ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1998
Seria: AUTOMATYKA z. 125 Nr kol. 1391

Piotr KADŁUCZKA
Akademia Górniczo Hutnicza, Kraków

# ALGORYTM GENETYCZNY DO ROZWIĄZYWANIA UOGÓLNIONEGO, WIELOKRYTERIALNEGO ZAGADNIENIA PODZIAŁU GRAFU

**Streszczenie.** W artykule zaprezentowano algorytm genetyczny dla uogólnionego, wielokryterialnego zagadnienia podziału grafu (GGP). Polega ono na podziale zbioru wierzchołków nieskierowanego, ważonego grafu na m rozłącznych zbiorów, tzw. punktów skupienia, w celu optymalizacji funkcji celu, przy równoczesnej optymalizacji ich rozmiaru.

W algorytmie zastosowano specjalną formalizację Pareto. Rozwiązania są odwzorowywane w przestrzeni 2D Euklidesa, gdzie pierwszą współrzędną obszaru jest rozmiar największego punktu skupienia (suma wag węzłów), a drugą współrzędną wartość funkcji celu (całkowita suma wag krawędzi należących do punktów skupienia).

W artykule przedstawiono model matematyczny zagadnienia, opis zastosowanego algorytmu genetycznego, wyniki badań komputerowych dla standardowych zadań testowych oraz wynikające z nich wnioski.

# GENETIC ALGORITHM FOR GENERALIZED MULTICRITERIAL GRAPH PARTITIONING PROBLEM

**Summary.** A genetics algorithm for the generalized multicriterial graph partitioning problem (GGP) is presented. The GGP problem consists in partitioning the set of nodes of an undirected weighted graph into m disjoint subsets of bounded size (called clusters), such that the objective function is maximized whereas the size of clusters is minimized.

In the presented approach the specific formalization of Pareto set was applied. The solutions are projected into to two dimensional Euclidean space where the first coordinate is the maximal cluster size (the sum of weights of nodes), the second is the value of objective function (the sum of edge weights in a cluster).

The paper presents the mathematical formalization of the problem, applied genetics algorithm and the results of computer experiments for the standard test cases.

## 1. Model zagadnienia

Uogólnione zagadnienie podziału grafu GGP (ang. generalized graph partitioning) w ujęciu wielokryterialnym może być przedstawione jako następujące zagadnienie optymalizacji kombinatorycznej. Dany jest nieskierowany, ważony graf G=(N,E). Należy znaleźć podział $X=\left(X_1, X_2, \ldots, X_i, \ldots, X_m\right)$, $X_i \subset N$, (i=1,...,m) zbioru N, który maksymalizuje funkcję celu:

$$f(X)=\sum_{i=1}^{m}\sum_{j,k\in X_i}c_{jk}\rightarrow \max \tag{1}$$

przy równoczesnej minimalizacji rozmiaru największego podzbioru spośród $X_i$ (i=1,...,m):

$$g(X)=\max\left\{\sum_{j\in X_1}a_{1j},\ldots,\sum_{j\in X_m}a_{mj}\right\}\rightarrow \min, \tag{2}$$

przy ograniczeniach:

$$\bigcup_{i=1}^{m}X_i=N,\qquad X_i\cap X_k=\varnothing \qquad \text{dla} \quad i\neq k, \tag{3}$$

gdzie:

$a_j=\left(a_{1j},a_{2j},\ldots,a_{ij},\ldots,a_{mj}\right)$: wektor wierzchołków j∈N,

$c_{jk}$: waga krawędzi jk∈E.

Ograniczenie (3) zapewnia, że każdy wierzchołek grafu jest umieszczony dokładnie w jednym podzbiorze $X_i$. Funkcja celu $f(X)$ określa całkowitą sumę wag krawędzi należących do punktów skupienia, natomiast funkcja $g(X)$ określa rozmiar największego, dla określonego podziału $X=\left(X_1,X_2,\ldots,X_i,\ldots,X_m\right)$, punktu skupienia, tj. sumę wag węzłów należących do największego spośród $X_i\subset N$, (i=1,...,m).

Zakładamy, bez straty ogólności, że dla każdego jk∈E: $c_{jk}>0$ i dla jk∉E: $c_{jk}=0$ oraz $a_{ij}=\infty$ w przypadku, gdy węzeł j nie ma być przydzielony do podzbioru $X_i$, w przeciwnym przypadku: $0<a_{ij}<\infty$ dla i=1,...,m; j=1,...,n. W dalszym ciągu rozdziału przez F(j) oznaczyliśmy zbiór dopuszczalny przydziałów dla wierzchołka j :

$$F(j)=\{\ i\in\{1,2,\ldots,m\}:\ a_{ij}<\infty\} \tag{4}$$

W ujęciu jednokryterialnym zamiast minimalizacji wielkości punktów skupienia (2) wprowadza się nierówności

$$\sum_{j\in X_i}a_{ij}\ \leq\ b_i,\qquad i=1,\ldots,m \tag{5}$$

przy zadanym zbiorze liczb $\left\{b_1,b_2,\ldots,b_m\right\}$, gdzie $b_i>0$ dla i=1,...,m, ograniczające rozmiar każdego podzbioru. W tym modelu $b_i$ oznacza zasób lub rozmiar punktu skupienia (podzbioru $X_i$) i. Jeżeli dla każdego i oraz j $a_{ij}=1$, to $b_i$ ogranicza liczbę wierzchołków punktu skupienia; w tym przypadku ograniczenie (5) przyjmuje postać $|X_i|\ \leq\ b_i$.

Model (1)-(3) nazywamy modelem zagadnienia GGP. Różni się on od standardowego zagadnienia GP, w którym dla każdego i zachodzi: $a_{ij}=a_j$, tak więc w ujęciu wielokryterialnym warunek (2) ma postać:

$$g(X)=\max\left\{\sum_{j\in X_1}a_j,\ldots,\sum_{j\in X_m}a_j\right\}\rightarrow \min \tag{2'}$$

lub w standardowym zagadnieniu jednokryterialnym GP (1),(5'),(3):

$$\sum_{j\in X_i}a_j\ \leq\ b_i \tag{5'}$$

W ujęciu jednokryterialnym, jeśli X jest rozwiązaniem zagadnienia GGP, które spełnia ograniczenia przydziału (3), natomiast nie zawsze ograniczenie rozmiaru punktów skupienia (5) lub (5') dla GP, możemy zdefiniować wartości $s_i$:

$$s_i = b_i - \sum_{j \in X_i} a_{ij} \qquad i=1,\dots,m \tag{6}$$

W tym podejściu wszystkie rozpatrywane rozwiązania spełniają ograniczenia przydziału (3) , natomiast rozwiązania niedopuszczalne mogą naruszać tylko ograniczenia (5). Stąd rozwiązaniami niedopuszczalnymi będą takie rozwiązania, dla których przynajmniej jedna wartość $s_i$ będzie ujemna. Miarę niedopuszczalności rozwiązań X definiujemy jako sumę ujemnych wartości $s_i$:

$$v(X) = \sum_{i=1}^{m} \min\{s_i, 0\}, \tag{7}$$

gdzie, żeby być ścisłym, rozwiązanie $X = (X_1, X_2, \dots, X_i, \dots, X_m)$ zawsze spełnia ograniczenia (3) oraz dla każdego $j \in X_i$ : i∈F(j).

W podejściu wielokryterialnym wszystkie rozwiązania spełniające warunek (3) są rozwiązaniami dopuszczalnymi. Można je odwzorować w przestrzeni 2D Euklidesa, gdzie pierwszą współrzędną obszaru jest rozmiar największego punktu skupienia - wartość funkcji g(X), a drugą współrzędną - wartość funkcji f(X).

Algorytmy genetyczne pracują z ciągami, dlatego wprowadziliśmy drugą postać problemu GGP: x=$(x_1, x_2, \dots, x_j, \dots, x_n)$, gdzie składowa $x_j, x_j \in F(j)$ definiuje przydział wierzchołka j do punktu skupienia (podzbioru $X_j$), gdzie $X_i = \{j : x_j = i\}$. W dalszej części pracy oznaczamy rozwiązanie problemu GGP symbolem x zamiast X.

## 2. Algorytm genetyczny

Dla zagadnienia GGP zaprojektowano specjalizowane operatory genetyczne. Operator optymalizacji lokalnej LO, operatory S1 i S2 opracowano i przetestowano w dwóch wariantach preferujących maksymalizację f(x) lub minimalizację g(x):

**1. Operator mutacji RM** (ang. random mutation operator)

Operator mutacji RM generuje jednego potomka $x^1$ modyfikując rodzica x =$(x_1, x_2, \dots, x_{j-1}, x_j, \dots, x_n)$ :

a) wybierz jedną liczbę j , $1 \le j \le n$ i podstaw $x_j^1$=RANDOM_UNIFORM(F(j))

b) podstaw $x^1 = (x_1, x_2, \dots, x_{j-1}, x_j^1, \dots, x_n)$

**2. Operator optymalizacji lokalnej LO** (ang. local optimization operator)

Niech

$S(x) = \{x' : x' = (x_1, x_2, \dots, x_{j-1}, x'_j, x_{j+1}, \dots, x_n) : x'_j \neq x_j,\ x'_j \in F(j),\ j \in \{1, \dots, n\}\}$

oznacza otoczenie rozwiązania x =$(x_1, x_2, \dots, x_j, \dots, x_n)$.

Operator LO generuje jednego potomka $x^1$ realizując proces przeszukiwania otoczenia wg jednego z wariantów, rozpoczynający się od otoczenia rodzica x:

- a) znajdź rozwiązanie $x' = \arg\max\{f(\hat{x}): \hat{x} \in S(x), \hat{x} \neq x$ oraz g(x) jest minimalne}
  b) jeżeli $f(x') > f(x)$, to podstaw x:=x' i powtórz krok a), inaczej zwróć $x^1$:=x,
- a) znajdź rozwiązanie $x' = \arg\min\{g(\hat{x}): \hat{x} \in S(x), \hat{x} \neq x$ oraz f(x) maksymalne},
  b) jeżeli $g(x') < g(x)$ to podstaw x:=x' i powtórz krok a), inaczej zwróć $x^1$:=x.

**3. Operator S1** (ang. search operator one)

Operator S1 generuje jednego potomka $x^1$ modyfikując rodzica x $=(x_1, x_2, \ldots, x_{i-1}, x_i, \ldots, x_{j-1}, x_j, \ldots, x_n)$:

a) wybierz jedną liczbę i, $1 \le i \le n$, oraz "najlepszą" liczbę j wg jednego z wariantów:

- $j = \arg\max_{k \neq i} \{f(x_1, x_2, \ldots, x_{i-1}, x_k, x_{i+1}, \ldots, x_{k-1}, x_i, x_{k+1}, \ldots, x_n): k \in \{1,2,\ldots,n\}$
  oraz g(x) jest minimalne}
- $j = \arg\min_{k \neq i} \{g(x_1, x_2, \ldots, x_{i-1}, x_k, x_{i+1}, \ldots, x_{k-1}, x_i, x_{k+1}, \ldots, x_n): k \in \{1,2,\ldots,n\}$
  oraz f(x) jest maksymalne}

b) przestaw element $x_i$ z elementem $x_j$ rodzica: $x^1 = (x_1, x_2, \ldots, x_{i-1}, x_j, \ldots, x_{j-1}, x_i, \ldots, x_n)$

**4. Operator S2** (ang. search operator two)

Operator S2 generuje jednego potomka $x^1$ modyfikując rodzica x $=(x_1, x_2, \ldots, x_{j-1}, x_j, \ldots, x_n)$

a) wybierz jedną liczbę j, $1 \le j \le n$, oraz "najlepszą" liczbę i wg jednego z wariantów:

- $f(x_1, x_2, \ldots, x_{j-1}, i, x_{j+1}, \ldots, x_n) = \max_{k \neq x_j} \{f(x_1, x_2, \ldots, x_{j-1}, k, x_{j+1}, \ldots, x_n): k \in F(j)$
  oraz g(x) jest minimalne}
- $g(x_1, x_2, \ldots, x_{j-1}, i, x_{j+1}, \ldots, x_n) = \min_{k \neq x_j} \{g(x_1, x_2, \ldots, x_{j-1}, k, x_{j+1}, \ldots, x_n): k \in F(j)$
  oraz f(x) jest maksymalne}

b) zastąp element $x_j$ rodzica liczbą i: $x^1 = (x_1, x_2, \ldots, x_{j-1}, i, \ldots, x_n)$

**5. Operator krzyżowania RX** (ang. random crossover operator)

Operator RX generuje dwa potomki $x^1$, $x^2$ dokonując krzyżowania dwóch rodziców: $x' = (x'_1, x'_2, \ldots, x'_j, x'_{j+1}, \ldots, x'_n)$, $x'' = (x''_1, x''_2, \ldots, x''_j, x''_{j+1}, \ldots, x''_n)$

a) wybierz jedną liczbę j, $1 \le j < n$

b) wygeneruj dwa potomki:

$$x^1 = (x'_1, x'_2, \ldots, x'_j, x''_{j+1}, \ldots, x''_n), \qquad x^2 = (x''_1, x''_2, \ldots, x''_j, x'_{j+1}, \ldots, x'_n)$$

Wszystkie operacje wyboru w operatorach genetycznych i opisanym poniżej algorytmie genetycznym są wykonywane za pomocą procedury losowania liczb całkowitych z zadanego zbioru liczb - RANDOM_UNIFORM ( rozkład wyboru - równomierny).

Dla rozważanego wcześniej podejścia jednokryterialnego, z którym wiązało się rozwiązanie niedopuszczalne (ze względu na ograniczenia (5)), najbardziej efektywne okazały się reguły szeregowania z zastosowaniem funkcji kary oraz „czasem życia" rozwiązania. W pierwszej przyjęto, że funkcja oceny EV(x) rozwiązania x ( funkcja przystosowania osobników ang. evoluation or fitness function ) składa się z funkcji celu f(x) oraz ważonej funkcji v(x) (por., wzór (7)):

$$EV(x) = f(x) + w * v(x), \tag{8}$$

gdzie $w$, $w \geq 0$ jest współczynnikiem (kary) funkcji oceny. W drugiej natomiast wprowadzono dodatkową funkcję - parametr t(x) - „czas życia" rozwiązania x, zależny podobnie jak poprzednio od funkcji celu f(x) oraz funkcji v(x). Określa on, w zależności od jakości rozwiązania, liczbę iteracji, jaką przebywa ono w dynamicznej populacji rozwiązań.

Dla zagadnienia wielokryterialnego zastosowano specjalną formalizację Pareto. Rozwiązania ze zbioru P (populacji rozwiązań) są odwzorowywane w przestrzeni 2D Euklidesa. Pierwszą współrzędną obszaru jest wartość funkcji g(x), a drugą współrzędną wartość funkcji celu f(x). Niezdominowane rozwiązania tworzą pierwszą warstwę Pareto. Pozostałe niezdominowane rozwiązania tworzą drugą warstwę, itd.

Rozwiązanie każdej warstwy Pareto można dodatkowo uszeregować za pomocą reguły BEST-WORST zgodnie z wartością funkcji g(x): pierwszym rozwiązaniem warstwy jest rozwiązanie o minimalnej wartości g(x), a ostatnim rozwiązanie o maksymalnej wartości g(x). Liniowo uszeregowane warstwy Pareto określają liniowy porządek w zbiorze P: pierwsze rozwiązanie pierwszej warstwy Pareto definiuje pierwsze rozwiązanie $x_{best}$ ,..., ostatnie rozwiązanie ostatniej warstwy Pareto definiuje ostatnie rozwiązanie $x_{worst}$ w populacji P. Poszukiwanymi rozwiązaniami jest zbiór rozwiązań potencjalnie Pareto-optymalnych (tj. rozwiązań niezdominowanych), tworzących pierwszą warstwę Pareto w ostatniej iteracji algorytmu.

**Algorytm GENGGP4**

W celu określenia zbioru rozwiązań potencjalnie Pareto-optymalnych wykonaj następujące kroki:

Krok 1

Wygeneruj M rozwiązań x=($x_1$,$x_2$,...$x_j$,...$x_n$), gdzie dla każdego j:
$x_j = RANDOM_UNIFORM(F(j))$ oraz oblicz funkcję celu f(x) oraz funkcję g(x) dla każdego z rozwiązań. Utwórz populację początkową P(0) szeregując wygenerowane rozwiązania za pomocą reguły BEST-WORST wg utworzonych warstw Pareto w ten sposób,
że pierwszym rozwiązaniem populacji (rozwiązaniem nr 1) jest rozwiązanie pierwsze pierwszej warstwy Pareto, a ostatnim rozwiązaniem (rozwiązaniem nr M) jest rozwiązanie ostatnie ostatniej warstwy.

Krok 2

Wybierz jeden operator genetyczny ze zbioru {RM, LO, S1, S2, RX}, gdzie prawdopodobieństwa selekcji operatorów wynoszą: $p_{RM}$, $p_{LO}$, $p_{S1}$, $p_{S2}$, $p_{RX}$ ($p_{RM} + p_{LO} + p_{S1} + p_{S2} + p_{RX} = 1$).

Krok 3

Dla wybranego operatora genetycznego wylosuj (rozkład równomierny) jednego lub dwóch rodziców (zależnie od typu operatora: jednego dla operatora unarnego i dwóch dla binarnego) ze "starego" zbioru populacja P(t-1).

Krok 4

Za pomocą wybranego operatora wyznacz potomki / potomka wybranych rodziców / rodzica $x^j$, $j \in \{1,2\}$ i utwórz nową populację P(t): wprowadź potomka $x^j$ między inne rozwiązania zgodnie z regułą szeregowania BEST-WORST (jeżeli potomek $x^j$ jest lepszy od najgorszego rozwiązania $x_{worst}$), usuń $x_{worst}$ z populacji i.

Krok 5

Powtórz kroki 2-4 zadaną liczbę iteracji L.

Zbiór rozwiązań tworzących pierwszą warstwę Pareto jest poszukiwanym zbiorem rozwiązań potencjalnie Pareto-optymalnych (tj. rozwiązań niezdominowanych).

Zauważmy, że rozmiar populacji M, liczba iteracji L i prawdopodobieństwa selekcji $p_{RM}$, $p_{LO}$, $p_{S1}$, $p_{S2}$, $p_{RX}$ są parametrami algorytmu GENGGP4.

## 3. Eksperymenty komputerowe

Algorytm GENGGP4 zaimplementowano w języku C w dwóch wariantach dotyczących konstrukcji operatora optymalizacji lokalnej LO oraz operatorów S1 i S2, preferujących maksymalizację f(x) (**wariant 1**) lub minimalizację g(x) (**wariant 2**). Przedstawione w tabeli 1 wyniki badań komputerowych wykonano dla wygenerowanych losowo zadań testowych o rozmiarach n=10, m=3; n=30, m=6; n=50, m=10 i następujących parametrów algorytmu: liczba iteracji L=15.000, wielkość populacji M.=100, prawdopodobieństwa selekcji operatorów $p_{RM} = p_{S1} = p_{S2} = p_{RX} = 0{,}23$ $p_{LO} = 0{,}08$. W tabeli podano wartości funkcji f(x) i g(x) dla pierwszego oraz ostatniego rozwiązania pierwszej warstwy Pareto oraz liczbę rozwiązań tworzących tę warstwę.

Tabela 1

Wyniki eksperymentów

| n / m | | wariant 1 | | wariant 2 | |
|---|---|---|---|---|---|
| | | min g(x) | max f(x) | min g(x) | max f(x) |
| 10 / 3 | f(x) | 1346 | 4292 | 1346 | 4292 |
| | g(x) | 12 | 56 | 12 | 56 |
| | l. rozw. | 17 | | 17 | |
| 30 / 6 | f(x) | 8988 | 42182 | 12828 | 20706 |
| | g(x) | 19 | 152 | 18 | 51 |
| | l. rozw. | 82 | | 21 | |
| 50 / 10 | f(x) | 22116 | 106558 | 24940 | 46616 |
| | g(x) | 28 | 209 | 21 | 57 |
| | l. rozw. | 93 | | 52 | |

Uzyskane wyniki eksperymentów komputerowych świadczą o tym, że dla zadań o małym rozmiarze algorytm znajduje identyczne rozwiązania potencjalnie Pareto-optymalne niezależnie od wariantu algorytmu i wielkości populacji. Dla większych zadań zbyt mała wielkość populacji ma krytyczne znaczenie przy wyznaczaniu poszukiwanego zbioru rozwiązań. Dla wariantu 1, preferującego rozwiązania maksymalizujące f(x), pierwsza warstwa Pareto zawiera więcej rozwiązań, niż w wariancie 2, kosztem jakości tych rozwiązań. Dla wariantu 2 lepiej spenetrowanym obszarem rozwiązań są rozwiązania dla mniejszych wartości g(x). Zasadnicze wydaje się zwiększenie dla tych zadań wielkości populacji tak, aby stale mieściło się wiele warstw Pareto, względnie wprowadzenie populacji dwuwymiarowej, pozwalającej na bardziej równomierny rozkład umieszczonych w niej rozwiązań.

## LITERATURA


1. Arabas J., Michalewicz Z., Mulawka J.: GaVaPS - a Genetic Algorithm with Varying Popoulatin Size. Proc. of The First IEE Conference on Evolutionary Computation, vol .1, 1994.
2. Chen C. C.: Placement and partitioning methods for integrated circuit layout. Berkeley. CA, Ph. D. Dissertation, Dep. of EECS, Univ. of California, 1986.
3. Feo T. A., Khellaf M.: A class of bounded approximation algorithm for graph partitioning. Networks, vol. 20, pp. 181-195, 1990.
4. Goldberg D.E.: Genetic algorithms in search, optimization and machine learning. Addison-Wesley, 1989.
5. Holland J.: Adaptation in natural and artificial systems. Univ. of Michigan Press, Ann Arbor, MI, 1975.

6. Kadłuczka P.: Algorytm genetyczny z dynamiczną populacją rozwiązań dla uogólnionego zagadnienia podziału grafu. Automatyka, t.1 z.1, 1997, pp. 209-218.
7. Kadłuczka P., Wala K.: Tabu search heuristic for generalized graph partitioning problem. Automatics vol. 64, Cracow, University of Mining and Metallurgy Press., 1993, pp. 473-487.
8. Kadłuczka P., Wala K.: New artificial genetic search algorithm for generalized graph partitioning problem. Proc. of Second Int. Symposium on Methods and Models in Automation and Robotics, vol. 2, Międzyzdroje, 1995, pp.725-730.
9. Kadłuczka P., Wala K.: Tabu search and genetic algorithms for generalized graph partitioning problem. Control and Cybernetics, vol. 24, No. 4, 1995, pp. 1-18.
10. Kadłuczka P., Wala K.: Two stage genetic search for generalized graph partitioning problem. Mat. I Krajowej Konf. Algorytmy Ewolucyjne KAE'96, 1996.
11. Kumar R. K., et al.: Grouping parts and components in flexible manufacturing system. Eur. J. Operations Res. vol. 24, pp. 387-397, 1986.
12. Kusiak A.: Group technology in flexible manufacturing systems,in Handbook of flexible manufacturing systems. Ed. N. K. Jha, Academic Press, Jnc., 1991.
13. Liepins G.E., Holliard M.R.: Genetic algorithms: foundations and applications. Annals of Operations Research, vol. 21, 1989, pp. 31-58.
14. Michalewicz Z.: Genetic algorithms+data structures=evolution programs, Springer-Verlag, Berlin 1992.
15. Wala K., Chmiel W.: Algorytmy uczące się dla NP-trudnych zagadnień optymalizacji dyskretnej. Materiały I Konferencji "Granty - Automatyka '95", vol. 1, Wyd. PIAP, 1995, pp.126-131.
16. Wala K., Werewka J.: Allocation of computational processes in a multiprocessor system. Archiwum Automatyki i Telemechaniki, vol. 34, No.1-2, 1989, pp. 197-215.

Recenzent: Dr hab.inż. Adam Janiak, prof.Pol.Wrocł.

**Abstract**

A genetics algorithm for the generalized multicriterial graph partitioning problem (GGP) is presented. The GGP problem consists in partitioning the set of nodes of an undirected weighted graph into m disjoint subsets of bounded size (called clusters), such that the objective function is maximized whereas the size of clusters is minimized.

GPP is NP-hard combinatorial problem. It serves as an useful model for many decision problems occurring in practice (for example data processing, storage and retrieval in large distributed or multiprocessor computer systems).

In the presented approach the specific formalization of Pareto set was applied. The solutions are projected into to two dimensional Euclidean space where the first coordinate is the

maximal cluster size (the sum of weights of nodes), the second is the value of objective function (the sum of edge weights in a cluster).

The first section of the paper introduces the mathematical formalization of the problem being the generalization of the classical graph partitioning problem (GP). The next section describes applied genetics algorithm, genetics operators and methods allowing efficient search of potentially Pareto-optimal solutions (i.e. not dominating solutions). The final section presents the results of computer experiments for the standard test cases and conclusions.