Dr inż. Radosław WINICZENKO
Wydział Inżynierii Produkcji, SGGW w Warszawie

OPTYMALIZACJA KOSZTÓW TRANSPORTU METODĄ BEZPOŚREDNIEGO POSZUKIWANIA®

Celem artykułu jest prezentacja ogólnej zasady działania metody bezpośredniego poszukiwania oraz jej zastosowanie w liniowym problemie transportowym.

W przeciwieństwie do tradycyjnych metod szukania punktu minimum, w których dla znalezienia minimum wykorzystuje się informację o gradiencie funkcji celu lub o pochodnych różnego rzędu, w algorytmie metody bezpośredniego poszukiwania analizuje się wprowadzony zbiór punktów wokół bieżącego punktu.

Algorytm bezpośredniego poszukiwania może być zastosowany do zadań związanych z kosztami stałymi, zmiennymi ograniczeniami w postaci równań i nierówności czy wymaganiami dotyczącymi źródeł zaopatrzenia. Powyższe zadania często występują w różnych gałęziach przemysłowych, w tym również w branży spożywczej.

WPROWADZENIE

Do najprostszych zadań kombinatorycznych z ograniczeniami można zaliczyć zadanie transportowe. W problemie tym poszukuje się planu najtańszego transportu z pewnej liczby punktów nadania do pewnej liczby punktów odbioru. Od zadania oczekuje się podania: poziomu zapasu towaru w każdym punkcie nadania, wielkości zapotrzebowania w każdym punkcie odbioru oraz kosztów transportu z każdego punktu nadania do każdego punktu odbioru [8].

Jeśli w zadaniu występuje tylko jeden towar, to punkty odbioru mogą uzupełniać swoje zapotrzebowanie z jednego lub więcej punktu nadania. Celem takiego planu jest wyznaczenie ilości towaru wysłanego z każdego punktu nadania do każdego punktu odbioru, aby zminimalizować całkowity koszt transportu [9, 14].

Jeżeli koszt przejazdu jest wprost proporcjonalny do ilości transportowanego towaru, mowa jest o zadaniu transportowym liniowym. W przeciwnym przypadku, jeśli ten warunek będzie niespełniony, to zadanie transportu będzie zadaniem nieliniowym.

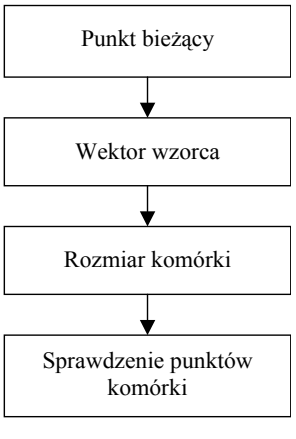
Jedną z metod optymalizacyjnych, które cieszą się znacznym zainteresowaniem, jest programowanie liniowe [5, 11, 13]. Metoda ta jest najbardziej przydatna do tworzenia sieci obiektów, przy czym wielkość popytu i podaży dla zakładów produkcyjnych, centrów dystrybucji lub poszczególnych rynków stanowią warunki ograniczające dla modelu. Przy danej funkcji celu, która zakłada na przykład minimalizację kosztu całkowitego, programowanie liniowe pomaga stworzyć optymalny wzorzec rozmieszczenia obiektów uwzględniający ograniczenia popytowo-podażowe.

Choć metoda programowania liniowego jest dość użyteczna, to jednak ma ograniczone zastosowania, ponieważ problem rozwiązywany za jej pomocą musi być sformułowany w sposób deterministyczny i powinien poddawać się liniowemu przybliżeniu. Ponadto w programowaniu liniowym nie można uwzględniać stałych i zmiennych kosztów funkcjonowania obiektów logistycznych [2, 3]. Takie rozwiązanie jest możliwe po zastosowaniu innych metod optymalizacyjnych opisanych w pracach [1, 4, 6, 7, 12, 15]. Jedną z nich jest metoda bezpośredniego poszukiwania, której zasadę działania opisano w niniejszym artykule. Może ona znaleźć zastosowanie w transporcie produktów spożywczych, gdzie mamy do czynienia z kosztami stałymi, zmiennymi ograniczeniami, jak również wymaganiami dotyczącymi źródeł zaopatrzenia.

METODA BEZPOŚREDNIEGO POSZUKIWANIA

Metoda bezpośredniego poszukiwania (z ang. Direct Search) jest metodą dla obliczania zadań optymalizacji, w której nie wykorzystuje się żadnej informacji o gradiencie funkcji celu. W przeciwieństwie do tradycyjnych metod szukania punktu minimum, w których dla znalezienia minimum wykorzystuje się informację o gradiencie funkcji celu lub o pochodnych różnego rzędu, w algorytmie metody bezpośredniego poszukiwania analizuje się wprowadzony zbiór punktów wokół bieżącego punktu. Tym punktem bieżącym w pierwszej iteracji jest losowo wybrany punkt startowy. Celem dalszego poszukiwania jest znalezienie takiego punktu, w którym wartość funkcji celu jest mniejsza niż wartość bieżącego punktu. Poszukiwanie kończy się wtedy, gdy osiągnięto minimalny rozmiar rozpatrywanego wzorca. Schemat działania algorytmu bezpośredniego poszukiwania przedstawiono na rysunku 1.

Powyższą metodę stosuje się najczęściej do zadań optymalizacji, gdy nie istnieje jakakolwiek informacja o różniczkowalności funkcji celu lub dla funkcji nieciągłej [10].



Rys. 1. Schemat działania algorytmu bezpośredniego poszukiwania.

Zasada działania algorytmu bezpośredniego poszukiwania

W algorytmach bezpośredniego przeszukiwania przeprowadza się numeryczne obliczenia takiej kolejności punktów, która zbiega się do punktu optymalnego. Na każdym kroku w algorytmie prowadzi się szukanie pewnego zbioru punktów, nazywanych komórką, wokół bieżącego punktu, który jest wynikiem obliczenia w poprzednim kroku. W danym algorytmie taką komórkę tworzy się drogą złożenia bieżącego punktu z pewnym skalarnym mnożnikiem z ustalonego zestawu wektorów, nazywanego *wzorcem* lub *szablonem* [10].

Wzorzec jest zbiorem wektorów, które są wykorzystywane w danym algorytmie do określenia punktów poszukiwania w każdej iteracji. Przykładem w zadaniu optymalizacyjnym mogą być dwie niezależne zmienne, dla których zaakceptowany domyślnie wzorzec składa się z następujących wektorów:

$$v\ 1 = [1\ 0]$$
$$v\ 2 = [0\ 1]$$
$$v\ 3 = [-1\ 0]$$
$$v\ 4 = [0\ -1]$$

Komórka jest pewnym zbiorem punktów. Algorytm formowania komórki ma następującą postać:

- Składniki wzorca (wektory) mnoży się przez pewien skalarny współczynnik, który nazywa się *rozmiarem komórki* (z ang. *mesh size*);
- Wektory uzyskane w wyniku mnożenia sumuje się z punktem bieżącym, czyli punktem o najlepszej wartości funkcji celu, znalezionej w poprzednim kroku.

Jeśli punkt bieżący będzie wynosił na przykład [1, 7 2, 4], wzorzec będzie składać się z przedstawionych powyżej wektorów, a bieżący rozmiar komórki jest równy na przykład 4, to wektory wzorca mnoży się przez współczynnik 4 a następnie sumuje się z punktem bieżącym, w wyniku czego powstaje komórka o następującej postaci:

$$[1, 7\ 2, 4] + 4\cdot[1\ 0] = [5, 7\ 2, 4]$$
$$[1, 7\ 2, 4] + 4\cdot [0\ 1] = [1, 7\ 6, 4]$$
$$[1, 7\ 2, 4] + 4\cdot [-1\ 0] = [-2, 3\ 2, 4]$$
$$[1, 7\ 2, 4] + 4\cdot [0\ -1] = [1, 7\ -1, 6]$$

Wektor wzorca, który został przyjęty jako punkt komórki, nazywa się *kierunkiem wzorca*.

Sprawdzanie punktów komórki

W każdej iteracji przeprowadza się sprawdzanie bieżących punktów komórki poprzez odpowiednie obliczanie wartości funkcji celu według przyjętego algorytmu. Procedura sprawdzania punktów komórki trwa do tego momentu, dopóki nie znajdzie wartości mniejszej niż dla bieżącego punktu. Po sukcesie znaleziony punkt staje się bieżącym punktem dla następnej iteracji. Następnie algorytm wpisuje ten drugi punkt w ciąg procesu poszukiwania. Jeśli w danej iteracji nie będzie takich punktów komórki, które mają wartość funkcji celu mniejszą niż wartość w punkcie poprzednim, to sprawdzenie będzie nieudane. W tym przypadku bieżący punkt już nie ulegnie zmianie dla następnej iteracji.

Procedura obliczania wartości funkcji celu może być przeprowadzona dla wszystkich bieżących punktów komórki. W tym przypadku przeprowadza się porównania wartości wszystkich bieżących punktów komórki z najmniejszą wartością funkcji celu. Jeśli w pewnym punkcie komórki znaleziona wartość funkcji celu będzie najmniejsza, to sprawdzenie punktu komórki odniosło sukces.

Bardziej szczegółowe informacje na temat metody bezpośredniego poszukiwania można znaleźć w literaturze [10].

Metoda bezpośredniego poszukiwania została zastosowana do typowego zadania transportowego, przedstawionego w poniższym przykładzie numerycznym. Podany przez autora przykład numeryczny został rozwiązany w programie Matlab w wersji 7.6.0 wyposażonym w pakiet do optymalizacji o nazwie „Algorytmy genetyczne i bezpośrednie poszukiwanie" (z ang. „Genetic Algorithms and Direct Serach").

Przykład numeryczny

Dwie hurtownie spożywcze: H_1 i H_2 dostarczają towar, którym jest cukier, do czterech sklepów zlokalizowanych w różnych miejscowościach: M_1, M_2, M_3, M_4. Jednostkowe koszty transportu (w zł), wielkości dostaw a_i (w tonach) oraz zapotrzebowanie sklepów b_i (w tonach) podano w tabeli 1.

Tabela 1. Jednostkowe koszty dla zadania transportowego hurtowni spożywczych

x_{ij}	M_1	M_2	M_3	M_4	a_i
H_1	6	3	3	7	800
H_2	2	6	9	8	800
b_i	100	300	500	700	1600

Celem niniejszego zadania optymalizacyjnego było opracowanie takiego planu transportu (przewozu) cukru, aby koszty były najmniejsze.

Model matematyczny dla tego zadania ma postać:

- Funkcja celu (funkcja kosztu):

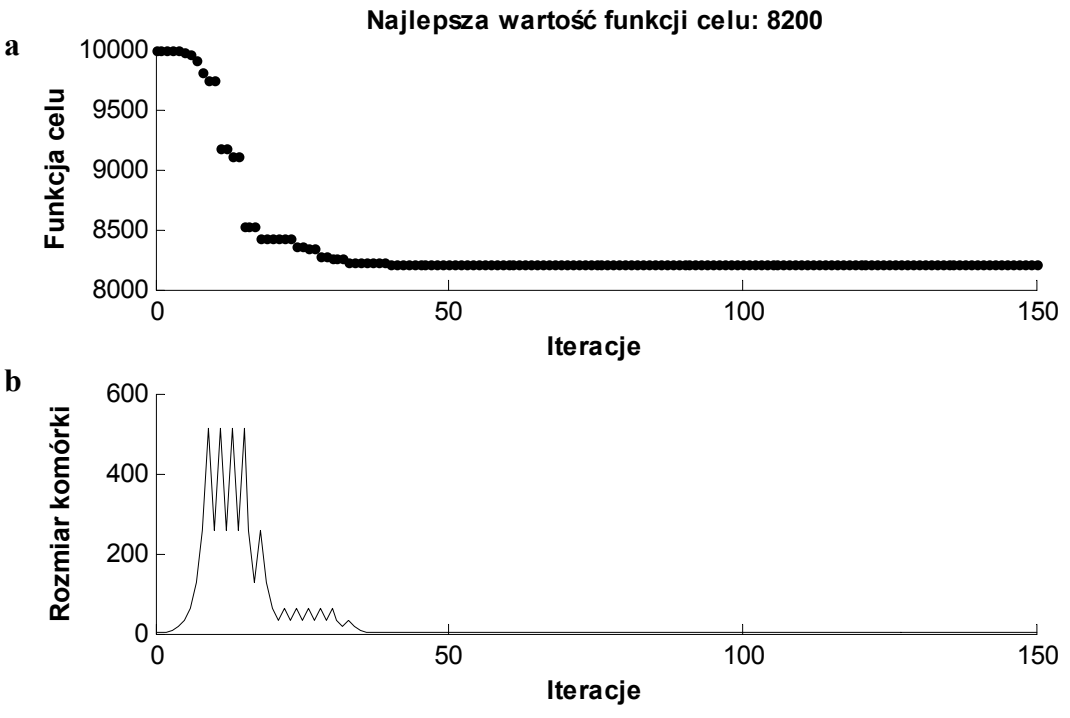
$$\min f(x) = 6\cdot x_{11} + 3\cdot x_{12} + 3\cdot x_{13} + 7\cdot x_{14} + 2\cdot x_{21} + 6\cdot x_{22} + 9\cdot x_{23} + 8\cdot x_{24};$$

- Równania ograniczające dla zmiennych decyzyjnych przyjmują postać:

$$x_{11} + x_{12} + x_{13} + x_{14} = 800;$$
$$x_{21} + x_{22} + x_{23} + x_{24} = 800;$$
$$x_{11} + x_{21} = 100;$$
$$x_{12} + x_{22} = 300;$$
$$x_{13} + x_{23} = 500;$$
$$x_{14} + x_{24} = 700;$$
$$x_{ij} \geq 0$$

Do obliczeń numerycznych przyjęto jako punkt startowy x_0=[10101010], który jest punktem początkowym. Przyjęto również procedurę sprawdzania punktów komórki, do momentu, dopóki algorytm nie znajdzie wartości mniejszej niż dla bieżącego punktu. Początkowy rozmiar komórki wyniósł 1.0.

Przebieg zmian funkcji celu (funkcji kosztu) w poszczególnych iteracjach pokazano na rys. 2a. Na wykresie widać, że funkcja osiągnęła wartość minimalną już w pierwszych iteracjach. Najlepsza wartość funkcji wyniosła f (x)=8200.



Rys. 2. Przebieg funkcji celu (kosztu) w poszczególnych iteracjach (rys 2-a) oraz rozmiar komórki w iteracjach (rys 2-b).

Na rysunku 2-a dla każdej iteracji wartości funkcji celu ulegają zmianie w ich najlepszym punkcie. Warto zauważyć, iż wartości funkcji celu dość szybko polepszają się we wczesnych iteracjach. Natomiast w miarę zbliżania się do optymalnej wartości, następuje ich wyrównanie.

Na rysunku 2-b ukazano rozmiar komórki dla każdej iteracji. W tym przypadku można zauważyć jak rozmiar komórki zwiększa się po każdej pomyślnej iteracji i zmniejsza po każdej niepomyślnej iteracji.

Z rozwiązania otrzymuje się: x_{11}=0, x_{12}=300, x_{13}=500, x_{14}=0, x_{21}=100, x_{22}=0, x_{23}=0, x_{24}=700 dla których funkcja celu (minimalizowana) wynosi f (x)= 8200. Otrzymane rozdysponowanie przewozów podano w tabeli 2.

Tabela 2. Rozdysponowanie przewozów zadania transportowego hurtowni spożywczych

c_{ij}	M_1	M_2	M_3	M_4	a_i
H_1	0	300	500	0	800
H_2	100	0	0	700	800
b_i	100	300	500	700	1600

Oznacza to, że minimalizację kosztów transportu otrzymuje się dla następującego planu przewozu: f (x) =3 · 300 + 3 ·500 + 2·100 + 8· 700 = 8200.

PODSUMOWANIE

Celem pracy zaprezentowanej w artykule było pokazanie zasady działania metody bezpośredniego poszukiwania na przykładzie ustalenia minimalnych kosztów transportu. Przy zastosowaniu odpowiedniego zapisu funkcji celu, warunków ograniczających oraz operatorów poszukiwań (np. wyboru poszukiwań, sposobu sprawdzenia komórki czy doboru rozmiaru komórki), algorytmy szybko znalazły rozwiązania w początkowych iteracjach.

Metoda bezpośredniego poszukiwania jest użyteczna dla obliczania zadań optymalizacji, w której nie wykorzystuje się żadnej informacji o gradiencie funkcji celu lub o pochodnych. W przeciwieństwie do tradycyjnych metod szukania punktu minimum, w metodzie bezpośredniego poszukiwania analizuje się wprowadzony zbiór punktów wokół bieżącego punktu.

Algorytm bezpośredniego poszukiwania może być zastosowany do zadań nieliniowych, problemów optymalizacyjnych związanych z kosztami stałymi, zmiennymi ograniczeniami potencjału, korzyściami skali oraz chociażby wymaganiami dotyczącymi źródeł zaopatrzenia. Powyższe zadania często występują w różnych gałęziach przemysłowych, w tym również w branżach przetwórstwa spożywczego.

LITERATURA

[1] Arabas J.: Wykłady z algorytmów ewolucyjnych, Warszawa, WNT, 2001.

[2] Biethahn J., Nissen V.: Evolutionary Algorithms in Management Applications, Berlin, Heidelberg, Springer-Verlag, 1995.

[3] Coyle J., Bardi J., Langley J.: Zarządzanie logistyczne, Warszawa, PWE, 2002.

[4] Cytowski J.: Algorytmy genetyczne: podstawy i zastosowania, Warszawa, PLJ, 1996.

[5] Gass Saull.: Programowanie liniowe, Warszawa, PWE, 1980.

[6] Goldberg D.E.: Algorytmy genetyczne i ich zastosowanie, Warszawa, WNT, 1998.

[7] Knosala R.: Zastosowanie metod sztucznej inteligencji w inżynierii produkcji, Warszawa, WNT, 2002.

[8] Michalewicz Z.: Algorytmy genetyczne + Struktura danych = Programy ewolucyjne, Warszawa, WNT, 1999.

[9] Nowak A.: Optymalizacja, Teoria i zadania, Gliwice, WPŚ, 2007.

[10] Ostanin A.: Informatyka z Matlabem, Rozprawy Naukowe Nr 147, Białystok 2007.

[11] Platt Cz.: Zastosowania programowania liniowego w rolnictwie i przemyśle spożywczym, Warszawa, PWE, 1990.

[12] Rutkowska D., Rutkowski L., Piliński M.: Sieci neuronowe, algorytmy genetyczne i systemy rozmyte, PWN, Warszawa 1999.

[13] Sierksma G.: Linear and Integer Programming, Theory and Practise, New York, Marcel Dekker, INC, 2002.

[14] Stachurski A., Wierzbicki A.: Podstawy optymalizacji, Warszawa, PW, 2001.

[15] Winiczenko R.: Algorytmy genetyczne i ich zastosowania, Postępy Techniki Przetwórstwa Spożywczego, 2008, Nr 1.

OPTIMIZATION OF COST TRANSPORTATION BY DIRECT SEARCH METHOD

SUMMARY

Direct search is a method for solving optimization problems which does not require any information about the gradient of the objective function. Unlike more traditional optimization methods which use information about the gradient or higher derivatives to search for an optimal point, a direct search algorithm searches a set of points around the current point, looking for one where the value of the objective function is lower than the value at the current point.

The paper presents a general principle of direct search method operation and their application in cost transportation problem in food industry.