Adam SŁOWIK, Marek POPŁAWSKI

Politechnika Koszalińska, Wydział Elektroniki i Informatyki, Poland

doi:10.15199/48.2018.07.18

Zastosowanie kartezjańskiego programowania genetycznego do projektowania filtrów cyfrowych do przetwarzania obrazów

Streszczenie. *W niniejszej pracy przedstawiono zastosowanie kartezjańskiego programowania genetycznego do projektowania filtrów cyfrowych do przetwarzania obrazów. Prezentowana metoda umożliwia projektowanie filtrów cyfrowych zarówno do przetwarzania obrazów kolorowych oraz do przetwarzania obrazów w odcieniach szarości. Jakość zaprojektowanych filtrów sprawdzono na przykładzie redukcji losowo generowanego szumu na wybranych obrazach. Wyniki uzyskane przy użyciu filtrów cyfrowych zaprojektowanych z wykorzystaniem kartezjańskiego programowania genetycznego porównano z wynikami uzyskanymi przy użyciu standardowych filtrów typu maksimum, typu minimum oraz przy użyciu filtrów medianowych. Cyfrowe obrazy poddane filtracji przy użyciu filtrów otrzymanych z wykorzystaniem kartezjańskiego programowania genetycznego cechują się mniejszym stopniem zaszumienia niż cyfrowe obrazy przetwarzane przy użyciu pozostałych filtrów cyfrowych.*

Abstract. *In this paper, the application of the Cartesian genetic programming to design of digital filters for image processing is presented. The digital filters for color images processing and the digital filters for shades of grey images processing can be designed by the use of the proposed method. The quality of designed filters was tested on the example of the reduction of randomly generated noise on the selected images. The results obtained using digital filters designed with the use of Cartesian genetic programming were compared with the results obtained using standard digital filters such as maximum filter, minimum filter, and median filter. The digital images subjected to filtration using digital filters designed using Cartesian genetic programming possesses less degree of noise than digital images which were processed with the use of other digital filters. (**Application of Cartesian Genetic Programming to Design of Digital Filters for Image Processing**).*

Słowa kluczowe: kartezjańskie programowanie genetyczne, projektowanie, filtry cyfrowe, przetwarzanie obrazów
Keywords: Cartesian genetic programming, design, digital filters, image processing

Wstęp

Filtracja obrazów cyfrowych jest zagadnieniem podstawowym w procesie przetwarzania obrazów. Istnieje duża liczba filtrów cyfrowych spełniających określone funkcje. Pośród takich filtrów możemy wymienić filtry typu minimum, filtry typu maksimum, filtry medianowe, filtry służące do detekcji krawędzi oraz wiele innych. Typowe filtry cyfrowe do przetwarzania obrazów bazują na różnego rodzaju maskach, w których zapisane są wartości wag dla poszczególnych pikseli. Również od pewnego czasu do tworzenia filtrów cyfrowych o zadanych cechach wykorzystuje się algorytmy ewolucyjne [1, 2, 17].

Wśród szeroko rozumianych algorytmów ewolucyjnych można także wymienić Programowanie Genetyczne (ang. Genetic Programming) [3, 4], którego pierwotną ideą była możliwość automatycznego tworzenia programów komputerowych służących do rozwiązywania określonych problemów przez odpowiednio zaprogramowane inne programy komputerowe.

Pod koniec lat 90-tych XX wieku rozwinęła się nowa koncepcja bazująca na Programowaniu Genetycznym, która została nazwana Kartezjańskim Programowaniem Genetycznym (ang. Cartesian Genetic Programming – CGP) [5, 6]. Jej twórcą jest Julian Miller. Generalnie CGP powstało jako naturalna konsekwencja zastosowania przetwarzania ewolucyjnego do projektowania układów cyfrowych [7, 8, 9].

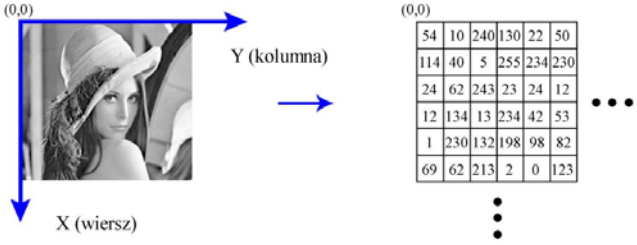
Od momentu utworzenia algorytmu CGP znalazł on liczne zastosowania praktyczne wśród których wymienić możemy np.: automatyczną klasyfikację anomalii występujących na zdjęciach mammograficznych [10], oraz projektowanie kontrolera robota [11].

W pracy [10] wykazano, że technika CGP może być stosowana do przetwarzania i klasyfikacji obrazów. Oczywiście istnieje duża liczba algorytmów służących do przetwarzania obrazów. Wśród algorytmów przetwarzania obrazów szczególne miejsce zajmują algorytmy filtracji, które znajdują zastosowanie w przeważnie każdym systemie związanym z przetwarzaniem obrazów. Pośród algorytmów filtracji można wymienić: filtry typu maksimum, filtry typu minimum oraz filtry typu medianowego. W niniejszym artykule postanowiono zastosować technikę CGP do automatycznego tworzenia filtrów cyfrowych wspomagających przetwarzanie obrazów. Przedstawiona metoda, którą nazwano CGPDFD (ang. Cartesian Genetic Programming for Digital Filters Design) pozwala na tworzenie filtrów cyfrowych dla obrazów kolorowych (jako model barw przyjęto model RGB – ang. Red Green Blue) oraz tworzenie filtrów cyfrowych dla obrazów w odcieniach szarości. Jakość utworzonych filtrów cyfrowych sprawdzono przy użyciu wybranych obrazów z ustalonym stopniem zaszumienia. Otrzymane rezultaty (filtracji wybranych obrazów) porównano z wynikami otrzymanymi przy użyciu filtrów typu maksimum, filtrów typu minimum oraz filtrów medianowych. Struktura artykułu jest następująca. W sekcji drugiej przedstawiono elementarne informacje związane z obrazami cyfrowymi, w sekcji trzeciej ukazano w skrócie algorytm CGP, w sekcji czwartej przedstawiono utworzoną metodę CGPDFD, w sekcji piątej ukazano reprezentację osobników w utworzonej metodzie CGPDFD, w sekcji szóstej przedstawiono funkcję celu przyjętą w metodzie CGPDFD, w sekcji siódmej zawarto wyniki eksperymentów. Ostatnia część niniejszego artykułu zawiera podsumowanie.

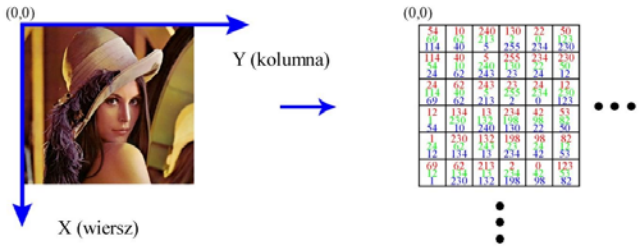
Obrazy cyfrowe

Obraz cyfrowy jest dwuwymiarową tablicą (o X wierszach i Y kolumnach) zwierającą wartości dodatnie i skończone. Punktom obrazu przypisuje się wartości funkcji *F*, która reprezentuje dany kolor.

Na rysunku 1 przedstawiono przykładowy obraz w odcieniach szarości łączenie z jego liczbową reprezentacją a na rysunku 2 przedstawiono obraz kolorowy także z jego reprezentacją liczbową (model RGB).



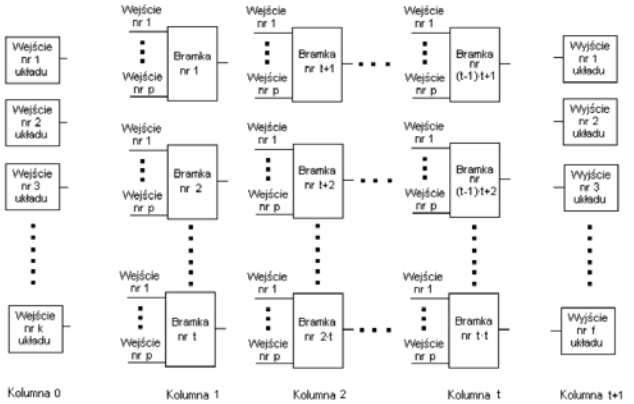
Rys. 1. Obraz w odcieniach szarości łącznie z jego liczbową reprezentacją [12]



Rys. 2. Obraz kolorowy łączenie z jego liczbową reprezentacją (model RGB) [12]

Algorytm CGP

Algorytm CGP pochodzi od algorytmu służącego do automatycznego projektowania układów cyfrowych przy użyciu technik ewolucyjnych. W tego rodzaju podejściu każdy układ reprezentowany jest przez szablon bramek przedstawiony na rysunku 3.



Rys. 3. Szablon bramek (metody ewolucyjnego projektowania układów cyfrowych) [13]

Koncepcja przedstawiona na rys. 3 polega na tym, że algorytm ewolucyjny poszukuje takich połączeń pomiędzy wejściami układu a jego wyjściami, oraz ustala takie funkcje logiczne dla każdej bramki z szablonu, aby dany układ spełniał zadaną przez użytkownika tablicę prawdy. Oczywiście mogą istnieć tutaj różnego rodzaju kryteria optymalności jedno- lub wielokryterialnej. Algorytm może dążyć do minimalizacji liczby bramek [14], do minimalizacji liczby tranzystorów [15] czy do jednoczesnej minimalizacji liczby tranzystorów oraz czasu propagacji sygnału przez dany układ cyfrowy [16].

Taka koncepcja (patrz rys. 3) została przyjęta w metodzie CGP w której także dąży się do ustalenia takich połączeń pomiędzy „bramkami" (w metodzie tej zamiast bramek logicznych występują funkcje matematyczne zdefiniowane przez użytkownika) i na takim doborze funkcji matematycznych, aby dana sieć spełniała przyjęte założenia projektowe. Proces przetwarzania informacji w metodzie CGP jest identyczny z procesem przetwarzania informacji w standardowym algorytmie ewolucyjnym.

Proponowana metoda CGPDFD

Proponowana metoda CGPDFD składa się z następujących kroków:

Krok 1: Utwórz *N* osobników z których każdy reprezentuje sobą szablon przedstawiony na rys. 3 i jest kodowany w formie chromosomu zawierającego kolejno wszystkie informacje przedstawione w szablone. W metodzie CGPDFD przyjęto, że wszystkie bramki (funkcje) są maksymalnie dwuwejściowe (dwuargumentowe). Wykaz wszystkich dostępnych funkcji z jakich może się składać tworzony filtr cyfrowy przedstawiono w tabeli 1.

Tabela 1. Zestaw funkcji przyjęty w metodzie CGPDFD

Numer funkcji	Realizowana funkcja (typ bramki)	Opis wartości wystawianej na wyjście bramki
0	255	Stała
1	x	Wartość x
2	255 – x	Inwersja
3	x v y	Bitowy OR
4	Not(x) v y	Bitowy (NOT x) OR y
5	x ∧ y	Bitowy AND
6	Not(x ∧ y)	Bitowy NAND
7	Xor(x, y)	Bitowy XOR
8	x >> 1	Przesunięcie w prawo o 1 bit
9	x >> 2	Przesunięcie w prawo o 2 bity
10	Swap(x, y)	Zamiana wartości miejscami
11	x+y	Dodawanie
12	x+s*y	Dodawanie z nasyceniem
13	(x+y) >> 1	Średnia
14	Max(x, y)	Maksimum
15	Min(x, y)	Minimum

Krok 2: Odkoduj każdy filtr reprezentowany przez poszczególnego osobnika i oceń jego wartość poprzez porównanie obrazu wzorcowego z zaszumionym obrazem przetworzonym przy użyciu otrzymanego filtru.
Krok 3: Wybierz z populacji osobnika w którym zapisany jest najlepszy filtr (filtr usuwający najwięcej zakłóceń z przetwarzanego obrazu) i zapamiętaj go w zmiennej *TheBest*.
Krok 4: Sprawdź czy można zatrzymać działanie algorytmu. Jako warunek stopu przyjęto osiągnięcie przez algorytm zadanej przez użytkownika liczby pokoleń. Jeśli tak wówczas idź do kroku 11. Jeśli nie idź do kroku 5.
Krok 5: Dokonaj selekcji chromosomów do nowej populacji. Jako metodę selekcji przyjęto selekcję turniejową z wielkością grupy turniejowej równą 2 chromosomom.
Krok 6: Dokonaj krzyżowania osobników z prawdopodobieństwem *PK*. W metodzie zastosowano krzyżowanie proste 1 punktowe [2].
Krok 7: Dokonaj mutacji osobników z prawdopodobieństwem *PM*. W metodzie zastosowano mutację prostą jednopunktową [2].
Krok 8: Sprawdź poprawność utworzonych osobników potomnych. Jeśli osobnik nie jest poprawny dokonaj korekty. Sprawdzanie poprawności osobników polega na tym aby do bramek z *i*-tej kolumny nie były podpinane wyjścia bramek z kolumny *i*+1.
Krok 9: Sprawdź czy w populacji znajduje się osobnik lepszy lub taki sam jak osobnik w zmiennej *TheBest*. Jeśli tak zapamiętaj go w zmiennej *TheBest*. Jeśli nie za najgorszego osobnika w populacji wstaw osobnika zapamiętanego w zmiennej *TheBest*.
Krok 10: Idź do kroku 2.
Krok 11: Jako wynik działania algorytmu zwróć filtr zapisany w zmiennej *TheBest*.

Reprezentacja osobników w metodzie CGPDFD

Osobnik w proponowanej metodzie CGPDFD składa się z występujących po sobie trójek liczbowych. W pojedynczej trójce liczbowej zapisane są: typ realizowanej funkcji oraz dwa argumenty będące adresami zmiennych wykorzystywanych przez zadaną funkcję. Dane adresowe muszą spełniać następującą zależność:

$$n_i+n_n<C_{ij} \quad (1)$$

gdzie: n_i – liczba wejść programowanych zależna od rozmiaru maski przyjętej do przetwarzania obrazu (w niniejszej pracy zastosowano maskę 3 na 3 piksele, czyli jest 9 wejść programowanych), n_n – numer aktualnej bramki

w szablonie, C_{ij} – wartość genu połączenia. W przypadku, gdy wartość genu połączenia jest mniejsza od wielkości stosowanej maski, wówczas dana używana przez bramkę (funkcję) w szablonie pochodzi z wejścia (maski przetwarzającej obraz). Indeksy maski przedstawiono na rysunku 4.

0	1	2
3	4	5
6	7	8

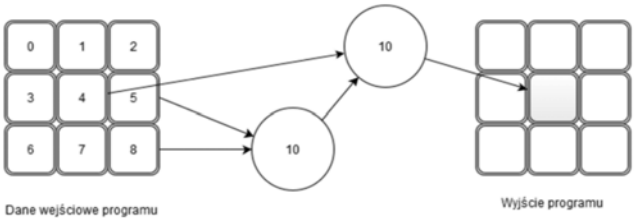
Rys. 4. Oznaczenie indeksów maski

Dla przykładu trójka liczbowa postaci <14; 0; 15> przedstawia sobą, że w danej bramce w szablonie realizowana jest funkcja numer 14 (patrz tabela 1 – funkcja maksimum). Pierwszy argument dla tej funkcji pochodzi z maski z pola o indeksie 0, a drugi argument jest wynikiem operacji wykonanej w bramce o indeksie 6 (w przypadku stosowania maski o wymiarze 3 na 3; 15 – 9 = 6).
W tabeli 2 przedstawiono przykładowego osobnika.

Tabela 2. Przykładowy osobnik złożony z 15 genów

Nr bramki	0			1			2			3			4		
Geny	3	0	1	5	4	3	10	8	5	6	1	4	10	11	4

Dekodując osobnika z tabeli 2 (dekodowanie przeprowadza się od końca osobnika) widać, że bramka numer 4 realizuje funkcję numer 10. Do funkcji numer 10 trafiają 2 argumenty: pierwszy z wyjścia bramki numer 2 (11 – 9 = 2), drugi z wejścia numer 4 (z danych z maski). Natomiast bramka numer 2 realizuje także funkcję numer 10. Tym razem jako argumenty do funkcji numer 10 trafiają dane: z wejścia numer 8 oraz z wejścia numer 5. W tym wypadku pozostałe geny już nie są brane pod uwagę. Na rysunku 5 przedstawiono filtr cyfrowy zapisany w osobniku z tabeli 2.



Rys. 5. Filtr cyfrowy zakodowany w tabeli 2 [12].

Funkcja celu

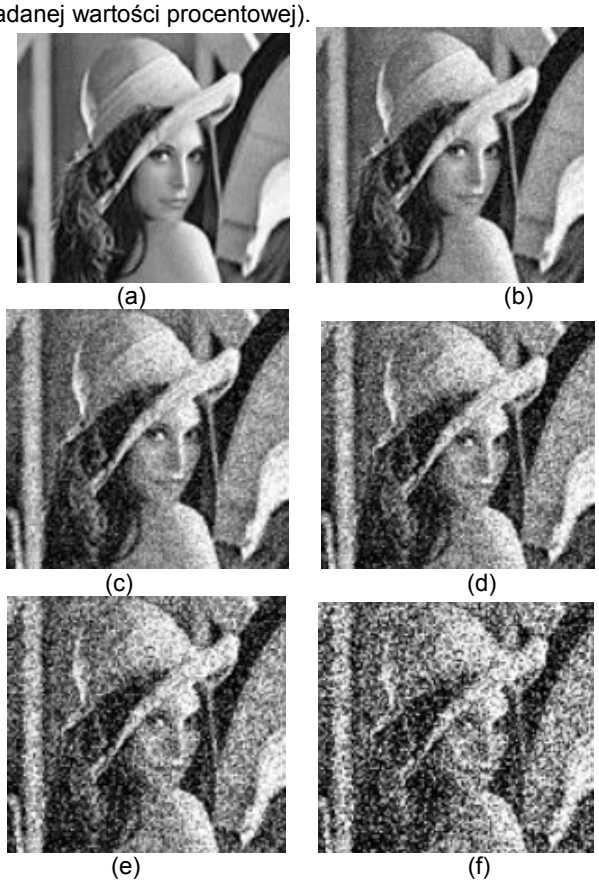
W proponowanej metodzie CGPDFD jako funkcję celu *FC* przyjęto średni błąd wartości pikseli pomiędzy generowanym obrazem *Fg*, a obrazem wzorcowym *Fw*, zgodnie z zależnością (2).

$$FC(O_i) = \frac{\sum_{x=0}^{X-1}\sum_{y=0}^{Y-1}\left|F_g(x,y) - F_w(x,y)\right|}{X \cdot Y} \tag{2}$$

Gdzie: O_i – i-ty osobnik, X – liczba wierszy w przetwarzanym obrazie, Y – liczba kolumn w przetwarzanym obrazie.
Podczas swojej pracy algorytm dąży do minimalizacji funkcji celu.

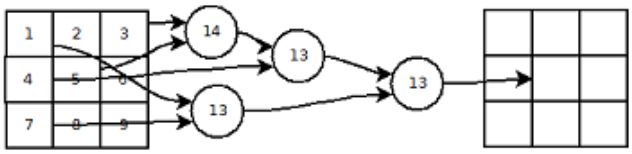
Przeprowadzone eksperymenty

W celu przeprowadzenia eksperymentów przyjęto następujące parametry w metodzie CGPDFD. Rozmiar szablonu bramek: 1 wiersz i 10 kolumn. Rozmiar populacji N=50, liczba pokoleń 1000, wielkość maski 3 na 3 piksele, współczynnik mutacji PM=0.3, współczynnik krzyżowania PK=0.7. Na rysunku 6 przedstawiono przyjęty obraz wzorcowy oraz obrazy zaszumione (szum typu sól-pieprz o zadanej wartości procentowej).



Rys. 6. Przyjęte obrazy: obraz wzorcowy bez szumu (a); obraz testowy z szumem: 5% (b), 10% (c), 15% (d), 20% (e), 30% (f)

Rys. 7. Uzyskane obrazy po filtracji wybranymi filtrami. Od lewej strony kolejno: filtr uzyskany metodą CGPDFD, filtr minimum, filtr maksimum, filtr medianowy (1 wiersz – szum 5%, 2 wiersz – szum 10%, 3 wiersz – szum 15%, 4 wiersz – szum 20%, 5 wiersz – szum 30%)



Rys. 8. Filtr cyfrowy otrzymany metodą CGPDFD dla obrazu testowego o stopniu zaszumienia 15%

Na rys. 7 przedstawiono kolejno (od lewej strony) efekty działania filtru: utworzonego przy użyciu metody CGPDFD, filtru minimum, filtru maksimum, filtru medianowego. Wartości liczbowe nad poszczególnymi obrazami wyznaczano przy użyciu zależności (2). Czym mniejsza wartość tym obraz jest mniej zaszumiony.

Z wyników przedstawionych na rysunku 7 widać, że filtry otrzymane przy użyciu proponowanej metody CGPDFD uzyskały najlepsze wartości (obraz został najlepiej odszumiony) dla poszczególnych obrazów testowych.

Dla przykładu na rysunku 8 ukazano strukturę otrzymanego filtru (przy użyciu metody CGPDFD) dla obrazu o stopniu zaszumienia 15%.

Metoda opisana w niniejszym artykule może być także stosowana do filtracji obrazów kolorowych. Wówczas przy użyciu metody CGPDFD tworzony jest zbiór 3 filtrów (każdy dla innej składowej koloru: Red, Green, Blue).

Podsumowanie

W niniejszym artykule przedstawiono metodę projektowania filtrów cyfrowych przeznaczonych do przetwarzania obrazów. Utworzona metoda bazuje na Kartezjańskim Programowaniu Genetycznym. Przy użyciu opisanej metody dokonano filtracji obrazów testowych wykazując, że filtry otrzymane metodą CGPDFD mogą efektywniej eliminować zakłócenia z obrazów cyfrowych w stosunku do filtrów typu maksimum, filtrów typu minimum i filtrów medianowych.

Podziękowanie

Autorzy pracy pragną podziękować Pani Dorocie Chyła za opracowanie aplikacji i przeprowadzenie testów opisanych w niniejszym artykule.

Autorzy:

dr hab. inż. Adam Słowik, dr inż. Marek Popławski, Politechnika Koszalińska, Wydział Elektroniki i Informatyki, Katedra Inżynierii Komputerowej, ul. Śniadeckich 2, 75-453 Koszalin, E-mail: aslowik@ie.tu.koszalin.pl

LITERATURA

[1] Michalewicz Z., Genetic Algorithms + Data Structures = Evolution Programs. Springer, Heidelberg (1992)
[2] Goldberg D.E., Genetic algorithms in search, optimization, and machine learning, Addison-Wesley Publishing Company Inc., New York (1989)
[3] Koza J., Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge, MA (1992)
[4] Arabas J., Wykłady z algorytmów ewolucyjnych. WNT, Warszawa (2001)
[5] Miller J. F., Turner A., Cartesian Genetic Programming. Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation, pp. 179-198, Madrid, Spain, (2015)
[6] Miller J. F., Thomson P., Cartesian Genetic Programming. Proceedings of the 3rd European Conference on Genetic Programming. Springer LNCS 1802, pp. 121-132, (2000)
[7] Kalganova T., Miller J. F, Fogarty T. C., Some Aspects of an Evolvable Hardware Approach for Multiple-Valued Combinational Circuit Design. Proceedings of the 2nd International Conference on Evolvable Systems: From Biology to Hardware (ICES98). Springer LNCS 1478, pp. 78-89, (1998)
[8] Vassilev V. K., Miller J. F., Fogarty T. C., On the Nature of Two-Bit Multiplier Landscapes. Proceedings of the First NASA/DOD Workshop on Evolvable Hardware (EH'99). IEEE Computer Society, pp. 36-45, (1999)
[9] Vassilev V. K., Miller J. F., The Advantages of Landscape Neutrality in Digital Circuit Evolution. Proceedings of the 3rd International Conference on Evolvable Systems: From Biology to Hardware. Springer LNCS 1801, pp. 252-263, (2000)
[10] Völk K., Miller J. F., Smith S. L., Multiple Networks CGP for the Classification of Mammograms. Proceedings of the 11th European Workshop on Image Analysis and Signal Processing (EvoIASP'09). Springer LNCS 5484, pp. 405-413. 5484, (2009)
[11] Harding S. L., Miller J. F., Evolution of Robot Controller Using Cartesian Programming. Proceedings of the 8th European Conference on Genetic Programming. Springer LNCS 3447 pp. 62-72, (2005)
[12] Chyła D., Ewolucyjne projektowanie filtrów do graficznej obróbki zdjęć. Praca inżynierska, Politechnika Koszalińska, Wydział Elektroniki i Informatyki, (2014)
[13] Słowik A., Projektowanie i optymalizacja cyfrowych układów elektronicznych przy użyciu algorytmów ewolucyjnych. Rozprawa doktorska. Politechnika Koszalińska, Wydział Elektroniki i Informatyki, (2007)
[14] Słowik A., Białko M., Design and Optimization of Combinational Digital Circuits Using Modified Evolutionary Algorithm. Proceedings of Seventh International Conference on Artificial Intelligence and Soft Computing, ICAISC 2004, Lecture Notes in Artificial Intelligence, Volume 3070/2004, pp. 468-473, Springer-Verlag, Zakopane, (2004)
[15] Słowik A., Białko M., Evolutionary Design and Optimization of Combinational Digital Circuits with Respect to Transistor Count. Bulletin of the Polish Academy of Sciences, Technical Sciences, Volume 54, Issue 4, pp. 437-442, (2006)
[16] Słowik A., Białko M., Design and Multi-Objective Optimization of Combinational Digital Circuits Using Evolutionary Algorithm with Multi-Layer Chromosomes. Ninth International Conference on Artificial Intelligence and Soft Computing, ICAISC 2008, Lecture Notes in Computer Science, Springer-Verlag, Volume 5097/2008, pp. 479-488, (2008)
[17] Słowik A., Białko M., Modified Version of Roulette Selection for Evolution Algorithm - The Fan Selection, Proceedings of Seventh International Conference on Artificial Intelligence and Soft Computing, ICAISC 2004, Lecture Notes in Artificial Intelligence, Volume 3070/2004, pp. 474-479, Springer-Verlag, Zakopane, (2004)