Polskie Towarzystwo Badań Operacyjnych i Systemowych
Instytut Badań Systemowych Polskiej Akademii Nauk
Wojskowa Akademia Techniczna

Redaktorzy:
Zbigniew Nahorski
Marian Chudy
Andrzej Straszak

OPTYMALIZACJA

Warszawa 1991

POLSKIE TOWARZYSTWO
BADAŃ OPERACYJNYCH I SYSTEMOWYCH

INSTYTUT BADAŃ SYSTEMOWYCH
POLSKIEJ AKADEMII NAUK

WOJSKOWA AKADEMIA TECHNICZNA

OPTYMALIZACJA

ZADANIA, METODY, ALGORYTMY

Redaktorzy

Zbigniew Nahorski, Marian Chudy, Andrzej Straszak

WARSZAWA 1991

OPTYMALIZACJA:
Zadania, metody, algorytmy
PTBOiS, Warszawa 1991.

MINIMALIZACJA POTRZEB PAMIĘCI PRZY OBLICZANIU WARTOŚCI WYRAŻEŃ ARYTMETYCZNYCH

Zygmunt Kaszubowski
Wojskowa Akademia Techniczna, Warszawa

Streszczenie: Rozważa się zagadnienie takiego ustalenia dopuszczalnej kolejności obliczania wartości wyrażenia arytmetycznego, przy którym potrzeby pamiętania parametrów wyrażenia oraz wyników pośrednich będą minimalne. Bazując na teorii grafów i sieci sprowadza się ten problem do rozwiązania zadania optymalnego uporządkowania wierzchołków dendrytu obrazującego to wyrażenie. Porównawczo sformułowano powyższy problem w języku programowania binarnego wykazując znaczną złożoność obliczeniową tego ujęcia.

1. Opis problemu.

Wyrażenia arytmetyczne mogą być obrazowane drzewem skierowanym z korzeniem (dendrytem), w którym z każdego wierzchołka wychodzą co najwyżej dwa łuki (dendryt binarny). Dane początkowe odpowiadają wierzchołkom wiszącym dendrytu, a pośrednie wyniki wierzchołkom wewnętrznym. Korzeń dendrytu odzwierciedla wartość obliczanego wyrażenia. Każdy wierzchołek wewnętrzny przedstawia binarną operację nad argumentami odpowiadającymi jego następnikom. Porządek, w którym realizowane są poszczególne operacje, może mieć znaczny wpływ na potrzeby pamięci. Przykładowo wyrażenie

$$\Big[(a+b) - c * d\Big] / \Big[e * (f-g)\Big]$$

przedtawia dendryt z rys.1.

Wybranie porządku obliczeń oznacza topologiczne uporządkowanie wierzchołków dendrytu obrazującego to wyrażenie. Dopuszczalna kolejność wykonywania operacji odzwierciedla się faktem, że łuki przy takim uporządkowaniu

prowadzą jedynie od wierzchołków dalszych do bliższych. Na rys. 2 przedstawiono dwa dopuszczalne uporządkowania dendrytu z rys. 1. Pierwsze z nich wymaga 7 komórek pamięci, a drugie odzwierciedlające optymalną kolejność obliczeń, wymaga 3 komórek pamięci. Łuki przechodzące nad wierzchołkiem v odpowiadają zapotrzebowaniu na pamięć dla danych lub wyników pośrednich. Jeśli liczba tych łuków wynosi $\omega(v)$, to musimy wykorzystywać w tym etapie obliczeń $\omega(v)+1$ komórek pamięci (jedna dla zapamiętania wyniku operacji zobrazowanej wierzchołkiem v).

Zadanie minimalizacji ilości komórek pamięci komputera potrzebnych przy obliczaniu wartości wyrażenia arytmetycznego można sprowadzić (patrz. [1]) do zadania uporządkowania wierzchołków dendrytu o minimalnej szerokości.

2. Optymalne porządkowanie wierzchołków dendrytu.

Obiektem rozważań jest dendryt $T = (V,E) = (V,\Gamma)$ z korzeniem r, gdzie: V – zbiór wierzchołków, $|V|=n$,

E – zbiór łuków, $|E|=m$,

Γ – funkcja określająca następniki wierzchołków.

W pracy [1] rozpatruje się tzw. numerację wierzchołków, polegającą na przyporządkowaniu każdemu z wierzchołków $v \in V$ jednego numeru ze zbioru liczb $N = \{1,2,\ldots,n\}$ wyczerpującą cały ten zbiór. W innej terminologii oznacza to uporządkowanie topologiczne wierzchołków dendrytu. Tę numerację (uporządkowanie) możemy traktować jako funkcję $F: V \xrightarrow{na} N$, gdzie $F(v)$ jest numerem uzyskanym przez v lub pozycją w uporządkowaniu. Uporządkowanie F nazywamy dopuszczalnym, gdy dla każdego $(x,y) \in E$ zachodzi nierówność $F(x) > F(y)$.

W uporządkowaniu F łuk (x,y) przechodzi nad wierzchołkiem z, gdy zachodzi : $F(y) < F(z) < F(x)$. Natomiast łuki (x,y) oraz (v,z) przecinają się, gdy zachodzi jeden z warunków:

a) łuk (x,y) przechodzi nad v oraz łuk (v,z) przechodzi nad y;

b) łuk (x,y) przechodzi nad z oraz łuk (v,z) przechodzi nad x.

Dopuszczalną numerację nazywamy płaską, gdy żadne z jego łuków nie przecinają się. Oznaczmy przez $\omega(x)$ liczbę łuków przechodzących nad wierzchołkiem x w uporządkowaniu F. Szerokością uporządkowania F jest funkcjonał $W(F) = \max_{x \in V} \omega(x)$. Uporządkowanie, którego szerokość jest minimalna ze względu na wszystkie dopuszczalne uporządkowania dendrytu T, nazywamy uporządkowaniem o minimalnej szerokości, a jego szerokość - szerokością dendrytu.

W pracy [1] podano algorytm wyznaczania uporządkowania o minimalnej szerokości w algolopodobnym języku. Algorytm ten opiera się na następujących intuicyjnie zasadnych przesłankach:

a) optymalne uporządkowanie powinno być płaskie;

b) rozpatrujemy jednokrotnie każdy z wierzchołków v dendrytu T i porządkujemy optymalnie poddendryt $T(v)$, dla którego v jest korzeniem;

c) kolejność rozpatrywania poszczególnych wierzchołków jest odwrotna do numeracji uzyskanej w tzw. poszukiwaniu wszerz z korzenia r;

d) dla każdego poddendrytu $T(v)$ porządkujemy jego wierzchołki w podciągi odpowiadające poddendrytom $T(z)$ wierzchołków $z \in \Gamma(v)$ ułożone w kolejności nierosnącej względem ich szerokości, zakończone wierzchołkiem v.

Iteracyjne rozpatrywanie poszczególnych poddendrytów oraz fakt jednokrotnego rozpatrywania każdego z wierzchołków zapewnia korzystną złożoność obliczeniową tego algorytmu.

3. Przedstawienie problemu uporządkowania dendrytu w języku programowania binarnego.

Definiujemy binarną macierz decyzyjną $X = (x_{ik})_{n \times n}$ określającą uporządkowanie F, której element

$$x_{ik} = \begin{cases} 1 - \text{gdy i-ty wierzchołek jest umieszczony na k-tym miejscu,} \\ 0 - \text{w przeciwnym przypadku.} \end{cases}$$

Dopuszczalność uporządkowania zadanego macierzą X jest określona zestawem warunków:

$$\begin{cases} \sum_{k=1}^{n} x_{ik} = 1, & \text{dla} \quad i = \overline{1,n} \\ \sum_{i=1}^{n} x_{ik} = 1, & \text{dla} \quad k = \overline{1,n} \\ \sum_{k=1}^{n} k \cdot x_{ik} - \sum_{k=1}^{n} k \cdot x_{jk} \geq 1, & \text{dla } (i,j) \in E \end{cases}$$

Niech $\bar{\omega}(k)$ oznacza ilość łuków przechodzących nad wierzchołkiem znajdującym się na k-tym miejscu w uporządkowaniu określonym X. Wielkość tę określa wyrażenie

$$\bar{\omega}(k) = \sum_{d=1}^{k-1} \sum_{m=k+1}^{n} \sum_{(l,j) \in E} x_{jd} \cdot x_{lm}, \quad \text{dla } 2 \leq k \leq n-1,$$

przy czym $\bar{\omega}(1) = \bar{\omega}(n) = 0$.
Szerokość uporządkowania określonego macierzą X wynosi

$$\bar{W}(X) = \max_{2 \leq k \leq n-1} \bar{\omega}(k)$$

Należy wyznaczyć macierz X^* określającą optymalne uporządkowanie z warunku

$$\bar{W}(X^*) = \min_{X \in \Omega} \max_{2 \leq k \leq n-1} \bar{\omega}(k)$$

gdzie

$\Omega = \{X=(x_{ik})_{n \times n} \mid x_{ik} \in \{0,1\}$, spełnione są war.(1),(2),(3)$\}$.

W celu uwolnienia się od minimaksowej postaci kryterium

(patrz [4]) wprowadzamy zmienną

$$x_o = \max_{2\le k\le n-1} \sum_{d=1}^{k-1} \sum_{m=k+1}^{n} \sum_{(l,j)\in E} x_{jd}\cdot x_{lm}$$

i rozszerzamy zestaw ograniczeń o warunek (4) otrzymując

$$\min_{(X,x_o)} x_o$$

$$\begin{cases} \text{przy ogr. (1),(2),(3)} \\ \text{oraz (4)} \ \sum_{d=1}^{k-1} \sum_{m=k+1}^{n} \sum_{(l,j)\in E} x_{jd}\cdot x_{lm} \le x_o \ , \ \text{dla } 2\le k\le n-1. \end{cases}$$

Dokonując binarnego rozwinięcia zmiennej x_o ($x_o \le n-2$)

$$x_o = \sum_{k=0}^{r_o-1} 2^k \cdot x_{ok} \ , \ \text{gdzie } x_{ok}\in\{0,1\}, \text{ a } r_o \text{ dobieramy z}$$

$$\text{z warunku } 2^{r_o-1} \le n-2 \le 2^{r_o}-1,$$

oraz linearyzując (patrz [5]) warunki (4) przez wprowadzenie nowych zmiennych $y_{jdlm}\in\{0,1\}$ w miejsce iloczynów $x_{jd}\cdot x_{lm}$ i po dwa dodatkowe ograniczenia

$$\begin{cases} x_{jd} + x_{lm} - y_{jdlm} \le 1 \\ x_{jd} + x_{lm} - 2y_{jdlm} \ge 0 \end{cases}$$

uzyskujemy równoważny problem programowania liniowego binarnego (PLB), który możemy rozwiązać w oparciu o algorytm Balasa.

Interesujące jest tu oszacowanie rozmiaru problemu:

- ilość zmiennych: $n^2 + \log_2(n-2) + \frac{1}{6}\, m\cdot n(n-1)(n-2)$
- ilość ograniczeń: $3(n-1) + m + \frac{1}{3}\, m\cdot n\,(n-1)(n-2)$.

Dla rozpatrywanego przykładu z rys.1, gdzie n=13, m=12 mamy:

- zmiennych 169 + 4 + 3432 = 3605
- ograniczeń 36 + 12 + 6864 = 6912

Trzecie z liczb występujących w oszacowaniach ilości zmiennych i ograniczeń wskazują wzrost tych wielkości z tytułu linearyzacji zagadnienia. Czyni to przedstawiany drugi sposób wyznaczania uporządkowania wierzchołków dendrytu o minimalnej szerokości praktycznie nieprzydatnym.

4. Uwagi końcowe.

Przedstawione rozważania wskazują przewagę algorytmu grafowego opisanego w punkcie 2 nad ewentualnym zastosowaniem algorytmu Balasa dla rozwiązania zagadnienia PLB przedstawionego w punkcie 3.

Literatura

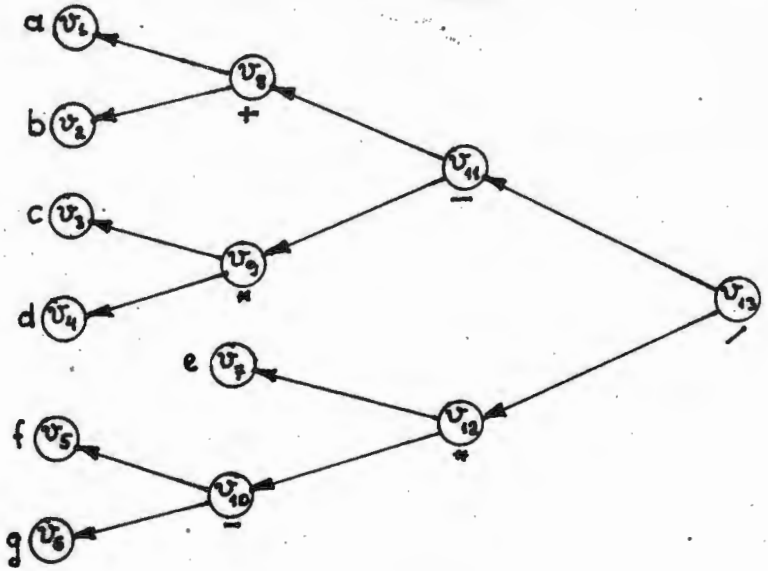
[1] Евстигнеев В.А.: Применение теории графов в программировании. Изд. ''Наука,, Москва 1985r.

[2] Deo N.: Teoria grafów i jej zastosowania w technice i informatyce. PWN, W-wa 1980r.

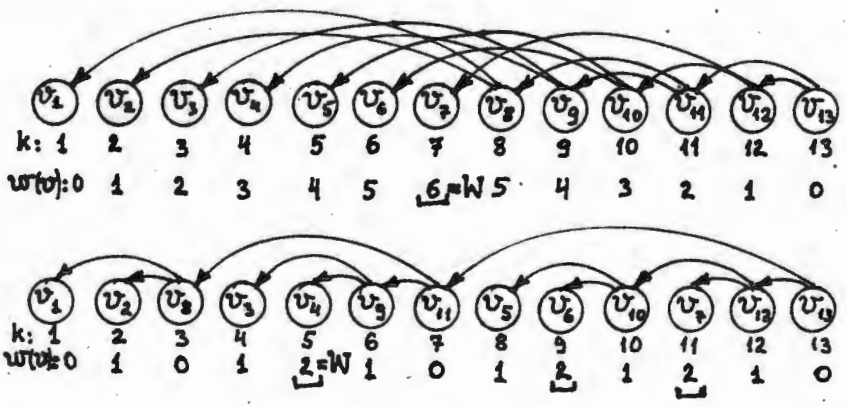
[3] Korzan B.: Elementy teorii grafów i sieci. WNT, W-wa 1978r.

[4] Grabowski W.: Programowanie matematyczne. PWE, W-wa 1980r.

[5] Walukiewicz S.: Programowanie dyskretne. PWN, W-wa 1986r.



Rys.1. Dendryt wyrażenia ((a+b)-c*d)/(e*(f-g))



Rys.2. Dopuszczalne uporządkowania dendrytu

ISBN 83-900412-1-9.