunki, przy których można pominąć ograniczenia wprowadzone przez listę tabu. Jeżeli atrybuty pewnego rozwiązania z otoczenia V_π są na liścia tabu, a jego wartość

funkcji celu jest mniejsza od wartości rozwiązania π, wówczas nie odrzucamy takiego rozwiązania.

Warunek końca

Parametr - ustalona liczba iteracji algorytmu.

Przy implementacji algorytmu wykorzystano uproszczoną wersję przeszukiwania z tabu, zamieszczoną w pracy Grabowskiego i Wodeckiego [6]. Celem przyśpieszenia obliczeń zastosowano jedynie „własności eliminacyjne" bloków z drogi krytycznej.

Ogólny schemat algorytmu memetycznego (hybrydowego algorytmu genetycznego), który stosowano do rozwiązywania problemu przepływowego można przedstawić następująco:

Algorytm memetyczny (**AM**)

$k \leftarrow 0$;
$\boldsymbol{P_k} \leftarrow$ Losowa_Populacja;
repeat
  Selekcja($\boldsymbol{P_k}$,$\boldsymbol{P'_k}$); {*Wybór rodziców*}
  Krzyżowanie($\boldsymbol{P'_k}$,$\boldsymbol{P''_k}$); {*Generowanie potomstwa*}
  Lokalna_Optymalizacja($\boldsymbol{P''}_k$, $\boldsymbol{P_k^*}$); {*Algorytm przeszukiwania z tabu*}
  $\boldsymbol{P_{k+1}} \leftarrow \boldsymbol{P_k^*}$; {*Nowa populacja*}
  $k \leftarrow k + 1$;
until Warunek_Końca;

W stosunku do klasycznego algorytmu genetycznego, pominięto operator mutacji. Na podstawie przeprowadzonych eksperymentów obliczeniowych stwierdzono, że lokalna optymalizacja daje wystarczające zróżnicowanie populacji.

4. System potokowy z niepewnymi parametrami

Dla wielu robót budowlanych istnieją duże trudności w jednoznacznym określeniu pewnych parametrów. Niepewność zazwyczaj wiąże się z unikalnością proces lub jego zależnością od zjawisk losowych. W pracy rozpatrujemy problem, w którym niepewne są czasy wykonywania prac $p_{i,j}$ ($i = 1,2,...,n,\ j = 1,2,...,m$). W praktyce niepewność ta może wynikać z losowości (np. pogoda, frekwencja pracowników, awaryjność maszyn, itp.) lub niemożności jednoznacznego określenia wartości (unikalność, brak norm, itp.).

4.1. Rozkłady prawdopodobieństwa

Zakładamy, że czasy wykonywania poszczególnych prac są niezależnymi zmiennymi losowymi o rozkładzie normalnym, czyli $\tilde{p}_{i,j} \sim N(p_{i,j}, \sigma_{i,j})$. Wartość oczekiwana $E(\tilde{p}_{i,j}) = p_{i,j}$. Wówczas, macierz zmiennych losowych $\tilde{p} = [\tilde{p}_{i,j}]_{n \times m}$ nazywamy danymi probabilistycznymi, a problem - probabilistycznym (w skrócie **PBP**).

Niech $\pi \in \Phi$ będzie pewną kolejnością wykonywania prac na obiektach. Dla uproszczenia obliczeń przyjmujemy, że momenty zakończenia poszczególnych prac mają wówczas także rozkład normalny

$$\tilde{C}_{\pi(i),j} \sim N(C_{\pi(i),j}, \alpha\sqrt{C^{(2)}_{\pi(i),j}}),$$

gdzie

$$C^{(2)}_{\pi(i),j} = \begin{cases} \sum_{k=1}^{i} p^2_{\pi(i),j}, & j = 1, \\ C^2_{\pi(i),j-1} + p^2_{\pi(i),j}, & i = 1,\ j > 1, \\ \max\{C^2_{\pi(i),j-1}, C^2_{\pi(i-1),j}\} + p^2_{\pi(i),j}, & i > 1,\ j > 1, \end{cases}$$

a parametr α jest wyznaczany eksperymentalnie. Po wykonaniu wstępnych obliczeń przyjęto $\alpha = 0,25$.

W tym przypadku jako kryterium porównawcze (aspiracji) rozwiązań będziemy stosowali funkcję

$$FP(\pi) = E(\tilde{C}_{\pi(n),m}) + D(\tilde{C}_{\pi(n),m}) = C_{\pi(n),m} + \alpha\sqrt{C^{(2)}_{\pi(n),m}}, \tag{4}$$

która jest sumą wartości oczekiwanej i odchylenia standardowego zmiennej losowej reprezentującej termin zakończenia całego przedsięwzięcia. Algorytm memetyczny, w którym będzie stosowana funkcja (4) nazywamy probabilistycznym i oznaczamy w skrócie przez **PAM**.

4.2. Liczby rozmyte

Niepewne czasy wykonywania prac będą reprezentowane przez trzypunktowe liczby rozmyte $\hat{p}_{i,j} = (p^{\min}_{i,j}, p^{\text{med}}_{i,j}, p^{\max}_{i,j})$, gdzie $p^{\min}_{i,j} \le p^{\text{med}}_{i,j} \le p^{\max}_{i,j}$ dla $(i = 1,2,\ldots,n,\ \ j = 1,2,\ldots,m)$. Wówczas termin wykonywania wszystkich prac (w kolejności π) jest także trzypunktową liczbą rozmytą

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

gdzie $C^{\min}_{\pi(n),m}, C^{\text{med}}_{\pi(n),m}, C^{\max}_{\pi(n),m}$ wyznacza się z zależności rekurencyjnych odpowiadających (1). Szczegółowe rozważania przedstawiono w pracy [2]. Dla porównywania rozmytych wartości funkcji celu została zastosowana tzw. słaba reguła porównania:

$$FN(\pi)) = \frac{1}{4}\left(C^{\min}_{\pi(n),m} + C^{\text{med}}_{\pi(n),m} + C^{\text{med}}_{\pi(n),m} + C^{\max}_{\pi(n),m}\right). \tag{5}$$

Algorytm memetyczny, w którym będzie stosowana funkcja (5) nazywamy rozmytym i oznaczamy w skrócie przez **RAM**.

5. Eksperymenty obliczeniowe

Przedstawione w pracy algorytmy zostały zaprogramowane w języku C++. Eksperymenty obliczeniowe wykonano na komputerze osobistym z procesorem 2.2GHz.

Dane deterministyczne

Dane deterministyczne pobrano z pliku zamieszczonego na stronie internetowej OR Library [14]. Każda z sześciu grup danych o rozmiarach $(n \times m)$: 20×5, 20×10, 20×20, 50×5, 50×10, 50×20 zawiera po 10 przykładów. W sumie, jest to 60 przykładów. Zbiór tych przykładów oznaczamy przez Ω. Celem eksperymentów było zbadanie stabilności algorytmów (deterministycznego **AM**, probabilistycznego **PAM** oraz rozmytego **RAM**), tj. wpływy zburzenie czasów wykonywania prac na wartości funkcji celu. Dane zaburzone

(deterministyczne) generowano w następujący sposób. Dla każdego przykładu danych deterministycznych $\delta \in \Omega$ wygenerowano 100 przykładów danych zaburzonych (elementy zbioru $D(\delta)$). Zaburzenie czasu $p_{i,j}$ ($i = 1,2,...,n,\ \ j = 1,2,...,m$) polega na zastąpieniu go nową wartością wylosowaną zgodnie z rozkładem normalnym $N(p_{ij}, p_{ij}/10)$. W sumie wyznaczono w ten sposób 6000 przykładów danych zaburzonych.

Dane probabilistyczne

Niech $\delta = [p_{i,j}]_{nxm}$ będzie przykładem danych deterministycznych. Przyjęto, że niepewne czasy wykonywania prac są niezależnymi zmiennymi losowymi o rozkładzie normalnym, tj. $\tilde{p}_{i,j} \square N(p_{i,j}, \sigma_{i,j})$, przy czym odchylenie standardowe $\sigma_{i,j} = p_{i,j}/30$.

Dane rozmyte

Do każdego przykładu danych deterministycznych $\delta = [p_{i,j}]_{nxm}$, niepewne czasy wykonywania prac $\hat{p}_{i,j}$, $i = 1,2,...,n,\ \ j = 1,2,...,m$ są reprezentowane przez tzw. trzypunktową liczbę rozmytą $(p_{i,j}^{\min}, p_{i,j}^{\text{med}}, p_{i,j}^{\max})$, gdzie

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

5.1. Stabilność algorytmu

Stabilność jest pewną miarą umożliwiającą oszacowanie wpływu zaburzeń danych na zmiany wartości funkcji celu. Niech **A**={**AM**, **PAM**, **RAM**}. Algorytmy **AM**, **PAM** i **RAM** są odpowiednio: deterministycznym, probabilistycznym i rozmytym. Przez $\pi_\delta^{\boldsymbol{A}}$ oznaczamy rozwiązanie (permutację) wyznaczoną przez algorytm **A** dla danych δ. Wartość wyrażenia $W(\pi_\delta^A, \varphi)$ jest wartością funkcji celu dla przykładu danych deterministycznych φ, gdy obiekty są wykonywane w kolejności (permutacji) $\pi_\varphi^{\boldsymbol{A}}$ (tj. w kolejności wyznaczonej przez algorytm **A** dla danych δ). Wówczas

$$\Delta(A, \delta, D(\delta)) = \frac{1}{|D(\delta)|} \sum_{\varphi \in D(\delta)} \frac{W(\pi_\delta^A, \varphi) - W(\pi_\varphi^{AM}, \varphi)}{W(\pi_\varphi^{AM}, \varphi)} \cdot 100\%,$$

nazywamy *stabilnością rozwiązania* $\pi_\delta^{\boldsymbol{A}}$ wyznaczonego przez algorytm **A** na zbiorze danych zaburzonych $D(\delta)$. Wyznaczając $\pi_\varphi^{\boldsymbol{TS}}$, za rozwiązanie startowe algorytmu *TS* przyjęto $\pi_\varphi^{\boldsymbol{A}}$ i wówczas $W(\pi_\delta^A, \varphi) - W(\pi_\varphi^{TS}, \varphi) \geq 0$. Stąd $\Delta(A, \delta, D(\delta)) \geq 0$. Wartość wyrażenia $\Delta(A, \delta, D(\delta))$ jest średnim błędem względnym najlepszego rozwiązania $\pi_\delta^{\boldsymbol{A}}$ do najlepszych rozwiązań wyznaczonych, dla każdego przykładu danych zaburzonych $\varphi \in D(\delta)$.

Niech Ω będzie pewnym zbiorem przykładów deterministycznych dla problemu **PB**. *Współczynnik stabilności algorytmu* **A** na zbiorze Ω definiujemy następująco:

$$S(A, \Omega) = \frac{1}{|\Omega|} \sum_{\delta \in \Omega} \Delta(A, \delta, D(\delta)). \tag{8}$$

Im ten współczynnik jest mniejszy, tym wyznaczone przez algorytm *A* rozwiązania są

stabilniejszej, tj. niewielkie zmiany wartości danych powodują niewielkie zmiany wartości funkcji celu.

5.2. Wyniki obliczeniowe

Przy uruchomianiu każdego algorytmu przyjęto następujące wartości parametrów:

1. Algorytm memetyczny:
   - liczebność populacji: $n/2$,
   - populację startową generowano losowo,
   - liczba iteracji algorytmu: $3n/2$.
2. Algorytm przeszukiwania z tabu:
   - długość listy tabu: $n/2$,
   - liczba iteracji algorytmu: n.

W pierwszej kolejności zbadano jakość wyznaczonych przez algorytmy rozwiązań. W tym celu, dla każdego rozwiązania przykładu danych deterministycznych wyznaczonego przez algorytm **A**={**AM**, **PAM**, **RAM**} obliczono procentowy błąd względny

$$\varepsilon(A) = \frac{W_A - W^*}{W^*} \cdot 100\%$$

gdzie W_A jest wartością rozwiązania wyznaczonego przez algorytm **A**, a W^* najlepszą obecnie znana wartością rozwiązania (optymalną lub przybliżoną). Średni błąd względny (dla każdej grupy danych) przedstawiono w Tabeli 1. Zgodnie z oczekiwaniami, najlepszy okazał się algorytm memetyczny **AM** (błąd $\varepsilon(A) = 5{,}8\%$). Dla najlepszych obecnie algorytmów metaheurystycznych błąd ten nie przekracza 0,5%. Dla każdego z algorytmów wyznaczono także współczynnik stabilności. Otrzymane wyniki zamieszczono w Tabeli 1. Czas obliczeń jednego przykłady, przez każdy z trzech opisanych algorytmów, nie przekracza kilkunastu sekund.

Tab. 1. Średni błąd względny $\varepsilon(A)$ oraz stabilność $S(A,\Omega)$ algorytmów: memetycznego **AM**, probabilistycznego **PAM** oraz rozmytego **RAM**.

Dane $n \times m$	Średni błąd względny $\varepsilon(A)$			Współczynnik stabilności $S(A,\Omega)$		
	Algorytm					
	AM	**PAM**	**RAM**	**AM**	**PAM**	**RAM**
20×5	1,8	3,4	6,6	16,1	6,8	8,4
20×10	8,7	10,6	11,3	17,8	5,4	11,0
20×20	6,5	9,9	13,0	21,3	6,1	7,9
50×5	5,8	8,7	10,4	13,4	8,3	8,4
50×10	5,3	9,1	9,8	15,7	7,4	12,1
50×20	6,1	10,2	14,2	15,1	9,2	9,1
Średnia	**5,7**	**8,6**	**11,0**	**16,6**	**7,2**	**9,5**

Na podstawie przedstawionych wyników można stwierdzić, że rozwiązania wyznaczone przez algorytm probabilistyczny **PAM** są w o wiele mniejszym stopniu podatne na zmiany danych wejściowych niż rozwiązanie wyznaczone przez algorytm deterministyczny **AM** lub rozmyty **RAM**. Po zaburzeniu danych, średnio wyniki są gorsze o 7,2% od najlepszych (suboptymalnych) wartości. Najmniej odporne na losowe zaburzenia danych są rozwiązania algorytmu deterministycznego **AM**. Średnie pogorszenie wyników wynosi 16,6%.

6. Podsumowanie

W pracy zaprezentowano popularną obecnie metodę metaheurystyczną stosowaną do rozwiązywania NP-trudnych problemów optymalizacyjnych - metodę algorytmu memetycznego. Mnogość zastosowań algorytmu genetycznego w różnych dziedzinach potwierdza jej szczególną łatwość adaptacji do nowych warunków i problemów. Jednakże duży wpływ na skuteczność tej metody, dla konkretnego problemu, ma sposób zaprojektowania i wykorzystania jej poszczególnych elementów. Uzyskane wyniki obliczeniowe pokazują przewagę wersji probabilistycznej algorytmu (z czasami wykonywania prac o rozkładzie normalnym), gdy dane są niepewne. Wykorzystanie elementów probabilistyki oraz teorii zbiorów rozmytych w adaptacjach metod metaheurystycznych pozwoliło na rozszerzenie zakresu zastosowania tych metod na przypadek, gdy informacja o danych wejściowych jest nieokreślona lub niejednoznaczna. Otwiera to nowe horyzonty w poszukiwaniach metod rozwiązywania dla wielu trudnych praktycznych problemów optymalizacyjnych i z pewnością pozwoli na konstruowanie w przyszłość coraz lepszych algorytmów.

Dodatkowe informacje

Praca częściowo finansowana z projektu badawczego MNiSW Nr N N514 232237.

Literatura

1. Bożejko W., Wodecki M.: Parallel genetic algorithm for the flow shop scheduling problem. Lecture Notes in Computer Science, 3019, Springer Verlag, 2004, 566-571.
2. Bożejko W., Hejducki Z., Rajba P., Wodecki M.: Project management In building process with uncertain tasks Times. Management and Production Engineering Review, vol.2, 2011, 3-9.
3. Dean B.C.: Approximation algorithms for stochastic scheduling problems. PhD thesis, MIT, 2005.
4. Garey M.R., D.S. Johnson, Seti R.: The complexity of flowshop and jobshop scheduling. Mathematics of Operations Research, 1, 1976, 117-129.
5. Goldberg D.E., Genetic Algorithms, Search, Optimization and Machine Learning. Addison-Wesley, 1989.
6. Grabowski J., Wodecki M.: A very fast tabu search algorithm for the permutation flow shop problem with makespan criterion. Computers & Operations Research, 31, 2004, 1891-1909.
7. Hejducki Z.: Sprzężenia czasowe w metodach organizacji złożonych procesów budowlanych. Wydawnictwo Politechniki Wrocławskiej, 2000.
8. Holland J.: Adaptation in natural and artificial systems. Univ. of Michigan Press, Ann Arbor, MI, 1975.

9. Ishibuschi H., Murata T.: Scheduling with Fuzzy Duedate and Fuzzy Processing Time, [in:] Scheduling Under Fuzziness, R.Słowiński and M.Hapke (eds), Springer-Verlag, 2000, 113–143.
10. Ishii H.: Fuzzy combinatorial optimization. Japanese Journal of Fuzzy Theory and Systems, vol. 4, no. 1, 1992.
11. Mrozowicz J.: Metody organizacji procesów budowlanych uwzględniające sprzężenia czasowe. Dolnośląskie Wydawnictwo Edukacyjne, Wrocław, 1997
12. Nowicki E., Smutnicki C.: A fast tabu search algorithm for the permutation flow-shop problem. European Journal of Operational Research, 91, 1996, 160–175.
13. Ogbu F., Smith D.: The Application of the Simulated Annealing Algorithm to the Solution of the n/m/Cmax Flowshop Problem. Computers & Operations Research, 17/3, 1990, 243–253.
14. OR Library http://www.ms.ic.ac.uk/info.html
15. Reeves C.: *A* Genetic Algorithm for Flowshop Sequencing. Computers & Operations Research, 22/1, 1995, 5–13.
16. Rogalska M., Bożejko W., Hajducki Z., Wodecki M.:, Harmonogramowanie robót budowlanych z zastosowaniem algorytmu tabu search z rozmytymi czasami wykonywania zadań. Przegląd Budowlany, Nr 7-8, 2009, 76-80.
17. Vondrák J.: Probabilistic methods in combinatorial and stochastic optimization, PhD, MIT, 2005.

Dr hab. Wojciech BOŻEJKO
Instytut Informatyki Automatyki i Robotyki
Politechnika Wrocławska
ul. Janiszewskiego 11/17, 50-372 Wrocław
e-mail: wbo@ict.pwr.wroc.pl

Dr hab. Zdzisław HAJDUCKI
Instytut Budownictwa
Politechnika Wrocławska
Pl. Grunwaldzki 11
e-mail: zdzislaw.hejducki@ict.pwr.wroc.pl

Mgr Paweł RAJBA
Dr hab. Mieczysław WODECKI
Instytut Informatyki
Uniwersytet Wrocławski
ul.Joliot-Curie 50-383 Wrocław
e-mail: pawel.rajba@ii.uni.wroc.pl
mwd@ii.uni.wroc.pl