ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1982

Seria: AUTOMATYKA z. 63 Nr kol. 735

Jarosław WARCZYŃSKI

Politechnika Poznańska

ZASTOSOWANIE PARAMETRYCZNEGO PROGRAMOWANIA LINIOWEGO Z ALGORYTMEM CHAČIJANA DO DWUKRYTERIALNYCH PROBLEMÓW ROZDZIAŁU ZASOBÓW

Streszczenie. W pracy przedstawiono metodę rozwiązania dwukryterialnych problemów rozdziału zasobów na drodze parametrycznego programowania liniowego, z zastosowaniem wielomianowego algorytmu Chacijana. Do wersji wyjściowej tego algorytmu wprowadzono kilka modyfikacji, które polepszają jego efektywność.

WPROWADZENIE

Problemy rozdziału zasobów dotyczą sytuacji, w których należy wykonać pewne powiązane ze sobą operacje, dysponując określoną ilością ograniczonych zasobów. Duże znaczenie praktyczne mają zwłaszcza badania dotyczące rozdziału zasobów odnawialnych, nieodnawialnych oraz tzw. podwójnie ograniczonych, rozpatrywanych we wspólnym modelu [6, 8]. Podejście takie znacznie wierniej odzwierciedla sytuacje praktyczne i stanowi obecnie jeden z najbardziej pożądanych kierunków badawczych.

Przedmiotem rozważań niniejszej pracy jest właśnie problem rozdziału zasobów odnawialnych i nieodnawialnych przy dyskretnych zapotrzebowaniach zasobowych operacji, w przypadku operacji podzielnych i niezależnych. W celu rozwiązania tego problemu proponujemy zastosowanie metody parametrycznego programowania liniowego z wielomianowym algorytmem Chačijana. Zastosowanie algorytmu Chačijana do rozwiązania zagadnienia parametrycznego programowania liniowego, wykorzystanego do rozwiązania dwukryterialnego programowania liniowego, do jakiego można sprowadzić wspomniany problem rozdziału zasobów, wydaje się szczególnie celowe ze względu na pewne właściwości tego algorytmu. Mianowicie, w odróżnieniu od algorytmu parametrycznego programowania liniowego, opartego na metodzie sympleksów, metoda oparta na algorytmie Chačijana pozwala na obserwację procesu dochodzenia do rozwiązań sprawnych. Cecha ta daje decydentowi możliwość lepszego wyboru rozwiązania zapewniającego najlepszy, w przyjętym sensie, kompromis pomiędzy kryteriami.

Rozdział 1 poświęcony jest prezentacji oryginalnej wersji algorytmu Chačijana. W rozdziale 2 przedstawiamy modyfikacje wprowadzone do tego algorytmu, które polepszają jego ogólną efektywność. W rozdziale 3 przedstawiamy metodę parametrycznego programowania liniowego opartą na algo-

rytmie Chačijana. W rozdziale 4 pokazano możliwość zastosowania, opisanej w rozdziale 3 metody do rozwiązania problemu rozdziału zasobów sformułowanego w postaci dwukryterialnego zadania programowania liniowego.

1. ALGORYTM CHAČIJANA

Od 1979 roku, kiedy L.G. Chačijan opublikował swój wielomianowy algorytm rozwiązania problemu programowania liniowego, rozstrzygnięta została kwestia przynależności problemu programowania liniowego do klasy problemów P. Fakt ten ma duże znaczenie teoretyczne w związku z tym, że wiele problemów decyzyjnych może być sprowadzonych do problemu programowania liniowego, a tym samym staje się możliwe rozwiązanie ich w czasie zależnym wielomianowo od rozmiaru.

Przedstawimy w skrócie ideę działania tego algorytmu. Algorytm Chacijana jest algorytmem stwierdzającym, czy układ nierówności liniowych:

$$A\,\underline{x} < \underline{b} \tag{1.1}$$

o współczynnikach całkowitych, posiada rozwiązanie. Jeśli tak, to algorytm znajduje je. Jak wiadomo, problem programowania liniowego może być przekształcony w układ nierówności liniowych. Idea algorytmu jest następująca:

Zbiór rozwiązań S układu nierówności (1.1), jeśli istnieje, to zawiera się w hiperkuli o promieniu zależnym od parametrów układu. Na każdej iteracji zbiór ten jest otaczany przez kolejne elipsoidy, których objętość maleje. Jednocześnie w każdym kroku sprawdza się, czy środek kolejnej elipsoidy należy do zbioru rozwiązań S. Jeśli należy, to algorytm znalazł rozwiązanie i zatrzymuje się. Jeżeli nie należy do zbioru S, to przeprowadza się przez niego hiperpłaszczyznę tnącą, równoległą do jednej z niespełnionych nierówności. Hiperpłaszczyzna ta dzieli daną elipsoidę na dwie części, z których tylko jedna zawiera zbiór S. Część tę otacza się kolejną elipsoidą o objętości mniejszej od poprzedniej, po czym powtarza się opisane postępowanie. W wypadku sprzeczności układu (1.1) algorytm zatrzymuje się, gdy objętość kolejnej utworzonej elipsoidy jest mniejsza niż najmniejsza teoretyczna objętość zbioru S. Ta sprzeczność wskazuje na to, że obszar S jest zbiorem pustym.

Przedstawimy obecnie w skrócie, wersję tego algorytmu podaną w [1].

Niech L będzie ilością bitów potrzebnych do zakodowania układu nierówności (1.1)

$$L = \left[\prod_{i=1}^{n} \sum_{j=1}^{n} \log_2(|a_{ij}|+1) + \sum_{i=1}^{n} \log_2(|b_i|+1) + \log_2 m\, n \right] + 1 \tag{1.2}$$

gdzie:

r - jest liczbą ograniczeń w (1.1),

n - jest liczbą zmiennych.

$[x]$ oznacza największą liczbę całkowitą, mniejszą lub równą x.

Oznaczmy przez:

$\underline{a}_i$ - wektor kolumnowy złożony z elementów a_{ij}, $j = 1,2,\ldots,n$,

k - numer iteracji algorytmu,

x^k - wektor n-elementowy, reprezentujący środek elipsoidy E^k w iteracji k,

B^k - macierz kwadratową $n \times n$ (symetryczną, dodatnio określoną).

Działanie algorytmu jest następujące:

Krok 1

$$x^0 := \underline{0}, \quad B^0 := 2^{2L} I, \quad K := 0.$$

Krok 2

Jeśli x^k jest rozwiązaniem dopuszczalnym układu (1.1), to zakończ obliczenia. Jeżeli $k < 4(n+1)^2 L$, to przejdź do kroku 3. W przeciwnym razie układ (1.1) nie ma rozwiązania.

Krok 3

Wybierz dowolną nierówność (1.1) nie spełnioną przez x^k, np. $\underline{a}_i^T x^k \geq \underline{b}_i$ i podstaw:

$$x^{k+1} := x^k - \frac{1}{n+1} \frac{B^k \underline{a}_i}{\sqrt{\underline{a}_i^T B^k \underline{a}_i}} \tag{1.3}$$

$$B^{k+1} := \frac{n^2}{n^2-1}\left(B^k - \frac{2}{n+1} \frac{(B^k \underline{a}_i)(B^k \underline{a}_i)^T}{\underline{a}_i^T B^k \underline{a}_i}\right) \tag{1.4}$$

$k := k+1$.

Wróć do kroku 2.

Dla symetrycznej, dodatnio określonej macierzy B oraz danego punktu $\underline{x}' \in R^n$, zbiór

$$E = \left\{\underline{x}: (\underline{x}-\underline{x}')^T B^{-1} (\underline{x}-\underline{x}') \leq 1\right\}, \tag{1.5}$$

określa elipsoidę ze środkiem w $\underline{x}'$. Oznaczmy przez $\frac{1}{2} E_a$ półelipsoidę:

$$\frac{1}{2} E_a = E \cap \left\{\underline{x}: \underline{a}_i^T (\underline{x}-\underline{x}') \leq 0\right\} \tag{1.6}$$

a przez E' zbiór:

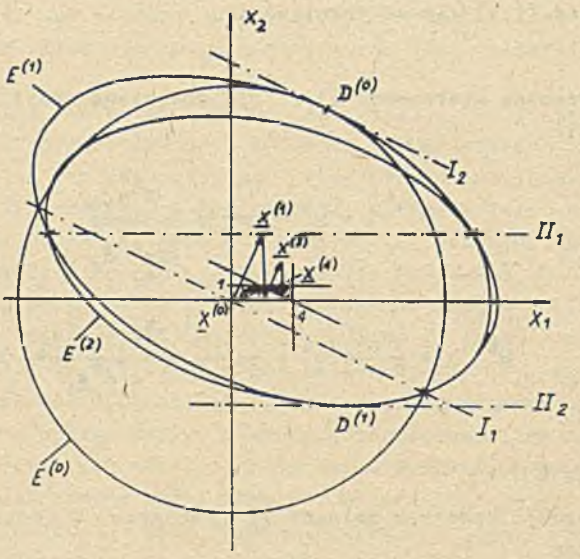
$$E' = \left\{ \underline{y} : (\underline{y}-\underline{y}')^T B'^{-1} (\underline{y}-\underline{y}') \leq 1 \right\}, \tag{1.7}$$

gdzie

$$\underline{y}' = \underline{x}' - \frac{1}{n+1} \frac{B\underline{a}_i}{\sqrt{\underline{a}_i^T B \underline{a}_i}} \tag{1.8}$$

$$B' = \frac{n^2}{n^2-1} \left(B - \frac{2}{n+1} \frac{(B\underline{a}_i)(B\underline{a}_i)^T}{\underline{a}_i^T B \underline{a}_i} \right) \tag{1.9}$$

geometrycznie oznacza to utworzenie elipsoidy E', stycznej do elipsoidy E w punkcie D, będącym punktem styczności elipsoidy E i hiperpłaszczyzny $\underline{a}_i^T \underline{x} = d$ $(d < \underline{a}_i^T \underline{x}_0)$, równoległej do naruszonego przez $\underline{x}$, i-tego ograniczenia oraz przecinającej hiperpłaszczyznę $\underline{a}_i^T \underline{x} = \underline{a}_i^T \underline{x}_0$ w tych samych punktach co elipsoida E (rys. 1).



Rys. 1. Geometryczna interpretacja działania algorytmu Chačijana

Jak wykazano w [1] elipsoida

$$E'(\underline{y}', B') \supseteq \tfrac{1}{2} E_a(\underline{x}', B) \tag{1.10}$$

oraz

$$\mathcal{A}(E') = c(n)\mathcal{A}(E) \tag{1.11}$$

przy czym

$$c(n) < 2^{-1/2(n+1)}, \tag{1.12}$$

gdzie:

$\mathcal{A}(E')$ i $\mathcal{A}(E)$ - objętości elipsoid E' i E.

Jeśli układ (1.1) jest niesprzeczny, to zbiór jego rozwiązań zawiera się w hiperkuli o promieniu mniejszym od 2^{L}, objętość tego zbioru

$$\mathcal{A}(S) \geq 2^{-(n+1)L} . \tag{1.13}$$

Z faktu, że w każdej iteracji zbiór rozwiązań S zawiera się wewnątrz bieżącej elipsoidy:

$$S \subseteq E^k \cap \left\{\underline{x}: \underline{a}_i^T(\underline{x}-x'^k) \leq 0\right\} \subseteq E^{k+1}$$

oraz z (1.11) i (1.13) wynika, że gorna granice liczby iteracji wynosi $k = 4(n+1)^2L$. Sprowadzenie problemu programowania liniowego do problemu rozwiązania układu nierówności może być przeprowadzone na podstawie teorii dualności programowania liniowego [1].

Jak wiadomo, jeśli problem prymalny programowania liniowego PL, maksymalizacji funkcji celu:

$$\max \underline{c}^T\underline{x} \tag{1.14}$$

przy ograniczeniach

$$A\underline{x} \leq \underline{b}, \quad \underline{x} \geq 0$$

posiada skończone rozwiązanie optymalne, to jest ono równe rozwiązaniu problemu dualnego minimalizacji funkcji:

$$\min \underline{b}^T\underline{x} \tag{1.15}$$

przy ograniczeniach

$$A^Ty \geq \underline{c}, \quad y \geq \underline{0}.$$

stąd układ nierówności:

$$\begin{aligned} \underline{c}^T\underline{x} &\geqslant \underline{b}^T\underline{y} \\ A\underline{x} &\leqslant \underline{b} \\ \underline{x} &\geqslant \underline{0} \\ A^T\underline{y} &\geqslant \underline{c} \\ \underline{y} &\geqslant \underline{0} \end{aligned} \tag{1.16}$$

ma rozwiązanie wtedy i tylko wtedy, gdy problem (1.14) ma skończone rozwiązanie optymalne.

2. MODYFIKACJE ALGORYTMU CHAČIJANA

Powyżej przedstawiona wersja algorytmu cechuje się kilkoma mankamentami, które można w pewnym stopniu przezwyciężyć na drodze wprowadzenia kilku modyfikacji. Modyfikacje te polepszają znacznie ogólną efektywność algorytmu.

2.1. Promień hipersfery początkowej

Liczba iteracji algorytmu w wypadku istnienia rozwiązania dopuszczalnego zależna jest między innymi od promienia hipersfery początkowej. Promień ten może być w sposób istotny zredukowany [2]. Jedynym ograniczeniem na długość tego promienia jest fakt, że powinien być on większy niż odległość od początku układu współrzędnych najbardziej odległego wierzchołka wielościanu rozwiązań dopuszczalnych. Współrzędne wierzchołków można oszacować stosując regułę Cramera do układu równań

$$A\underline{x} = \underline{b} \tag{2.1}$$

dla którego $m \geqslant n$, powstałego z układu (1.1). Współrzędne te można wyrazić jako:

$$\underline{x}_i = \frac{D_i}{D}, \quad i = 1,2,\ldots,n, \tag{2.2}$$

gdzie:

D - jest wyznacznikiem macierzy $n \times n$ współczynników odpowiednich równań,

D_i - jest wyznacznikiem tej samej macierzy, w której kolumna i zawiera odpowiednie elementy wektora $\underline{b}$.

Ponieważ współczynniki macierzy A oraz wektora $\underline{b}$ są liczbami całkowitymi, to jeżeli

$$D \neq 0 \quad \text{to} \quad x_i \leq D_i, \quad i = 1,\ldots,n. \tag{2.3}$$

Wartość wyznacznika D_i można oszacować w oparciu o nierówność Hadamarda:

$$|D|^2 = \prod_{i=1}^{n} \sum_{k=1}^{n} |a_{ik}|^2. \tag{2.4}$$

Stąd wartość wyznacznika D_i może być oszacowana jako iloczyn n-elementowych norm wektorów macierzy A. W celu znalezienia maksymalnej współrzędnej najdalszego (od początku układu współrzędnych) wierzchołka wielościanu rozwiązań dopuszczalnych, należy znaleźć iloczyn (n-1) maksymalnych n-elementowych norm wektorów macierzy A i pomnożyć go przez maksymalną n-elementową normę wektora $\underline{b}$. Ponieważ chcemy znaleźć długość promienia R hiperkuli zawierającej wierzchołek o maksymalnej współrzędnej, należy otrzymany wynik pomnożyć jeszcze przez $\sqrt{n}$.

Dla przykładowego problemu:

$$x_1 + x_2 < 2$$

$$x_1 < 1$$

$$x_1 + 2x_2 < 2$$

wartość promienia R, obliczonego w opisany sposób, wynosi $R = 7$, wobec $R = 2^6 = 4164$, dla wersji oryginalnej algorytmu.

2.2. Stabilność numeryczna

Miarą stabilności numerycznej algorytmu Chačijana jest liczba warunkowa macierzy B^k - $\mathrm{cond}(B^k)$ [4] zdefiniowana jako iloczyn norm B^k i jej macierzy odwrotnej. Jeśli liczba warunkowa macierzy B^k zwiększa się, to zwiększa się również dokładność niezbędna do obliczenia $x^{k+1}-x^k$ w (1.3) gdyż obliczana jest wówczas mała różnica dwóch wektorów o dużych składowych. Początkowo liczba warunkowa, $\mathrm{cond}(B^0) = 1$ we wszystkich przypadkach, po czym rośnie ona w kolejnych iteracjach algorytmu aż osiągnie swoją wartość maksymalną. $\mathrm{Cond}(B^k)$ nie zmniejsza się praktycznie nigdy poniżej 1/2 swojej wartości maksymalnej. Wykazano, że

$$\mathrm{Cond}(B^k) = \left[\mathrm{Cond}(A)\right]^2 \tag{2.5}$$

Zmniejszenie liczby warunkowej $\mathrm{cond}(B^k)$ możliwe jest dzięki zamianie macierzy B^k przez iloczyn $J^k(J^k)^T$, gdzie J^k jest macierzą kwadratową o rozmiarach $n \times n$ [5, 10]. W związku z powyższym wzór na uaktualnienie macierzy B^k można napisać w formie następującej:

$$J^{k+1}(J^{k+1})^T = \frac{n^2}{n^2-1} J^k \left[I - \frac{2}{n+1} \frac{(J^k)^T \underline{a}_i \underline{a}_i^T J^k}{\underline{a}_i^T J^k (J^k)^T \underline{a}_i} \right] (J^k)^T . \tag{2.6}$$

Oznaczając przez

$$z^k = \frac{(J^k)^T \underline{a}_i}{\| J^k \underline{a}_i \|} \tag{2.7}$$

$$\theta = \frac{n^2}{n^2-1} \tag{2.8}$$

można zapisać (2.6) w następującej formie:

$$J^{k+1}(J^{k+1}) = \theta J^k \left[I - \frac{2}{n+1} z^k (z^k)^T \right] (J^k)^T =$$

$$= \sqrt{\theta} J^k \left[I + \left(\sqrt{\frac{n-1}{n+1}} - 1 \right) z^k (z^k)^T \right]^2 (J^k)^T \sqrt{\theta} . \tag{2.9}$$

Z (2.9) wynika, że

$$J^{k+1} = \sqrt{\theta} J^k \left[I + t z^k (z^k)^T \right], \tag{2.10}$$

gdzie:

$$t = \frac{2}{1 + n + \sqrt{n^2-1}} \tag{2.11}$$

Stąd $\underline{x}^{k+1}$ może być określone jako:

$$\underline{x}^{k+1} = \underline{x}^k - \frac{1}{n+1} J^k z^k . \tag{2.12}$$

Macierz J^0 winna być określona jako:

$$J^0 = \sqrt{R} I , \tag{2.13}$$

gdzie R jest promieniem hipersfery początkowej określonym jak w rozdziale 2.1.

Opisana modyfikacja powoduje lepsze uwarunkowanie macierzy algorytmu.

$$\mathrm{Cond}(J^k) \sim \mathrm{Cond}(A) \tag{2.14}$$

W związku z tym, że obliczenia prowadzone przez maszyny cyfrowe dokonywane są ze skończoną dokładnością, na skończonej długości rejestrach, zdarza się, jak pokazują eksperymenty obliczeniowe, że macierz $J(J)^T$ przestaje być macierzą symetryczną.

W przypadku takiego wyradzania się macierzy $J(J)^T$ zostaje ona zamieniona na macierz diagonalną $\sqrt{\lambda_{max}}\,I\,\sqrt{\lambda_{max}}$, gdzie λ_{max} jest maksymalną wartością własną poprzedniej, jeszcze nie wyrodzonej, macierzy $J(J)^T$. Geometrycznie oznacza to, że w momencie wyrodzenia się macierzy $J(J)^T$ utworzona zostaje, nie kolejna elipsoida, lecz kula o promieniu równym najdłuższej półosi poprzedniej elipsoidy [3].

2.3. Efektywniejsze cięcia

Nietrudno przekonać się, że sprowadzenie zagadnienia programowania liniowego do układu nierówności liniowych poprzez połączenie ograniczeń problemu prymalnego i dualnego oraz dodanie nierówności:

$$\underline{b}^T\underline{y} \leqslant \underline{c}^T\underline{x} \tag{2.15}$$

stanowi najmniej korzystny przypadek z punktu widzenia algorytmu Chočijana. Algorytm musi bowiem, w tym przypadku, znaleźć jeden punkt będący rozwiązaniem, to znaczy, że środek kolejnej utworzonej elipsoidy musi pokryć się z punktem stanowiącym rozwiązanie, lub też, trafić w obszar o promieniu ε, jeśli (2.15) zapiszemy w postaci:

$$\underline{c}^T\underline{x} - \underline{b}^T\underline{y} \leqslant |\varepsilon| , \tag{2.16}$$

gdzie:

ε - jest dokładnością z jaką algorytm powinien działać.

Z doświadczeń z algorytmem Chačijana wiadomo, że dla znalezienia rozwiązania leżącego w pewnym obszarze nie potrzebuje on zbyt wielu iteracji. Stąd efektywniejszym będzie nieco inne postępowanie w wypadku rozwiązywania zagadnienia programowania liniowego. Mianowicie, jeśli kolejne $\underline{x}^k$ nie leży w obszarze rozwiązań dopuszczalnych danego zadania programowania liniowego, to postępowanie jest analogiczne jak w wersji oryginalnej. Jeśli jednak $\underline{x}^k$ należy do zbioru rozwiązań dopuszczalnych, to hiperpłaszczyzną tnącą będzie w tym wypadku hiperpłaszczyzna równoległa do funkcji celu, tj.

$$\underline{c}^T\underline{x} = \underline{c}^T\underline{x}^k.$$

Kolejna elipsoida otoczy tę połowę poprzedniej, w której może być spełniona nierówność:

$$\underline{c}^T\underline{x} > \underline{c}^T\underline{x}^k.$$

Zagadnienie programowania liniowego zostało więc w tym przypadku sprowadzone do układu nierówności:

$$\begin{aligned} A\underline{x} &\leq \underline{b} \\ \underline{x} &\geq 0 \\ \underline{c}^T\underline{x} &\geq \underline{c}^T\underline{x}^k \end{aligned} \tag{2.17}$$

3. METODA PARAMETRYCZNEGO PROGRAMOWANIA LINIOWEGO OPARTA NA ALGORYTMIE CHAČIJANA

Parametryczne programowanie liniowe, w odniesieniu do zmian współczynników kosztu, pozwala badać zachowanie się rozwiązań zagadnienia programowania liniowego przy zmianach współczynników w funkcji celu.

Rozpatrzmy następujące zagadnienie programowania liniowego:

$$\mathrm{Max}\left[\lambda\underline{c}^T + (1-\lambda)\underline{c}'^T\right]\underline{x}, \quad \lambda\in\langle 0, 1\rangle$$

przy ograniczeniach:

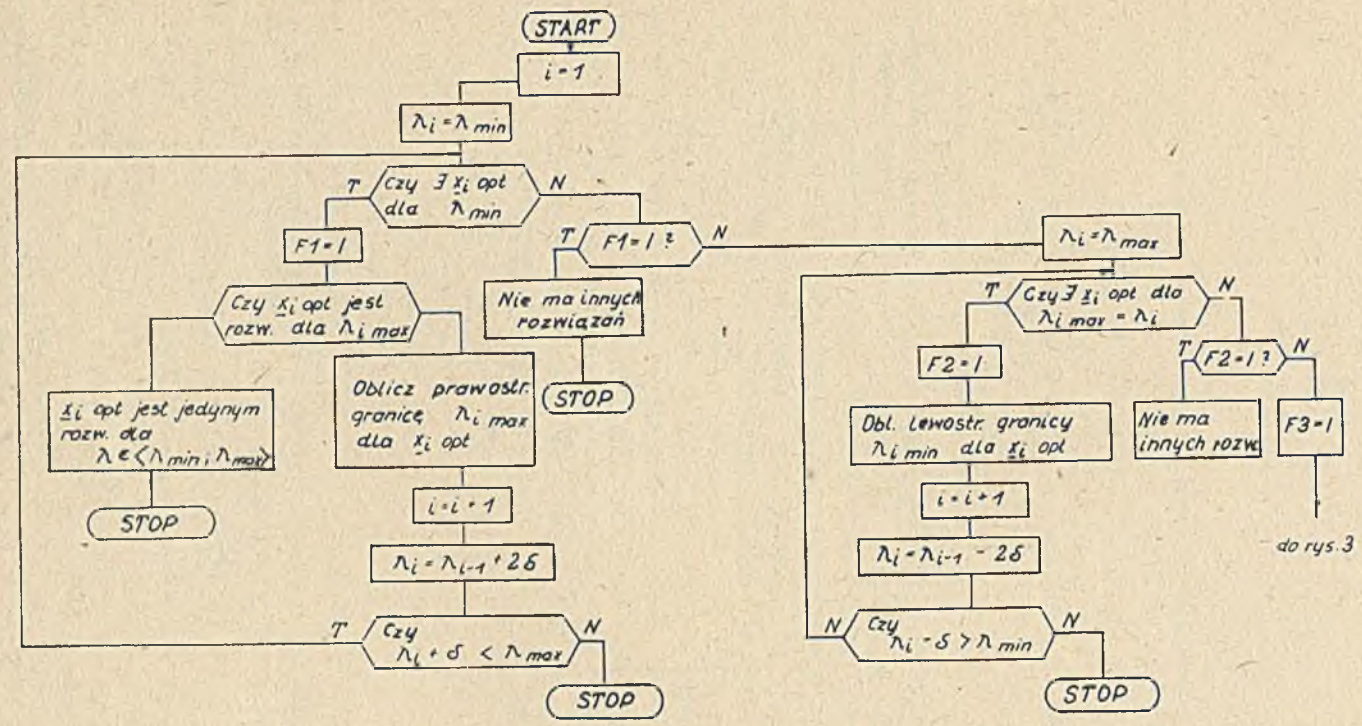
$$\begin{aligned} A'\underline{x} &\leq \underline{b}' \\ \underline{x} &\geq \underline{0}\ . \end{aligned} \tag{3.1}$$

Sprowadzamy zagadnienie (3.1) do układu nierówności:

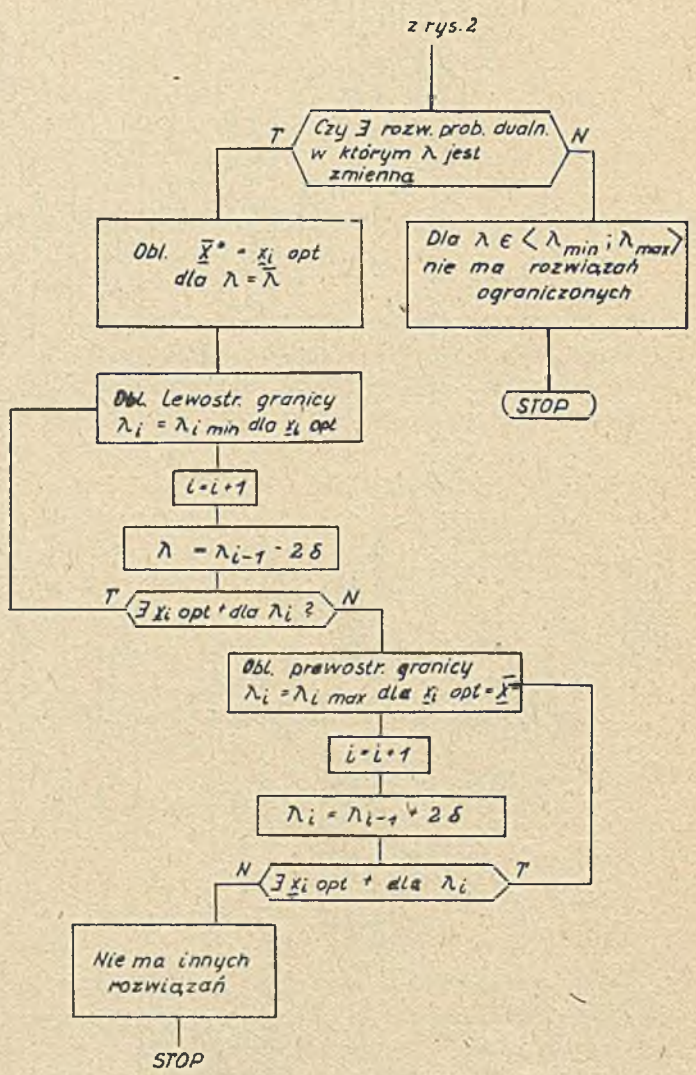
$$\begin{aligned} \left[\lambda\underline{c}^T + (1-\lambda)\underline{c}'^T\right]\underline{x} &\geq \left[\lambda\underline{c}^T + (1-\lambda)\underline{c}'^T\right]\underline{x}^k \\ A'\underline{x} &\leq \underline{b}' \\ \underline{x} &\geq \underline{0}\ . \end{aligned} \tag{3.2}$$

Do rozwiązania tego problemu proponujemy zastosowanie algorytmu, który poniżej przedstawiamy. Schemat blokowy tego algorytmu przedstawia rys. 2 i 3.

Idea algorytmu oparta jest na znanej zasadzie podziału odcinka. W pierwszym kroku badamy, czy istnieje rozwiązanie optymalne dla parametru $\lambda = \lambda_{min}$. Wykorzystujemy do tego celu algorytm Chačijana. W celu wcześniejszego wykrycia nieistnienia skończonego rozwiązania, równolegle prowadzone jest poszukiwanie rozwiązania układu dualnego, nierówności:



Rys. 2. Schemat blokowy algorytmu metody parametrycznego programowania liniowego z algorytmem Chačijana



Rys. 3. Schemat blokowy algorytmu metody parametrycznego programowania liniowego z algorytmem Chačijana

$$A^T \underline{y} \geqslant \underline{0}$$

$$\underline{y} \geqslant \underline{0} \qquad (3.3)$$

$$b^T \underline{y} < \underline{0},$$

gdzie

A jest macierzą powstałą z macierzy A' poprzez dodanie wiersza:

$$\lambda \underline{c}^T + (1-\lambda)\underline{c}'^T,$$

$\underline{b}$ jest wektorem powstałym z wektora $\underline{b}'$ poprzez dodanie elementu:

$$\left[\lambda \underline{c}^T + (1-\lambda)\underline{c}'^T\right]\underline{x}^k,$$

Jak wiadomo, układ nierówności (3.3) posiada rozwiązanie wtedy i tylko wtedy, gdy układ (3.2) rozwiązania nie posiada. Stąd, algorytm zarówno w przypadku sprzeczności, jak i niesprzeczności układu (3.2) znajduje rozwiązanie po liczbie iteracji mniejszej niż maksymalna liczba - konieczna do określenia problemu wyjściowego.

Jeśli rozwiązanie problemu (3.2) istnieje, to algorytm naszej metody znajduje je, po czym bada, czy rozwiązanie to - $\underline{x}^1$ jest rozwiązaniem optymalnym dla $\lambda = \lambda_{max}$. Z teorii programowania liniowego wiadomo, że przedział parametru, w którym istnieją rozwiązania optymalne, skończone jest przedziałem domkniętym i spójnym. Stąd, jeśli znalezione w kroku 1 rozwiązania optymalne, jest rozwiązaniem optymalnym także dla λ_{max} to znaczy, że jest ono rozwiązaniem optymalnym dla wszystkich λ z przedziału λmin, λmax. Jeśli $\underline{x}^1$ nie jest rozwiązaniem optymalnym dla λmax, to algorytm bada czy $\underline{x}^1$ jest rozwiązaniem optymalnym dla

$$\lambda_1 = \frac{\lambda_{min} + \lambda_{max}}{2}.$$

Jeśli nie, to postępowanie powtarza się, tzn. $\lambda_2 = \frac{\lambda_{min} + \lambda_1}{2}$. W wypadku znalezienia λ, dla którego $\underline{x}^1$ jest rozwiązaniem optymalnym, następuje poszukiwanie z zadaną dokładnością ε granicznej wartości parametru $\lambda = \lambda_{graniczne}$, dla której $\underline{x}^1$ jest rozwiązaniem optymalnym.

W kroku następnym do znalezionej wartości $\lambda = \lambda_{granicz.}$ dodaje się wielkość 2ε, po czym następuje powrót do kroku pierwszego, tzn. poszukiwanie rozwiązania optymalnego dla nowej wartości $\lambda = \lambda_{granicz.} + 2\varepsilon$. Postępowanie to powtarza się dopóki algorytm nie osiągnie końcowej wartości λ z zadaną dokładnością ε lub zostaje przerwane z chwilą, gdy dla kolejnego $\lambda = \lambda^k_{granicz.} + 2\varepsilon$ nie istnieje skończone rozwiązanie optymalne,

co oznacza, że rozwiązania optymalne nie istnieją w następnych przedziałach parametru.

Jeśli w kroku pierwszym algorytmu okaże się, że nie istnieje rozwiązanie optymalne dla $\lambda = \lambda_{min}$, to następuje badanie czy istnieje rozwiązanie optymalne dla $\lambda = \lambda_{max}$. Jeśli tak, to całe, opisane powyżej postępowanie powtarza się, z tym, że tym razem przedział zmienności parametru jest badany od przeciwległego krańca.

Jeżeli nie istnieją rozwiązania optymalne ani dla $\lambda = \lambda_{min}$, ani dla λ_{max}, wówczas poszukuje się rozwiązania dopuszczalnego ograniczeń problemu dualnego, w których, jak wiadomo λ występuje w wektorze prawych stron. Parametr λ traktujemy jako zmienną. Jeśli rozwiązanie takie nie istnieje, to znaczy, że nie istnieją skończone rozwiązania optymalne w całym przedziale parametru $\lambda \in \langle 0,1 \rangle$.

Jeśli rozwiązanie zostaje znalezione, to tym samym znaleziona zostaje wartość parametru $\lambda = \bar{\lambda}$, dla której musi istnieć skończone rozwiązanie optymalne problemu prymalnego (3.2). Wartość $\lambda = \bar{\lambda}$, zostaje zapamiętana, po czym następuje rozwiązanie problemu PL (3.2) dla danego $\lambda = \bar{\lambda}$. Po znalezieniu rozwiązania $\underline{\bar{x}}^1$, optymalnego dla $\bar{\lambda}$ następuje, analogicznie jak w krokach opisanych powyżej, poszukiwanie lewostronnej granicy wartości parametru, przy której znalezione rozwiązanie $\underline{\bar{x}}^1$ jest jeszcze optymalne.

W kolejnych krokach poszukuje się granicznych wartości parametru dla kolejnych rozwiązań optymalnych.

Z chwilą stwierdzenia, że nie istnieje rozwiązanie optymalne dla kolejnej wartości parametru:

$$\lambda = \bar{\lambda}^k_{granicz.} - 2\varepsilon,$$

następuje powrót do $\bar{\lambda}$ i $\underline{x} = \underline{\bar{x}}^1$, dla którego znaleziona zostaje prawostronna granica parametru. Następnie, analogicznie jak w krokach poprzednich, poszukuje się nowych rozwiązań optymalnych odpowiadających przedziałom parametru położonym na prawo od $\bar{\lambda}$ i granicznych wartości parametru, dla których rozwiązania te pozostają optymalnymi.

W momencie nie znalezienia rozwiązania optymalnego dla kolejnej wartości parametru:

$$\bar{\lambda}^k = \bar{\lambda}^{k-1}_{granicz.} + 2\varepsilon,$$

algorytm zatrzymuje się.

W celu sprawdzenia czy dane rozwiązanie optymalne $\underline{x} = \underline{x}^k_{opt.}$ jest rozwiązaniem optymalnym dla następnej wartości parametru λ stosujemy procedurę z algorytmem Chačijana, która bada układ nierówności:

$$A^T\underline{y} \geq \left[\lambda \underline{c}^T + (1-\lambda)\underline{c}'^T\right]\underline{x}$$

$$\underline{b}^T\underline{y} \leq \left[\lambda \underline{c}^T + (1-\lambda)\underline{c}'^T\right]\underline{x}_{opt.} + \varepsilon \qquad (3.4)$$

$$\underline{y} \geq \underline{0} .$$

Jeśli układ jest niesprzeczny, to $\underline{x} = \underline{x}^k_{opt.}$ pozostaje rozwiązaniem optymalnym również dla $\lambda = \lambda^{k+1}$.

4. ZASTOSOWANIE PARAMETRYCZNEGO PROGRAMOWANIA LINIOWEGO Z ALGORYTMEM CHAČIJANA DO DWUKRYTERIALNEGO PROBLEMU ROZDZIAŁU ZASOBÓW

Jak zostało wspomniane na wstępie, przedstawiona w rozdziale 3 metoda może być wykorzystana do rozwiązania pewnej klasy problemów rozdziału zasobów. Mianowicie, do problemu rozdziału zasobów odnawialnych i nieodnawialnych, przy dyskretnych zapotrzebowaniach zasobowych operacji, w przypadku operacji podzielnych. Tutaj ograniczymy się do przypadku operacji niezależnych.

Przypomnijmy, że problem rozdziału polega na znalezieniu takiego przydziału zasobów ze zbioru $\mathcal{R}$ do operacji ze zbioru $\mathcal{A}$, który zapewni wykonanie wszystkich operacji, przy zachowaniu nałożonych ograniczeń i który zapewni najlepszy, w przyjętym sensie, kompromis pomiędzy kryteriami ze zbioru Q [9].

Zbiór $\mathcal{R}$ zawiera zasoby odnawialne, tzn. takie, na które nałożone są ograniczenia na dostępność w każdej chwili oraz zasoby nieodnawialne, na które nałożone są ograniczenia zużycia w pewnym przedziale czasowym.

Zbiór $\mathcal{A}$ zawiera niezależne i podzielne operacje charakteryzujące się:

- dyskretnymi zapotrzebowaniami zasobowymi,
- modelem matematycznym w postaci wektora czasów wykonywania.

Zbiór Q zawiera kryterium typu czasowego, które jest związane z wykorzystaniem zasobów odnawialnych oraz jedno lub więcej kryteriów typu kosztowego, związanych z wykorzystaniem zasobów nieodnawialnych. Kryteria kosztowe i czasowe są sobie przeciwstawne.

W [7] pokazano, że problem taki może być sprowadzony do problemu wielokryterialnego programowania liniowego. Jeśli kryteria typu kosztowego zagreguje się w jedno kryterium, to otrzymamy problem dwukryterialny, który można rozwiązać za pomocą przedstawionej metody parametrycznego programowania liniowego.

Kryteria kosztu K i czasu T wykonania zbioru operacji $\mathcal{A}$ przedstawimy w postaci funkcji parametrycznej:

$$Z = \lambda K + (1-\lambda)T. \qquad (4.1)$$

W wyniku zastosowania zaproponowanej metody otrzymamy wówczas zbiór rozwiązań sprawnych (pareto-optymalnych), który w przestrzeni kryteriów przedstawia się w postaci krzywej K(T), z której decydent będzie mógł wybrać rozwiązanie optymalne w sensie najlepszego kompromisu między obydwoma kryteriami.

LITERATURA

[1] Aspvall R., Stone R.E.: Khachiyan's linear programming algorithm. Report STAN-CS-79-776. Department of Computer Science. Stanford University.

[2] Berresford G.C., Rockett A.M., Stevenson J.C.: Khachiyan's algorithm and the microcomputer. Byte. Aug. 1980.

[3] Kaliński J.: Referat wygłoszony na seminarium IA Politechnika Poznańska. Czerwiec 1981

[4] Rosen J.B., Frawley G.: A computational test of Khachiyan's algorithm for linear inequalities. Technical Report 79-28, Computer Science Department, University of Minesota.

[5] Ruszczyński A.: The relation between the Khachiyan's algorithm and the subgradient method with space dilation. Raport IA Politechnika Warszawska.

[6] Słowiński R.: Two approaches to problems of resource allocation among project activities - a comparative study. J. Opi. Res. Soc. 31/1980/ n. 2.

[7] Słowiński R.: Multiobjective network scheduling withefficient use of renewable and nenrenewable resources. EJOR 1981, n. 3.

[8] Węglarz J.: New models and procedures for resource allocation problems. Proc. of 6-th Internet Congres. vo 12, VDI GmbH Dusseldorf, 1979.

[9] Węglarz J.: Sterowanie w systemach typu kompleks operacji. PWN, Warszawa-Poznań 1981.

[10] Wolfe Ph.: A bibliography for the ellipsoid algorithm. Research Report. RC 8237 (35829) 4/39/80. Mathematical Science Department. IBM Th. J. Watson Research Center.

Recenzent: Doc. dr inż. Wojciech TARNOWSKI

Wpłynęło do Redakcji 15.05.1982 r.

ПРИМЕНЕНИЕ ПАРАМЕТРИЧЕСКОГО ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ С АЛГОРИТМОМ ХАЧИЯНА К ДВУХ КРИТЕРИАЛЬНОЙ ПРОБЛЕМЕ РАСПРЕДЕЛЕНИЯ РЕСУРСОВ

Р е з ю м е

В настоящей работе рассмотрена проблема распределения по операциям обновляемых и необновляемых ресурсов, в случае дискретности запросов операций на ресурсы.

Представлен метод решения двухкритериальной проблемы распределения ресурсов путём параметрического линейного программирования с применением модифицированного полиномиального алгоритма Хачияна. Действие метода основано на известном принципе разделения отрезка.

APPLICATION OF PARAMETRIC LINEAR PROGRAMMING WITH KHACHIYAN'S ALGORITHM TO TWO-OBJECTIVE RESOURCE ALLOCATION PROBLEM

S u m m a r y

We presented a method of solving certain allocation problem based on a modification of the polynomial Khachiyan's algorithm. Two categories of constrained resources are considered: renevable and nonrenevable. The general idea of our method is based on the well known sector division principle.