O METODĂ DE PROGRAMARE A PRODUSULUI A DOUĂ MATRICI

DE

E. MUNTEANU și T. RUS

(Cluj)

Lucrarea prezentată la sesiunea asupra problemelor teoretice și practice privitoare la automatizările industriale, noiembrie 1961, București

Să considerăm două matrici

$$\mathbf{A} = (a_{ij}) \quad (i = 1, 2, \ldots, n;\ j = 1, 2, \ldots, m), \tag{1}$$

$$\mathbf{B} = (b_{ki})(k = 1, 2, \ldots, p) \tag{2}$$

și produsul lor $\mathbf{C} = \mathbf{A} \cdot \mathbf{B} = (c_{jk})$, unde

$$c_{jk} = \sum_{i=1}^{n} a_{ij} b_{ki}. \tag{3}$$

Pentru a realiza produsul acestor două matrici cu o mașină electronică de calcul, sînt necesare, în afara celulelor ocupate de program, $m \times n + p \times n = n(p + m)$ celule pentru păstrarea elementelor celor două matrici inițiale. Se întîmplă adeseori că pentru mașinile electronice, cu memorii relativ mici, problema produsului să constituie o dificultate din punctul de vedere al păstrării în memorie a elementelor celor două matrici. În multe probleme, se întîmplă că printre elementele celor două matrici, există multe elemente nule. În aceste cazuri, o bună parte din celulele memoriei care păstrează elementele nule rămîn neutilizate.

În cele ce urmează, vom da o metodă de programare a produsului a două matrici folosind economic memoria mașinii, prin păstrarea exclusivă a elementelor diferite de zero.

Vom presupune că matricea $\mathbf{A}$ are r elemente diferite de zero iar matricea $\mathbf{B}$, q elemente diferite de zero. Elementele diferite de zero ale matricii $\mathbf{A}$, le înscriem în memorie pe coloane, în celulele succesive

$$a + 1,\ a + 2,\ \ldots,\ a + r, \tag{4}$$

iar elementele diferite de zero ale matricei **B** le înscriem în memorie pe linii, în celule succesive

$$b+1,\ b+2,\ \ldots,\ b+q. \tag{5}$$

Pentru a determina linia și coloana în care se află un element diferit de zero într-una din cele două matrici, vom atașa fiecăreia dintre ele cîte un șir de indici în felul următor: fiecărui element diferit de zero, facem să-i corespundă 1 și fiecărui element nul facem să-i corespundă 0. În felul acesta obținem pentru matricea A pe coloane, șirul

$$i_1,\ i_2,\ i_3,\ \ldots,\ i_{n\times m}, \tag{6}$$

unde

$$i_k = \begin{cases} 1, & \text{dacă elementul al } k\text{-lea al matricii A, numărat pe coloane, este diferit de zero,} \\ 0, & \text{în caz contrar;} \end{cases}$$

iar pentru matricea B șirul

$$j_1,\ j_2,\ j_3,\ \ldots,\ j_{p\times u}, \tag{7}$$

unde

$$j_k = \begin{cases} 1, & \text{dacă elementul al } k\text{-lea al matricei B, numărat pe linii, este diferit de zero,} \\ 0, & \text{în caz contrar.} \end{cases}$$

Elementele șirului (6) fiind 0 sau 1, le vom înscrie pe ordine succesive în celulele

$$s+1,\ s+2,\ \ldots\ s+h_1, \tag{8}$$

unde

$$h_1 = \frac{m\times n}{H}, \tag{8'}$$

H fiind numărul ordinelor unei celule, iar elementele șirului (7) le vom înscrie deasemenea pe ordine succesive în celulele

$$t+1,\ t+2,\ \ldots,\ t+h_2, \tag{9}$$

unde

$$h_2 = \frac{n\times p}{H}. \tag{9'}$$

De asemenea în memorie se păstrează un pas n, care reprezintă numărul elementelor unei coloane pentru matricea **A**, respectiv a unei linii pentru matricea **B**, un pas p, care reprezintă numărul liniilor matricei **B** și un pas m care reprezintă numărul coloanelor matricii **A**. Pentru a realiza produsul a două elemente corespunzătoare din matricile **A** și **B**, mașina va descifra în șirurile respective (6) și (7) dacă indicii i și j, corespunzători acestor elemente, sînt sau nu diferiți de zero. Dacă sînt diferiți de zero, elementele respective se caută și se face produsul lor, dacă nu, se trece mai departe. Cînd se obține un element al matricei **C**, acesta se tipărește și numai pe urmă se trece mai departe. Tipărirea elementelor matricei produs, în momentul obținerii lor, duce la o economisire a unui număr destul de mare de celule, ceea ce în cazul mașinilor cu memorie mică este esențial. Dacă este necesar ca matricea produs **C** să fie păstrată în memorie, acest lucru se realizează destul de ușor.

Schema logică operatorială a algoritmului care realizează produsul a două matrici după metoda de mai sus este

$$\overset{12}{\downarrow}[(s+1)\to\alpha_1]\overset{13}{\downarrow}\overset{6}{\downarrow}\mathrm{A}_1\mathrm{F}(K_a)p(K_a<H+1)\overset{1}{\uparrow}\overset{10}{\downarrow}\times$$
$$\times\ [(t+1)\to\beta_1]\,\mathrm{A}_2\mathrm{F}(K_b)p(K_b<H+1)\overset{2}{\uparrow}p((\alpha)=1)\overset{3}{\uparrow}\mathrm{F}(r)\times$$
$$\times\ p((\beta)=1)\overset{4}{\uparrow}\mathrm{F}(q)\,\Phi(\mathrm{A}_3)\mathrm{A}_3\overset{3}{\downarrow}\overset{4}{\downarrow}\mathrm{F}(i)p(i<n)\overset{5}{\uparrow}\mathrm{A}_{i_1}\times$$
$$\times\overset{9}{\downarrow}\mathrm{A}_5\Phi(\mathrm{A}_3)\mathrm{O}\overset{6}{\uparrow}\overset{5}{\downarrow}p(c_{jk}\neq 0)\overset{7}{\uparrow}\mathrm{T}(c_{jk},1]\overset{2}{\downarrow}\mathrm{T}(0)\mathrm{F}(k)\times$$
$$\times\ p(k<p)\overset{8}{\uparrow}\mathrm{F}^{-n}(i)(0\to<K_a>][0\to<r>][0\to<c_{kj}>]\times$$
$$\times\overset{9}{\uparrow}\overset{2}{\downarrow}\mathrm{F}^{-(H+1)}(K_b)\mathrm{O}\overset{10}{\uparrow}\overset{8}{\downarrow}\mathrm{F}(j)p(j<m)\overset{11}{\uparrow}\mathrm{F}^{-n}(i)\mathrm{F}^{-p}(k)\times$$
$$\times\ [0\to<q>][0\to<K_b>][0\to(c_{jk}>]\Phi([(t+1)\to\beta_1])\mathrm{A}_6\times$$
$$\times\ \Phi(\mathrm{A}_3)\mathrm{O}\overset{13}{\uparrow}\overset{1}{\downarrow}\mathrm{F}(s)\mathrm{F}^{-(H+1)}(K_a)\mathrm{O}\overset{12}{\uparrow}\overset{11}{\downarrow}\ \text{Stop.}$$

Operatorul $[(s+1)\to\alpha_1]$ transportă prima celulă care conține pe ordine elemente din șirul (6) într-o celulă operativă α_1.

Operatorul A_1 face ca primul ordin al celulei α_1 să fie pus în evidență pentru a ne putea da seama dacă este 0 sau 1. El este un operator aritmetic.

$F(K_a)$ este un operator de redresare care mărește pe K_a cu o unitate, K fiind inițial zero. Acesta numără ordinele unei celule și ne spune cînd trebuie să trecem la o celulă următoare care conține elementele șirului (6).

Operatorii $[(t+1)\to\beta_1]$, A_2, $F(K_b)$ sînt operatori absolut analogi cu cei descriși mai sus, ei se referă la celulele care conțin termenii șirului (7). Condiția logică $p(K_b<H+1)$ stabilește de asemenea momentul alegerii celulei următoare care conține elemente ale șirului (7).

Condiția logică $p[(\alpha)\equiv 1]$ stabilește dacă primul ordin al celulei operative α este 1 sau 0. Dacă este 1, operatorul $F(r)$ mărește indicele r cu o unitate, ceea ce duce la determinarea elementului matricii **A** diferit de zero din șirul (4). În mod analog, acționează condiția logică $p[(\beta)\equiv 1]$ și operatorul $F(q)$ pentru șirul (5), relativ la matricea **B**.

Operatorul $\Phi(A_3)$ este un operator de formare a operatorului aritmetic A_3, care execută produsul a două elemente corespunzătoare din șirurile (4), respectiv (5) și adună acest produs la o celulă fixă $<c_{jk}>$. Acest operator este singurul operator de calcul și intră în funcțiune numai atunci cînd avem de a face cu două elemente corespunzătoare, diferite de zero, ale celor două matrici.

Operatorul $F(i)$ mărește indicele i cu o unitate, ceea ce înseamnă că se trece la un element următor al unei coloane din matricea **A**. Condiția logică $p(i < n)$ verifică dacă am epuizat sau nu o coloană a matricii **A**. Operatorii A_4 și A_5 fac ca ordinele următoare ale celulelor operative α_1, respectiv β_1, să devină primele lor ordine. Ei sînt operatori aritmetici.

Condiția logică $p(c_{jk} \neq 0)$ stabilește dacă, după ce am efectuat produsul unei coloane a matricii **A** cu o linie a matricii **B**, rezultatul este zero sau nu. Dacă rezultatul este diferit de zero, se va tipări conținutul celulei $< c_{jk} >$ urmat de tipărirea unității. Dacă conținutul celulei $< c_{jk} >$ este zero, se tipărește 0. În acest fel rezultatul apare sub forma unui șir în care, după fiecare element diferit de zero, urmează un 1, iar după fiecare element zero, urmează un element diferit de zero. Din acest șir se poate atunci extrage șirul

$$c + 1,\ c + 2,\ \ldots,\ c + l, \tag{10}$$

care conține elementele diferite de zero ale matricii produs **C** și șirul

$$p_1,\ p_2,\ p_3,\ \ldots,\ p_{m \times p}, \tag{11}$$

unde

$$p_k = \begin{cases} 1, & \text{dacă al } k\text{-lea element al matricii } \mathbf{C} \text{ numărat pe} \\ & \text{coloane este diferit de zero;} \\ 0, & \text{în caz contrar.} \end{cases}$$

Astfel, matricea produs **C** apare complet determinată.

Operatorul $F(k)$ este un operator de redresare care numără liniile matricii **B**. Condiția logică $p(k < p)$ ne spune dacă liniile matricii **B** s-au epuizat sau nu.

Operatorul $F^{-n}(i)$ este un operator de restabilire. El reface valoarea indicelui i la cea inițială.

Operatorii $[0 \rightarrow < K_a >]$, $[0 \rightarrow < r >]$, $[0 \rightarrow < c_{jk} >]$ sînt operatori de restabilire a conținutului celulelor $<K_a>$, $<r>$, $<c_{jk}>$, la valoarea lor inițială, dar deoarece nu se știe cantitatea cu care trebuiesc modificate aceste celule, ei nu se pot nota în mod obișnuit ca și operatorii de restabilire.

Operatorul A_6 este un operator aritmetic care face ca ordinul următor al celulei operative α_1 să devină primul său ordin. Restul operatorilor sînt de aceeași natură ca și cei explicați mai sus.

Pe baza schemei operatoriale date și descrise mai sus, vom întocmi acum programul pentru calculatorul electronic CIFA-2.

001	$01 - s + 1$	$11 - \alpha_1$
002	$01 - \alpha_1$	$14 - <30>$
003	$11 - \alpha$	$01 - <K>$
004	$02 - <1>$	$11 - <K>$
005	$03 - <32>$	$06 - 058$

006	$01 - l + 1$	$11 - \beta_1$
007	$14 - <30>$	$11 - \beta$
008	$01 - <K>$	$02 - <1>$
009	$11 - <K>$	$03 - <32>$
010	$06 - 045$	$01 - \alpha_1$

011	$03 - <1>$	$07 - 023$
012	$01 - <r>$	$02 - <1>$
013	$11 - <r>$	$01 - \beta$
014	$03 - <1>$	$07 - 023$
015	$01 - <q>$	$02 - <1>$
016	$11 - <q>$	$01 - <r>$
017	$14 - <6>$	$02 - 021$
018	$11 - 021$	$01 - <q>$
019	$14 - <21>$	$02 - 021$
020	$11 - 021$	$13 - 000$
021	$01 - a$	$04 - b$
022	$02 - <c_{jk}>$	$11 - <c_{jk}>$

023	$01 - < i >$	$02 - <1>$
024	$11 - < i >$	$03 - <n>$
025	$06 - 030$	$01 - \alpha_1$
026	$15 - <1>$	$11 - \alpha_1$
027	$01 - \beta_1$	$15 - <1>$
028	$11 - \beta_1$	$01 - <c>$
029	$11 - 021$	$10 - 002$

030	$01 - <0>$	$03 - <c_{jk}>$
031	$06 - 035$	$13 - 000$
032	$01 - <c_{jk}>$	$16 - 000$
033	$01 - <1>$	$16 - 000$
034	$10 - 038$	$13 - 000$
035	$01 - <c_{jk}>$	$03 - <0>$
036	$07 - 032$	$01 - <0>$
037	$16 - 000$	$13 - 000$

038	$01 - <k>$	$02 - <1>$
039	$11 - <k>$	$03 - <p>$
040	$06 - 048$	$01 - s + 1$
041	$11 - \alpha_1$	$01 - <0>$
042	$11 - < i >$	$11 - <r>$
043	$11 - <c_{jk}>$	$11 - <K_a>$
044	$10 - 027$	$13 - 000$

045	$01 - 006$	$02 - <n_1>$
046	$11 - 006$	$01 - <0>$
047	$11 - <K_b>$	$10 - 006$

048	$01 - < j >$	$02 - <1>$
049	$11 - < j >$	$03 - <m>$
050	$06 - 061$	$01 - <0>$
051	$11 - <q>$	$11 - <K_b>$
052	$11 - < i >$	$11 - <k>$
053	$11 - <c_{jk}>$	$01 - <m_1>$
054	$11 - 006$	$01 - \alpha_1$
055	$15 - <1>$	$11 - \alpha_1$
056	$01 - <c>$	$11 - 021$
057	$10 - 002$	$13 - 000$

058	$01 - 001$	$02 - <n_1>$
059	$11 - 001$	$01 - <0>$
060	$11 - (K_a>$	$10 - 001$
061	$00 - 000$	

Pe lîngă cele 61 celule ocupate de program în cazul mașinii CIFA-2, mai avem nevoie de încă 4 celule operative și de 20 de celule pentru păstrarea anumitor constante după cum urmează:

062 —	α	
063 —	β	
064 —	α_1	
065 —	β_1	
066 —	K_a	inițial $K_a = 0$
067 —	K_b	inițial $K_b = 0$
068 —	1	
069 —	30	
070 —	32	
071 —	r	inițial $r = 0$
072 —	q	inițial $q = 0$
073 —	6	
074 —	21	
075 —	i	inițial $i = 0$
076 —	n	
077 —	c	$c = 01 - \mathrm{a} \quad 04 - b$
078 —	0	
079 —	c_{jk}	c_{jk} inițial este zero
080 —	n_1	$n_1 = 00 - 001 \quad 00 - 000$
081 —	j	inițial $j = 0$
082 —	m	
083 —	m_1	$m_1 = 01 - l + 1 \quad 11 - \beta_1$
084 —	k	inițial $k = 0$
085 —	p	

Programul împreună cu celulele operative şi constantele auxiliare folosite ocupă deci 85 de celule ale memoriei. Pentru ca această metodă să fie aplicabilă, este necesar ca, dacă M este numărul celulelor memoriei maşinii, să avem

$$85 + h_1 + h_2 + r + q \leqslant M.$$

Dacă înlocuim pe h_1 şi h_2 cu valorile lor (8') şi (9'), unde $H = 31$, avem

$$85 + \frac{n}{31}(m + p) + r + q \leqslant M.$$

Dacă K este numărul celulelor ocupate de un program obişnuit de înmulţire a două matrici şi de constantele auxiliare şi celulele operative folosite de acest program, atunci pentru ca metoda de mai sus să fie eficace trebuie să avem satisfăcută condiţia

$$85 + \frac{n}{H}(m + p) + r + q \leqslant K + n(p + m).$$

De aici scoatem

$$r + q \leqslant n(m + p) - \frac{n}{H}(p + m) + K - 85$$

sau

$$r + q \leqslant n(m + p)\left(1 - \frac{1}{H}\right) + K - 85.$$

Pentru a determina procentul elementelor $r + q$ diferite de zero, vom nota cu C diferenţa $K - 85$, care în general este o constantă negativă, şi vom obţine

$$r + q \leqslant n(m + p)\left(1 - \frac{1}{H}\right) + C,$$

sau, în procente, $r + q$ poate fi cel mult $\left[\left(1 - \frac{1}{H}\right) + \frac{C}{n(m + p)}\right] 100\%$ din numărul total al elementelor celor două matrici.

МЕТОД ПРОГРАММИРОВАНИЯ ПРОИЗВЕДЕНИЯ ДВУХ МАТРИЦ

РЕЗЮМЕ

В работе даестя метод программирования для электронной вычислительной машины произведения двух матриц (1) и (2); при этом в память машины вводятся только отличные от нуля элементы этих матриц. Для определения места, отличного от нуля элемента матрицы множителя, используется ряд индексов (6) и соответственно (7), элементы которых равны 0 или 1, смотря по тому, равен ли нулю или нет соответственный элемент матрицы. На основе этих рядов индексов строится логическая операторная схема алгоритма, осуществляющего произведение двух матриц, и дается и соответствующая программа с использованием кода электронной вычислительной машины CIFA-2.

В заключение находится условие, при котором этот метод эффективнее обычных классических методов.

MÉTHODE DE PROGRAMMATION DU PRODUIT DE DEUX MATRICES

RÉSUMÉ

Les auteurs indiquent une méthode de programmation pour les machines à calculer électroniques du produit de deux matrices (1) et (2), en n'inscrivant dans la mémoire de cette machine que les éléments différents de zéro de ces matrices. Pour déterminer la place d'un élément différent de zéro et appartenant à une matrice du produit, on utilise une série d'indices (6), respectivement (7), dont les éléments sont 0 ou 1, selon que l'élément correspondant de la matrice este 0 ou non. Se fondant sur ces séries d'indices, les auteurs ont construit le schéma logique opérationnel de l'algorithme qui réalise le produit de deux matrices et ils donnent aussi le programme correspondant, en utilisant le code de la machine électronique à calculer CIFA-2.

La conclusion contient la condition nécessaire pour que cette méthode soit plus efficace que les méthodes classiques habituelles.

BIBLIOGRAFIE

1. Leapunov A. A., *Despre schemele logice de programare* (traducere din limba rusă). Probleme de cibernetică, Bibl. Analelor Rom.-Sov., Ser. tehnică, *69—70* (1959).

Primit la XII. 1961.