FOLIA 97

Annales Universitatis Paedagogicae Cracoviensis

Studia Technica V (2012)

Wiktor Hudy

Porównanie algorytmów optymalizacyjnych: symulowanego wyżarzania, algorytmu zachłannego, metody największego spadku oraz algorytmu ewolucyjnego

Wstęp

Obliczenie maksimum globalnego w wielowymiarowej przestrzeni poszukiwań jest zadaniem niełatwym. Podstawową trudnością jest fakt „nieomijania" przez algorytmy ekstremów lokalnych i „utykanie" w tych ekstremach. Dodatkowym utrudnieniem są ograniczenia nałożone na funkcję oceny. Jednocześnie często występuje fakt nieciągłości przestrzeni poszukiwań. Z podobnymi problemami spotykamy się przy poszukiwaniach minimum globalnego. Istnieje wiele różnych algorytmów poszukujących ekstrema globalne. W niniejszej pracy przedstawiono i porównano cztery z nich, najczęściej stosowane. Ich zalety i wady przedstawiono na dwóch przykładach.

Algorytmy optymalizacyjne

Symulowane wyżarzanie

Algorytm symulowanego wyżarzania występuje pod kilkoma angielskimi nazwami: *simulated annealing* (najczęściej), *Monte Carlo annealing*, *probabilistic Hill climbing*, *stochastic relaxation* [3]. Ta metoda optymalizacyjna bazuje na zjawiskach z dziedziny termodynamiki, a w szczególności na zjawisku zamarzania cieczy i krystalizowania się jej lub na zjawiskach jakie zachodzą w strukturze metalu podczas zamarzania i wyżarzania. W wysokiej temperaturze i odpowiednim ciśnieniu, gdy dany materiał jest w formie płynnej, molekuły cieczy poruszają się swobodnie. W wyniku schładzania tej cieczy ruchy cząsteczek ulegają spowolnieniu.

Jeśli schładzanie jest przeprowadzane odpowiednio wolno, wówczas w cieczy tworzą się uporządkowane struktury zwane kryształami. Kryształ jest stanem o minimalnej wartości energii tego materiału. Jeśli schładzanie przeprowadzane jest wystarczająco szybko, to chłodzona ciecz nie osiąga stanu krystalicznego. Pozostaje wówczas w stanie amorficznym (tzw. szklistym), o wyższej wartości energii. By wyjaśnić to zjawisko, należy odwołać się do rozkładu Boltzmana, który określa prawdopodobieństwo P znalezienia się układu utrzymywanego w temperaturze T w stanie energii E.

$$P(E) \sim e^{-\frac{E}{kT}} \tag{1}$$

gdzie:
k – stała Boltzmana.

Ze wzoru (1) wynika, że istnieje niezerowe prawdopodobieństwo zdarzenia, że dany układ znajdzie się w stanie o wyższej energii. Jednak prawdopodobieństwo tego szybko maleje wraz z obniżaniem temperatury układu T. Jeżeli jednak materiał w początkowej fazie znajdował się w stanie ciekłym (o wyższej energii), to jego proces krystalizacji (dążenia do najniższego poziomu energii) może ulec spowolnieniu lub nawet zatrzymaniu (układ znajdzie się w stanie pośrednim, w stanie szklistym). Powodem tego są bariery energetyczne, na których pokonanie zwykle potrzeba pewnego czasu. Z tego powodu stan materiału zwany kryształem wymaga odpowiednio wolnego schładzania.

Na podstawie tych faktów stworzono algorytm poszukujący minimum badanej funkcji w założonej przestrzeni poszukiwań. Poszukiwane minimum globalne to odpowiednik stanu układu o najniższej energii. Minima lokalne to odpowiednio pośrednie stany układu – stany szkliste układu. Algorytm w podstawowej implementacji przetwarza jedno rozwiązanie, które jest w danej chwili czasowej rozwiązaniem o najmniejszej dotychczas znalezionej wartości funkcji oceny. Algorytm jest algorytmem iteracyjnym. W każdej iteracji oblicza się w losowo wybranym punkcie przestrzeni poszukiwań jedno rozwiązanie. Jeśli rozwiązanie to ma wartość funkcji oceny mniejszą niż wartość funkcji oceny dotychczas znalezionego punktu, to przyjmuje się go z prawdopodobieństwem P określonym relacją:

$$P(E) = e^{-E \cdot a} \tag{2}$$

gdzie:
a – parametr kontrolny algorytmu, którego wartość zostaje zwiększana podczas pracy programu (odpowiednik temperatury dla rozkładu Boltzmana).

W przeciwnym przypadku rozwiązanie jest odrzucane i losuje się w następnej iteracji kolejny punkt. Parametr a jest stopniowo zwiększany podczas pracy algorytmu. Zawiera on informacje o wielkości przestrzeni wokół dotychczas znalezionego rozwiązania. Nowe rozwiązanie w danej iteracji jest losowane zawsze z tego ograniczonego obszaru, który podczas trwania procesu optymalizacji ulega stopniowemu zawężaniu.

Algorytm zachłanny

Algorytm zachłanny (ang. *greedy algorithm*) jest algorytmem iteracyjnym [1]. W każdym kroku dokonywany jest wybór rozwiązania częściowego, które jest najlepsze z punktu widzenia lokalnego (działanie zachłanne). Algorytm dokonuje decyzji lokalnie optymalnej, bez zwracania uwagi na kroki poprzednie (dokonuje wyboru rozwiązania wydającego się w danym kroku najlepszym, np. o minimalnej wartości funkcji oceny). W dalszej kolejności następuje rozwiązanie podproblemu wynikającego z dokonanego wyboru.

Dotychczas nie udowodniono, czy dla danego problemu stosowanie algorytmu zachłannego jest najkorzystniejsze, czyli czy algorytm zawsze znajdzie poszukiwane minimum lokalne. Istnieje wiele problemów, przy rozwiązywaniu których stosowanie algorytmów zachłannych jest wystarczające.

Metoda największego spadku

Metoda największego spadku (ang. *steepest descent*) jest najprostszą z metod gradientowych [2, 4]. Kierunek w jakim należy szukać rozwiązania o mniejszej wartości kryterium jakości jest odwrotny do gradientu funkcji jakości F i określany ogólnie wzorem dla funkcji n-wymiarowej:

$$-\nabla F(x_1, x_2, \dots, x_n) = -\left[\frac{\partial F}{\partial x_1}, \frac{\partial F}{\partial x_2}, \dots, \frac{\partial F}{\partial x_n}\right] \tag{3}$$

W kolejnych krokach algorytmu oblicza się gradient funkcji oceny. Na podstawie wartości gradientu oblicza się nowe rozwiązanie. Kroki algorytmu powtarza się aż do osiągnięcia warunku końca określonego wzorem:

$$\nabla F * \nabla F = \|\nabla F\|^2 \leq \varepsilon \tag{4}$$

gdzie:
ε – założona dokładność obliczanego rozwiązania

Algorytm ewolucyjny

Algorytm ewolucyjny (ang. *evolutionary algorithm*) jest algorytmem iteracyjnym, obliczającym wiele możliwych rozwiązań danego problemu [2, 4]. Naśladuje on naturalną ewolucję. W każdej iteracji zwanej pokoleniem z grupy osobników (możliwych rozwiązań danego problemu) wybierane są osobniki do reprodukcji. Na ich podstawie generuje się nowe osobniki (nowe rozwiązania tego problemu), używając operatorów ewolucyjnych, takich jak krzyżowanie i mutacja. Powiększona populacja (zbiór wszystkich osobników) poddawana jest operatorowi selekcji, który wybiera osobniki do nowej populacji spośród dostępnych osobników. Po wyborze proces ewolucji się powtarza.

W zależności od rodzaju problemu do rozwiązania należy wybrać sposób reprezentacji osobników. W zadaniach identyfikacji parametrów modelu matematycznego silników indukcyjnych oraz optymalizacji parametrycznej układów z tymi silnikami osobniki są przedstawione jako zbiór wartości zmiennopozycyjnych. Istnieją zadania, takie jak np. problem komiwojażera (ang. TSP – *Travelling Salesman Problem*), gdzie osobniki są reprezentowane jako liczby stałopozycyjne dodatnie. Ponadto dla problemu zwanego dylematem więźnia (ang. *Prisoner's dilemma*) osobniki koduje się jako symbole lub dla innych zadań stosuje się kodowanie binarne. Operatory ewolucyjne są odpowiedzialne za generowanie nowych rozwiązań danego problemu i dzieli się je na:

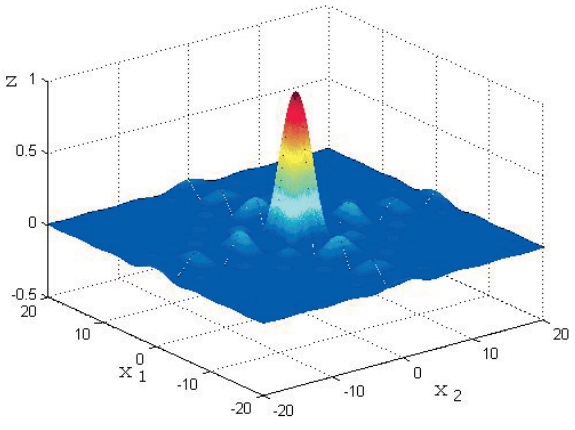
- mutację zmiennopozycyjną (operator nieznacznie zmieniający cechy osobnika). W wyniku jej działania otrzymujemy z jednego rodzica jednego potomka. Potomek powinien z większym prawdopodobieństwem znajdować się w otoczeniu osobnika, z którego powstał, a z mniejszym prawdopodobieństwem w znacznej odległości od niego. Operator musi jednak umożliwić stworzenie potomka w możliwie największym obszarze wokół rodzica, z odpowiednio dobranym prawdopodobieństwem,
- mutację postępową zmiennopozycyjną (odmiana mutacji zmiennopozycyjnej). Wybrany w sposób losowy osobnik próbuje, wykonując pewien krok w losowym kierunku, polepszyć swoją funkcję oceny. Jeżeli krok przyniósł pozytywny rezultat, czyli uzyskano zwiększenie wartości funkcji oceny, to krok ten jest zachowywany, w przeciwnym przypadku osobnik powraca do miejsca wyjścia,
- krzyżowanie zmiennopozycyjne (operator wieloargumentowy, tzn. z kilku osobników, najczęściej dwóch, tworzy się rozwiązanie lub rozwiązania potomne),
- selekcję (operator odpowiedzialny za wybór osobników do nowej populacji spośród osobników z populacji powiększonej o osobniki będące wynikiem operatorów mutacji i krzyżowania). Najczęściej stosuje się selekcję metodą ruletki, turnieju, metodą deterministyczną.

Maksymalizacja funkcji dwóch zmiennych

Do rozważań przyjęto funkcję $F(x_1,x_2)$ opisaną wzorem:

$$F(x_1, x_2) = \frac{\sin(x_2)}{x_1}\frac{\sin(x_2)}{x_2} \tag{5}$$

w przedziałach:
$x_1 = \langle -20,20 \rangle$
$x_2 = \langle -20,20 \rangle$
przy czym punkt $[x_{10}, x_{20}]=[0,0]$ nie należy do dziedziny tej funkcji. Wykres rozpatrywanej funkcji przedstawiono na rys 1.



Rys. 1. Wykres rozpatrywanej funkcji $F(x_1,x_2)$

Jak wynika z rys 1, rozpatrywana funkcja F w rozpatrywanej dziedzinie nie ma maksimum globalnego (punkt [0,0] nie należy do dziedziny funkcji), lecz wiele maksimów lokalnych, co stanowi dodatkową trudność. W niniejszej pracy zajmowano się problemem maksymalizacji funkcji $F(x_1,x_2)$. Jako rozwiązanie należy przyjąć punkty w otoczeniu punktu [0,0] o wartościach bliskich 1. Każdy z algorytmów uruchamiano 10-krotnie.

Symulowane wyżarzanie

Parametry algorytmu zawarto w tabeli 1.

Tab. 1. Parametry algorytmu symulowanego wyżarzania

Maksymalny czas działania algorytmu	10 s
Temperatura początkowa	100
Temperatura końcowa	95
Prędkość chłodzenia	0.96

Wyniki obliczeń zawarto w tabeli 2.

Tab. 2. Wyniki maksymalizacji funkcji F przy zastosowaniu algorytmu symulowanego wyżarzania

Lp.	x_1	x_2	$F(x_1,x_2)$
1.	–4.75	–5.28	0.0335961
2.	–13.94	–0.31	0.0692249
3.	0.45	–19.99	0.0439447
4.	17.25	–10.84	0.0052811
5.	–14.25	–0.72	0.0638589
6.	–19.86	0.14	0.0425127
7.	–7.82	8.24	0.0143688
8.	–13.57	–7.84	0.0079270
9.	–10.99	4.92	0.0180968
10.	10.85	4.47	0.0198043

Jak wynika z tabeli 2, punkt o możliwie największej wartości funkcji oceny F jest punktem z pozycji 2. Wartość funkcji w tym punkcie jest, z uwagi na wartości funkcji w całym przedziale poszukiwań, znacznie mniejsza niż 1. Punktem startowym algorytmu był każdorazowo losowo wybrany punkt, mieszczący się w założonej dziedzinie funkcji. Z uwagi na swój charakter działania i znaczną liczbę maksimów lokalnych, algorytm „utykał" za każdym razem w maksimum lokalnym. Nie potrafił „samodzielnie" ominąć tych maksimów. Do rozwiązywania tego i podobnych problemów ten rodzaj algorytmu nie jest trafnym wyborem.

Algorytm zachłanny

Parametr algorytmu umieszczono w tabeli 3.

Tab. 3. Parametr algorytmu zachłannego

Maksymalny czas działania algorytmu	10 s

Wyniki obliczeń zestawiono w tabeli 4.

Tab. 4. Wyniki maksymalizacji funkcji *F* przy zastosowaniu algorytmu zachłannego

Lp.	x_1	x_2	$F(x_1,x_2)$
1.	19.67	–0.07	0.0371556
2.	–17.23	–11.68	0.0038453
3.	–13.78	–19.83	0.0028483
4.	11.24	–11.02	0.0078310
5.	–5.02	4.64	0.0408093
6.	13.04	13.65	0.0022644
7.	–10.85	–10.82	0.0082984
8.	–4.74	11.23	0.0182655
9.	0.10	–0.45	0.9649800
10.	1.11	0.04	0.8067210

Algorytm zachłanny, podobnie jak algorytm symulowanego wyżarzania, jako punkt startowy przyjmuje losowo wybrany punkt z dziedziny funkcji *F*. Wobec powyższego, wynik silnie zależy od tego punktu. Najlepszym rozwiązaniem jest rozwiązanie z pozycji 9 z tabeli 4. I jest ono akceptowalne ze względu na wartość funkcji oceny. Należy zwrócić uwagę na fakt, że algorytm kończył swoje działanie dużo szybciej niż algorytm symulowanego wyżarzania.

Metoda największego spadku

Jedynym parametrem jest z góry ograniczony czas działania algorytmu. Czas ten został ustawiony na 10 s, podobnie jak dla poprzednich metod. Wyniki obliczeń zestawiono w tabeli 5.

Tab. 5. Wyniki maksymalizacji funkcji *F* przy zastosowaniu metody największego spadku

Lp.	x_1	x_2	$F(x_1,x_2)$
1.	4.01	–10.60	0.0165707
2.	0.21	–19.78	0.0402429
3.	0.10	0.51	0.9556160
4.	16.40	–11.06	0.0035107
5.	4.59	17.37	0.0123970
6.	–14.34	–0.84	0.0605514

7.	14.42	–0.91	0.0577750
8.	14.15	–19.10	0.0009169
9.	7.66	–13.31	0.0065153
10.	4.68	–16.43	0.0085908

Algorytm wykorzystujący metodę największego spadku, tak jak poprzednie dwa algorytmy, jako punkt startowy przyjmował losowo wybrany punkt z przestrzeni poszukiwań. Podobnie jak poprzednio wynik końcowy zależał w znacznym stopniu od punktu początkowego. Najlepszym rozwiązaniem jest wynik z pozycji 3 z tabeli 5. Wynik ten uważa się za zadowalający. Czas działania algorytmu zawiera się między czasem działania algorytmu symulowanego wyżarzania i algorytmu zachłannego.

Algorytm ewolucyjny

Parametry algorytmu zestawiono w tabeli 6.

Tab. 6. Parametry algorytmu ewolucyjnego

Liczba pokoleń	200
Wielkość populacji	200
Liczba krzyżowań na pokolenie	30
Liczba mutacji na pokolenie	40

Wyniki obliczeń zawarto w tabeli 7.

Tab. 7. Wyniki maksymalizacji funkcji *F* przy zastosowaniu algorytmu ewolucyjnego

Lp.	x_1	x_2	$F(x_1,x_2)$
1.	0.02	–0.02	0.999837
2.	–2.01*10–4	–4.43*10–5	0.999999
3.	–2.04*10–4	1.59*10–4	0.999999
4.	1.12*10–4	–1.45*10–6	0.999999
5.	2.70*10–4	9.50*10–5	0.999999
6.	5.76*10–5	3.47*10–4	0.999999
7.	2.58*10–4	7.83*10–5	0.999999
8.w	–5.45*10–5	–3.24*10–4	0.999999
9.	3.36*10–4	1.40*10–4	0.999999
10.	–3.38*10–5	–2.37*10–4	0.999999

Punkty obliczone przez algorytm ewolucyjny w każdej próbie mieściły się w otoczeniu punktu [0,0]. Granica funkcji *F* jest równa

$$\lim_{\substack{x_1 \to 0 \\ x_2 \to 0}} F(x_1, x_2) = 1 \tag{6}$$

Rozwiązania przedstawione w pozycjach od 2 do 10 z tabeli 7, przy przyjętej dokładności obliczeń, należy traktować jako jednakowo dokładne. Wartość znalezionego maksimum z pozycji 1 z tabeli 7 jest mniejsza niż przedstawionych w pozycjach od 2 do 10 tej samej tabeli, ale uznaje się go za zadowalający. Algorytm ewolucyjny mimo znacznie dłuższego czasu działania niż pozostałe algorytmy znalazł każdorazowo rozwiązania lepsze (o wyższej wartości funkcji oceny). Jedną z zalet algorytmów ewolucyjnych jest „omijanie" lokalnych maksimów i znajdowanie punktów w otoczeniu punktu maksimum globalnego. Do maksymalizacji funkcji $F(x_1,x_2)$ i podobnych o znacznej liczbie maksimów i minimów lokalnych spośród rozpatrywanych metod najlepiej nadaje się algorytm ewolucyjny.

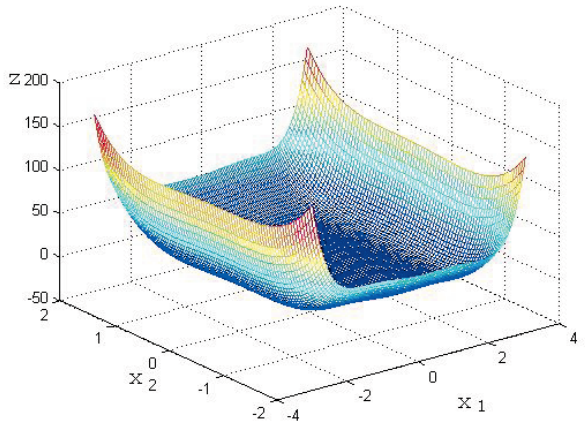
Minimalizacja funkcji dwóch zmiennych

Do rozważań przyjęto funkcję $G(x_1, x_2)$ (ang. *Six-hump camel back function*) opisaną wzorem:

$$G(x_1,x_2) = (4 - 2.1 \cdot x_1^2 + x_1^{4/3}) \cdot x_1^2 + x_1x_2 + (-4 + 4 \cdot x_2^2) \cdot x_2^2 \qquad (7)$$

w przedziałach zmiennych niezależnych:
$x_1 = \langle -3,3 \rangle$
$x_2 = \langle -2,2 \rangle$

Wykres rozpatrywanej funkcji przedstawiono na rys 2.



Rys. 2. Wykres rozpatrywanej funkcji $G(x_1,x_2)$

Jak wynika z rys. 2, rozpatrywana funkcja G ma minimum globalne, równe w zaokrągleniu –1.03163. W obszarze wokół minimum globalnego znajduje się obszar „spłaszczenia" wykresu funkcji G, co stanowi dodatkową trudność. W dalszej części pracy rozpatrywano problem minimalizowania funkcji $G(x_1,x_2)$. Każdy z algorytmów uruchamiano 10-krotnie.

Symulowane wyżarzanie

Parametry algorytmu zawarto w tabeli 1. Wyniki obliczeń przedstawiono w tabeli 8.

Tab. 8. Wyniki minimalizacji funkcji *G* przy zastosowaniu algorytmu symulowanego wyżarzania

Lp.	x_1	x_2	$G(x_1,x_2)$
1.	1.660	–0.340	1.06305
2.	1.540	0.840	2.58054
3.	1.200	–0.820	0.54161
4.	0.150	–0.720	–1.01632
5.	–1.680	–0.330	2.22467
6.	–1.960	–1.140	7.01942
7.	1.490	0.450	2.19854
8.	–1.100	1.320	6.02654
9.	–0.730	–0.470	1.23411
10.	0.350	–0.790	–0.75367

Rozwiązaniami rozpatrywanego zadania o najmniejszej wartości funkcji oceny są rozwiązania o liczbie porządkowej 4 oraz 10 z tabeli 8. Z uwagi na założoną dokładność akceptowalnym rozwiązaniem jest tylko rozwiązanie 4 z tabeli 8.

Algorytm zachłanny

Parametr algorytmu przedstawiono w tabeli 3. Wyniki obliczeń zestawiono w tabeli 9.

Tab. 9. Wyniki minimalizacji funkcji G przy zastosowaniu algorytmu zachłannego

Lp.	x_1	x_2	$G(x_1,x_2)$
1.	–1.710	0.860	–0.16629
2.	0.102	–0.765	–1.00759
3.	–1.718	0.714	–0.14358
4.	1.770	–0.764	–0.15442
5.	0.076	–0.813	–0.93511
6.	1.638	–0.229	1.47915
7.	–1.548	–0.579	2.11806
8.	1.759	–0.594	0.18766
9.	0.610	–0.674	–1.01790
10.	1.620	0.427	2.15446

Stosując algorytm zachłanny, otrzymano dwa rozwiązania akceptowalne (pozycje 2 oraz 9 z tabeli 9). Jak wynika z tabel 8 i 9, w wyniku działania algorytmu zachłannego otrzymano punkty o mniejszych wartościach ich funkcji oceny niż w wyniku działania algorytmu symulowanego wyżarzania.

Metoda największego spadku

Jedynym parametrem jest z góry ograniczony czas działania algorytmu. Czas ten został ustawiony na 10 s podobnie jak dla poprzednich metod. Wyniki obliczeń zestawiono w tabeli 10.

Tab. 10. Wyniki minimalizacji funkcji *G* przy zastosowaniu metody największego spadku

Lp.	x_1	x_2	$G(x_1,x_2)$
1.	0.123	–0.698	–1.02516
2.	1.590	–0.820	–0.10828
3.	1.613	0.591	2.10653
4.	–1.625	–0.554	2.10639
5.	–0.045	0.724	–1.02388
6.	–0.082	0.732	–1.02809
7.	–1.701	0.770	–0.20807
8.	1.698	–0.792	–0.21500
9.	–0.133	0.690	–1.01938
10.	–0.100	0.672	–1.01803

W wyniku pracy algorytmu opartego na metodzie największego spadku otrzymano 5 rozwiązań akceptowalnych ze względu na przyjętą dokładność obliczeń, co jest rezultatem lepszym od wyników otrzymanych przy zastosowaniu algorytmu symulowanego wyżarzania i algorytmu zachłannego.

Algorytm ewolucyjny

Parametry algorytmu podano w tabeli 6. Wyniki obliczeń zamieszczono w tabeli 11.

Tab. 11. Wyniki minimalizacji funkcji *G* przy zastosowaniu algorytmu ewolucyjnego

Lp.	x_1	x_2	$G(x_1,x_2)$
1.–10.	0.090	–0.713	–1.03163

Dla przyjętej dokładności obliczeń, po zaokrągleniach, wyników poszczególnych ewolucji nie da się odróżnić od siebie. Wszystkie są akceptowalne, bliskie rozwiązaniu najlepszemu (po zaokrągleniu równe rozwiązaniu najlepszemu). Dla tego zadania algorytm ewolucyjny okazał się najbardziej efektywny spośród rozpatrywanych metod.

Podsumowanie

W pracy porównano cztery metody optymalizacyjne z zastosowaniem różnych algorytmów: algorytmu symulowanego wyżarzania, algorytmu zachłannego, metody największego spadku oraz algorytmu ewolucyjnego. Rozpatrywano dwie różne funkcje o wielu ekstremach lokalnych. Jedna z nich była funkcją nieciągłą. W pierwszym przypadku rozpatrywano maksymalizowanie wskaźnika jakości, w drugim minimalizowanie wskaźnika jakości. Dla każdego zadania poszczególne algorytmy uruchamiano 10-krotnie. Wyniki przedstawiono w tabelach. Okazuje się, że dla rozpatrywanych zadań najlepszą metodą optymalizacyjną jest algorytm ewolucyjny. Mimo znacznie dłuższego czasu działania niż pozostałe algorytmy odnalazł każdorazowo akceptowalne rozwiązania, z uwagi na przyjętą dokładność. Rozwiązania te nie są optymalne, ale nadają się jako dane wejściowe dla innych metod optymalizacyjnych, w tym prezentowanych w niniejszym artykule.

Literatura

[1] Bang-Jensen J., Gutin G., Yeo A., *When the greedy algorithm fails*, Discrete Optimization 1, 2004, s. 121–127

[2] Goldberg D.E., *Algorytmy genetyczne i ich zastosowania*, WNT, Warszawa 2003

[3] Kirkpatrick S., Gelatt C.D., Vecchi M.P., *Optimization by Simulated Annealing*, Science 220 1983, s. 671–680,

[4] Michalewicz Z., *Algorytmy genetyczne + struktury danych = programy ewolucyjne*, WNT, Warszawa 2003

Comparing operational research algorithms: simulated annealing, greedy algorithm, steepest descent and evolutionary algorithm

Abstract

In this paper four algorithms are compared: simulated annealing, greedy algorithm, steepest descent, and evolutionary algorithm. The target of using the algorithms is assignment of the global extreme of the examined process. Two functions were analyzed with many numbers of local maximum and minimum. One of them is discontinuous function. They made for 10 independent tests. The results were presented in tables.

Keywords: simulated annealing, greedy algorithm, steepest descent, evolutionary algorithm

Wiktor Hudy
Uniwersytet Pedagogiczny w Krakowie
Instytut Techniki
ul. Podchorążych 2
30–084 Kraków