Dostrajanie klasyfikatora FUZZY LOGIC z użyciem wybranej procedury optymalizacji inspirowanej naturą.

Projekt wykonany w ramach przedmiotu Metody inteligencji obliczeniowej

Mirosław Kołodziej

Mateusz Nizio

13.06.2022

1. Opis i założenia projektu

Głównym założeniem projektu było wykorzystanie układu FUZZY LOGIC, którego zadaniem byłaby klasyfikacja danych (IRIS, WINE oraz SEEDS) ze zbioru benchmarkowego UCI MLR (http://archive.ics.uci.edu/ml/datasets.php).

Program miał wykorzystać ogólną zasadę budowy układu FL, a następnie doprecyzować parametry używając wybranego algorytmu z Evolutionary Computation Bestiary (https://github.com/fcampelo/EC-Bestiary). W naszym przypadku był to Crow Search Optimization.

Otrzymany rezultat należało porównać z rozwiązaniem generowanym automatycznie.

2. Realizacja projektu

Link do repozytorium: https://github.com/mirek0707/WFiIS-MIO-PROJEKT

Program został napisany za pomocą oprogramowania MATLAB w wersji R2020a oraz R2022a. W projekcie skorzystaliśmy z następujących toolboxów:

- Fuzzy Logic Toolbox
- Bioinformatics Toolbox
- Deep Learning Toolbox

Realizację projektu rozpoczęliśmy od wczytania i zainicjalizowania zbiorów z danymi. Działaliśmy na następujących zbiorach:

- IRIS – każda próbka opisywana jest za pomocą czterech cech (układ FL będzie miał 4 wejścia)
- WINE – każda próbka opisywana jest za pomocą trzynastu cech.

- SEEDS – każda próbka opisywana jest za pomocą siedmiu cech.

W kolejnym kroku randomizujemy kolejność danych, a następnie dzielimy dane na dwa zbiory, testujący i uczący w proporcji 1:4.

Po uzyskaniu zbiorów, inicjalizujemy Fuzzy Inference System z użyciem opcji Subtractive Clustering, która zapewniła nam automatyczną budowę układu rozmytego za pomocą metody klasteryzacji. Następnie sprawdzamy skuteczność klasyfikacji i wypisujemy ją w konsoli dla zbioru testującego i uczącego oraz generujemy macierz pomyłek. Wykonujemy również walidację za pomocą algorytmu Cross-Validation CV-5.

W kolejnym kroku skorzystaliśmy z funkcji *getTunableValues*, która pozwoliła nam na uzyskanie parametrów systemu wnioskowania rozmytego FIS. Przekazaliśmy je do algorytmu Crow Search Optimization(CSO):

https://www.mathworks.com/matlabcentral/fileexchange/56127-crow-search-algorithm?s_tid=srchtitle_crow_1

za pomocą którego dokonaliśmy ich optymalizacji. Musieliśmy jednak lekko zmodyfikować wybrany przez nas algorytm, ze względu na to, że w oryginalnej wersji pracował on na losowo generowanych danych.

Algorytm na wejściu tworzy 20 kopii parametrów a następnie dodaje do nich szum, aby nie powtórzyć wyniku uzyskanego przed działaniem CSO. Później w 10 (lub innej wybranej przez nas liczbie) iteracjach próbuje uzyskać jak najlepszy wynik, którego przystosowanie sprawdza za pomocą napisanej przez nas funkcji fitness, zwracającej błąd przystosowania.

Po zakończeniu działania algorytmu CSO, korzystamy z funkcji *setTunableValues*, która pozwala nam na użycie nowo uzyskanych parametrów w systemie wnioskowania rozmytego FIS. Na koniec ponownie sprawdzamy skuteczność klasyfikacji, wyświetlamy macierz pomyłek oraz wykonujemy walidację przy użyciu CV-5.

Każde z podejść (użycie rozwiązania automatycznego, użycie algorytmu CSO), testowaliśmy co najmniej trzykrotnie, dla zróżnicowania wyników. Edytowaliśmy także parametry naszego algorytmu, aby zbadać jak wpłynie to na końcowe rozwiązanie.

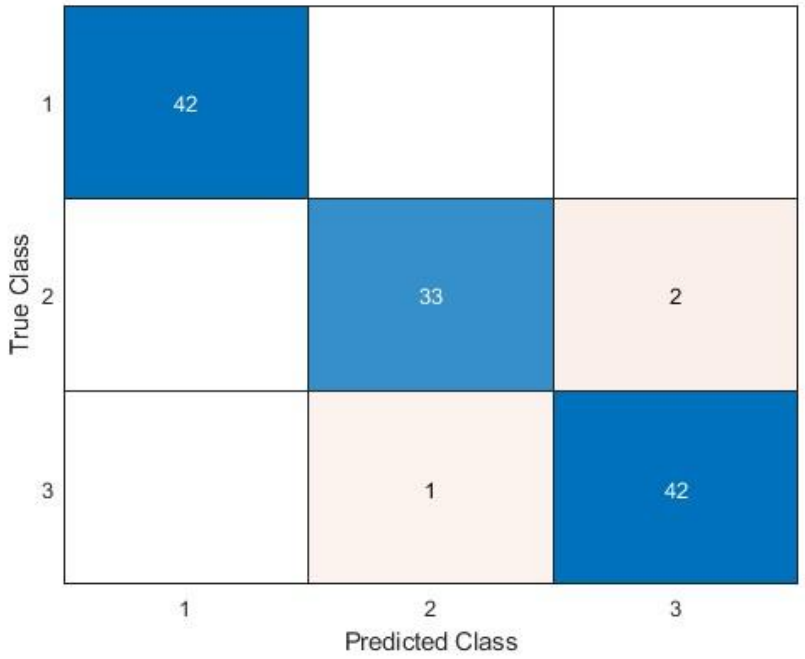
3. Opis wybranego algorytmu

Algorytm Crow Search Optimization jest jednym z metaheurystycznych algorytmów optymalizujących. Symuluje on zachowania wron w składowaniu nadmiaru jedzenia i odnajdywaniu go, gdy nadejdzie taka potrzeba. W tej metodzie optymalizacji wrona poszukuje jedzenia, jej otoczenie jest przestrzenią poszukiwań, zaś losowe przechowywanie lokalizacji jedzenia jest prawdopodobnym rozwiązaniem. Lokalizacja z największą ilością jedzenia spośród wszystkich jest rozważana jako globalnie optymalne rozwiązanie, natomiast jako funkcję celu traktujemy ilość jedzenia. Poprzez symulację inteligentnego zachowania wron CSO próbuje znaleźć optymalne rozwiązanie dla różnych problemów optymalizacji. Algorytm zyskał na popularności ze względu na prostotę implementacji, małą ilość parametrów oraz elastyczność.

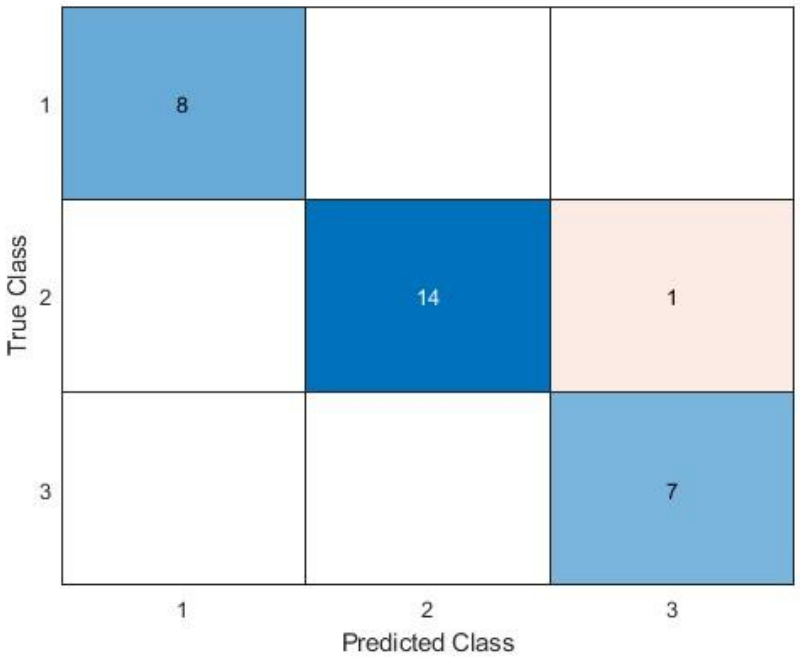
4. Wyniki

4.1 Zbiór IRIS

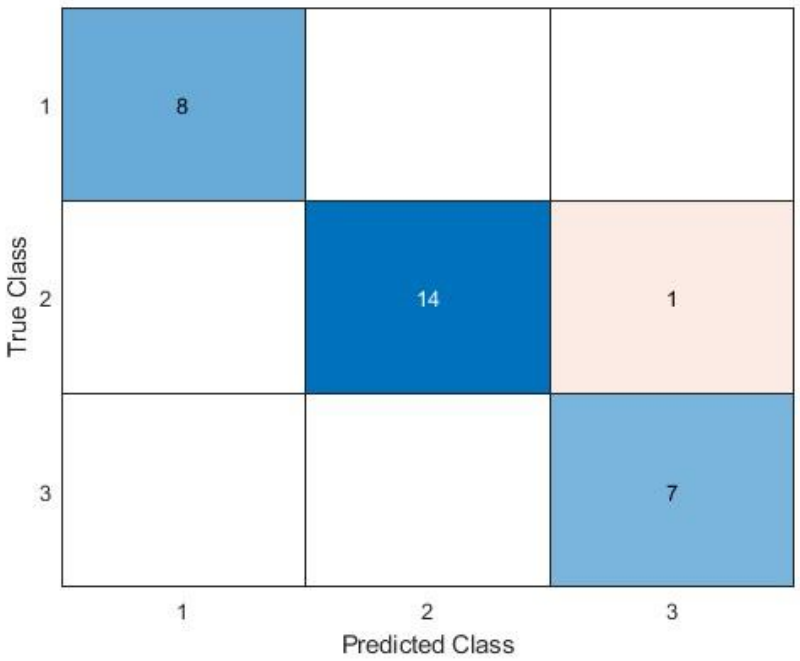
Użyteczność naszego rozwiązania testowaliśmy na kilka sposobów. Najpierw sprawdziliśmy poprawność wyniku generowanego automatycznie (Substractive Clustering) zarówno na zbiorze testującym jak i uczącym. Następnie zbadaliśmy jak zastosowanie algorytmu CSO wpłynie na jakość rozwiązania. Każdy program uruchomiliśmy trzykrotnie dla zapewnienia większej różnorodności rozwiązań. Zbiór IRIS składa się ze 150 osobników, należących do trzech różnych klas, a każdy z nich reprezentowany jest przez 4 różne cechy.



rys1. Macierz pomyłek dla rozwiązania automatycznego oraz zbioru uczącego (zbiór IRIS)



rys2. Macierz pomyłek dla rozwiązania automatycznego oraz zbioru testującego (zbiór IRIS)



rys3. Macierz pomyłek dla algorymtu CSO oraz zbioru testującego (zbiór IRIS)

	Substractive Clustering			CSO	
num.	zbiór uczący	zbiór testujący	błąd średniokwadratowy	zbiór testujący	błąd średniokwadratowy
1	97.5	96.667	0.0506	93.333	0.045
2	95.833	96.667	0.0352	98	0.044
3	96.677	100	0.09	96.667	0.049

tab1.

suma	348.012	88.00	-	86.4	-
średnia	116.0	29.33	-	28.8	-
odch. stand.	0.680	1.57	-	1.96	-

tab2.

Powyższa tabela **(tab1)** przedstawia procent poprawnie zakwalifikowanych przez program osobników do danego zbioru oraz obliczony błąd średniokwadratowy rozwiązania. Widzimy zatem, że dla pierwszej iteracji algorytm podał poprawne rozwiązanie ewaluując zbiór testujący w ponad 96% przypadków. Dla drugiej iteracji algorytm CSO poprawnie wybrał aż 98% osobników.

W tabeli **(tab2**) możemy także odczytać łączną sumę poprawnie zakwalifikowanych osobników po trzech iteracjach (suma), średnią ilość odczytanych osobników w jednej iteracji (średnia) oraz odchylenie standardowe rozwiązania(odch. stand.).

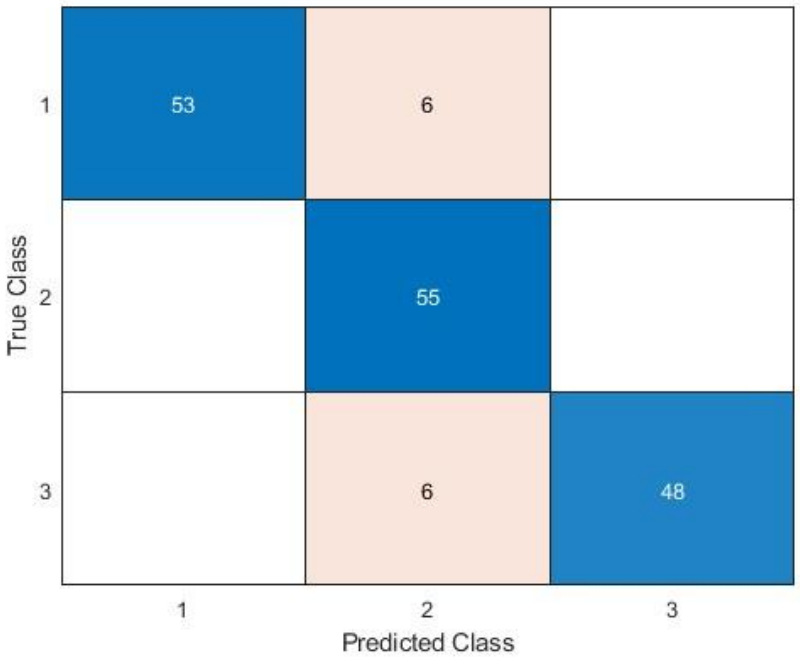
Wartość parametrów	Procent poprawnie zakwalifikowanych danych(po CSO)
AP=0.1 FL=2	91.3
AP=0.9 FL=3	83.33
AP=0.2 FL=15	100

tab3.

Na jakość rozwiązania ma także wpływ wartość parametrów użytych w samym algorytmie (Awarness probability - AP, Flight length - FL). Podczas kolejnych trzech iteracji zmienialiśmy losowo wartości tych parametrów, co prezentuje **tab3**.

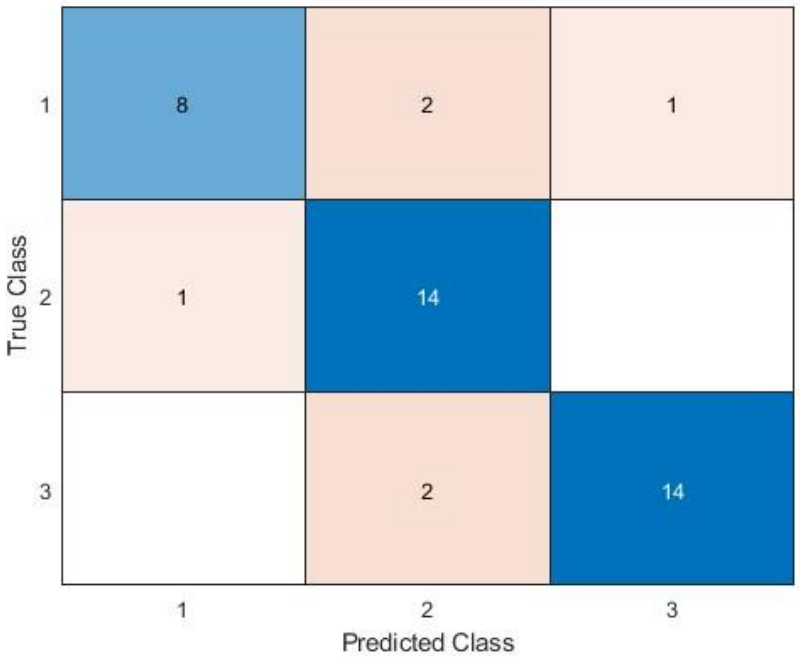
4.2 Zbiór SEEDS

Podczas testowania programu na zbiorze danych SEEDS działaliśmy tak samo jak w poprzednim podpunkcie. Zbiór SEEDS różni się od zbioru IRIS tym, że zawiera 210 osobników, należących do trzech różnych klas, a każdy z nich reprezentowany jest przez 7 cech.



rys4. Macierz pomyłek dla rozwiązania automatycznego oraz zbioru uczącego (zbiór SEEDS)

rys5. Macierz pomyłek dla rozwiązania automatycznego oraz zbioru testującego (zbiór SEEDS)



rys6. Macierz pomyłek dla algorymtu CSO oraz zbioru testującego (zbiór SEEDS)

	Substractive Clustering			CSO	
num.	zbiór uczący	zbiór testujący	błąd średniokwadratowy	zbiór testujący	błąd średniokwadratowy
1	90.857	80.952	0.29	85.714	0.85
2	93.452	88.095	0.09	92.857	0.08
3	88.69	92.095	0.19	83.333	0.09

tab4.

suma	458.63	105.472	-	109.99	-
średnia	152.87	35.15	-	36.19	-
odch. stand.	1.946	3.13	-	4.04	-

tab5.

Tab4 przedstawia procent poprawnie zakwalifikowanych przez program osobników do danego zbioru oraz obliczony błąd średniokwadratowy rozwiązania. Widzimy na przykład, że dla pierwszej iteracji algorytm podał poprawne rozwiązanie ewaluując zbiór testujący w 85% przypadków.

W tabeli **tab5** możemy także odczytać łączną sumę poprawnie zakwalifikowanych osobników po trzech iteracjach (suma), średnią ilość odczytanych osobników w jednej iteracji (średnia) oraz odchylenie standardowe rozwiązania (odch. stand.).

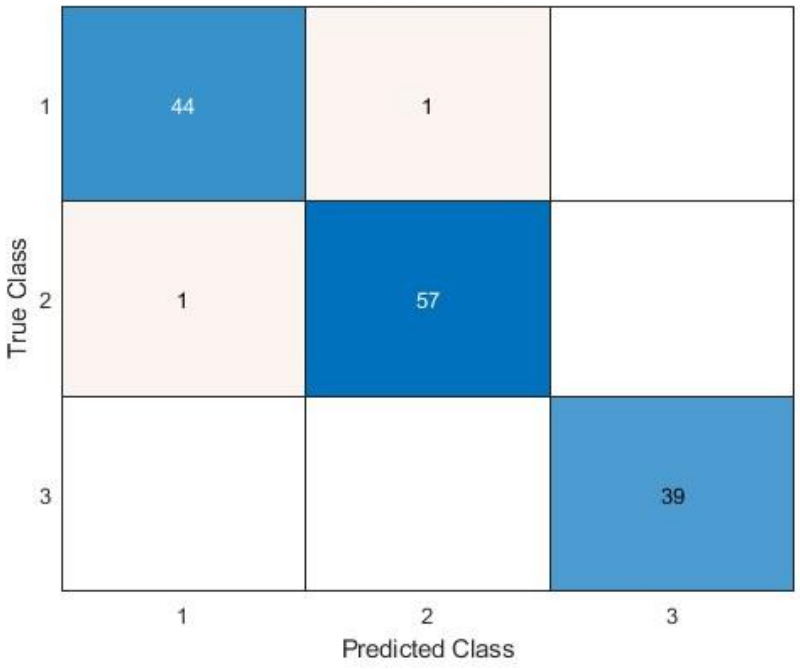
Wartość parametrów	Procent poprawnie zakwalifikowanych danych (po CSO)
AP=0.1 FL=2	85.33
AP=0.9 FL=3	80.33
AP=0.2 FL=15	87.69

tab6.

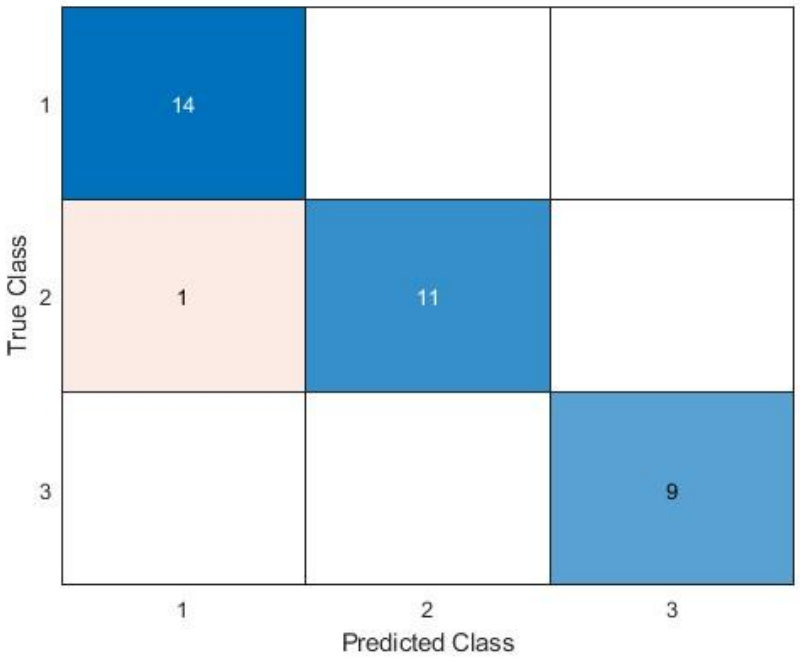
Tab6 prezentuje wpływ zmiany parametrów algorytmu CSO (Awarness probability - AP, Flight length - FL) na jakość otrzymanego rozwiązania.

4.3 Zbiór WINE

W przypadku badania zbioru WINE mamy do czynienia z 178 osobnikami a każdego z nich opisuje aż 13 cech. Tak duża liczba wariancji znacząco wpłynęła na czas rozwiązania zadania, więc do celów testowania postanowiliśmy zrezygnować z trzech ostatnich cech. Końcowo więc każdego osobnika opisuje 10 cech.



rys7. Macierz pomyłek dla rozwiązania automatycznego oraz zbioru uczącego (zbiór WINE)



rys8. Macierz pomyłek dla rozwiązania automatycznego oraz zbioru testującego (zbiór WINE)

rys9. Macierz pomyłek dla algorymtu CSO oraz zbioru testującego (zbiór WINE)

	Substractive Clustering			CSO	
num.	zbiór uczący	zbiór testujący	błąd średniokwadratowy	zbiór testujący	błąd średniokwadratowy
1	100	5.71	13.82	83.13	13.94
2	98	61.13	13.92	57.14	13.28
3	100	81.3	15.15	85.71	14.08

tab7.

suma	415.76	79.01	-	79.09	-
średnia	138.92	26.67	-	26.36	-
odch. stand.	3.62	11.17	-	12.90	-

tab8.

Tab7 przedstawia procent poprawnie zakwalifikowanych przez program osobników do danego zbioru oraz obliczony błąd średniokwadratowy rozwiązania.. Widzimy na przykład, że dla pierwszej iteracji algorytm podał poprawne rozwiązanie ewaluując zbiór testujący w 83.13% przypadków.

W tabeli **tab8** możemy także odczytać łączną sumę poprawnie zakwalifikowanych osobników po trzech iteracjach (suma), średnią ilość odczytanych osobników w jednej iteracji (średnia) oraz odchylenie standardowe rozwiązania (odch. stand.).

Wartość parametrów	Procent poprawnie zakwalifikowanych danych (po CSO)
AP=0.1 FL=2	83.13
AP=0.9 FL=3	91.42
AP=0.2 FL=15	80.00

tab9.

Na jakość rozwiązania ma także wpływ wartość parametrów użytych w samym algorytmie (Awareness probability - AP, Flight length - FL). Podczas kolejnych trzech iteracji zmienialiśmy losowo wartości tych parametrów, co prezentuje **tab9**.

5. Wnioski

Analizując wyniki uzyskane za pomocą programu możemy zauważyć, że algorytm CSO nie zawsze potrafi uzyskać lepsze wyniki niż używanie opcji Substractive Clustering.

Przykładowo, dla zbioru IRIS za pomocą CSO nie zawsze udawało się uzyskać lepsze rozwiązanie, dla trzech prób tylko w jednym przypadku uzyskaliśmy lepszy procent skuteczności klasyfikacji. Większa jest za to wartość odchylenia standardowego. Oznacza to, że Fuzzy Inference System zbudowany przy użyciu danych uzyskanych z CSO jest mniej stabilny.

W przypadku zbioru WINE uzyskaliśmy dla każdej z trzech prób podobny procent klasyfikacji oraz większe odchylenie standardowe.

Możemy dojść do wniosku, że dla używanych zestawów danych uzyskane wyniki nie zawsze będą lepsze dla algorytmu optymalizującego. Dużo zależy od losowości. Możemy również stwierdzić, że tworzenie Fuzzy Inference System z opcją Subtractive Clustering daje zadowalające wyniki, które niekoniecznie trzeba poprawiać. Możliwe jest również, że użyliśmy zbyt małej liczby iteracji, aby rozwiązania używając CSO były lepsze od rozwiązania automatycznego. Na wyniki wpływ mają także na pewno parametry algorytmu CSO, których dobry dobór może znacząco poprawić jakość końcowego rozwiązania.