ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1992
Seria: AUTOMATYKA z. 109 Nr kol. 1175

Wojciech Chmiel, Konrad Wala
Akademia Górniczo-Hutnicza

DYNAMIKA ALGORYTMU GENETYCZNEGO NA PRZYKŁADZIE PROBLEMU QAP
GENETIC ALGORITHM DYNAMICS ON THE EXAMPLE OF THE QAP PROBLEM
DIE DYNAMIK EINES GENETISCHEN ALGORITHMUS AUF DEN BEISPIEL QAP PROBLEMS

Streszczenie: W pracy przedstawiono wyniki badań eksperymentalnych procesu optymalizacji realizowanego za pomocą algorytmu genetycznego, na przykładzie kwadratowego zagadnienia przydziału. Celem badań jest określenie wpływu parametrów algorytmu na efektywność procesu optymalizacji.

Summary: This paper presents the results of experimental examination of the optimization process, realized with the aid of genetic algorithm, on an example of the quadratic assignment problem. The purpose of these investigations is a determination of the algorithm parameters influence on the optimization process efficiency.

Zusammenfassung: In dieser Forschungsarbeit werden die Ergebnisse der Experimentaluntersuchungen des Optimierungsvorganges vorgestellt, der mit Hilfe eines genetischen Algorithmus verwirklichen wird, auf den Beispiel des Quadratproblems der Zuteilung. Das Ziel der Untersuchungen ist die Bestimmung des Einflusses der Kennwerten des Algorithmus auf die Effektivität des Optimierungsvorganges.

1. WROWADZENIE

Problem przydziału przy kwadratowym wskaźniku jakości QAP (ang.Qadratic Assigment Problem) jest zagadnieniem optymalizacji kombinatorycznej należącym do klasy zagadnień NP-trudnych. Problem ten występuje w licznych zagadnieniach praktycznych w takich dziedzinach jak elektronika, informatyka, ekonomia, ergonomia, architektura itd. Przegląd ważniejszych zastosowań i metod rozwiązywania zagadnienia QAP można np. znaleźć w [4],[6].

Zagadnienie QAP sformalizujemy w następujący sposób. Dany jest zbiór $\mathbb{N}=\{1,\dots,n\}$ i dwie $(n\times n)$ wymiarowe macierze $A=[a_{ik}]$, $B=[b_{jl}]$, należy znaleźć permutację $\Pi=(\Pi(1),\dots,\Pi(n))$ zbioru $\mathbb{N}$, która minimalizuje funkcję celu $f(\Pi)$ w postaci :

$$f(\Pi^*) = \min_{\Pi\in\Pi} f(\Pi) = \sum_{i=1}^{n}\sum_{k=1}^{n} a_{ik} * b_{\Pi(i)\Pi(k)}$$

W terminologii alokacji obiektów zbiór $\mathbb{N}$ opisuje zbiór pozycji alokacji, a $\Pi(i)\in\mathbb{N}$, $i=1,\dots,n$, określa numer pozycji, do której jest przydzielony i-ty obiekt. Macierz A jest wtedy macierzą odległości pomiędzy pozycjami rozmieszczenia obiektów, podczas gdy macierz B opisuje powiązania występujące pomiędzy obiektami. Natomiast funkcja celu $f(\Pi)$, $\Pi\in\Pi$, gdzie Π jest zbiorem n-elementowych dopuszczalnych permutacji, określa koszt globalny eksploatacji systemu.

W zagadnieniu QAP, podobnie jak w standardowym zagadnieniu komiwojażera, poszukujemy optymalnej permutacji $\Pi^* \in \Pi$, ale jest ono obliczeniowo trudniejszym problemem. Warto odnotować fakt, że obecnie metodami ścisłymi można rozwiązywać zagadnienie komiwojażera stosunkowo dużych rozmiarów (n może wynosić kilkaset), natomiast w przypadku QAP uzyskanie rozwiązania dla n=20 jest w praktyce niemożliwe, zaś dla algorytmów przybliżonych dla n>30 stanowi już poważny problem obliczeniowy.

Jest to powodem licznych prac (na przykład [2],[3],[5]) w zakresie rozwoju metod przybliżonych, dla zagadnień QAP większego rozmiaru (n>15). Zwykle, w celu uzyskania zadowalającego użytkownika rozwiązania, stosuje się dwuetapowy proces optymalizacji (por. na przykład [4]) polegający na generacji za pomocą procedur inauguracyjnych, w pierwszym etapie, rozwiązań startowych dla etapu drugiego. Procedury inauguracyjne wyznaczają zwykle rozwiązania startowe za pomocą reguł szeregowania listowego lub na drodze losowania. Drugi etap optymalizacji polega na zastosowaniu iteracyjnych procedur poprawiania rozwiązań takich jak algorytmy optymalizacji lokalnej, termodynamiczne i, ostatnio intensywnie badane, algorytmy genetyczne. W pracy zaprezentowano wyniki badań dynamiki algorytmu genetycznego w zakresie poprawiania rozwiązań - permutacji zagadnienia QAP.

2. CECHY ALGORYTMU GENETYCZNEGO

Zaproponowany w 1975 roku przez Hollanda [9] algorytm maszynowego uczenia się rozwiązywania zagadnień formalizowanych w postaci binarnej został nazwany algorytmem genetycznym. W następnych latach algorytm genetyczny zimplementowano do rozwiązywania permutacyjnych zagadnień optymalizacji, w tym i zagadnienia QAP (por. [2], [8], [10], [12], [13]).

Badany w pracy algorytm genetyczny rozwiązywania permutacyjnych zagadnień optymalizacji uczy się rozwiązywać konkretne zagadnienia QAP (instancje), wykonując szereg eksperymentów obliczeniowych i obserwując wyniki tych eksperymentów - wartości funkcji celu sprawdzanych rozwiązań. Uczenie się na drodze wykonywania eksperymentów (ang. lerning by doing, por. [1]) bez nauczyciela oraz bez analizy rozwiązywanego zagadnienia (tj. bez realizacji procesu rozumowania) jest jedną z podstawowych cech działania algorytmu genetycznego.

Za drugą cechę funkcjonalną algorytmu genetycznego możemy przyjąć sposób gromadzenia doświadczenia (wiedzy) z przeprowadzonych eksperymentów. Mianowicie, rezultatem procesu uczenia się algorytmu nie jest jedno, najlepsze znane, rozwiązanie problemu, ale zbiór rozwiązań nazywany dalej krótko populacją. Tak więc wyniki uczenia się algorytmu w zakresie rozwiązywania konkretnego zagadnienia na drodze realizacji eksperymentów obliczeniowych są gromadzone w populacji, przy czym w zbiorze tym nie znajdują się wszystkie rozwiązania badane podczas

eksperymentów, ale tylko m (m>2) najlepszych rozwiązań, gdzie m jest liczbą naturalną ustaloną przez analityka zagadnienia (użytkownika programu). Zauważmy, że druga cecha jest szczególnie ważna w przypadku rozwiązywania zagadnień optymalizacji kombinatorycznej, ponieważ często poprawianie gorszego rozwiązania prowadzi do uzyskania zadowalającego wyniku końcowego.

Prowadzenie eksperymentu obliczeniowego polega na generowaniu od jednego (Π^1) do czterech (Π^1,Π^2,Π^3,Π^4) nowych rozwiązań-hipotez przydziału za pomocą wybranego operatora genetycznego i obserwacji przydatności nowego rozwiązania (lub rozwiązań) na drodze obliczenia odpowiadających mu (lub im) wartości funkcji celu ($f(\Pi^1)$ lub $f(\Pi^1),f(\Pi^2),f(\Pi^3),f(\Pi^4)$). Jeżeli wartość funkcji celu nowego rozwiązania (nowych rozwiązań) jest lepsza (tj. mniejsza) od jej wartości dla najgorszego rozwiązania populacji, to ostatnie jest usuwane z populacji i na to miejsce jest wprowadzane nowe - lepsze rozwiązanie. Czynności te powodują, że w miarę trwania procesu realizacji eksperymentów obliczeniowych i obserwacji wartości funkcji celu uzyskanych rozwiązań populacja zawiera coraz lepsze rozwiązania. Zauważmy,że w ten sposób jest realizowana zasada słabych ogniw : "chcąc poprawić pewną całość (strukturę), poprawiamy kolejno słabe ogniwa struktury", którą można uznać za trzecią cechę algorytmu genetycznego.

W przypadku zagadnienia sformalizowanego jako permutacyjne zagadnienie optymalizacji można stosować następujące operatory genetyczne :

1. Operator inwersji

Operator inwersji tworzy, w oparciu o jedno wybrane rozwiązanie (przodka), jednego potomka w następujący sposób :

a) wybiera dwie liczby naturalne i, j takie, że $1 \le i < j \le n$.

b) zamienia miejscami elementy stojące na miejscach pomiędzy i oraz j w ten sposób, że element stojący na miejscu i jest wymieniany z elementem j, (i+1) z (j-1) itd. Jeżeli przodek jest rozwiązaniem o postaci :

$\Pi=(\Pi(1),\Pi(2),\ldots,\Pi(i),\Pi(i+1),\ldots,\Pi(j),\Pi(j+1),\ldots,\Pi(n))$,

to potomek ma postać:

$\Pi^1=(\Pi(1),\Pi(2),\ldots,\Pi(j),\Pi(j-1),\ldots,\Pi(i+1),\Pi(i),\ldots,\Pi(n))$.

2. Operator mutacji

Operator mutacji tworzy, oparciu o jedno wybrane z populacji rozwiązanie (przodka), jednego potomka w następujący sposób :

a) wybiera dwie liczby naturalne i, j takie, że $1 \le i < j \le n$.

b) zamienia miejscami dwa elementy rozwiązania stojące na miejscach i oraz j. Jeżeli przodek jest rozwiązaniem :

$\Pi=(\Pi(1),\Pi(2),\ldots,\Pi(i),\Pi(i+1),\ldots,\Pi(j),\Pi(j+1),\ldots,\Pi(n))$,

to potomek ma postać:

$\Pi=(\Pi(1),\Pi(2),\ldots,\Pi(j),\Pi(i+1),\ldots,\Pi(i),\Pi(j+1),\ldots,\Pi(n))$.

W algorytmie genetycznym zastosowano dwa operatory krzyżowania. Każdy z nich, w oparciu o dwa wybrane z populacji rozwiązania (dwóch przodków), tworzy dwa rozwiązania *wstępne*, które są następnie poprawiane - uzupełniane do n-elementowej permutacji - za pomocą pomocniczej procedury ALFA. Procedura ALFA, poprawiając jedno rozwiązanie wstępne, generuje do dwóch nowych rozwiązań - potomków.

3. Operator krzyżowania

a) wybierz liczbę naturalną i, $2\le i\le n-1$ oraz dwóch przodków :

$\Pi_1=(\Pi(1),\Pi(2),\ldots,\Pi(i),\Pi(i+1),\ldots,\Pi(j),\Pi(j+1),\ldots,\Pi(n))$

$\Pi_2=(\Pi(1),\Pi(2),\ldots,\Pi(i),\Pi(i+1),\ldots,\Pi(j),\Pi(j+1),\ldots,\Pi(n))$,

b) utwórz dwa rozwiązania wstępne :

$\Pi^1=(\Pi_1(1),\Pi_1(2),\ldots,\Pi_1(i),\Pi_2(i+1),\ldots,\Pi_2(n))$

$\Pi^2=(\Pi_2(1),\Pi_2(2),\ldots,\Pi_2(i),\Pi_1(i+1),\ldots,\Pi_1(n))$,

c) każde rozwiązanie wstępne popraw za pomocą procedury ALFA.

4. Operator krzyżowania wewnętrznego

a) wybierz liczby naturalne i, j, $2\le i<j\le n-1$ oraz dwóch przodków:

$\Pi_1=(\Pi_1(1),\Pi_1(2),\ldots,\Pi_1(i),\Pi_1(i+1),\ldots,\Pi_1(j),\Pi_1(j+1),\ldots,\Pi_1(n))$

$\Pi_2=(\Pi_2(1),\Pi_2(2),\ldots,\Pi_2(i),\Pi_2(i+1),\ldots,\Pi_2(j),\Pi_2(j+1),\ldots,\Pi_2(n))$,

b) utwórz dwa wstępne rozwiązania :

$\Pi^1=(\Pi_1(1),\Pi_1(2),\ldots,\Pi_1(i-1),\Pi_2(i),\ldots,\Pi_2(j),\Pi_1(j+1),\ldots,\Pi_1(n))$

$\Pi^2=(\Pi_2(1),\Pi_2(2),\ldots,\Pi_2(i-1),\Pi_1(i),\ldots,\Pi_1(j),\Pi_2(j+1),\ldots,\Pi_2(n))$,

c) każde rozwiązanie wstępne popraw za pomocą procedury ALFA.

Jeżeli rozwiązanie wstępne, uzyskane w wyniku realizacji operacji krzyżowania, jest permutacją n-elementową, to procedura ALFA nie zmienia jego postaci i w tym przypadku rozwiązanie to jest bezpośrednio potomkiem. W przeciwnym przypadku procedura ALFA poprawia rozwiązanie wstępne do permutacji n-elementowej generując dwie permutacje - dwóch potomków.

PROCEDURA ALFA :

a) Jeżeli rozwiązanie wstępne Π^j (j=1,2) jest permutacją n-elementową, to rozwiązanie Π^j jest potomkiem ⟶POWRÓT.

W przeciwnym wypadku przejdź do kroku b).

b) Sprawdzaj elementy ciągu Π^j od lewej strony do prawej. Pierwszy taki element, który występuje dwukrotnie zastąp przez element α, gdzie α jest najmniejszym elementem zbioru N, który nie występuje w ciągu Π^j. Otrzymanego w ten sposób potomka oznacza się symbolem Π^{j+2}.

c) Sprawdzaj elementy ciągu Π^j od prawej strony do lewej. Pierwszy taki element, który w ciągu występuje dwukrotnie, zastąp przez element α.

Otrzymanego w ten sposób potomka oznacz także symbolem Π^i ——→POWRÓT.

Występująca w operatorach genetycznych czynność "wybór" oznacza wybór arbitralny. W dalszej części pracy przyjmuje się, jak czyni się to powszechnie, że czynność wyboru jest realizowana za pomocą mechanizmu losowania.

Zauważmy, że wszystkie wymienione powyżej operatory genetyczne generują nowe rozwiązania - hipotezy przydziału (które następnie są sprawdzane na drodze obliczenia wartości funkcji celu), wykorzystując doświadczenie (wiedzę) zgromadzoną w populacji w następujący sposób :

z populacji wybierane są w sposób losowy rozwiązania (są to nośniki informacji o zdobytym doświadczeniu), a następnie rozwiązania te podlegają częściowym przekształceniom przypadkowym, w wyniku czego otrzymujemy nowe rozwiązania, które nazywamy potomkami. Jest to czwarta własność algorytmu genetycznego.

3. OPIS PAKIETU DOŚWIADCZALNEGO

W celu przeprowadzenia badań dynamiki algorytmu genetycznego zaprojektowano i wykonano pakiet doświadczalny oprogramowania zakodowany w języku C na komputery zgodne ze standardem IBM PC. Pakiet składa się z modułów pomocniczych takich jak moduł sterujący działaniem całego pakietu, edytor (przeznaczony do wczytywania macierzy A i B rozwiązywanego zagadnienia QAP oraz parametrów algorytmu genetycznego, edycji wczytanych danych oraz wyników pośrednich i końcowych obliczeń), losowy generator rozwiązań początkowych - permutacji - o równomiernym rozkładzie, moduł graficzny (zbiór programów pozwalający na graficzne przedstawienie wyników obliczeń w postaci wykresu zmian wartości funkcji celu, podczas pracy pakietu, najlepszego i najgorszego rozwiązania oraz średniej wartości funkcji celu całej populacji od liczby wygenerowanych nowych rozwiązań) oraz moduł algorytmu genetycznego.

Po wprowadzeniu do zbioru populacji m rozwiązań początkowych (m - parametr algorytmu genetycznego) wyznaczonych przez losowy generator rozwiązań początkowych, badany algorytm genetyczny funkcjonuje w następujący sposób.

Jeden krok algorytmu genetycznego polega na wyborze jednego operatora genetycznego, przy czym prawdopodobieństwo wyboru operatora pierwszego wynosi p_1, drugiego p_2 itd., gdzie liczby p_1, p_2, p_3, p_4 $(p_1+p_2+p_3+p_4=1)$ są parametrami algorytmu. Następnie jest realizowany wybór jednego lub dwóch przodków z populacji (rozkład równomierny) w zależności od tego, ilu przodków wymaga operator genetyczny do wygenerowania potomków. Operator kopiuje wybrane z populacji rozwiązanie/rozwiązania oraz generuje potomków. Jeżeli wartości funkcji celu wygenerowanych rozwiązań są lepsze od najgorszych rozwiązań w populacji, to rozwiązania te są zastępowane

potomkami.

Warunkiem zatrzymania się algorytmu może być wykonanie zadanej liczby (parametr algorytmu) iteracji.

4. EKSPERYMENTY OBLICZENIOWE

Tablica 1. Wartość funkcji celu Q_{10} i Q_{30} dla testu Nugent'a, n=30.

	L1	L2	L1	L2	L1	L2	L1	L2	L1	L2	L1	L2	L1	L2
Q_{10}	6344	6342	6648	6648	7102	7102	6488	6488	6458	6238	6574	6420	6498	6320
Q_{30}	6444	6380	6636	6612	6868	6868	6400	6284	6302	6302	6564	6424	6416	6364
p_1	0		1.0		0		0.1		0.2		0.7		0.5	
p_2	1.0		0		0		0.7		0.5		0.1		0.2	
p_3	0		0		0.5		0.1		0.2		0.1		0.2	
p_4	0		0		0.5		0.1		0.1		0.1		0.1	

Tablica 2. Wartość funkcji celu Q_{10} dla testu Nugent'a, gdy liczba iteracji wynosi L2.

Q_{10}	2660	2754	2878	2708	2666	2602	2702
p_1	0	1.0	0	0.1	0.2	0.7	0.5
p_2	1.0	0	0	0.7	0.5	0.1	0.2
p_3	0	0	0.5	0.1	0.2	0.1	0.2
p_4	0	0	0.5	0.1	0.1	0.1	0.1

Obecnie przedstawiamy wybrane wyniki numeryczne ilustrujące proces optymalizacji zagadnienia QAP za pomocą algorytmu genetycznego. Eksperymenty obliczeniowe wykonano dla testów Nugent'a [11] rozmiaru n=30 (tablica 1 oraz rys. 1 i 2) oraz i n=20 (tablica 2 oraz rys.3).

Przyjęto następujące oznaczenia :

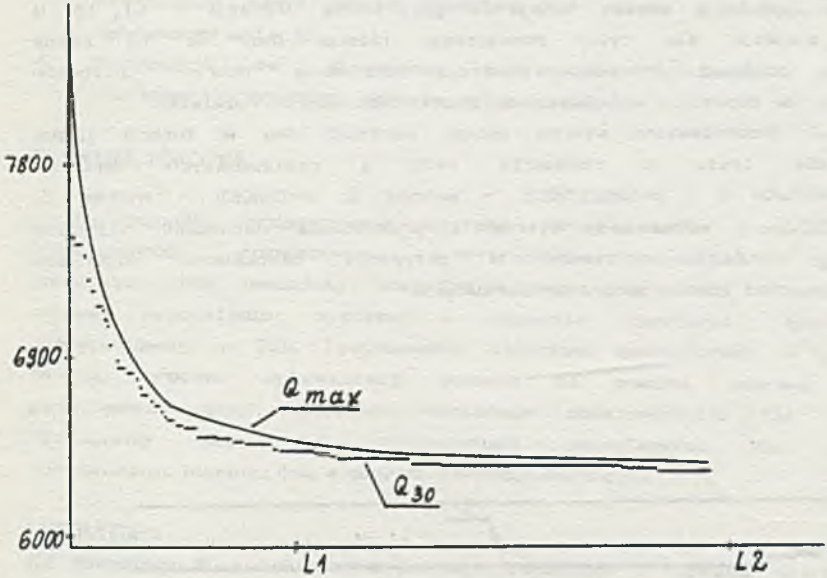
L1, L2 - liczba iteracji, odpowiednio 10.000 oraz 30.000, wykonywanych przez algorytm genetyczny w czasie procesu optymalizacji,

Q_{10}, Q_{30} - minimalna wartość funkcji celu rozwiązań populacji określona w przypadku rozmiaru populacji odpowiednio, m=10 i m=30,

$p=(p_1,p_2,p_3,p_4)$ - wektor prawdopodobieństw losowania operatorów genetycznych ; p_1,p_2,p_3,p_4 - prawdopodobieństwa losowania następujących operatorów genetycznych ; inwersja, mutacja, krzyżowanie i krzyżowanie wewnętrzne.

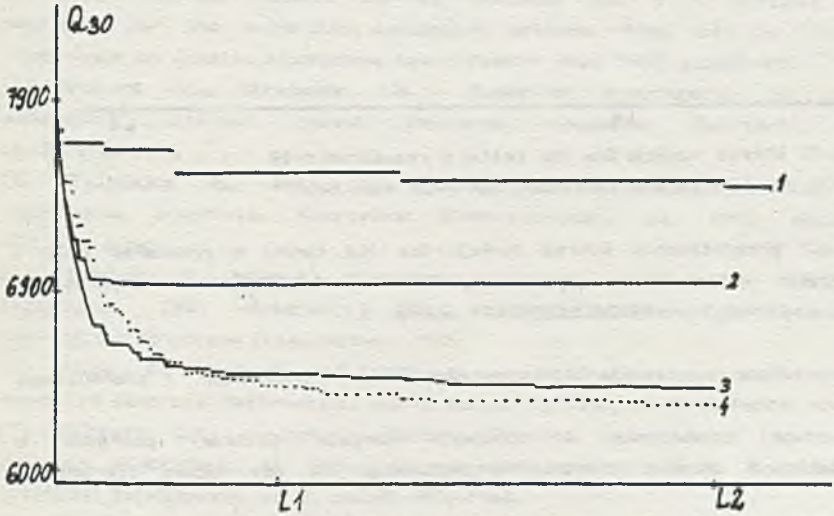
Przyjęto, że liczba iteracji algorytmu genetycznego jest równa liczbie generowanych potomków. Rozwiązania początkowe (startowe) były generowane w sposób losowy przy równomiernym rozkładzie, przy czym każdy eksperyment rozpoczynał się od tych samych rozwiązań startowych - tej samej, wygenerowanej losowo, populacji startowej. Funkcja celu najlepszego

rozwiązania startowego w przypadku testu rozmiaru n=30 wynosiła 7702, natomiast dla n=20 równała się 3166. Wartości funkcji celu najlepszych znanych rozwiązań wynoszą 6124 i 2570, odpowiednio dla n=30 i n=20 [6].



Rys. 1. Wykres funkcji celu Qso i Qmax dla testu o rozmiarze n=30

Fig. 1. Diagram of Qso and Qmax functions for n=30 size test

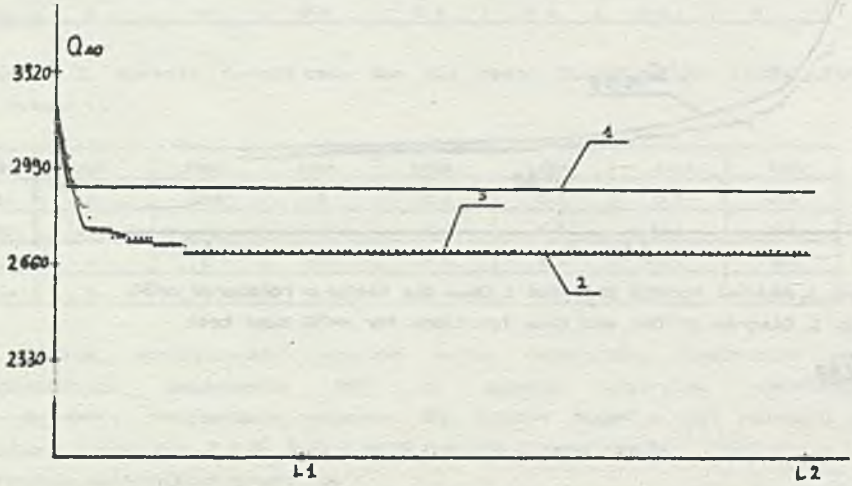


Rys. 2. Wykres funkcji Qso dla testu o rozmiarze n=30

Fig. 2. Diagram of Qso functions for n=30 size test

Na rys.1 przedstawiono wykres zmiany wartości minimalnej (Q_{30}) i maksymalnej (Q_{max}) funkcji celu populacji w funkcji liczby iteracji dla wektora p=(0.2;0.5;0.2;0.1) i testu rozmiaru n=30. Zauważmy, że różnica $Q_{max}-Q_{30}$ w populacji szybko maleje i gdy liczba iteracji > L1, to w populacji znajdują się tylko rozwiązania bliskie Q_{30}. Są to różne rozwiązania, ponieważ procedura kontroli wpisywania "dobrych" potomków do populacji nie dopuszcza wpisywania do zbioru tych samych rozwiązań.

Na rys.2 przedstawiono wykres zmiany wartości Q_{30} w funkcji liczby iteracji dla testu o rozmiarze n=30 i następujących wektorów prawdopodobieństw p : p=(0;0;0.5;0.5) - wykres 2, p=(0;1;0;0) - wykres 3, p=(0.2;0.5;0.2;0.1) - wykres 4. Wykres 1 przedstawia natomiast najlepsze rozwiązanie populacji otrzymane w przypadku zastąpienia algorytmu genetycznego przez losowy generator permutacji.



Rys. 3. Wykres funkcji Q_{40} dla testu o rozmiarze n=10

Fig. 3. Diagram of Q_{40} functions for n=20 size test

Na rys.3 przedstawiono wykres funkcji Q_{40} dla testu o rozmiarze n=20 i następujących wektorów prawdopodobieństw : p=(0;0;0.5;0.5) - wykres 1, p=(0;1;0;0) - wykres 2 i p=(0.2;0.5;0.2;0.1) - wykres 3.

Przeprowadzone badania eksperymentalne pozwalają sformułować następujące wnioski :

1. Wszystkie rozwiązania w populacji ulegają szybkiej poprawie w początkowym okresie procesu optymalizacji, tj. gdy liczba iteracji < L1.
2. Podstawowym operatorem genetycznym, decydującym o powodzeniu procesu optymalizacji, jest operator mutacji, który powinien być dostatecznie

często stosowany (np. p_2>0.4).

3. Sam operator krzyżowania powoduje zmniejszenie wartości funkcji celu populacji tylko w początkowym okresie procesu optymalizacji. Praktycznie rzecz biorąc, powinien być zawsze stosowany łącznie z operatorem mutacji.
4. Rezygnacja ze stosowania operatora krzyżowania wewnętrznego nie pogarsza przebiegu procesu optymalizacji.

5. UWAGA KONCOWA

W przypadku projektowania systemu oprogramowania przeznaczonego do przybliżonego rozwiązywania NP-trudnych zagadnień optymalizacji kombinatorycznej metodami wieloiteracyjnego poprawiania rozwiązań jest celowe wyposażenie systemu w algorytm genetyczny (por. system zaprezentowany w [5]). Zastosowanie algorytmu genetycznego w początkowym okresie procesu optymalizacji pozwala na znaczną poprawę rozwiązań początkowych przy niedużych nakładach obliczeniowych. Tak poprawione rozwiązania mogą być rozwiązaniami początkowymi dla algorytmów optymalizacji lokalnej lub algorytmu termodynamicznego.

LITERATURA

[1] **Bernstein M.** : Learning by doing, application of genetic search to the design of control strategies. Proc. of IEE Int. Symposium on Intelligent Control 1987, Philadelphia, Pennsylvania, pp.501-505.

[2] **Brown D. E., Huntley Ch. L., Spollane A.R.** : A parallel genetic heuristic for the quadratic assignment problem. Proc. of the Third Int. Conference on Genetic Algorithms, Georg Manson Univ. 1989, pp.406-415.

[3] **Brukard R.E., Stratmann K.H.** : Numerical investigation on quadratic assignment problems. Naval Research Logistics Qaurterly, vol.25, pp.129-147, 1978.

[4] **Filipowicz B., Wala K.** : Algorytmy optymalizacji kwadratowego zagadnienia przydziału. Kwartalnik Elektrotechniki, z.1, 1992, Wydawnictwo AGH w Krakowie.

[5] **Filipowicz B., Wala K.** :Komputerowy system wspomagający rozwiązywanie zagadnienia QAP. Materiały VIII Krajowej Konferencji Automatyzacji Dyskretnych Procesów Przemysłowych '92.

[6] **Finke G., Burkard R.E.,Rendl F.** : Quadratic assignment problems. Annals of Discrete Mathematics, vol.31, pp. 61-82, 1987, North-Holland.

[7] **Goldberg D.E.** : Dynamic System Control Using Rule Learning and Genetics Algorithms, in Proc. of the International Joint Conference on Artificial Intelligence, vol.9, pp.588-592, 1985.

[8] **Goldberg D. E.** : Genetic algorithms in search, optimization and machine learning, Addison-Wesley, 1989.

[9] **Holland J.** : Adaptation in Natural and Artificial Systems. Univ. of

Michigan Pres, Ann Arbor, MI, 1975.

[10] **Jog P., Suh J.Y,Van Guht D.** : The effects of population size, heuristic crossover and local improvement on genetic algorithm for the TSP. Proc. of the Third Int. Conference on Genetic Algorithms, Georg Manson Univ. 1989, pp.110 - 115.

[11] **Nugent C. E., Vollman T.E., Ruml J.** : An experimental comparison of techniques for the assignment of facilities to locations. Operation Research, vol.16, pp.150-173, 1968.

[12] **Syswerda G.** : Uniform Crossover in genetic algorithms, Georg Manson Univ. 1989, pp 2-9.

[13] **Vignaux G. A.,Michalewicz Z.** : Genetic algorithms for the transportation problem. Methodologies for Intelligent Systems, vol.4, pp.252-259, ed. Z.W. Ras, Elsevier Sc. Pub. Co.,Inc. 1989.

Recenzent: Prof.dr inż. Henryk Kowalowski

Wpłynęło do Redakcji do 30.04.1992 r.

Abstract: There are a number of discrete decision problems that can be formalized as the permutational optimization problems, for instance the problem of traveling salesman, quadratic assignment problem (QAP) and so forth. Usually these ones are NP-hard problems.

The paper presents the results of computer (numerical) investigation of the optimization process realized with the aid of genetic algorithm on the example of the quadratic assignment problem. The investigation has been realized by means of experimental software package which is specially prepared for IBM PC compatibile computer to serve these purpose. The aim of these investigation is the determination of the algorithm parameter influence on the optimization process efficiency, and in this way, to get instructions connected with algorithm parameters adjustement to the investigated problem. We present a collection of numerical results for Nugent's (n=20, n=30) test tasks which give the right characteristics of subjection of the optimization process dynamic to algorithm parameters condition.