AUTOMATYKA
CZASOPISMO TECHNICZNE
TECHNICAL TRANSACTIONS
AUTOMATIC CONTROL
WYDAWNICTWO POLITECHNIKI KRAKOWSKIEJ
1-AC/2012
ZESZYT 25
ROK 109
ISSUE 25
YEAR 109

DOMINIKA FALKIEWICZ*, SZYMON ŁUKASIK**

MODELOWANIE ROZMYTE Z ZASTOSOWANIEM ALGORYTMU OPTYMALIZACJI ROJEM CZĄSTEK

FUZZY MODELING WITH THE PARTICLE SWARM OPTIMIZATION ALGORITHM

Streszczenie

Głównym celem niniejszego artykułu jest opracowanie algorytmu klasteryzacyjnego opartego o inspirowany biologicznie algorytm optymalizacji rojem cząstek i dedykowanego dla zagadnienia modelowania rozmytego. W pracy omówiona została idea heurystycznego algorytmu rojowego, z uwzględnieniem wybranych jego modyfikacji. Zawarte zostały wyniki eksperymentalnej ewaluacji, zarówno wybranej techniki optymalizacji, jak i opracowanej z jej uwzględnieniem metody modelowania rozmytego, w odniesieniu do istniejącego już algorytmu k-średnich oraz realizacji procesu sterowania rozmytego.

Słowa kluczowe: modelowanie rozmyte, eksploracja danych, klasteryzacja, algorytm optymalizacji rojem cząstek

Abstract

The main goal of this paper is a description of clustering algorithm based on the particle swarm optimization algorithm, inspired on social behavior of animals and its application in fuzzy modeling. In the paper the idea of the heuristic swarm-based algorithm was presented, including a few modifications. Moreover, the results of the experimental evaluation were shown, both a selected optimization technique and its synthesis with a fuzzy modeling method referring to the k-means algorithm and the fuzzy control process.

Keywords: fuzzy modeling, exploratory data analysis, cluster analysis, particle swarm optimization algorithm

* Mgr inż. Dominika Falkiewicz, Katedra Automatyki i Technik Informacyjnych, Wydział Inżynierii Elektrycznej i Komputerowej, Politechnika Krakowska oraz Studia Doktoranckie, Instytut Badań Systemowych, Polska Akademia Nauk.

** Dr inż. Szymon Łukasik, Katedra Automatyki i Technik Informacyjnych, Wydział Inżynierii Elektrycznej i Komputerowej, Politechnika Krakowska oraz Instytut Badań Systemowych, Polska Akademia Nauk.

1. Wstęp

Rozwój nowoczesnych technologii sprawił, że ludzie są dzisiaj zasypywani ogromną ilością danych z różnorakich dziedzin. Ich eksploracja, rozumiana również jako analiza danych, polega na umiejętnym połączeniu technik: statystycznych, sztucznej inteligencji i obliczeń wysokiej wydajności oraz pozwala na wydobycie z rozpatrywanego zbioru tych informacji, które można uznać za cenne i użyteczne, tj. ważnych relacji, wzorców i trendów. Jednym z narzędzi analizy danych jest klasteryzacja, określana również mianem analizy skupień, która ma na celu wykrycie wewnętrznej struktury zbioru danych i wygenerowanie współzależności między jej elementami. Zadaniem procesu klasteryzacji jest więc podział zbioru danych na określoną liczbę podgrup, wewnątrz których elementy są w jak największym stopniu podobne, natomiast w zestawieniu z elementami z innych podzbiorów – jak najbardziej zróżnicowane [11].

Autorzy podejmują się próby rozwiązania omawianego problemu w aspekcie zagadnienia modelowania rozmytego i z użyciem algorytmu klasteryzacyjnego opartego na optymalizacji rojem cząstek [3, 6, 12, 15] (ang. *Particie Swarm Optimization – PSO*), która stanowi jedną z najnowocześniejszych heurystycznych metod optymalizacji [18].

Celem niniejszego artykułu jest omówienie idei algorytmu PSO uwzględniające jego wybrane modyfikacje oraz przedstawienie zaimplementowanego w środowisku MATLAB algorytmu klasteryzacyjnego, który realizuje syntezę dwóch innowacyjnych technik: inteligencji rojowej i modelowania rozmytego, w jeden efektywny system. W artykule zawarto wyniki eksperymentalnej ewaluacji zarówno wybranej techniki optymalizacji, jak i opracowanej z jej uwzględnieniem metody modelowania, także w odniesieniu do istniejących procedur klasycznych (algorytm *k*-średnich) oraz wybranych danych testowych.

Istnieje wiele wyspecjalizowanych metod służących do efektywnego rozwiązywania złożonych zadań optymalizacyjnych. Omawiany w niniejszym artykule problem klasteryzacji był już z powodzeniem rozwiązywany z użyciem takich algorytmów heurystycznych, jak: algorytmy genetyczne, ewolucyjne, sztuczne sieci neuronowe czy algorytmy mrówkowe [3, 11]. PSO ma wiele podobieństw z wymienionymi metodami obliczeniowymi inspirowanymi systemami biologicznymi, które coraz częściej określane są mianem „sztucznego życia" (ang. *artificial life*), jednak zawiera inny mechanizm przepływu informacji, przez co częstokroć jest w stanie osiągnąć lepsze rezultaty. Zastosowanie tej metody do modelowania rozmytego rozważono dotychczas w niewielkiej liczbie opracowań [19, 20], stąd też ambicją autorów jest stworzenie punktu wyjścia dla dalszych badań w tym zakresie.

2. Algorytm optymalizacji rojem cząstek

2.1. Podstawy funkcjonowania

Algorytm optymalizacji rojem cząstek jest innowacyjną metodą w przestrzeni inteligencji obliczeniowej inspirowaną rojami organizmów żywych [3, 16]. PSO opiera się na autonomicznych współpracujących ze sobą agentach. Każdy z tych agentów nazywany jest cząstką, która porusza się w przestrzeni poszukiwań w celu znalezienia optymalnej w sensie rozważanej funkcji celu pozycji. W problemie klasteryzacji algorytm ten może realizować podział

zbioru danych na możliwie spójne podzbiory, w których różnorodność obserwacji wewnątrz grup (ang. *within-cluster variation* – WCV) jest minimalizowana, a różnorodność względem obserwacji spoza grupy (ang. *between-cluster variation* – BCV) maksymalizowana.

Rozważmy *d*-wymiarową przestrzeń, w której cząstka roju o indeksie *i* może być wyrażona przez *d*-wymiarowy wektor położenia $x_i = (x_{i1}, x_{i2}, ..., x_{id})$, co w ujęciu klasteryzacji stanowi reprezentację położenia środka klastra. Prędkość cząstki jest oznaczona przez wektor $v_i = (v_{i1}, v_{i2}, ..., v_{id})$. Dodatkowo wyróżnia się najlepszą znalezioną przez cząstkę pozycję jako $x_{i,\text{best}} = (x_{i1}, x_{i2}, ..., x_{id})$ oraz najlepszą pozycję znalezioną przez wszystkie cząstki w roju: $x_{g,\text{best}} = (x_{g1}, x_{g2}, ..., x_{gd})$ [12].

Algorytm rozpoczyna swoje działanie od zainicjowania losowo położeń cząstek (potencjalnych rozwiązań). W kolejnych krokach realizuje poszukiwanie optimum poprzez uaktualnianie ich konfiguracji. Przedstawia to następujące równanie aktualizacji prędkości:

$$v_i^{t+1} = v_i^t + c_1 r_1 (x_{i,\text{best}}^t - x_i^t) + c_2 r_2 (x_{g,\text{best}}^t - x_i^t) \tag{1}$$

gdzie:

r_1, r_2 – liczby losowe z rozkładu jednostajnego w przedziale [0, 1], mające na celu zapewnienie różnorodności roju,

c_1, c_2 – stałe współczynniki, określające wpływ poszczególnych składników aktualizacji prędkości, adekwatnie nazwane parametrami: kognitywnym (c_1) i socjalnym (c_2).

Położenie *i*-tej cząstki jest uaktualniane z użyciem wcześniej zdefiniowanego wektora prędkości w następujący sposób:

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{2}$$

Jakość poszczególnego rozwiązania określana jest za pomocą funkcji ewaluacji obliczanej w trakcie poszczególnych iteracji. W niniejszym artykule, z wielu dostępnych wskaźników jakości klasteryzacji, wybrano indeks Daviesa-Bouldina, który można zdefiniować jako funkcję określającą stosunek rozproszenia wewnątrz klastrów do rozproszenia pomiędzy różnymi klastrami [2].

Indeks DB przyjmuje wartości w przedziale [0, 1]. Ponieważ pożądane jest znalezienie klastrów o możliwie małym rozproszeniu wewnętrznym i leżących możliwie daleko od siebie, to celem klasteryzacji powinno być minimalizowanie tego indeksu [16].

Istnieją liczne modyfikacje algorytmu PSO, które poprawiają jego zbieżność do globalnego optimum i dotyczą głównie alternatywnych metod obliczania nowego położenia i prędkości cząstki. Autorzy zastosowali modyfikację pierwotnej wersji algorytmu, wprowadzając do równania uaktualniającego prędkość *i*-tej współczynnik inercji *w*, zgodnie z równaniem:

$$v_i^{t+1} = w v_i^t + c_1 r_1 (x_{i,\text{best}}^t - x_i^t) + c_2 r_2 (x_{g,\text{best}}^t - x_i^t) \tag{3}$$

$$w = \left(w_{\text{start}} - \frac{w_{\text{start}} \cdot t - w_{\text{end}}}{t-1} \right) \cdot l + \left(\frac{w_{\text{start}} \cdot t - w_{\text{end}}}{t-1} \right) \tag{4}$$

gdzie w_{start}, w_{end} oznaczają krańcowe ustalone wartości przedziału, w którym następuje liniowy spadek inercji, $l = 1., ..., t$.

Liniowa redukcja współczynnika w pozwala na balansowanie pomiędzy globalną (większa wartość współczynnika w_{start}) a lokalną (mniejsza wartość współczynnika w_{end}) eksploracją przeprowadzaną przez populację cząstek.

3. Modelowanie rozmyte

Klasyczna teoria mnogości nie zapewnia odpowiednich narzędzi do analizy złożonych systemów, w których cele oraz zależności wejścia-wyjścia są często nieprecyzyjnie określone, a co za tym idzie, trudne do ilościowego ujęcia. Stąd też nastąpił znaczny postęp w zakresie zastosowań metod opartych na logice rozmytej (ang. *fuzzy logic*). Jej techniki bazują bowiem na wnioskowaniu zbliżonym do rozumowania ludzkiego, w związku z czym mają szerokie spektrum praktycznych zastosowań, zwłaszcza w zakresie zagadnień modelowania i sterowania.

3.1. Klasteryzacja w modelowaniu rozmytym

Pierwotna idea tworzenia modeli rozmytych na bazie wiedzy eksperckiej, dotyczącej danego systemu wykazała wiele niedoskonałości. Stąd też opracowano metody tworzenia samostrojących się modeli rozmytych, co realizowane jest wyłącznie na podstawie dostępnych danych pomiarowych. Strojenie modelu wiąże się tu z określeniem takich parametrów funkcji przynależności wejść i wyjść, aby zminimalizować błąd modelu względem modelowanego systemu (w sensie np. błędu średniokwadratowego), określany na podstawie dostępnych danych uczących. Jedną z metod optymalizacji parametrów modelu rozmytego stanowią metody klasteryzacyjne. Wykorzystywany jest w nich fakt automatycznego wykrywania pewnych grup punktów pomiarowych i charakterystycznych wzorców zachowania się układu, które mogą być reprezentowane przez jedną regułę lub ich spójny zbiór [14].

Jedną z najpopularniejszych i najczęściej stosowanych metod klasteryzacyjnych używanych w modelowaniu rozmytym jest klasteryzacja subtraktywna [4] (ang. *subtractive clustering*). Z założenia metoda ta próbuje wybrać spośród n punktów pomiarowych k potencjalnych środków klastrów. W tym celu definiuje się następującą miarę gęstości D_i dla każdego punktu x_i:

$$D_i = \sum_{i,j=1}^{n} e^{-\frac{4}{r_a^2}(\|x_i - x_j\|^2)} \tag{5}$$

gdzie r_a oznacza dodatnią stałą określająca zasięg sąsiedztwa dla pojedynczego elementu.

Pierwszym środkiem klastra (x_{k1}) zostaje ten punkt, dla którego (5) osiąga największą wartość – D_{k1}. Następnie wielkość ta obliczana jest ponownie dla każdego punktu x_i, $i \neq k_1$, zgodnie ze wzorem:

$$D_i = D_i - D_{k_1} e^{-\frac{4}{r_a}(\|x_i - x_{k_1}\|^2)} \tag{6}$$

Stała r_a wpływa tu na spadek wartości (6), tj. dla punktów znajdujących się najbliżej pierwszego środka x_{k_1} miara gęstości D_i znacząco się redukuje. Po zmianie wartości D_i kolejnym środkiem klastra zostaje ten punkt, dla którego wielkość ta jest największa. Cały proces powtarzany jest do momentu aż osiągnięta zostanie zadana liczba klastrów k [8].

Idea stosowania analizy skupień przy tworzeniu sterownika rozmytego opiera się na obserwacji, że dane pomiarowe rozpatrywanego systemu grupują się zwykle w charakterystycznych punktach w obszary, tzw. klastry. Dla każdego klastra można ustalić regułę determinującą wyjście w zakresie tego właśnie obszaru. Innymi słowy, można połączyć środek każdego klastra regułą rozmytą z punktami, które przynależą do omawianego klastra.

Realizacja procesu modelowania rozmytego odbywa się więc na podstawie zbioru wejść $u_1, ..., u_{n_x}$ i wyjść $y_1, ..., y_{n_y}$ rozpatrywanego systemu. Aby go przeprowadzić z użyciem algorytmu klasteryzacyjnego, należy wygenerować za pomocą opisanej powyżej klasteryzacji subtraktywnej zbiór $\{x_1, ..., x_k\}$ środków k-klastrów, gdzie:

$$x_i = (u_{k_i 1}, u_{k_i 2}, ..., u_{k_i n_x}, y_{k_i 1}, y_{k_i 2}, ..., y_{k_i n_y}) \tag{7}$$

Reguły wnioskowania tworzone są według modelu Takagi-Sugeno. Ogólna postać reguły rozmytej przedstawia się następująco:

$$r_k\text{: if } x \text{ is } A_k \text{ then } y \text{ is } g_k(x) \tag{8}$$

gdzie:

$x \in X$ – zmienna wejściowa,
$y \in Y$ – zmienna wyjściowa,
A_k – zbiór rozmyty w obszarze rozważań X,
$g_k(x)$ – liniowa funkcja zmiennej wejściowej x.

Uwzględniając fakt, iż ekstrakcja reguł odbywa się na podstawie danych pomiarowych rozpatrywanego systemu, każdy wektor u_{k_i} wygenerowany w równaniu (7) tworzy i-tą regułę wnioskowania, która ma formę [14]:

$$\text{if } u \text{ is } \mu_{k_i} \text{ then } y = Au + B \tag{9}$$

gdzie A, B są dodatkowymi parametrami.

Funkcja przynależności wejść dla i-tej reguły ma w tym przypadku postać:

$$\mu_i(u) = e^{-\frac{4}{r_a^2}\|u - u_{k_i}\|^2} \tag{10}$$

gdzie r_a oznacza współczynnik generalizacji dla utworzonego zbioru reguł.
Zasadniczo, wartość funkcji przynależności określa stopień bliskości zmiennej wejściowej względem rozważanego środka klastra.

Konkluzja utworzonej reguły nie zawiera zbioru rozmytego, ale funkcję, która przyjmuje postać funkcji liniowej zmiennych wejściowych. W rezultacie wektor wyjścia modelowanego systemu dany jest jako:

$$y = \frac{\sum_{i=1}^{k} \mu_i(u) y_{k_i}}{\sum_{i=1}^{k} \mu_i(u)} \tag{11}$$

Więcej informacji na temat zastosowania procedury klasteryzacji subtraktywnej w modelowaniu rozmytym można znaleźć w pracy [5].

Autorzy niniejszego artykułu proponują, by zakładaną koncepcję zrealizować z użyciem algorytmu PSO, który ma służyć lokalizacji środków klastrów. Realizacja tego podejścia wraz z weryfikacją jego skuteczności zostanie omówiona w dalszej części artykułu.

3.2. Modelowanie rozmyte z wykorzystaniem algorytmu optymalizacji rojem cząstek

Rozwiązanie procesu klasteryzacji na podstawie algorytmu optymalizacji rojem cząstek (przy założeniu, że pojedynczym rozwiązaniem x_i^t w iteracji t są współrzędne środków klastrów) uzyskano na podstawie procedury:

---

for każda cząstka
    Zainicjuj cząstkę (położenie początkowe, prędkość)
end
  do
    for każda cząstka
      Oblicz wartość indeksu DB
      Jeżeli obecna wartość indeksu DB(x_i^t) jest mniejsza niż znaleziona dotychczas DB($x^t_{i,best}$), wtedy ustaw x_i^t jako $x^t_{i,best}$
    end
      Wybierz cząstkę z całej populacji z najniższą wartością indeksu DB i przypisz jej $x^t_{g,best}$
    for każda cząstka
      Oblicz prędkość każdej cząstki roju
      Uaktualnij położenie każdej cząstki roju
    end
while kryterium zatrzymania nie zostało osiągnięte

Rys. 1. Procedura rozwiązania procesu klasteryzacji na podstawie algorytmu PSO

Fig. 1. Procedure for clusterization process solution based on PSO algorithm

Koszt rozwiązania jest określany każdorazowo przez wartość indeksu Daviesa-Bouldina. Proces poszukiwania optimum globalnego kończy się z chwilą, gdy algorytm wykona liczbę iteracji ustaloną *a priori* przez użytkownika.

Uzyskane w ten sposób środki klastrów posłużyły jako prototypy reguł układu modelowania rozmytego w podobny sposób, jak opisany w rozdziale 3.1. Podczas badań przyjęto wartość współczynnika generalizacji określoną według wzoru $r_a = 1/liczba\_klastrow$. Wynikało to z intencji automatycznego doboru jego wartości i zostało pozytywnie zweryfikowane w toku przeprowadzonych eksperymentów obliczeniowych.

4. Wyniki badań eksperymentalnych

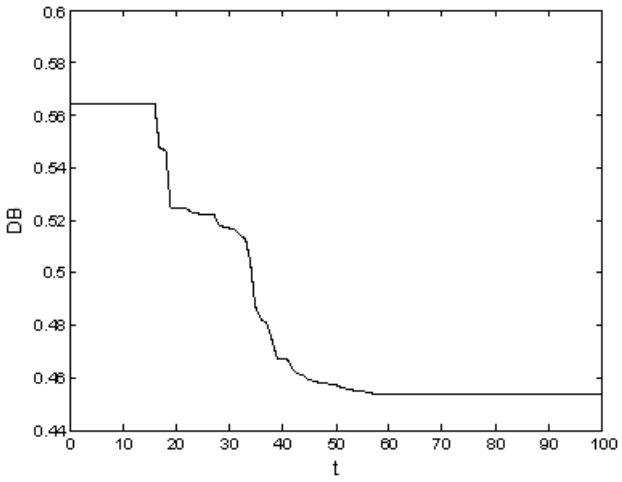
4.1. Wybór najbardziej efektywnego wariantu algorytmu PSO

Na wstępie przeprowadzono badania dotyczące skuteczności wybranych wariantów algorytmu PSO w problemie klasteryzacji, zgodnie z przedstawioną w rozdziale 2.1. koncepcją.

Wnioskowanie o efektywności poszczególnych modyfikacji PSO przeprowadzono, analizując zbiór rozwiązań uzyskanych po 20 uruchomieniach zaimplementowanego algorytmu. Każdorazowo otrzymywano więc zbiór 20 cząstek wybranych z całego roju, które uzyskały najniższą wartość indeksu Daviesa-Bouldina.

Analizując wybrane miary pozwalające na opis wartości danego wskaźnika (minimum, maksimum, średnia arytmetyczna, mediana) oraz miary zmienności reprezentowane przez odchylenie standardowe, wskazano na wersję z liniową redukcją współczynnika inercji uwzględniającą: $w_{start} = 0{,}9$ oraz $w_{end} = 0{,}4$ [12]. Przyjęto też ustalone przez Kennedy'ego i Eberharta [9] wartości parametru kognitywnego i socjalnego: $c_1, c_2 = 2$ oraz potwierdzono największą efektywność algorytmu dla 40 cząstek i 100 iteracji, przy przyjętej stałej liczbie ewaluacji funkcji celu równej 4000 [6].

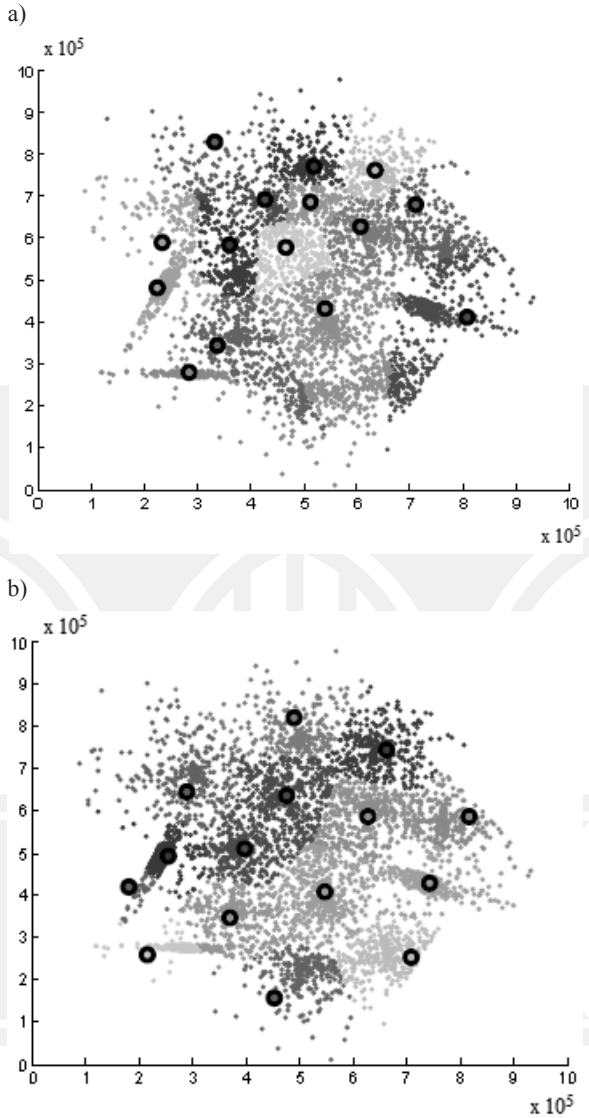
Poniżej przedstawiono przebieg zmienności wskaźnika kosztu dla jednego z przykładowych rozwiązań otrzymanych po 100 iteracjach, dla którego indeks DB stabilizował się na wartości 0,4532.



Rys. 2. Wykres zmian wartości indeksu DB w funkcji liczby iteracji *t* z wykorzystaniem modyfikacji z liniowym spadkiem współczynnika inercji

Fig. 2. Diagram of changing DB index values in the function of *t* iterations, with the use of modification with a linear decrease of inertia coefficient

Początkowe ustawienie większej wartości współczynnika *w* promuje wędrówkę cząstki po całej przestrzeni poszukiwań, natomiast stopniowa redukcja tego współczynnika pozwala na uzyskiwanie coraz doskonalszych rozwiązań.



Rys. 3. Porównanie rozkładu środków klastrów dla pierwszej iteracji (a) i rozwiązania po 100 iteracjach (b) algorytmu PSO z liniową redukcją współczynnika inercji

Fig. 3. Comparison of the distribution of clusters' centers for the first iteration (a) and after 100 iterations (b) of the PSO algorithm with a linear reduction of inertia coefficient

Rysunki 3a i 3b przedstawiają wizualizację przykładowych najlepszych rozwiązań w toku działania algorytmu PSO. Danymi użytymi do przeprowadzenia opisanego eksperymentu były zbiory *S-sets* dedykowane dla weryfikacji algorytmów klasteryzacyjnych, pobrane ze strony internetowej [7]. Dane te są dwuwymiarowe i obejmują 5000 elemen-

tów, z sugerowaną liczbą klastrów równą 15. Na rys. 3a i 3b zilustrowano zbiór *S-sets* w wariancie bardziej jednolitym pod względem przenikania się poszczególnych klastrów (klastry nie są od siebie odseparowane). Podział na klastry, z wyraźnym wyszczególnieniem ich środków przedstawiony w początkowym (rys. 3a) i finalnym (rys. 3b) stadium działania algorytmu stanowi dowód poprawności przedstawionej tu koncepcji.

4.2. Porównanie możliwości wybranego algorytmu rojowego z klasycznym algorytmem *k*-średnich w problemie klasteryzacji

Na wstępie warto zaznaczyć, że zarówno algorytm optymalizacji rojem cząstek, jak i algorytm *k*-średnich pozwalają spojrzeć na problem klasteryzacji bardziej z perspektywy rozlokowania środków poszczególnych klastrów niż dokonania właściwego podziału zbioru danych.

Pierwszą różnicą, którą należy wskazać jest złożoność rozpatrywanych metod. W przypadku implementacji algorytmu PSO, należy uwzględnić dużą liczbę parametrów, m.in.: rozmiar roju, początkowy rozkład środków klastrów, początkowe prędkości cząstek, rodzaj topologii etc. Algorytm *k*-średnich także wymaga podania określonych parametrów pracy, jednak ich liczba jest zdecydowanie mniejsza [15].

Kolejną kwestią, którą należy podjąć rozpatrując działanie obu algorytmów, jest liczba iteracji niezbędna do osiągnięcia rozwiązania końcowego. I choć potwierdzono, że działanie klasycznego algorytmu klasteryzacyjnego zależy od wyboru początkowego rozkładu środków klastrów, to jednak bez względu na dokonany wybór, algorytm *k*-średnich jest szybszy niż innowacyjna heurystyka rojowa. Istotna wydaje się zatem odpowiedź na pytanie: czy szybkość algorytmu *k*-średnich idzie w parze ze stabilnością uzyskiwanych rozwiązań? Do badań weryfikacyjnych wykorzystano uzyskiwaną jako wynik działania algorytmu *k*-średnich sumę odległości obserwacji przynależących do danego klastra od jego środka. Po 20 uruchomieniach zaimplementowanego algorytmu połowa rozwiązań określała niemal identyczna wartość tej sumy. Wynik ten wskazuje na eksplorację bardzo wąskiego sąsiedztwa otaczającego początkowo wylosowane środki, co przekłada się na wysoce prawdopodobne utknięcie tej metody w minimum lokalnym.

Rozważania te wskazały na dodatkową zaletę alternatywnego rozwiązania w postaci PSO, przeprowadza ono mianowicie poszukiwanie globalnego optimum przy równoległym udziale wielu cząstek, z których każda posiada inne położenie początkowe. W rezultacie, daje to możliwość uzyskania różnorakich rozwiązań w tym samym czasie, a więc eksploracji większych obszarów przestrzeni poszukiwań.

Analizując uzyskane wyniki należy wziąć pod uwagę również fakt, że ani algorytm PSO, ani metoda *k*-średnich nie jest pozbawiona wad. I choć nie są one na tyle istotne, aby przesłonić skuteczność i dokładność wymienionych wyżej technik, to autorzy dostrzegają możliwość zastosowania rozwiązania hybrydowego. Połączenie obu algorytmów przyczynia się do powstania zupełnie nowego podejścia do klasteryzacji, a cel, który można w ten sposób osiągnąć, to połączenie zalet obu technik oraz wyeliminowanie, a przynajmniej ograniczenie ich wad. Rozwiązanie to, choć dość innowacyjne, stało się już obiektem eksperymentów obliczeniowych. Wgląd do uzyskanych w ten sposób wyników [1] potwierdza tylko tezę, że hybryda obu algorytmów pozwala na uzyskanie lepszych rezultatów niż rozważanie obu rozwiązań z osobna.

4.3. Weryfikacja autorskiej procedury modelowania rozmytego układu nieliniowego

Głównym przedmiotem rozważań w toku badań eksperymentalnych było opracowanie algorytmu klasteryzacyjnego, opartego na inspirowanym biologicznie algorytmie optymalizacji rojem cząstek dla zagadnienia modelowania rozmytego. Wstępna analiza koncepcji algorytmu PSO oraz szerokie badania bibliograficzne pozwoliły autorom na przypuszczenia, że wykorzystanie nowoczesnej metody heurystycznej zwiększy efektywność procedury klasteryzacji – w szczególności w odniesieniu do zagadnienia modelowania rozmytego.

Pierwszy eksperyment weryfikacyjny polegał na sprawdzeniu efektywności autorskiej procedury w odniesieniu do modelowania rozmytego układu nieliniowego. W tym celu posłużono się danymi użytymi w pracy [10], pozyskanymi na podstawie analizy układu nieliniowego opisanego równaniem różnicowym drugiego rzędu:

$$y(k) = \frac{y(k-1)y(k-2)[y(k-1)+2,5]}{1+y^2(k-1)+y^2(k-2)} + u(k) \tag{12}$$

Zadaniem układu modelującego jest określenie wartości $y(k)$ na podstawie wartości sygnału sterującego u i wyjścia y dla poprzednich k kroków. Rozważany model zawiera trzy wejścia $\{u(k), y(k-2), y(k-1)\}$ i jedno wyjście $y(k)$. Dane uczące obejmują 500 elementów $\{u(k), y(k-2), y(k-1), y(k)\}$, z $u(k)$ wylosowanym z rozkładu jednostajnego w przedziale [–2, 2]. Dane testujące zostały wygenerowane dla dwóch zbiorów: $u(k) = \sin(2\pi k)/25$ (zbiór 1) oraz $u(k) = 1{,}6\cos(2\pi k/25)$ (zbiór 2). Wyniki dotyczą wartości błędu średniokwadratowego (RMSE) dla danych uczących i testujących zarówno dla modelu rozmytego wykorzystującego klasteryzację subtraktywną, jak i modelu wykorzystującego klasteryzację opartą na algorytmie PSO.

Tabela 1

Porównanie wartości RMSE dla modelowania rozmytego układu nieliniowego przy użyciu dwóch różnych metod klasteryzacji

Metoda	Klasteryzacja subtraktywna	Klasteryzacja oparta na PSO
Liczba reguł	10	10
RMSE dla danych uczących	0,2032	0,2297
RMSE dla danych testujących (zbiór 1)	0,3344	0,3303
RMSE dla danych testujących (zbiór 2)	0,5529	0,2879

Parametry algorytmu realizującego klasteryzację subtraktywną oraz klasteryzację opartą na PSO zostały dobrane tak, aby oba modele rozmyte zawierały identyczną liczbę reguł. Choć w obu przypadkach rezultaty na poziomie danych uczących i danych testujących zbioru 1 są porównywalne, to dla zbioru 2 wartość błędu średniokwadratowego modelu opartego na autorskiej metodzie jest niemal dwukrotnie mniejsza.

4.4. Synteza układu rozmytego z PSO dla problemu sterowania pracą dysku twardego

Główna część niniejszego artykułu obejmuje zagadnienie syntezy układu rozmytego z heurystycznym algorytmem rojowym dla problemu sterowania wybranym obiektem dynamicznym. Rozpatrzono przypadek sterowania pracą dysku twardego. Uwzględniono następujący model serwomechanizmu w postaci równań stanu [17]:

$$\begin{bmatrix} \dot{y}(t) \\ \dot{v}(t) \end{bmatrix} = \begin{bmatrix} 1 & 1,664 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} y(t) \\ v(t) \end{bmatrix} + \begin{bmatrix} 1,384 \\ 1,664 \end{bmatrix} u(t)$$
$$z(t) = [1 \quad 0] \begin{bmatrix} y(t) \\ v(t) \end{bmatrix} \tag{13}$$

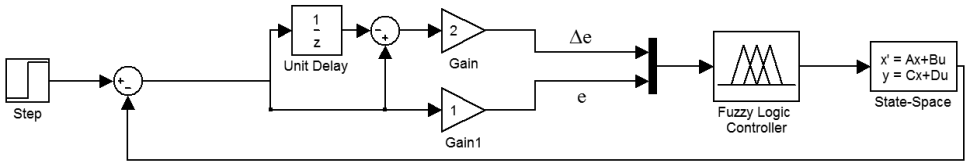
gdzie:

u – napięcie na wejściu serwomechanizmu (podawane w woltach),
y, v – odpowiednio: położenie (ścieżka) i prędkość głowicy dysku twardego.

Omawiany problem dokładnego pozycjonowania uwzględnia wyjście $z(t) = y(t)$.

Na wstępie rozważono standardowy 49-regułowy regulator PD zrealizowany z użyciem metod logiki rozmytej (PD FLC – ang. *Proportional Derivative Fuzzy Logic Controller*) [13].

Schemat blokowy zamkniętego układu regulacji z tym regulatorem przedstawiono na rys. 4.



Rys.4. Schemat blokowy zamkniętego układu regulacji z uwzględnieniem regulatora rozmytego

Fig. 4. Block diagram of close loop regulation system including fuzzy logic controller

Wartości sygnału wyjściowego regulatora określane są na bazie reguł rozmytych, zdefiniowanych poprzez uchyb wielkości regulowanej (e) i jego zmiany (Δe). Forma reguł przedstawia się następująco:

$$\text{if } e \text{ is } E \text{ and } \Delta e \text{ is } \Delta E \text{ then } u \text{ is } U \tag{14}$$

Regulator PD FLC został skonfigurowany tak, aby szybko odpowiadać na sygnał odniesienia, w tym przypadku skok jednostkowy. Uzyskano zbiór składający się z 201 trójek $\{e, \Delta e, u\}$, które posłużyły jako dane wejściowe i wyjściowe do autorskiego algorytmu klasteryzacyjnego. W rezultacie stworzono nowy regulator opierający się na zasadach logiki rozmytej ze zredukowaną liczbą reguł. Wyboru regulatora rozpatrywanego w dalszej części badań dokonano na podstawie analizy wielkości błędu średniokwadratowego dla danych uczących przy różnej liczbie reguł (tab. 2).

Tabela 2

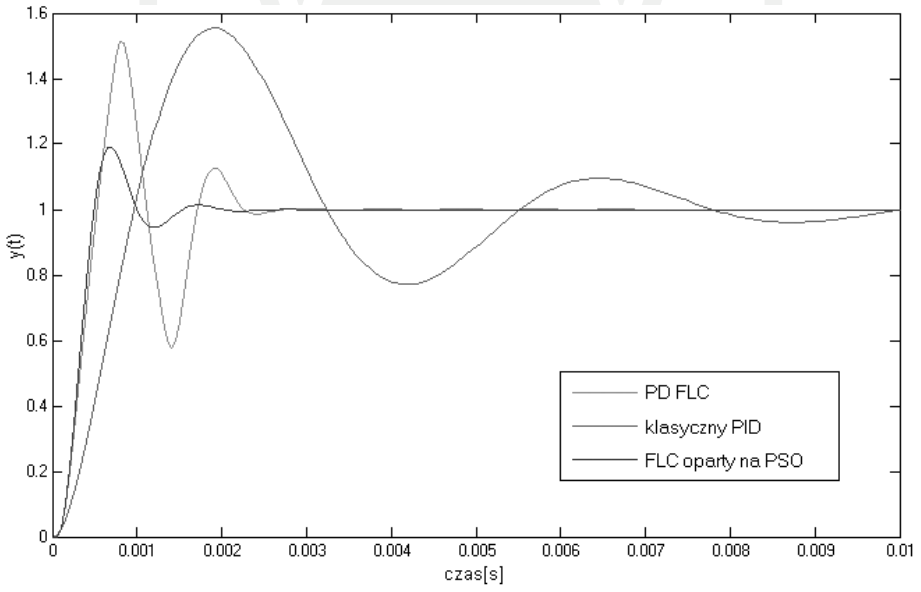
Porównanie wartości RMSE dla różnej liczby reguł algorytmu logiki rozmytej opartego na PSO

Liczba reguł	20	25	30	35	40	45	49
RMSE	0,0127	0,0124	0,0127	0,1196	0,1189	0,1186	0,1188

W konsekwencji, w dalszych rozważaniach uwzględniono regulator oparty na 25 regułach. Ponadto rozpatrzono też układ zawierający klasyczny regulator PID przedstawiony w pracy [10, 13]. Parametry regulatora PID dobrane metodą Zieglera-Nicholsa wynoszą odpowiednio:

$$K_p = 2,07;\ T_i = 1,8;\ T_d = 0,45 \tag{15}$$

Analiza wyników zawiera porównanie trzech układów, w których sterowanie odbywało się kolejno z użyciem każdego z wymienionych regulatorów. Aby dokładnie ocenić ich efektywność, uwzględniono: liczbę reguł regulatora, wielkość błędu średniokwadratowego (RMSE), czas ustalenia ($T_{s,2\%}$), maksymalne przeregulowanie (OV) wyrażone w procentach (tab. 3) oraz uzyskaną przez nie odpowiedź układu na skok jednostkowy, co zbiorczo przedstawiono na rys. 5.



Rys. 5. Przebieg regulacji pracą dysku twardego z użyciem trzech rozpatrywanych regulatorów

Fig. 5. Hard driver servo motor control plot with the use of three analyzed controllers applications

Analizując tabelę 3 oraz rysunek 5 stwierdzono, że wybrane do porównania regulatory oparte na logice rozmytej działają efektywniej niż klasyczny PID. Wykazały one znacznie mniejszy błąd średniokwadratowy, jednak najistotniejszy okazał się fakt, iż ich czas ustalenia T_s

Tabela 3

Porównanie wyników uzyskanych dla klasycznego PID, PD FLC oraz FLC opartego na PSO

Regulator	Liczba reguł	Wartość RMSE	$T_{s,2\%}$ [s]	OV [%]
Klasyczny PID	–	0,2909	0,0096	56
PD FLC	49	0,1981	0,0020	52
FLC oparty na PSO	25	0,1647	0,0029	19

był prawie pięciokrotnie mniejszy. Biorąc pod uwagę wyłącznie regulatory zbudowane na zasadach logiki rozmytej, stwierdzono, że regulator FLC oparty na algorytmie klasteryzacyjnym PSO miał równie mały błąd RMSE oraz czas osiągnięcia wartości zadanej, co zbudowany na tych samych zasadach regulator PD. Fundamentalną różnicą w działaniu obu regulatorów był jednak fakt, że ten pierwszy potrzebował dwukrotnie mniejszej liczby reguł do otrzymania podobnego rezultatu, osiągając przy tym znacznie mniejszą wartość przeregulowania. Stąd wniosek, iż klasteryzacja powoduje redukcję obszernego zbioru reguł regulatora rozmytego i uzyskanie takiego ich zestawu, który cechuje się większą efektywnością w badanych problemach sterowania.

5. Wnioski

W niniejszym artykule nie tylko opisano heurystyczny algorytm rojowy, ale pokazano również, że jest on w stanie sprostać zadaniu analizy skupień zarówno na podstawie wygenerowanych danych numerycznych, jak też w praktycznym zagadnieniu modelowania rozmytego. Badania wykazały, że efektywność nowatorskiej procedury jest porównywalna, a często nawet lepsza niż zaprezentowane w pracy metody klasyczne. Stwierdzenie to zostało potwierdzone zarówno przy realizacji zadań modelowania, jak i zadań automatycznej regulacji.

Przeprowadzane badania potwierdziły też, że rozpatrywany algorytm heurystyczny nie jest rozwiązaniem idealnym i pomimo wielu zalet, ma też wiele wad, m.in. wspólny dla wszystkich metod heurystycznych brak gwarancji otrzymania najlepszego rozwiązania danego problemu. Cechują go wszakże z drugiej strony: potencjał rozwiązywania trudnych problemów optymalizacyjnych, intuicyjna inspiracja mechanizmami biologicznymi oraz możliwość adaptacji z klasycznymi metodami, które samodzielnie, choć skuteczne, nie nadążają za tempem rozwoju nowoczesnych technik obliczeniowych.

Biorąc pod uwagę ciągłe zainteresowanie tematyką metod rojowych, a także szeroki zakres zastosowań algorytmu optymalizacji rojem cząstek, można potwierdzić, że technika ta stanowi perspektywiczny przedmiot dalszych badań naukowych w zakresie rozwiązania trudnych problemów optymalizacyjnych.

Badania Dominiki Falkiewicz współfinansowane są przez Fundację na Rzecz Nauki Polskiej w ramach PhD Projects in Intelligent Computing (projekt finansowany przez Unię Europejską w programach Innowacyjna Gospodarka oraz Europejski Fundusz Rozwoju Regionalnego).

Badanie zrealizowano dzięki dofinansowaniu w ramach stypendium naukowego z projektu pn. „Technologie informacyjne: badania i ich interdyscyplinarne zastosowania" współfinansowanego ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego, Program Operacyjny Kapitał Ludzki (Umowa nr UDA-POKL.04.01.01-00-051/10-00).

Literatura

[1] Ahmadyfard A., Modares H., *Combining PSO and k-means to Enhance Data Clustering*, Proceedings of the IEEE International Symposium on Telecommunications, 2008, 688-691.

[2] Bandyopadhyay S., Maulik U., *Genetic clustering for automatic evolution of clusters and application to image classification*, Proceedings of the Pattern Recognition, vol. 35, 2002, 1197-1200.

[3] Chan F.T.S., Tiwari M.K., *Swarm Intelligence. Focus on Ant and Particle Swarm Optimization*, I-Tech Education and Publishing, Vienna 2007, 507-513.

[4] Chiu S.L., *Fuzzy Model Identification Based on Cluster Estimation*, Journal of Intelligent and Fuzzy Systems, vol. 2, No. 3, 1994, 267-284.

[5] Chopra S., Mitra R., Kumar V., *Fuzzy Controller: choosing and Appropriate and Smallest Rule Set*, International Journal of Computational Cogniton, vol. 3, No. 4, 2005.

[6] Clerc M., *Particle swarm optimization*, ISTE Ltd., Great Britain 2006.

[7] Clustering datasets (online), homepage: http://cs.joensuu.fi/sipu/datasets/ (date of access: 2012-10-30).

[8] Hammouda K.M., Karray F., *A Comparative Study of Data Clustering Techniques*, [in:] *Tools of Intelligent Systems Design. Course Project*, 2000.

[9] Kennedy J., Eberhart R.C., Particle Swarm Optimization, Proceedings of the 1995 IEEE International Conference on Neutral Networks, vol. 4, Perth, Australia 1995, 1942-1948.

[10] Kowalski P.A., Łukasik Sz., Charytanowicz M., Kulczycki P., *Data-Driven Fuzzy Modeling and Control with Kernel Density Based Clustering Technique*, Polish Journal of Environmental Studies, vol. 17, No. 4C, 2008, 83-87.

[11] Krzyśko M., Wołyński W., Górecki T., Skorzybut M., *Systemy uczące się. Rozpoznawanie wzorców, analiza skupień i redukcja wymiarowości*, wyd. I, WNT, Warszawa 2008.

[12] Lazinica A., *Particle Swarm Optimization*, In-Tech, 2009, 2-3, 51-53, 89-90, 293-294, 341-343.

[13] Mudi R., Pal N.R., *A Robust Self-Tuning Scheme for PI and PD Type Fuzzy Controllers*, Proceedings of the IEEE Trans. Fuzzy Syst., 1999.

[14] Piegat A., *Modelowanie i sterowanie rozmyte*, Akademicka Oficyna Wydawnicza EXIT, Warszawa 2003.

[15] Poli R., Kennedy J., Blackwell T., *Particle swarm optimization: an overview*, Swarm Intelligence Journal, vol. 1, No. 1, 2007, 33-40, 50-53.

[16] Su M.C., *A New Index of Cluster Validity* (online), homepage: www.cs.missouri.edu/~skubicm/8820/ClusterValid.pdf (date of access: 2012-10-30).

[17] Tan K.C., Sathikannan R., Tan W.W., Loh A.P., *Evolutionary Design and Implementation of a Hard Disk Drive Servo Control System*, Soft Computing, vol. 11, No. 2, 2007, 131-139.

[18] Trojanowski K., *Metaheurystyki praktyczne*, WIT, Warszawa 2005.

[19] Wang D., Wang G., Hu R., *Parameters optimization of fuzzy controller based on PSO*, Intelligent System and Knowledge Engineering, vol. 1, 2008, 599-603.

[20] Wong C.C., Wang H.Y., Li S.A., *PSO-based motion fuzzy controller design for mobile robots*, International Journal of Fuzzy Systems, vol. 10, No. 1, 2008, 284-292.