Studia i Materiały. Miscellanea Oeconomicae
Rok 16, Nr 2/2012
Wydział Zarządzania i Administracji
Uniwersytetu Jana Kochanowskiego w Kielcach

Zarządzanie i finanse

Rafał Prońko[1]

ZASTOSOWANIE KLASYCZNEGO ALGORYTMU GENETYCZNEGO DO ROZWIĄZANIA ZBILANSOWANEGO ZAGADNIENIA TRANSPORTOWEGO

1. Opis Algorytmu Genetycznego

Algorytmy genetyczne w ciągu ostatnich 30 lat stały się przedmiotem intensywnych badań naukowych, jako element niezbędny przy rozwiązywaniu wielu zadań. Dzięki algorytmom genetycznym udało się rozwiązać lub choćby przyspieszyć rozwiązanie wielu różnych problemów np.: dylemat więźnia, zadanie komiwojażera, zadanie transportowe, zadania harmonogramowania, podziału obiektów i grafów, czy choćby optymalizację parametrów sieci neuronowych.

W uproszczeniu algorytmy genetyczne można porównać do samej genetyki. Załóżmy, że mamy populację lisów wiadomo, że lisy muszą być szybkie, zwinne, mieć rozwinięte zmysły – wszystkie te elementy będziemy nazywać cechami gatunku (w teorii algorytmów genetycznych będziemy je nazywać chromosomami). Na początku istnienia populacji lisów (zwana rozwiązaniem początkowym). Cechy te są rozwinięte na pewnym poziomie (poziom rozwinięcia tych cech został przez naturę „wylosowany"). Jednak aby lisy mogły coraz sprawniej polować cechy te muszą z każdym pokoleniem coraz bardziej się rozwijać. Pierwszą możliwością rozwoju tych genów jest łączenie się lisów w pary. Następuje wtedy wymiana genów pomiędzy dwoma lisami (w algorytmach genetycznych nazywa się to krzyżowaniem). Jak wiadomo z pary lisów nie powstanie jeden „super lis" obdarzony najlepszymi cechami obu rodziców. W genetyce jest tak, że powstaje tych lisów trochę więcej każdy otrzymał pewne cechy jednego rodzica i pewne cechy drugiego (wcale nie są one najlepsze). Pytanie powstaje czy w ten sposób lisy

[1] Mgr Rafał Prońko, doktorant Wydziału Matematyki i Informatyki Uniwersytetu Łódzkiego.

z każdym pokoleniem będą się rozwijać? Odpowiedź jest oczywista i brzmi **nie.** Zatem co potrzeba aby cechy lisów w każdym pokoleniu były coraz lepsze? Z pomocą jednak przychodzi matka natura wprowadzając taki element jak mutację genów co powoduje ich rozwój, czasem degenerację. Mutacja polega na jak najmniejszej zmianie pojedynczego genu. Dzięki mutacji w każdym pokoleniu lisów otrzymujemy geny różniące się trochę od najlepszych genów rodziców. Podczas procesu mutacji można powiedzieć, że matka natura gra w ruletkę. Niekiedy te zmiany są na lepsze, ale czasem są na gorsze. W ten sposób z jednego pokolenia lisów otrzymujemy w następnym pokoleniu osobniki obdarzone lepszymi jak również gorszymi cechami odpowiedzialnymi za polowanie. Zwierzęta obdarzone gorszym cechami nierzadko wymierają, ponieważ nie są w stanie polować tak dobrze jak osobniki obdarzone lepszymi genami (w teorii algorytmów ewolucyjnych takie zachowanie nazywa się selekcją). Nie wszystkie jednak lisy obdarzone gorszymi genami wymierają zdarza się, że lisy takie przeżyją i także mają możliwość rozmnażania się co powoduje, że w każdym pokoleniu lisów występują cechy gorsze i cechy lepsze. Dla genetyki czy też algorytmów genetycznych jest to jak najbardziej dobre ponieważ przeprowadzenie mutacji na lepszym genie może dać gorsze wyniki niż przeprowadzenie mutacji na genie gorszym. W ten sposób otrzymujemy możliwość dodania genów, które w pewnych warunkach mogą okazać się dużo lepsze niż geny cały czas rozwijane.

Tak właśnie działa algorytm genetyczny. Posiadamy pewną grupę rozwiązań (nie wiemy czy są one dobre czy nie), nazywamy tę grupę przestrzenią rozwiązań, następnie rozwiązania krzyżujemy ze sobą. Nowo powstałe rozwiązania zmienia się nieznacznie w sposób losowy (mutacja). Z rozwiązań jakich dostajemy w wyniku krzyżowania i mutacji wybieramy pewną grupę a resztę usuwamy. Wybrana grupa jest znów poddawana działaniom krzyżowania i mutacji i tak do momentu uzyskania najlepszych rozwiązań. Oczywiście wybór grupy osobników nie jest przypadkowy, podlega on prawom rachunku prawdopodobieństwa, każdy osobnik ma prawo być wybrany ale tylko z pewnym prawdopodobieństwem. To zapewnia nam zmienność genów (mogą do następnej populacji przejść osobniki słabiej przystosowane ale po skrzyżowaniu z innymi lepiej dopasowanymi mogą dać rezultaty lepsze niż gdyby skrzyżować osobniki lepiej dopasowane).

Należy zdać sobie sprawę, że podstawowe operatory genetyczne takie jak krzyżowanie (łączenie się lisów w pary), czy też mutację najprościej wykonywać jest na łańcuchach binarnych, choć nie zawsze jest to możliwe. Algorytm genetyczny, gdzie reprezentacja danych jest w postaci binarnej nazywa się algorytmem klasycznym.

Parametrami algorytmu genetycznego są: prawdopodobieństwo krzyżowania p_c i prawdopodobieństwo mutacji p_m.

Początkową populację osobników dobieramy w sposób całkowicie losowy. Chromosomy (osobniki) w obecnej populacji oznaczamy jako v_i natomiast od-

powiadające im punkty przestrzeni (rozkodowane łańcuchy binarne) oznaczamy jako: $\overline{v_i}$.

Kroki algorytmu genetycznego[2]

1. Należy obliczyć wartość dopasowania dla każdego chromosomu $eval(v_j) = f(\overline{v_j})$ dla *j=1... n*, gdzie n oznacza rozmiar populacji (zwyczajowo oznaczone jako *popsize*)
2. Obliczyć całkowite przystosowanie populacji $F = \sum_{j=1}^{popsize} eval(v_j)$.
3. Obliczyć prawdopodobieństwo wyboru każdego chromosomu $p_j = \frac{eval(v_j)}{F}$, gdzie *j=1...popsize*.
4. Obliczyć dystrybuantę (prawdopodobieństwo skumulowane) dla każdego chromosomu $q_j \sum_{i=1}^{j} p_i = \sum_{i=1}^{j} \frac{eval(v_i)}{F}$.
5. Proces selekcji – polega na obrocie ruletką *popsize* razy (aby wybrać znów *popsize* chromosomów do nowej populacji) i wyborze jednego chromosomu do nowej populacji w następujący sposób:
   - wygenerować liczbę z zakresu $r \in \langle 0,1 \rangle$;
   - wybierz chromosom v_j dla którego zachodzi $q_{j-1} < r < q_j$;

   oczywiście pewne chromosomy (lepiej przystosowane) mogą zostać wybrane więcej niż jeden raz,
6. proces krzyżowania przeprowadzany jest w następujący sposób:
   - wygeneruj zmienno przecinkową liczbę z przedziału $r \in \langle 0,1 \rangle$;
   - jeśli $r < p_c$ to wybierz rozpatrywany chromosom do krzyżowania:

   Oczekiwana liczba chromosomów, które należy skrzyżować to $posize \cdot p_c$.

Następnie dla każdej pary chromosomów losujemy liczbę $pos \in \langle 1, m-1 \rangle$, która wyznacza miejsce krzyżowania dwóch chromosomów. Załóżmy, że mamy dwa chromosomy *(10010011)* i *(00111001)* losujemy pewną liczbę $pos = 3$ (jak wiadomo musi być ona z przedziału od 1 do 7 ponieważ wielkość naszych chromosomów, liczba zer i jedynek w nich, wynosi 8). Następnie rozdzielamy oba chromosomy po trzecim bicie $(100|10011), (001|11001)$. W ten sposób tworzy się dwa potomki, najpierw biorąc część pierwszą (tą przed linią) z pierwszego chromoso-

[2] Zb. Michalewicz, *Algorytmy genetyczne + struktury danych = programy ewolucyjne*, WNT, Warszawa 1999.

mu, następnie wybiera się część drugą z drugiego chromosomu (tą po linii pionowej), a później na odwrót. Potomki jakie powstały z wybranych chromosomów to: $(100|11001), (001|10011)$.

7. dla każdego chromosomu i dla każdego bitu w chromosomie wykonaj:
   - losujemy liczbę $r \in \langle 0,1 \rangle$;
   - jeśli $r < p_m$ to zmutuj bit (czyli zmień z 0 na 1 lub odwrotnie);

   Uwaga: każdy bit ma taką samą szansę na to by został zmutowany, zmieniony)
8. następnie oceniamy populację i wybieramy losowo chromosomy, które są najlepiej dopasowane (z pewnym prawdopodobieństwem)

Całość algorytmu genetycznego powtarzamy do momentu aż nie zostaną osiągnięte warunki stopu (często jest to pewna z góry ustalona ilość powtórzeń).

Zapiszmy teraz schemat kroków w postaci algorytmu:

program AlgorytmGenetyczny
t:= 0;
wybierz populację początkową P(t)
oceń P(t)
dopóki (nie jest spełniony warunek stopu) wykonaj
początek pętli
t:=t+1;
wybierz P(t) z P(t-1) // jest to selekcja
zmień P(t) // operatory genetyczne (krzyżowanie, mutacje)
oceń P(t)
koniec pętli
wyświetl najlepszy wynik z P(t)
Koniec programu.

Znając już zasadę działania algorytmów genetycznych można przejść do rozwiązywania zadania transportowego za pomocą algorytmu genetycznego.

2. Definiowanie zagadnienia transportowego

Zagadnienie transportowe polega na zaplanowaniu przewozów towarów od pewnej liczby dostawców (magazynów) do pewnej liczby odbiorców (sklepów), w taki sposób aby koszty transportu były jak najniższe. Założenia zagadnienia transportowego:

1. w każdym punkcie dostawy można odebrać nie więcej towaru niż pojemność takiego punktu a_i
2. do każdego punktu odbioru można dostarczyć nie więcej towaru niż jest on w stanie przyjąć b_i

3. transport odbywa się tylko wzdłuż zaplanowanych tras, oraz znane są koszty transportu pomiędzy dowolnymi punktami.

Zadanie to po sformułowaniu można zapisać w postaci matematycznej:

$$K = \sum_{i=1}^{n}\sum_{j=1}^{m} c_{ij} \cdot x_{ij} \rightarrow \min \tag{1}$$

gdzie: K – koszty transportu

c_{ij} - jednostkowy koszt transportu towaru na trasie *i-j;*

x_{ij} - ilość towaru przewożona na trasie *i-j*.

Funkcja ta oznacza funkcję celu (funkcja kosztów).

Funkcje ograniczeń:

$$\sum_{j=1}^{m} x_{ij} \leq a_i \quad \text{podaż dostawców nie może być przekroczona} \tag{2}$$

$$\sum_{i=1}^{n} x_{ij} \geq b_j \quad \text{popyt odbiorców musi być co najmniej zaspokojony} \tag{3}$$

Tak skonstruowane zadanie jest zadaniem ogólnym, aby rozwiązać go metodami analitycznymi (z wykorzystaniem rachunku macierzowego) należy dokonać zbilansowania zadania, czyli zastąpić znaki nierówności znakami równości poprzez wprowadzenie fikcyjnych punktów odbioru i źródła.

Zagadnienie transportowe można rozwiązywać za pomocą metod analitycznych takich jak metoda simpleks, kąta północno-zachodniego, najmniejszego elementu, VAM[3].

3. Przykładowe zagadnienie transportowe

Załóżmy, że pewna firma ma trzy magazyny i trzy sklepy oznaczone odpowiednio m_i i s_i gdzie $i = 1,2,3$. Ilość towaru w poszczególnych magazynach opisana jest wektorem:

$$m = \begin{bmatrix} 20 \\ 10 \\ 30 \end{bmatrix}$$

Natomiast zapotrzebowanie poszczególnych sklepów opisane jest wektorem:

$$s = \begin{bmatrix} 15 \\ 30 \\ 15 \end{bmatrix}$$

[3] J. Prońko, *Szczególne przypadki programowania liniowego*, wykład kursowy dla studentów Wydziału Zarządzania UJK w Kielcach, Kielce 2011.

W prezentowanym przykładzie mamy do czynienia z zagadnieniem transportowym zbilansowanym, ponieważ potrzeby sklepów są równe możliwościom magazynów.

Koszty jednostkowe transportu pomiędzy poszczególnymi magazynami i sklepami opisuje tabela 1.

Tabela 1. Jednostkowe koszty transportu między poszczególnymi magazynami i sklepami.

	s_1	s_2	s_3
m_1	1	3	4
m_2	5	2	1
m_3	3	2	8

Źródło: Opracowanie własne.

Sformułowany problem można zapisać w następującej postaci macierzowej:

$$f \cdot x \rightarrow \min \tag{4}$$

Przy warunkach:

$$A \cdot x = b \tag{5}$$

Do rozwiązania tak sformułowanego problemu posłużono się pakietem oprogramowania Matlab, w którym funkcja rozwiązująca zadanie programowania liniowego przyjmuje postać:

$$[x, fval] = linprog(f, A, b, Aeq, 1b, ub) \tag{6}$$

gdzie: x – wektor rozwiązań;

fval – wartość przy której funkcja $f \cdot x$ osiąga minimum;

f – wektor jednostkowych kosztów transportu;

A i b – współczynniki ograniczeń przy zagadnieniu niezbilansowanym;

Aeq i beq – macierze współczynników ograniczeń;

1b – dolne ograniczenie dla x (od jakiej wartości x musi być większe)

ub – górne ograniczenie dla x (od jakiej wartości x musi być mniejsze)

Jak wynika z funkcji (6) rozwiązanie tego zadania z wykorzystaniem pakietu matlab wymaga zdefiniowania:

wektora kosztów: $f = [1 \quad 3 \quad 4 \quad 5 \quad 2 \quad 1 \quad 3 \quad 2 \quad 8]$ (7)

macierzy współczynników ograniczeń: $$Aeq = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \quad (8)$$

wektora ograniczeń: $$beq = \begin{bmatrix} 20 & 10 & 30 & 15 & 30 & 15 \end{bmatrix} \quad (9)$$

Podstawiając powyższe dane do funkcji (6) otrzymamy:

$$fval = 105 \quad (10)$$

$$x = \begin{bmatrix} 15 \\ 0 \\ 5 \\ 0 \\ 0 \\ 10 \\ 0 \\ 30 \\ 0 \end{bmatrix} \quad (11)$$

Interpretacja wyników. Transportując towar z magazynów do sklepów w ilościach przedstawionych w tabeli 2 koszty tej operacji będą najmniejsze i wyniosą 105.

Tabela 2. Najtańszy sposób transportu towarów z magazynów do sklepu.

	s_1	s_2	s_3
m_1	15	0	5
m_2	0	0	10
m_3	0	30	0

Źródło: Opracowanie własne.

4. Rozwiązanie zagadnienia transportowego za pomocą algorytmu genetycznego

W klasycznym algorytmie genetycznym, jak już było wspominane, każdy chromosom jest zapisywany w postaci binarnej. Należy zatem najpierw zdefiniować, jak powinny wyglądać rozwiązania. Wiadomo, że w rozwiązując zadanie transportowe otrzymujemy wynik w postaci wektora składającego się z kilku liczb. Należy zatem każdą z tych liczb zapisać w postaci binarnej (czyli zero jedynkowej). Każde rozwiązanie w zadaniu transportowy będzie się zatem składać z pewnej liczby wektorów zawierających zera i jedynki.
Załóżmy, że rozwiązaniem zagadnienia transportowego jest wektor:

$$x = \left(x_1, x_2, \ldots, x_m\right)^T \tag{12}$$

Zatem, aby taki wektor był reprezentowany w postaci binarnej należy każdy z $x_{11}, x_{12}, \ldots, x_{nm}$ zapisać również w postaci binarnej:

$$x_{11} = \left(w_0^1, w_1^1, \ldots, w_p^1\right) \text{ gdzie } w_j \in \{0,1\} \tag{13}$$

Parametr p oznacza największą liczbę całkowitą jaką mogę te wektory reprezentować:

$$2^{p+1} - 1$$

Łatwo zauważyć, że taka reprezentacja danych spełnia ograniczenie $x_{ij} \geq 0$, ponieważ każdy z wektorów binarnych reprezentuje tylko liczby dodatnie. Wartości każdego x_{ij} przy tej reprezentacji należą do zbioru liczb całkowitych.

Funkcję oceny przy tej reprezentacji można zapisać jako:

$$eval\left(decimal\left(x_1\right), decimal\left(x_2\right), \ldots, decimal\left(x_m\right)\right) = \sum_{j=1}^{m} f_j \cdot x_j \tag{14}$$

Funkcja *decimal* oznacza funkcję zamiany z postaci binarnej na postać dziesiętną. Może być ona zapisana w takiej postaci[4]:

```
function decimal ($liczba)
{
$ilosc_miejsc = strlen($liczba);
$liczb_dziesietna = 0;

for ($i=$ilosc_miejsc;$i>-1;$i--)
{

$liczba_dziesietna += $liczba[$i-1]*pow(2,$ilosc_miejsc-$i);
}
return $liczba_dziesietna;
}
```

[4] Opracowanie własne, język programowania php.

Załóżmy, że operatory genetyczne definiowalibyśmy tak, jak w krokach algorytmu genetycznego. Okazuje się, że jeśli użyjemy klasycznej mutacji genów pojawia się problem. Przypomnijmy, że mutacja to jest minimalna zmiana genów najlepiej na pojedynczym bicie. Załóżmy teraz, że mamy bit postaci (0001) co odpowiada liczbie 1. Jeśli zrobimy minimalną zmianę w tym bicie np.: na miejscu pierwszym wstawimy 1 czyli otrzymamy reprezentację w postaci (1001) co odpowiada liczbie 10. Zmiana okazuje się już nie taka mała jak na początku. Mało tego przy próbie zachowania takiej mutacji powstaje konieczność poprawienia co najmniej dwóch innych genów aby ograniczenia zostały zachowane. To wymusiło skonstruowanie funkcji kontroli. Po wprowadzeniu tej funkcji mój algorytm genetyczny spisywał się bardzo dobrze (zawsze po pewnej liczbie iteracji powstawały odpowiedzi zbliżone często identyczne jak wynik, który otrzymałem w matlabie). Niestety funkcja poprawiająca wyniki wydłużyła bardzo czas działania algorytmu, co stawia pod znakiem zapytania opłacalność stosowania algorytmu genetycznego do rozwiązania tego zadania transportowego. Doszedłem jednak do wniosku, że można poprawić działania tego algorytmu dodając procedurę opisaną przez Zbigniewa Michalewicza, którą nazwał on inicjalizacja:

procedure inicjalizacja;
begin
ustaw wszystkie pozycje od 1 do m\cdot n jako nie odwiedzone
repeat
wybierz losowo liczbę q z zakresu 1 do m\cdot n i ustaw odpowiednią
pozycję jako odwiedzoną
podstaw (wiersz) i=(q-1)/k + 1 // zaokrąglamy do dołu
podstaw (kolumnę) j =(q-1) mod k+1
podstaw val=min(s[i],d[j])
podstaw x_i = val
podstaw s[i] = s[i]-val
podstaw d[j] = d[j] - val
until wszystkie wartosci będą odwiedzone

Po zastosowaniu tej procedury mój algorytm genetyczny stał się dużo bardziej wydajny (choć nadal dużo słabszy niż rozwiązanie w matlab).

5. Wnioski

Z moich obserwacji wynika, że nieopłacalnym jest konstruowanie algorytmu genetycznego dla liniowego zagadnienia transportowego klasy wskazanej w przykładzie. Chociaż otrzymywane wyniki są porównywalne z rozwiązaniami metodami analitycznymi, to jednak algorytmy genetyczne w tym przypadku są dużo wolniejsze.

Sądzę, iż warto się zastanowić nad zmianą reprezentacji danych w zagadnieniu transportowym z binarnego na bardziej naturalną reprezentację macierzową, powinno to poprawić wyniki i przyspieszyć działanie algorytmu genetycznego.

Zagadnienie przedstawione w przykładzie jest najprostszą formą zagadnienia transportowego. Spore zastrzeżenia, przy tak sformułowanym zadaniu, budzi statyczny opis jednostkowych kosztów transportu. W rzeczywistości kosztów tych nie da się tak opisać, ponieważ zależność między ilością transportowanego towaru a kosztami tego transportu jest nieliniowa i często również skokowa. Jest to związane z ładownością środków transportu. Jeżeli dysponujemy środkiem transportu o ładowności np.: 5t, to transport na określonej trasie ładunku 5 kg i 5 ton będzie niemal taki sam. Natomiast gdybyśmy zamierzali tym środkiem transportu przewieźć 6 ton to koszt będzie dwukrotnie większy, ponieważ musimy wykonać dwa kursy. Rozwiązanie takiego zagadnienia metodami analitycznymi jest bardzo skomplikowane, o ile w ogóle możliwe. Natomiast w przypadku zastosowania algorytmów genetycznych możemy takie zagadnienie rozwiązać.

Bibliografia:

1. Arabas J., *Wykłady z algorytmów ewolucyjnych*, WNT, Warszawa 2003.
2. Goldberg David E., *Algorytmy genetyczne i ich zastosowania*, WNT, Warszawa 1995.
3. Michalewicz Zb., *Algorytmy genetyczne + struktury danych = programy ewolucyjne*, WNT, Warszawa 1999.
4. Prońko J., *Szczególne przypadki programowania liniowego*, wykład kursowy dla studentów Wydziału Zarządzania UJK w Kielcach, Kielce 2011.
5. Rutkowski L., *Metody i techniki sztucznej inteligencji*, PWN, Warszawa 2006.
6. Studniarski M., *Teoria algorytmów ewolucyjnych*, wykład kursowy dla doktorantów, Łódź 2012.

Abstrakt

Artykuł zawiera podstawowy opis algorytmów genetycznych oraz ich zasadę działania. W dalszej części definiowane jest zagadnienie transportowe, oraz sposób jego rozwiązania metodami analitycznymi. W ostatniej części artykułu umieszczone jest rozwiązanie liniowego zbilansowanego zagadnienia transportowego za pomocą algorytmu genetycznego, oraz wnioski płynące z tego rozwiązania.

Use of Classical Genetic Algorithm for solving Balanced Transportation problems

The article contains an introductory description of genetic algorithms, and the principle of their operation. Later in the article, the transportation problem is defined, as well as the way to solve it by means of analytical methods. In the last part of the article, a solution of linear balanced transportation problem by use of the genetic algorithm is presented, together with the conclusions which may be drawn from this solution.

MBA Rafał Prońko, post-graduate student, University of Lodz.