AUTOMATYKA • 2010 • Tom 14 • Zeszyt 3/2

Bogusław Filipowicz*, Joanna Kwiecień*

Zastosowanie przykładowego algorytmu stadnego w optymalizacji kombinatorycznej

1. Wprowadzenie

Do rozwiązania większości współczesnych problemów optymalizacyjnych konieczne jest stosowanie algorytmów, które łatwo dostosowują się do ograniczeń, niezależnie od liczby zmiennych i rozmiarów przestrzeni rozwiązań.

W ostatnich latach rozwija się nowy dział metod optymalizacyjnych zwanych algorytmami bazującymi na inteligencji roju (*Swarm optimization*), których zasady działania zostały zaczerpnięte z obserwacji natury. Naśladując zachowania istniejące w świecie zwierząt, ptaków i roślin, dostarczają bardzo wydajnych algorytmów. Do tej klasy algorytmów należą m.in. algorytmy mrówkowe (ACO – *Ant colony optimization*), algorytmy pszczele (BA – *Bee algorithms*) i algorytmy roju cząstek (PSO – *Particle swarm optimization*). Nie bazują one na pojedynczym aktualnym rozwiązaniu, jak klasyczne metody optymalizacji, lecz na pewnym zbiorze rozwiązań.

Przedstawiona w pracy metoda roju cząstek wzorowana jest na zachowaniu osobników (cząstek) tworzących zorganizowane populacje [1, 3]. Środowisko bytowania oraz cząstki wpływają na zachowania pozostałych. Każdy osobnik posiada zdolność zapamiętywania swojego położenia i przystosowania się do środowiska. Dzięki tym cechom osobniki wyszukują nowe obszary o bardziej sprzyjających właściwościach. Przykładem rojów są między innymi stada ptaków czy ławice ryb. Algorytmy te znajdują szereg zastosowań m.in. w symulacji tłumu, w analizie ruchu w sieci. Metoda optymalizacji rojem cząstek stanowi olbrzymi potencjał do rozwiązywania zagadnień optymalizacji kombinatorycznej. W pracy [2] przedstawiono zastosowanie PSO do rozwiązania kwadratowego problemu przydziału. Algorytmy te stosowane są również z powodzeniem do rozwiązania problemów szeregowania zadań [4–6]. W pracy [8] zastosowano PSO do rozwiązania problemu marszrutyzacji pojazdów z dostawą w wymaganych oknach czasowych. Niniejsza praca przedstawia zastosowania metody optymalizacji rojem cząstek do rozwiązania wielowymiarowego problemu plecakowego.

* Katedra Automatyki, Akademia Górniczo-Hutnicza w Krakowie, kwiecien@ia.agh.edu.pl

2. Algorytm optymalizacji rojem cząstek

Algorytm optymalizacji rojem cząstek (PSO), nazywany także algorytmem ptasim, opracowany został przez Kennedy'ego i Eberharta w 1995 roku [1, 3]. Algorytm ten opiera się na obserwacji zachowania całej populacji (roju, stada), przy możliwości komunikowania się między osobnikami i dzielenia się informacjami. Każdy osobnik należący do danej populacji traktowany jest jako cząstka. Cząstki przemieszczają się do nowych położeń, poszukując optimum. Podobnie jak stado ptaków, rój podąża za przywódcą (najlepszym rozwiązaniem) przyspieszając i zmieniając kierunek, jeśli lepsze rozwiązanie zostanie znalezione. Przywódcą roju zostaje osobnik o najlepszym dotychczas znanym położeniu. Osobniki podejmują decyzje na podstawie najbliższego otoczenia, jednak dzięki przesyłaniu informacji stado zachowuje swoją dynamikę. Każda cząstka posiada określone położenie, prędkość i zwrot, oraz zna swoich sąsiadów, wartość funkcji ewaluacyjnej dla swojego położenia. Pamięta również najlepsze położenia, jakie udało jej się osiągnąć. Zarówno położenie oraz prędkość i-tej cząstki w d-wymiarowej przestrzeni można przedstawić jako wektory: $x_i = [x_{i1}, x_{i2}, ..., x_{id}]$ oraz $v_i = [v_{i1}, v_{i2}, ..., v_{id}]$. Każda cząstka ma swoją własną najlepszą pozycję $p_i = [p_{i1}, p_{i2}, ..., p_{id}]$ odpowiadającą najlepszej uzyskanej dotychczas wartości funkcji celu, natomiast najlepsza pozycja cząstki-przywódcy w całym roju określona jest jako $p_d = [p_{d1}, p_{d2}, ..., p_{dd}]$.

Podstawowy algorytm PSO można przedstawić następująco:

1) Inicjalizacja losowej pozycji cząstek i ich prędkości początkowych,
2) Ocena położenia cząstek za pomocą funkcji dopasowania,
3) Porównanie zachowania każdej cząstki z jej najlepszym (do tej pory) zachowaniem, wybór przywódcy stada,
4) Uaktualnienie prędkości każdej cząstki w każdym kroku k:

$$v_i(k) = \omega v_i(k-1) + c_1 r_1 [p_i(k-1) - x_i(k-1)] + c_2 r_2 [p_d(k-1) - x_i(k-1)] \tag{1}$$

gdzie:

ω – współczynnik inercji ruchu cząstki,
v_i – wektor prędkości i-tej cząstki,
x_i – wektor położenia i-tej cząstki,
p_i – najlepsze położenie i-tej cząstki,
p_d – najlepsze położenie dowolnej cząstki,
c_1 – stała dodatnia, tzw. wskaźnik samooceny,
c_2 – stała dodatnia, wskaźnik społecznościowy (zaufanie położeniu sąsiadów),
r_1, r_2 – losowe liczby o rozkładzie równomiernym w przedziale [0, 1],

5) Uaktualnienie położenia każdej cząstki:

$$x_i(k) = x_i(k-1) + v_i(k) \tag{2}$$

Etapy 2–5 wykonywane są do momentu spełnienia kryterium stopu. Współczynnik inercji określa, w jakim stopniu aktualna prędkość cząstki jest uzależniona od wartości poprzedniej. Duża wartość ukierunkowuje na globalne przeszukiwania. Wskaźnik samooceny określa, jak bardzo dana cząstka ufa kierunkowi do swojego najlepszego położenia. Dobór parametrów algorytmu ma istotny wpływ na zbieżność algorytmu.

3. Zastosowanie binarnej wersji PSO do binarnego wielowymiarowego problemu plecakowego

3.1. Model matematyczny wielowymiarowego problemu plecakowego

Wielowymiarowy problem plecakowy to jeden z głównych problemów optymalizacji kombinatorycznej. Problem ten polega na zapakowaniu do plecaka jak najcenniejszego zbioru przedmiotów, nie przekraczając przy tym jego ograniczonej pojemności. Może być rozważany jako problem alokacji m zasobów do n obiektów. Każdy zasób ma określony budżet M_i, obiekt przynosi zysk p_j oraz zużywa w_{ij} zasobu i. Maksymalizowana funkcja celu zdefiniowana jest następująco:

$$f(x) = \sum_{j=1}^{n} p_j x_j \tag{3}$$

przy ograniczeniach:

$$\sum_{j=1}^{n} w_{ij} x_j \le M_i, i = 1, ..., m$$

$$x_j \in \{0,1\}, p_j > 0, w_{ij} \ge 0, M_i \ge 0$$

W procesach transportowych, przy problemie załadunku towarów, w zagadnieniach optymalizacji ilości towaru pod względem maksymalizacji zysków transportu (minimalizacji kosztów) występuje wielowymiarowy problem plecakowy.

3.2. Binarny algorytm optymalizacji rojem cząstek

W 1997 r. Kennedy i Eberhart zaproponowali binarną wersję PSO (BPSO) do rozwiązywania problemów dyskretnych [1, 7], w której wektory przyjmują binarne wartości. Cząstka może więc poruszać się tylko w przestrzeni zero-jedynkowej. Za pomocą funkcji sigmoidalnej opisanej równaniem (4), otrzymujemy transformację zawężającą wartości prędkości do zakresu [0, 1] oraz ustawiającą wartość położenia na 0 lub 1. Funkcja ta przedstawia prawdopodobieństwo ustawienia bitu na wartość 1.

$$S(v_{ij}) = 1/(1 + e^{-v_{ij}}) \tag{4}$$

Równania aktualizacji mogą być zrealizowane w dwóch etapach. Pierwszy polega na aktualizacji prędkości cząstki zgodnie z zależnością (1) i wykorzystaniu funkcji sigmoidalnej do określenia ograniczeń prędkości w przedziale [0, 1]. W drugim etapie, wartość każdego bitu x_{ij} jest określona przez porównanie jego prawdopodobieństwa z wartością losową, i przyjmuje wartości określone zależnością:

$$x_{ij} = \begin{cases} 1, \text{ dla } rand() \le S(v_{ij}) \\ 0 \end{cases} \tag{5}$$

przy czym *rand()* jest liczbą z zakresu [0, 1].

Istnieje kilka modyfikacji binarnej wersji PSO. Jedną z nich jest PBPSO (*probability binary particle swarm optimization*), która wprowadza nową formułę określającą binarny bit px_{ij} [7]:

$$L(x_{ij}) = (x_{ij} - R_{\min})/(R_{\max} - R_{\min})$$

$$px_{ij} = \begin{cases} 1, \text{ dla } rand() \le L(x_{ij}) \\ 0 \end{cases} \tag{6}$$

gdzie $L(x)$ jest funkcją liniową o wyniku należącym do (0, 1), *rand()* jest liczbą losową z zakresu [0, 1], $R_{\max}$ i $R_{\min}$ jest predefiniowaną maksymalną i minimalną wartością x_{ij}.

Procedurę rozwiązania wielowymiarowego problemu plecakowego za pomocą PBPSO można przedstawić następująco:

1) Inicjalizacja:
   – stałej populacji roju,
   – parametrów początkowej prędkości i położenia (x, v, p_i, p_d),
   – $\omega = 0{,}8$; $c_1 = c_2 = 2{,}0$.
2) Ocena rozwiązania według funkcji dopasowania opisanej równaniem (3).
3) Aktualizacja lokalnych optimów i globalnego optimum.
4) Aktualizacja cząstek według zależności (1, 2, 6).
5) Sprawdzenie warunku zakończenia obliczeń. Jeśli uzyskane rozwiązanie jest dobre – koniec algorytmu, w przeciwnym wypadku – powrót do kroku 2.

4. Wyniki przeprowadzonych eksperymentów

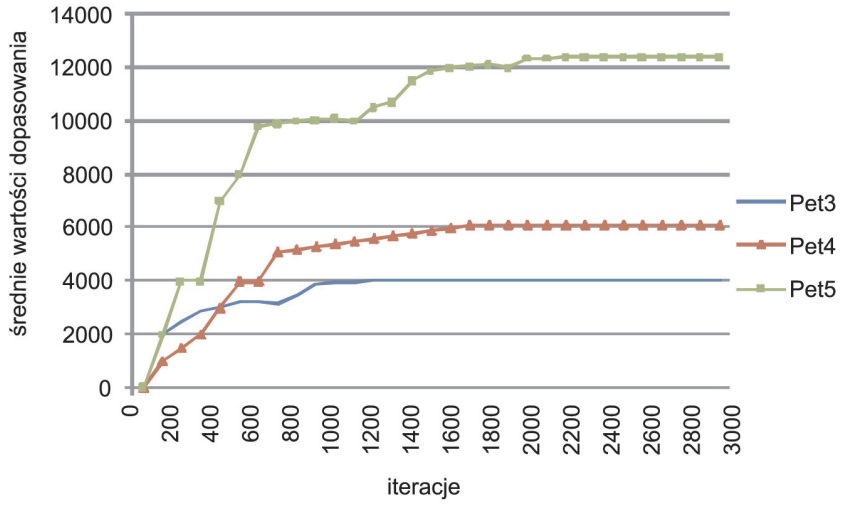
Eksperymenty przeprowadzono dla zadań testowych zaczerpniętych z biblioteki zawierającej instancje testowe opisujące wielowymiarowy problem plecakowy, dostępnej online [9]. Instancje te przedstawione w postaci plików .dat i zebrane z różnych źródeł literaturowych, zawierają informacje dotyczące liczby obiektów i zasobów oraz ich ograniczeń.

Wyniki badań eksperymentalnych algorytmu PBPSO zastosowanego do rozwiązania wielowymiarowego problemu plecakowego przedstawiono w tabeli 1. Zagadnienia podzielone są na dwie grupy. Dane Pet* obrazują problemy składające się z 10 zasobów, natomiast dane Weish* dotyczą problemów złożonych z 5 zasobów. W kolumnie drugiej umieszczone zostały prawidłowe rozwiązania, natomiast kolumna trzecia zawiera procent najlepszych rozwiązań dla PBPSO. Algorytm zaimplementowany został w Matlabie. Przeprowadzono po 20 eksperymentów dla każdej instancji testowej. Parametry algorytmu, które zostały przyjęte: stały rozmiar populacji składającej się z 50 cząstek, kryterium stopu – wykonanie 3000 iteracji, $c_1 = c_2 = 2{,}0$, $R_{max} = 50$ oraz $R_{min} = -50$.

Tabela 1
Wyniki przeprowadzonych eksperymentów

Instancja	Znane rozwiązanie	% prawidłowych rozwiązań
Pet3, m = 10, n = 15	4015	100
Pet4, m = 10, n = 20	6120	80
Pet5, m = 10, n = 28	12400	60
Weish03, m = 5, n = 30	4115	55
Weish04, m = 5, n = 40	4561	50
Weish10, m = 5, n = 50	6339	40

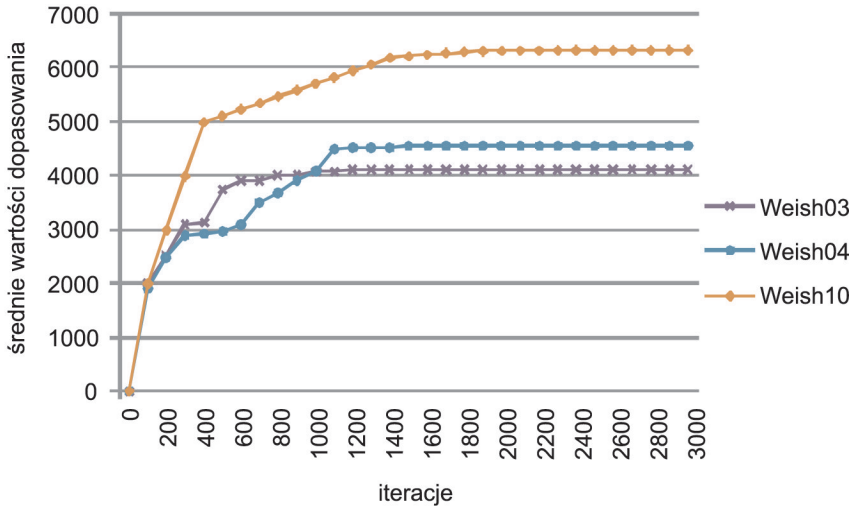
Rysunek 1 przedstawia średnie wartości dopasowania algorytmu PBPSO dla danych Pet.*.



Rys. 1. Średnie wartości dopasowania algorytmu PBPSO dla danych Pet3, Pet4 i Pet5

Dla danych Pet3 prawidłowe rozwiązanie zostało znalezione w 1178 iteracji, dla Pet4 w 1705 iteracji, natomiast dla Pet5 – w 2179 iteracji. Są to uśrednione wartości dla 20 eksperymentów. W niektórych przypadkach minimalna liczba iteracji była znacznie mniejsza.

Na rysunku 2 przedstawiono średnie wartości dopasowania algorytmu PBPSO dla danych Weish03, Weish04, Weish10. Najdłużej poszukiwane było rozwiązanie dla danych składających się z 5 zasobów i 50 obiektów (dane Weish10). Jeśli rozwiązanie zostało znalezione, to dopiero w okolicach 2100 iteracji. Dla danych Weish03 algorytm PBPSO potrafił wyznaczyć optymalne rozwiązanie już w 1217 iteracji.



Rys. 2. Średnie wartości dopasowania algorytmu PBPSO w zależności od liczby iteracji, dla danych eksperymentalnych Weish03, Weish04, Weish10

5. Podsumowanie

Przedstawiony w pracy algorytm optymalizacji rojem cząstek może być stosowany do rozwiązania wielu problemów optymalizacji kombinatorycznej. Jednym z nich jest wielowymiarowy problem plecakowy, który został rozwiązany za pomocą binarnej wersji PSO. Uzyskane wyniki dla wybranych instancji testowych wskazują na olbrzymi potencjał metody PBPSO. Dzięki wymianie informacji między cząstkami w trakcie wykonywania algorytmu, a tym samym ograniczeniu liczby wywołań funkcji dopasowania, metoda ta może być stosowana do rozwiązywania wielu problemów wymagających znacznych nakładów obliczeniowych. Interesującym obszarem badań byłyby zagadnienia związane z wprowadzeniem innych, nowych formuł aktualizacji cząstek.

Literatura

[1] Eberhart R., Shi Y., Kennedy J., *Swarm Intelligence*. Morgan Kaufman, San Francisco, 2001.
[2] Gong T., Tuson A.L., *Particle swarm optimization for quadratic assignment problems – a forma analysis approach*. International Journal of Computational Intelligence Research, 4, 2008, 177–185.
[3] Kennedy J., Eberhart R., *Particle Swarm Optimization*. Materiały IEEE International Conference on Neural Networks, 4, 1995, 1942–1948.
[4] Lian Z., Gu X., Jiao B., *A similar particle swarm optimization algorithm for permutation flowshop scheduling to minimize makespan*. Applied Mathematics and Computation, 175, 2006, 773–785.
[5] Sha D.Y., Hsu Ch.Y., *A new particle swarm optimization for the open shop scheduling problem.* Computers & Operations Research, 35, 2008, 3243–3261.
[6] Tasgetiren M.F., LiangY.C., Sevkli M., Gencyilmaz G., *A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing*. European Journal of Operational Research, 177, 2007, 1930–1947.
[7] Wang L., Wang X., Fu J., Zhen L., *A novel probability binary particle swarm optimization algorithm and its application.* Journal of Software, 3, 2008, 28–35.
[8] Zhu Q., Qian L., Li Y., Zhu S., *An improved particle swarm optimization algorithm for vehicle routing problem with time windows*. Materiały IEEE Congress on Evolutionary Computation, 2006, 1386–1390.
[9] Zbiór instancji testowych problemu plecakowego: http://elib.zib.de/pub/Packages/mp-testdata/ip/sac94-suite/index.html (styczeń 2010).