ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 2002
Seria: AUTOMATYKA z. 136 Nr kol. 1556

Wojciech CHMIEL
Akademia Górniczo-Hutnicza

ALGORYTM TABU SEARCH ZE ZMIENNYM CZASEM PAMIĘCI KRÓTKOTERMINOWEJ*

Streszczenie. W pracy przedstawiono wyniki badań eksperymentalnych procesu optymalizacji zagadnienia przydziału z kwadratową funkcją celu realizowanego za pomocą algorytmu ewolucyjnego, w którym zastosowano operator oparty na Tabu Search. Zastosowano w nim zmienny czas pamięci krótkoterminowej, zależny od wartości oczekiwanej częściowego rozwiązania.

ALGORITHM TABU SERCH WITH FLEXIBLE TIME OF SHORT-MEMORY

Summary. This paper presents results of experimental examination of optimization process realized on the example of quadratic assignment problem, with the aid of evolution algorithm, where an operator based on Tabu Search is used. This algorithm uses flexible time of short-memory depending on conditional expectation of fitness function for partial solution.

1. Wprowadzenie

Problem QAP jest problemem generalizującym wiele zagadnień, których rozwiązania można sformułować w postaci permutacji. Obecnie stanowią one ważną klasę problemów decyzyjnych i są przyczynkiem do wielu prac w zakresie rozwoju metod przybliżonych dla tego zagadnienia ([4], [8], [9]).

Klasycznymi typami problemów permutacyjnych są: kwadratowy problem przypisania, kolorowania grafu, zagadnienia harmonogramowania, zagadnienia komiwojażera, *backboard wiring problem* (minimalizacja długości połączeń pomiędzy obiektami

* Praca sfinansowana przez KBN nr grantu: 11.11.120.227 (AGH, Kraków)

umieszczonymi na płytce elektronicznej) oraz wiele innych ważnych zagadnień optymalizacyjnych.

2. Heurystyki dla problemu QAP

Istnieje wiele, niestety dotąd nie w pełni efektywnych oraz deterministycznych (zależnych od m.in. warunków startowych), metod rozwiązywania problemu *QAP*. Jest to przyczyną ciągłego poszukiwania nowych, bardziej wydajnych algorytmów, pozwalających na znajdowanie rozwiązań przybliżonych dla problemów o dużym rozmiarze.

W przypadku *QAP* o rozmiarze $n>20$ znalezienie dokładnego rozwiązania jest niezwykle trudne ze względu na rozmiar przestrzeni rozwiązań. Ogólnie algorytmy przybliżone można podzielić na: algorytmy konstrukcyjne, algorytmy poprawcze oraz algorytmy bazujące na teorii grafów. W ostatnim czasie coraz częściej są stosowane metaheurystyki odwołujące się praw rządzących teorią ewolucji.

Algorytmy konstrukcyjne iteracyjnie przypisujące obiekty do położeń w bazują na różnego typu heurystykach, m.in. „najbliższego sąsiedztwa". Należą do nich między innymi takie algorytmy, jak: HC66 [15], procedura Gilmore'a [11], PLANET [7], SHARPE [14] i inne. Algorytm konstrukcyjny tworzy rozwiązania bez korzystania z rozwiązań startowych oraz posiada względnie krótki czas obliczeń. Niestety, tego typu algorytmy nie dostarczają dobrej jakości rozwiązań [20].

Algorytmy poprawcze w sposób systematyczny dokonują wymiany obiektów przypisanych do położeń, co prowadzi do iteracyjnej ewaluacji rezultatu. Jakość rozwiązań otrzymywanych na podstawie tych algorytmów często zależy od użytych rozwiązań startowych. Jedną z najbardziej popularnych metod jest *SDPIP* („*Steepest Descent Pairwise Interchange Procedure*" [11]), będąca jednocześnie podstawową procedurą w pakiecie CRAFT [2]. W algorytmie tym dany jest pewien zestaw rozwiązań wstępnych. Na ich podstawie algorytm dokonuje zmiany położeń par obiektów wybierając te, które powodują największą redukcję kosztów. H63 [15] jest innym algorytmem poprawczym, opartym na tzw. tablicy ruchów pożądanych („*move desirability table*"). Algorytm ten został przebadany oraz poprawiony przez Hillera i Francisa [15] oraz White'a [10]. Spośród innych procedur poprawczych warto wspomnieć jeszcze o heurystykach opartych na „*biased sampling*"

Nugeta oraz o VERBES [2], korzystających z procedur posiadających trzy lub cztery różne metody zmiany położeń elementów.

Algorytmy oparte na teorii grafów są podejściem pozwalającym na obliczenie dolnego oraz górnego ograniczenia rozwiązań dopuszczalnych na podstawie specyficznego problemu dualnego do *QAP* [10]. Jednym z pierwszych podejść bazujących na teorii grafów były prace Seppanena i Morra [19] we wczesnych latach 70.

Algorytmy ewolucyjne powstały z relaksacji założeń i wprowadzanie „innowacji" w klasycznych algorytmach genetycznych *(AG)*. Zostały one zaproponowane w 1975 roku przez Hollanda jako ogólnego przeznaczenia metaheurystyki odwołujące się do praw ewolucji [16]. Dzięki swoim cechom dają one możliwość przeszukiwania wielkich przestrzeni rozwiązań, wobec których klasyczne metody optymalizacji okazują się bezradne.

Opisany w pracy algorytm należy do procedur *modGA*, zmodyfikowanych w stosunku do klasycznego AG. Został on zaproponowany przez Michalewicza w monografii [17] i należy do klasy algorytmów *Steady State GA*.

Połączenie możliwości algorytmów ewolucyjnych, a szczególnie umiejętności przeszukiwania dużych „obszarów" przestrzeni rozwiązań, z lokalnymi procedurami popraw, pozwalającymi na dokładne przeglądnięcie otoczeń wcześniej wygenerowanego rozwiązania, w świetle uzyskanych wyników wydaje się niezwykle obiecujące. Jako procedurę lokalnej optymalizacji można zastosować algorytm Tabu Search, w którym wprowadzono zmienny czas zabronień krótkoterminowych, których wartość zależy od „jakości" dokonanej modyfikacji rozwiązania startowego.

Nakładanie zabronień ma na celu nie tylko przeciwdziałanie powrotowi algorytmu do poprzednio przeglądniętych rozwiązań, ale dodatkowo ma pozwolić na preferowanie tych obszarów, które są perspektywiczne, tj. wartość oczekiwana częściowo ustalonego rozwiązania jest najbardziej obiecująca.

Poniższe opracowanie prezentuje rezultaty poszukiwań przybliżonych algorytmów ewolucyjnych i możliwość ich połączenia z procedurami lokalnych popraw, takimi jak *Tabu Search* ze zmiennym czasem tabu krótkoterminowego. Niniejsza publikacja jest pierwszą, w której opisano algorytm *Tabu Search* ze zmiennym czasem zabronień opartym na warunkowej wartości oczekiwanej funkcji celu.

3. Zagadnienie QAP

Zagadnienie *QAP* sformalizujemy w następujący sposób. Dany jest zbiór $N=\{1,...,n\}$ i dwie $(n\times n)$ wymiarowe macierze $A=[a_{i,k}]$, $B=[b_{j,l}]$. Należy znaleźć permutację $\pi=(\pi(1),...,\pi(n))$ elementów zbioru N, która minimalizuje funkcje celu $\phi(\pi)$ o postaci:

$$\phi(\pi)=\sum_{i=1}^{n}\sum_{k=1}^{n} a_{i,k}\, b_{\pi(i),\pi(k)} \tag{1}$$

W terminologii alokacji obiektów zbiór N jest zbiorem numerów obiektów, a $\pi(i)\in N$, $i=1,...,n$ określa numer obiektu przydzielonego do pozycji i. Macierz A jest wtedy macierzą odległości pomiędzy pozycjami rozmieszczenia obiektów, podczas gdy macierz B opisuje powiązania występujące pomiędzy obiektami. Natomiast funkcja celu $\phi(\pi), \pi\in\Pi$, gdzie Π jest zbiorem permutacji zbioru N, określa globalny koszt eksploatacji systemu.

4. Warunkowa wartość oczekiwana dla problemu QAP

Niech $m\geq k$ oraz $[k, m]$ określają zbiór $\{k,k+1,...,m\}$. Niech m będzie liczbą całkowitą dodatnią i niech D będzie podzbiorem $[1, m]^n=\{(x_1,x_2,..,x_n), x_i\in[1,m], i = 1, ..,n\}$.
Niech $f: [1, m]^n \rightarrow \Re$. Rozważmy następujący problem optymalizacji:

$$\min\{f(x_1,x_2,..,x_n), (x_1,x_2,..,x_n)\in D\} \tag{2}$$

$x^0_{i_1},...,x^0_{i_k}\in[1,...,m]$ gdzie $1\leq i_s\neq i_t\leq n, 1\leq s<t\leq k$.
Wtedy $D(x^0_{i_1},...,x^0_{i_k})=\{(x_1,...,x_n)\in D: x_{i_1}=x^0_{i_1},...,x_{i_k}=x^0_{i_k}\}$ Niech dla k = 0, $D(x^0_{i_1},...,x^0_{i_k})=D$.
Rozważmy D jako przestrzeń prawdopodobieństwa poprzez przypisanie każdemu elementowi $x^0=(x^0_1,...,x^0_n)$ przestrzeni D nieujemnej wartości – prawdopodobieństwa $P(x^0)$ takiego, że $\sum_{x^0\in D}P(x^0)=1$. Przy takiej interpretacji f jest zmienną losową, dzięki czemu możemy posługiwać się wartością oczekiwaną Ef funkcji f. Taka interpretacja D oraz f umożliwia zastosowanie metody derandomizacji warunkowego prawdopodobieństwa ([1], [23]). Dodatkowo, zakłada się równomierny rozkład prawdopodobieństwa $P(x) = 1/|D|$ dla każdego $x\in D$.

Problem QAP może być sformułowany jako problem typu (2): znajdź $\min\{\phi(\pi(1), .., \pi(n): \pi\in S_n\}$, gdzie S_n określa grupę symetryczną permutacji. Zakładamy, że $P(\pi\in S_n)=1/n!$.

W takim przypadku możemy, bez utraty ogólności, określić warunkową wartość oczekiwaną $E(\phi/\pi(s_1)=c(s_1),\ldots,\pi(s_i)=c(s_i),\ldots,\pi(s_k)=c(s_k))$, gdzie $0\le s_1,\ldots, s_i,\ldots, s_k\le n$ oraz $c(s_1),\ldots,c(s_i),\ldots,c(s_k)$ są różnymi wartościami z zakresu $[1,\ldots,n]$.

Określmy zbiór liczb naturalnych zakresu 1 do n jako $L=[1,\ldots,n]$, zbiór $M=[1,\ldots,n]\setminus\{c(s_1),\ldots, c(s_i),\ldots, c(s_k)\}$, i zbiór $H=\{s_1,\ldots, s_i,\ldots, s_k\}$. W takim przypadku warunkowa wartość oczekiwana dla rozwiązania problemu *QAP* przy k ustalonych pozycjach, korzystając z własności liniowości wartości oczekiwanej, wyniesie:

$$E(\phi/\pi(s_1)=c(s_1),\ldots,\pi(s_i)=c(s_i),\ldots,\pi(s_k)=c(s_k))=\sum_{i\in H}\sum_{j\in H}a_{ij}b_{\pi(i)\pi(j)}+$$
$$\frac{1}{n-k}\sum_{i\in H}\sum_{j\in L\setminus H}a_{ij}\sum_{m\in M}b_{\pi(i)m}+\frac{1}{n-k}\sum_{i\in L\setminus H}\sum_{j\in H}a_{ij}\sum_{m\in M}b_{m\pi(j)}+ \tag{3}$$
$$\frac{1}{n-k}\sum_{i\in L\setminus H}a_{ii}\sum_{m\in M}b_{mm}+\frac{1}{n-k}\frac{1}{n-k-1}\sum\{a_{ij}:j\neq i\in L\setminus H\}\sum\{b_{ij}:j\neq i\in M\}$$

Wielkość warunkowej wartości oczekiwanej można obliczyć w czasie $O(n^2)$. Określmy zbiór $T=\{(i,j): i\neq j,\ i,j\in L\}$. Na podstawie równania (3) można dowieść, że wartość oczekiwana dla całego problemu QAP wynosi:

$$E_\phi=\frac{1}{n(n-1)}\sum_{(i,j)\in T}a_{ij}\sum_{(i,j)\in L}b_{ij}+\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}b_{jj} \tag{4}$$

Gutin i Yeo w [21] udowodnili następujące twierdzenie dla dowolnego, wystarczająco dużego n, a dotyczące liczebności rozwiązań o wartości funkcji celu większej niż E_ϕ dla problemu *QAP*:

Twierdzenie 1

Niech $\beta>1$ będzie dowolne. Wtedy dla wystarczająco dużego n istnieje co najmniej $n!/\beta^n$ permutacji takich, że $\phi(\omega)\ge E_\phi$

Powyższe twierdzenie zostało udowodnione na podstawie prac Zassenhausa opublikowanych w 1935 roku, dotyczących własności grup *sharply 2-transitive*, oraz twierdzenia opublikowanego w *„Permutation Group"* D.S. Passmana (Tw. 20.3 [22]), dotyczącego przypadku, w którym n jest potęgą liczby pierwszej.

5. Algorytm Tabu Search

Metoda Tabu Search została zaproponowana przez Glovera w 1989 roku [12], [13]. Algorytmy oparte na niej należą do najskuteczniejszych algorytmów popraw. Generuje ona

dla rozwiązania bazowego π^t sąsiedztwo $N(V,\pi^t)\subset\Pi$, $\pi^t\notin N(V,\pi^t)$. $V(\pi)$ jest zbiorem ruchów v, generującym otoczenie rozwiązania π, spełniających warunek: $\pi_v \neq \pi_{v'}$ dla każdego v, $v'\in V(\pi)$. Z sąsiedztwa wybierane jest najlepsze rozwiązanie π^*, które staje się rozwiązaniem bazowym dla następnej iteracji. Rozwiązanie π^* porównywane jest pod względem wartości funkcji celu z najlepszym rozwiązaniem π^{TS} znalezionym w poprzednich iteracjach oraz zapamiętywane, jeśli $\phi(\pi^*)$ jest lepsze od $\phi(\pi^{TS})$.

W celu zapobieżenia powrotom do rozwiązań już przebadanych wprowadza się mechanizm zabronień (tabu). W wyniku jego zastosowania pewne rozwiązania z sąsiedztwa uważa się za zabronione - są one pomijane podczas wyznaczania rozwiązania π^*. W zależności od czasu jego działania może mieć swoje odmiany: długoterminową, średnioterminową i krótkoterminową. Dodatkowo wprowadza się kryterium aspiracji, pozwalające w szczególnych przypadkach odrzucić ograniczenia nakładane przez mechanizm tabu.

Poniżej przedstawiamy algorytm typu Tabu Search, nazwany przez nas *CS_TS (Conditional Expectation Tabu Search)*, ze zmiennym czasem pamięci krótkoterminowej. W celu wyznaczenia czasu zabronień *CS_TS* korzysta z procedury *CAQAP*. Algorytm *CS_TS* jest wywoływany przez algorytm *ETABGen* jako procedura optymalizacji lokalnej dedykowanej dla problemu *QAP*.

Algorytm CS_TS (Conditional Expectation Tabu Search)
Oznaczenia:

- $\pi(\pi(i), \pi(j), i, j)$- rozwiązanie z otoczenia rozwiązania $\pi = (\pi(1), \pi(2), ..., \pi(i)..., \pi(j), ..., \pi(n))$, otrzymane w wyniku przestawienia elementów $\pi(i)$, $\pi(j)$ znajdujących się na miejscach i oraz j, tj. $\pi' = (\pi(1), \pi(2), ..., \pi(j)..., \pi(i),..., \pi(n))$;
- CA- zmienna logiczna określająca czy zastosowano kryterium aspiracji;
- α - współczynnik różnicowania poszukiwania;
- GLT(a, b, i, j) – pozostały czas tabu pary a, b na miejscach i, j (pamięć krótkoterminowa);
- DLT(a, b, i, j) – liczba przestawień elementów a, b pomiędzy miejscami i, j w procesie poszukiwania lepszego rozwiązania (pamięć długoterminowa);
- T – podstawowy czas tabu krótkoterminowego;
- K – liczba iteracji procesu poszukiwania algorytmu;
- $N(v,\pi)$ – otoczenie powstałe w wyniku wszystkich możliwych przestawień miejscami dwóch elementów permutacji π;
- π_{min} – aktualne najlepsze rozwiązanie;
- ϕ_{min} – wartość funkcji celu dla aktualne najlepszego rozwiązania;
- π_{start} – rozwiązanie startowe.

Krok 1. Podstaw $\pi := \pi_{min} := \pi_{start}$, oraz $\phi := \phi_{min} := \phi(\pi_{start})$, $CA = FALSE$.

Krok 2. Dla $k=1$, do K wykonaj:

a) Wyznacz najlepsze rozwiązanie w sąsiedztwie $N(v,\pi)$:

$$\pi(\pi(i)^*,\pi(j)^*,i^*,j^*) = \arg\min \{\phi(\pi(\pi(i),\pi(j),i,j))+\alpha*DLT(\pi(i),\pi(j),i,j)/k: GLT(\pi(i),\pi(j),i,j)=0\},$$

$$\pi(\pi(i)',\pi(j)',i',j') = \arg\min \{\phi(\pi(\pi(i),\pi(j),i,j)): GLT(\pi(i),\pi(j),i,j)>0\}.$$

b) Wykonaj ruch podstawiając $\pi := \pi(\pi(i)^*,\pi(j)^*,i^*,j^*)$, $\phi := \phi(\pi)$.

c) Popraw najlepsze rozwiązanie:

Jeżeli $\phi_{min} > \phi(\pi)$, to podstaw $\phi_{min} := \phi(\pi)$, $\pi_{min} := \pi$ i przejdź do e). W przeciwnym przypadku przejdź do d).

d) Zastosuj kryterium aspiracji.

Jeżeli $\phi_{min} > \phi(\pi(\pi(i)',\pi(j)',i',j'))$, to podstaw $\pi := \pi(\pi(i)',\pi(j)',i',j')$, $\pi_{min} := \pi$, $\phi_{min} := \phi(\pi)$, $CA = TRUE$.

e) Korekta stanu pamięci:

Dla $\forall(\pi(i),\pi(j),i,j)$ podstaw $GLT(\pi(i),\pi(j),i,j) := \max\{0, GLT(\pi(i),\pi(j),i,j)-1\}$.

Jeżeli $CA=TRUE$ (zastosowano kryterium aspiracji), to podstaw:

$GLT(\pi(i)',\pi(j)',i',j') := CAQAP(\pi(i)',\pi(j)',i',j',T)$

$DLT(\pi(i)',\pi(j)',i',j') := DLT(\pi(i)',\pi(j)',i',j')+1$,

$CA=FALSE$.

W przeciwnym przypadku, tj. $CA = FALSE$, podstaw:

$GLT(\pi(i)^*,\pi(j)^*,i^*,j^*) := CAQAP(\pi(i)^*,\pi(j)^*,i^*,j^*,T)$

$DLT(\pi(i)^*,\pi(j)^*,i^*,j^*) := DLT(\pi(i)^*,\pi(j)^*,i^*,j^*)+1$.

Procedura *CAQAP* wyznacza wartość czasu tabu na podstawie warunkowych wartości oczekiwanych, przy ustalonych wartościach $\pi(i),\pi(j)$ na pozycjach i, j permutacji π

Procedura $CAQAP(\pi(i), \pi(j), i, j, T)$

- *EX* – wartość oczekiwana rozwiązania dla całego problemu *QAP*, obliczana na podstawie równania (4).
- *maxEX* – maksymalna wielkość wartości oczekiwanej rozwiązania uzyskana w dotychczasowym procesie poszukiwania;
- *minEX* – minimalna wielkość wartości oczekiwanej rozwiązania uzyskana w dotychczasowym procesie poszukiwania;
- *pEx* – wartość oczekiwana dla problemu QAP przy ustalonych wartościach $\pi(i)$, $\pi(j)$ na miejscach i oraz j, obliczana na podstawie równania (3).
- T – podstawowa wartość czasu tabu;
- $\beta\in(0,1]$ – współczynnik wzmocnienia;
- *TTime* – wartość czasu tabu;

$$TTime = \begin{cases} T*\left(\tanh\left(\beta*\dfrac{EX - pEX}{maxEX - EX}\right)+1\right) & EX \le pEX \\ T*\left(\tanh\left(\beta*\dfrac{EX - pEX}{EX - minEX}\right)+1\right) & EX > pEX \end{cases}$$

6. Algorytm ewolucyjny

Algorytmem nadrzędnym dla procedury *CS_TS* jest algorytm ewolucyjny *ETABGen*, wykorzystujący dwa operatory unarne i dwa operatory krzyżowania generujące rozwiązania dopuszczalne problemu *QAP*, spośród których w sposób losowy jest wybierane rozwiązanie startowe poprawiane przez algorytm CS_TS.

W przeprowadzonych badaniach komputerowych zastosowano następujący unarny operator genetyczny:

1. Operator mutacji RM (random mutation [5])

Dla rozwiązań, w których ważna jest kolejność elementów, operatory krzyżowania są bardziej skomplikowane, ponieważ klasyczne (ślepe) krzyżowanie może prowadzić do wyznaczenia rozwiązań niedopuszczalnych. Tak więc, należy zastosować ściśle określone heurystyczne operatory krzyżowania. W eksperymentach komputerowych stosowano dwa operatory krzyżowania:

2. Operator PMX (partially matched crossover [5])

3. Operator OX (order crossover [5])

Jako jeden z operatorów zastosowano algorytm optymalizacji lokalnej *CS_TS*, który z formalnego punktu widzenia jest operatorem unarnym.

4. Algorytm CS_TS (Conditional Expectation Tabu Search)

Algorytm ETABGen:

Aby wyznaczyć rozwiązanie przybliżone π_{approx} wykonaj następującą procedurę:

Krok 1. Wyznaczenie populacji początkowej

Wygeneruj, w sposób losowy, M permutacji π oraz określ funkcje przystosowania $f(\pi)$ dla każdej z nich. Z wygenerowanych rozwiązań utwórz populację początkową P o rozmiarze M, uporządkowaną wg wartości funkcji przystosowania $f(\pi)$, tzn. permutacja nr 1 jest najlepsza ($\pi_{best} = \arg\min \{f(\pi) : \pi \in P\}$) i ostatnia permutacja nr M jest najgorsza. Wyzeruj tablice *GLT* oraz *DLT*.

Krok 2. Wybór operatora genetycznego

Wylosuj typ operatora genetycznego ze zbioru {*RM, OX, PMX, CS_TS* }, gdzie każdy z operatorów jest losowany z prawdopodobieństwem $p_i \geq 0$, $i \in \{RM, OX, PMX, CS_TS\}$, przy czym $p_{RM} + p_{LO} + p_{OX} + p_{PMX} + p_{CS_TS} = 1$.

Krok 3. Wybór rodziców/rodzica

Dla wybranego operatora wylosuj, zgodnie z rozkładem równomiernym, ze zbioru P, zależnie od typu operatora, jedno (w przypadku operatora unarnego) lub dwa (w przypadku operatora krzyżowania) rozwiązania, które nazywamy rodzicami.

Krok 4. Generowanie potomków

Za pomocą wylosowanego operatora genetycznego dokonaj modyfikacji rozwiązania-rodzica/rozwiązań-rodziców i wyznacz w ten sposób rozwiązanie-potomka/rozwiązania-potomków.

Krok 5. Poprawa rozwiązań w zbiorze P

Dla każdego potomka oblicz wartość funkcji przystosowania. Jeżeli ta wartość jest lepsza od wartości najgorszego rozwiązania zbioru *P*, to umieść takiego potomka w zbiorze *P*, usuwając zarazem z tego zbioru rozwiązanie najgorsze.

Krok 6. Warunek STOP-u.

Jeżeli wygenerowano zadaną liczbę *L* potomków, to STOP zwróć najlepsze rozwiązanie zbioru *P* - $\pi_{approx} = arg\ min\ \{\phi(\pi): \pi \in P(O)\}$. W przeciwnym przypadku idź do kroku 2.

Wielkości, takie jak rozmiar populacji M, liczba iteracji algorytmu *L*, prawdopodobieństwa selekcji operatorów *RM* - p_{RM}, *LO* - p_{LO}, OX - p_{OX}, *PMX* - p_{PMX}, *CS_TS* – p_{CS_TS} są parametrami algorytmu *ETABGen*.

7. Wyniki eksperymentu

Na podstawie algorytmu *ETABGen* wykonano eksperymentalny program komputerowy, zakodowany w języku C++ pod systemem W2000, zaimplementowany dla zagadnienia *QAP*. Obliczenia wykonano na procesorze Pentium 500MHz. W tabeli 1 przedstawiono wyniki badań eksperymentalnych dla zagadnienia *QAP*, zbioru 37 testów o rozmiarze n=26-64, zaczerpniętych z biblioteki QAPLIB-A [3]. Wszystkie eksperymenty komputerowe wykonane w oparciu o algorytm *ETABGen* korzystają z następującego zbioru parametrów: L=10000 iteracji, M = 100 (rozmiar populacji), zbiór operatorów *H={RM, CS_TS, OX, PMX}*. Przy ustalonych powyżej parametrach wykonano serie obliczeń dla operatorów ze zbioru *H*. Przyjęto $p_{CS_TS} = 0.5$, $p_{OX} = 0.2$, $p_{PMX} = 0.2$, $p_{RM} = 0.1$. Dla algorytmu *CS_TS* przyjęto następujące wartości parametrów: $\alpha=10$, $T=50$, $\beta=0.8$. Otrzymane wyniki porównano z przypadkiem, w którym w algorytmie *ETABGen* wyeliminowano procedurę określającą czas tabu w oparciu o wartości oczekiwane dla rozwiązania częściowego i przyjęto stały czas tabu $T=50$. Przyjęto oznaczenia:

$f_{QAPLIB\text{-}A}$ – najlepsza znana wartość funkcji przystosowania zaczerpnięta z biblioteki QAPLIB-A,

f_{start} – wartość funkcji przystosowania najlepszego rozwiązania w populacji startowej $\pi_{start} = arg\ min\ \{f(\pi): \pi \in P(O)\}$,

f_{ap} – wartość funkcji przystosowania najlepszego znalezionego rozwiązania dla π_{approx},

$E = 100\% * (f_{ap} - f_{QAPLIB\text{-}A}) / f_{QAPLIB\text{-}A}$,

L_{ap} – czas w sekundach, po którym zostało znalezione najlepsze rozwiązanie π_{approx}.
TT_{flx} – wyniki dla zmiennego czasu tabu,
TT_{cnst} – wyniki dla stałego czasu tabu.

Tablica 1

Nazwa	$f_{QAPLIB-A}$	F_{start}	TT_{flx}			TT_{const}		
			f_{ap}	E [%]	Lap	f_{ap}	E [%]	Lap
BUR26A	5426670	5666080	5428220	0,02856	9,78	5431890	0,09619	53,11
BUR26B	3817852	4008670	3818480	0,01645	25,16	3825070	0,18906	24,71
BUR26C	5426795	5735520	5426800	0,00009	43,61	5426990	0,00359	94,14
BUR26D	3821228	4079800	3821260	0,00084	19,88	3821250	0,00058	13,4
BUR26E	5386879	5667490	5386880	0,00002	15,32	5386880	0,00002	22,8
BUR26F	3782044	4030400	3782250	0,00545	11,09	3782040	-0,00011	8,96
CHR22A	6156	10508	6324	2,72904	46,36	6396	3,89864	15,6
CHR22B	6194	10940	6406	3,42267	26,48	6308	1,84049	17,3
CHR25A	3796	14076	4440	16,96523	34,6	4070	7,21812	33,94
ESC32A	130	370	130	0,00000	48,45	142	9,23077	19,33
ESC32B	160	408	188	17,50000	6,1	168	5,00000	14,23
ESC32C	642	752	642	0,00000	3,29	642	0,00000	3,35
ESC32D	200	312	200	0,00000	11,81	200	0,00000	9,18
ESC32E	2	6	2	0,00000	1	2	0,00000	1
ESC32F	2	24	2	0,00000	3,18	2	0,00000	3,24
ESC32G	6	12	6	0,00000	3,24	6	0,00000	3,3
ESC32H	438	630	438	0,00000	4,45	438	0,00000	74,21
ESC64A	116	226	116	0,00000	15,38	116	0,00000	5,05
KRA30A	88900	124630	90850	2,19348	73,88	91420	2,83465	9,72
KRA30B	91420	127200	91910	0,53599	114,68	91910	0,53599	80,46
LIPA30A	13178	13726	13356	1,35074	44,38	13178	0,00000	6,2
LIPA30B	151426	191360	151426	0,00000	7,58	151426	0,00000	5,87
LIPA40A	31538	32525	31863	1,03050	436,5	31863	1,03050	241
LIPA40B	476581	604619	476581	0,00000	17,47	476581	0,00000	31,2
LIPA50A	62093	63823	62721	1,01139	1068,7	62674	0,93569	802,4
LIPA50B	1210244	1524310	1210240	-0,00033	391,73	1412500	16,71200	765,7
LIPA60A	107218	109816	108155	0,87392	1683,9	108144	0,86366	2472
LIPA60B	2520135	3219390	2984630	18,43135	2372,2	2979790	18,23930	1425
SKO42	15812	19294	15882	0,44270	548,43	15982	1,07513	373,5
SKO49	23386	28164	23598	0,90653	1098,7	23510	0,53023	872,6
SKO56	34458	41110	34642	0,53398	989,48	34512	0,15671	1249
STE36A	9526	41056	9678	1,59563	144,02	9816	3,04430	332,1
STE36B	15852	18426	15980	0,80747	222	15980	0,80747	61,9
STE36C	8239110	51728	8239110	0,00000	214,43	8356160	1,42066	267,4
THO30	149936	15616200	150704	0,51222	182,79	150606	0,44686	40,7
THO40	240516	309522	243004	1,03444	581,06	240638	0,05072	524
WIL50	48816	54250	48868	0,10652	1223,7	48998	0,37283	1387

8. Interpretacja wyników

Porównanie kolumn zawierających wyniki dla zmiennego oraz stałego czasu tabu zdaje się wskazywać, że ten pierwszy generuje statystycznie lepsze rozwiązania nie wydłużając czasu obliczeń. Podczas swojego działania procedura testując warunkowe wartości oczekiwane częściowego rozwiązania wpływa na preferowanie tych rozwiązań, które z punktu widzenia problemu *QAP* są bardziej perspektywiczne od innych. Tablica

zawierająca czasy tabu jest dodatkowym źródłem informacji dla algorytmu o problemie uzyskanym podczas procesu optymalizacji. Potwierdzenie pozytywnego wpływu zmiennego czasu tabu na wartości otrzymanych rozwiązań otwiera nowe możliwości dla algorytmów przybliżonych dedykowanych dla zagadnienia *QAP*, które powstaną w przyszłości na bazie metod statystycznych stosowanych w badaniach własności grafów.

LITERATURA

1. Alon, N., Spencer J.: The probabilistic method. Willey, New York 1992.
2. Burkard, R. E., Stratmann, K.H.: Numerical investigations on quadratic assignment problem. Naval Research Logistics Quarterly 25 (1978), s. 129-148.
3. Burkard R.E., Karisch S.E., Rendl F.: QAPLIB-A Quadratic Assignment Problem Library. European Journal of Operational Resarch, 55, 1991, s.115-119.
4. Brukard R.E., Stratmann K.H.: Numerical investigation on quadratic assignment problems. Naval Research Logistics Quaterly, vol.25, 1978, s.129-147.
5. Chmiel W.: Algorytm ewolucyjny z ograniczonym wyborem operatorów genetycznych. Zeszyty Naukowe Pol. Śl., z.125, Gliwice 1998, s.125-134.
6. Chmiel W., Kadłuczka P.: Algorytm hybrydowy z długoterminową pamięcią częstotliwościową. University of Mining and Metallurgy Press, Automatyka vol.3, Kraków 1999, s. 59-71.
7. Deisenroth, M.P., Apple J. M.: A computerized Plant layouts analysis and evaluation techniques. Technical Paper, Annual AIIE conference, Norcross, GA, 1972.
8. Filipowicz B., Wala K.: Algorytmy optymalizacji kwadratowego zagadnienia przydziału. Kwartalnik Elektrotechnki, z.1, 1992, Wydawnictwo AGH w Krakowie.
9. Finke G., Burkard R.E, Rendl F.: Quadratic assignments problems. Annals of Discrete Mathematics, vol.31, North-Holland, 1987, s. 61 -82.
10. Francis L.R., White J. A.: Facility layout and location: An analytical approach. Pretence-Hall, Englewood Hall, NJ, 1974.
11. Gilmore P., L.: Optimal and suboptimal algorithms for QAP. SIAM Journal of Applied Mathematics 10 (1962) 305-313.
12. Glover, F.: Tabu Search, Part I. ORSA Journal of Computing 1989, 1, 190-206.
13. Glover, F.: Tabu Search, Part II. ORSA Journal of Computing 1990, 2, 4-32.
14. Hassan, M. M D., Hogg G.L., Smith, D.,R.: SHAPE: A construction algorithm for area placement evaluation. Int. Journal of Production Research 24 (1986), 1283-1295.
15. Hillier, F.S., Connors, M.M.: Quadratic Assignment Problem algorithms and the location of indivisible facilities. Management Science 13 (1966) 42-57.
16. Holland J.: Adaptation in natural and artificial systems. Univ. of Michigan Press, Ann Arbor, MI, 1975.
17. Michalewicz Z.: Genetic algorithms + data structures = evolution programs. Springer-Verlag, Berlin 1992.

18. Moor J. M.: Computer aided facilities design: an international survey. International Journal of Production Research 12/1 (1974).
19. Seppanen, J.J., Moore, J.M.: Facilities Planning with graph theory. Management Science 17/4 (1970) 185-187.
20. Sherali, H.D., Rajgopal P.: A flexible polynomial-time, construction and improvement heuristics for the quadratic assignment problem. Computers & Operation Research 13 (1986) 587-600.
21. Gutin, G., Yeo A.: Polynomial approximation algorithms for TSP and the QAP with factorial domination number.
22. Passman D.S.: Permutation Group. Benjamin Inc., NY 1968 36.
23. Raghavan P.: Probabilistic construction of deterministic algorithms: approximating packing integer programs. J. Computer and System 37 (1988) 130-143.
24. Wala K., Chmiel W.: Evolution Algorithm for Quadratic Assignment Problem. University of Mining and Metallurgy Press, Elektrotechnika 1,1, Kraków, 1997, s. 409-414.

Recenzent: Prof. dr hab. inż. Adam Janiak

Abstract

Permutation problems are an important class of decision problems in the combinatorial optimization domain. Classical instances of permutation problems include quadratic assignment problem *(QAP)*, graph coloring, production scheduling problems, as well as variety of design problems. The paper presents the results of computer investigation of approximate algorithm, realized evolution artificial search process, for *QAP* as a hard instance of permutation problems and, on the other hand, there is a rich Quadratic Assignment Problem Library, called QAPLIB-A, with test task of this problem for approximate algorithm examinations. Approximate algorithm uses local search operator, developed on the base of Taboo Search with feasible time of short memory. The basic idea of Taboo Search is to replace an element X with element $Y \in N(X) \setminus \{X\}$ such that Y is feasible and $f(Y)$ is minimum among all such feasible elements. It might happen that $f(Y)<f(X)$, so we can escape from locally optimal solution X by means of this method. However, having replaced X by Y in this case, it is highly likely that next step would be to replace Y by X. This is not desirable, since the algorithm would enter in infinite loop from which cannot escape. Hence, we need to find a way to avoid this situation and other similar problems such cycling. This is accomplished by means of tabu list. By assigning to each solution nonnegative weight (probability - which sum all is equal 1), we can consider domain of *QAP* problem as probability space, where f (fitness function) is random variable. In such situation, we can deal with the expectation Ef of f as well as its conditional expectation. Such interpretation of domain of *QAP* and f allows us to apply the derandomization method of conditional probabilities. Using this conditional expectation of fitness function, we can determine flexible tabu time.