ТЕХНИЧЕСКИ УНИВЕРСИТЕТ СОФИЯ

Синтез и анализ на алгоритми

КУРСОВА РАБОТА

Тема:

Матрици – алгоритъм за умножение. Алгоритъм на Щрасен (Strassen) за ускорено умножение на матрици

Изготвили : Стефан Стоилков - 123221012

Кристиян Иванов - 123221009

1. *Въведение*

Матрицата е правоъгълна таблица от числа (в общия случай правоъгълна таблица от произволни обекти).

Числата m и n определят размерността ($m \times n$) на матрицата: т.е. показват, че тя се състои от m реда и n стълба ($n \geq 1$, $m \geq 1$). Когато $m = n$ матрицата се нарича квадратна. Всеки елемент a_{ij} на матрицата се характеризира с двоен индекс: първото число в него определя номера на реда, а второто — номера на стълба, където е разположен (фигура 1).

$A_{m \times n}$ =

a_{11}	a_{12}	...	a_{1n}
a_{21}	a_{22}	...	a_{2n}
...			
a_{m1}	a_{m2}	...	a_{mn}

Фигура 1. Матрица $m \times n$

Матриците намират широко приложение в алгебрата, при решаване на системи от уравнения, в компютърната анимация, където матрици с размерност 2×2, 3×3, 4×4 се използват за транслация и ротация на обекти, за постигане на различни графични ефекти (като изглед в перспектива в тримерен терен) и др. Използването на матрици в тези и други случаи води до значително по-ефективни алгоритми и съответно до по-висока производителност.

В Си правоъгълна таблица от числа може да се представи чрез двумерен масив:

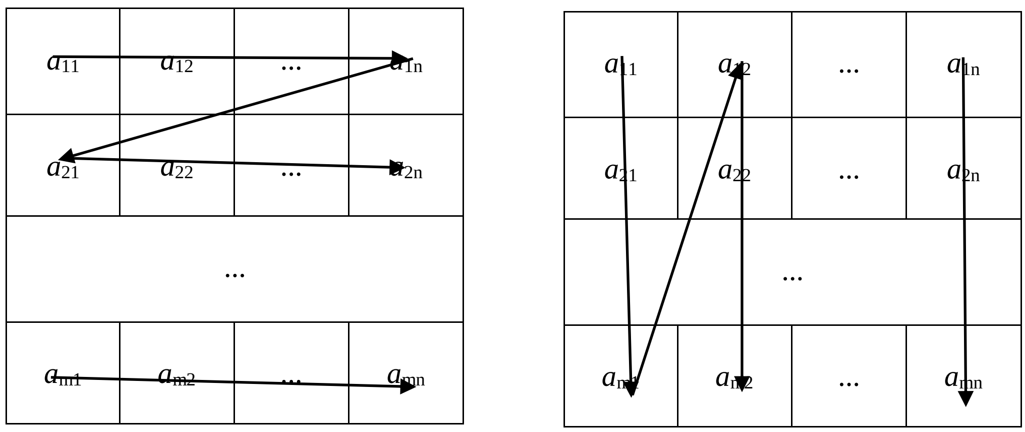
```
int A[m][n];
```

Възможен е и по-общият случай, когато елементи на матрицата са произволни обекти от някакъв предварително дефиниран тип struct data, например:

```
struct data {
  int a;
  int b;
  ...
} A[m][n] ;
```

Въвеждане и отпечатване на матрица

Елементите на матрица могат да бъдат прочетени/отпечатани по най-разнообразни начини. Двата най-прости са матрица по редове или по стълбове (фигура 2).



а) б)

Фигура 2. Обхождане на елементите на матрица: (а) по редове и (б) по стълбове

```
/* Въвеждане по редове */
for (i = 0; i < m; i++)
  for (j = 0; j < n; j++)
    scanf("%d", &A[i][j]);
/* Въвеждане по стълбове */
for (i = 0; i < n; i++)
  for (j = 0; j < m; j++)
    scanf("%d", &A[j][i]);
/* Отпечатване на матрица по редове */
for (i = 0; i < m; i++) {
  for (j = 0; j < n; j++)
    printf("%.3d", A[i][j]);
  printf("\n");
```

Литература :

1. Учебник : Програмиране =++Алгоритми; (Programming =++Algorithms;) Пето преработено издание, Преслав Наков, Панайот Добриков, 2015. (стр. 43 и 44)

Изготвил : Кристиян Иванов 123221009

2. ***Умножение на матрици***
$A = (a_{is}) \in M_{m\times k}(\mathbb{R})$ *и* $B = (b_{sj}) \in M_{k\times n}(\mathbb{R})$, взети в този ред, се нарича матрицата $AB = (c_{ij}) \in M_{m\times n}(\mathbb{R})$.

$$AB = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1k} \\ a_{21} & a_{22} & \dots & a_{2k} \\ a_{m1} & a_{m2} & \dots & a_{mk} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} & \dots & b_{1k} \\ b_{21} & b_{22} & \dots & b_{2k} \\ b_{m1} & b_{m2} & \dots & b_{mk} \end{pmatrix}$$

с елементи c_{ij}, получени по правилото

$$c_{ij} = \sum_{s=1}^{k} a_{is}b_{sj} = a_{i1}b_{1j} + a_{i2}b_{2j} + \dots + a_{ik}b_{kj}.$$

Действието, при което от матриците A и B се получава произведението AB, се нарича *умножение на матрици*.

Правилото за умножение на матрици е известно като правило "***ред по стълб***", т. е. всеки ред на първата матрица A се умножава последователно с всички стълбове на втората матрица B, за да се получат всички елементи на произведението AB. За да получим елементите от даден ред на AB, трябва да умножим реда със същия пореден номер на A последователно със всички стълбове на B съгласно обичайното правило за скаларно умножение на вектори (скаларно умножение относно ортонормирана координатна система). Елементът c_{ij} в i-тия ред и j-тия стълб на матрицата $C = AB$ се получава като скаларно произведение на i-тия ред на A и j-тия стълб на B. Затова броят на координатите (елементите) в i-тия ред на A и j-тия стълб на B трябва да бъде равен.

По-просто казано две матрици могат да бъдат умножени, само ако броят на стълбовете на първата матрица е равен на броя на редовете на втората.

Произведението им е матрица, на която броят на редовете е равен на броя на редовете на първия множител, а броят на стълбовете е равен на броя на стълбовете на втория множител.

Пример 1. Да се намери произведението на матриците

$$A = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 2 & 3 \\ 1 & -3 & 1 \\ 4 & 0 & 2 \end{pmatrix}, \quad B = \begin{pmatrix} 2 & 1 \\ -1 & 0 \\ 3 & 1 \end{pmatrix}.$$

Матриците A и B могат да бъдат умножени, тъй като първата от тях е от тип (4×3), а втората е от тип (3×2). Следователно произведението им $C = AB$ е (4×2)-матрица.

За намирането на елементите от първия ред на $C = (c_{ij})$ умножаваме последователно първия ред на A с всички стълбове на B. Аналогично постъпваме и за получаването на останалите три реда на C.

$c_{11} = 1.2 + (-1)(-1) + 0.3 = 3$, $c_{12} = 1.1 + (-1).0 + 0.1 = 1$,

$c_{21} = 0.2 + 2.(-1) + 3.3 = 7$, $c_{22} = 0.1 + 2.0 + 3.1 = 3$,

$c_{31} = 1.2 + (-3)(-1) + 1.3 = 8$, $c_{32} = 1.1 + (-3).0 + 1.1 = 2$,

$c_{41} = 4.2 + 0.(-1) + 2.3 = 14$, $c_{42} = 4.1 + 0.0 + 2.1 = 6$,

Така получихме

$$C = AB = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 2 & 3 \\ 1 & -3 & 1 \\ 4 & 0 & 2 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ -1 & 0 \\ 3 & 1 \end{pmatrix} = \begin{pmatrix} 3 & 1 \\ 7 & 3 \\ 8 & 2 \\ 14 & 6 \end{pmatrix}$$

Нека отбележим, че същите матрици не могат да бъдат умножени, ако разменим реда им, т.е. **не съществува** матричното произведение BA

$$BA = \begin{pmatrix} 2 & 1 \\ -1 & 0 \\ 3 & 1 \end{pmatrix} \begin{pmatrix} 1 & -1 & 0 \\ 0 & 2 & 3 \\ 1 & -3 & 1 \\ 4 & 0 & 2 \end{pmatrix} = \ ?\ ,$$

тъй като редовете на B са наредени 2-ки, а стълбовете на A са наредени 4-ки и нямаме правило за скаларно умножение, с което да умножаваме вектори с различен брой координати.

От този пример е ясно, че за произволни матрици A и B:

$$AB \neq BA.$$

Пример 2.

Нека $A = \begin{pmatrix} -1 & 1 \\ 2 & 0 \end{pmatrix}$ и $B = \begin{pmatrix} 1 & -1 \\ 3 & 2 \end{pmatrix}$.

$$AB = \begin{pmatrix} -1.1 + 1.3 & -1(-1) + 1.2 \\ 2.1 + 0.3 & 2(-1) + 0.2 \end{pmatrix} = \begin{pmatrix} 2 & 3 \\ 2 & -2 \end{pmatrix}$$

$$BA = \begin{pmatrix} 1 & -1 \\ 3 & 2 \end{pmatrix} \begin{pmatrix} -1 & 1 \\ 2 & 0 \end{pmatrix} = \begin{pmatrix} -3 & 1 \\ 1 & 3 \end{pmatrix}$$

Пример 3. Повдигане на квадратна матрица на степен.

k-та степен на квадратна матрица A от n-ти ред се дефинира по следния начин

$$A^k = \underbrace{A.A\ldots A}_{k},$$

където k е естествено число. Имаме $A^2 = AA$, $A^3 = A^2A$ и т.н.

Ето един пример с квадратна матрица от 2-ри ред:

$$A = \begin{pmatrix} 1 & 2 \\ -1 & 1 \end{pmatrix}.$$

$$A^2 = AA = \begin{pmatrix} 1 & 2 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ -1 & 1 \end{pmatrix} = \begin{pmatrix} -1 & 4 \\ -2 & -1 \end{pmatrix},$$

$$A^3 = A^2A = \begin{pmatrix} -1 & 4 \\ -2 & -1 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ -1 & 1 \end{pmatrix} = \begin{pmatrix} -5 & 2 \\ -1 & -5 \end{pmatrix}.$$

Литература

2. Интернет страница : http://web.uni-plovdiv.bg/marta/tema-7.pdf (стр. 2-10)

   Изготвил : Кристиян Иванов 123221009

Умножението на матрици притежава следните свойства:

1) AB ≠ BA – умножението на матрици **не е** комутативно. В общия случай, ако AB съществува, BA може изобщо да не дефинирано (Пример 1).
2) $(AB)C = A(BC)$ (асоциативен закон).
3) Съществува квадратна матрица E от n-ти ред, наречена единична матрица, такава че за всяка квадратна матрица A от n-ти ред е изпълнено $\boldsymbol{AE = EA = A}$. Матрицата $\boldsymbol{E}$ има вида

$$E = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 \end{pmatrix}$$

Единичната матрица е диагонална матрица с единици по главния диагонал и нули под и над него. Единичната квадратна матрица от втори ред има вида

$$E = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

За всяка квадратна матрица от втори ред А

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

е изпълнено

$$AE = \begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = A, \quad EA = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} a & b \\ c & d \end{pmatrix} = A\,.$$

4) $\lambda(AB) = (\lambda A)B = A(\lambda B)$ за всяко $\lambda \in \mathbb{R}$.

5) $A(B + C) = AB + AC$, $(A + B)C = AC + BC$ (ляв и десен дистрибутивен закон).

6) $\det(AB) = \det A \det B$ (A, B са квадратни матрици от един и същи ред).

7) $(AB)^{\mathrm{T}} = B^{\mathrm{T}}A^{\mathrm{T}}$.

Литература :

3. Интернет страница : http://web.uni-plovdiv.bg/marta/tema-7.pdf
   (стр. 10-13)
   Изготвил : Стефан Стоилков 123221012

3. *Алгоритъм на Щрасен за ускорено умножение на матрици*

Умножението на матрици е нова възможност за демонстрация на възможностите на стария римски принцип. Ще припомним, че класическият алгоритъм за умножение на матрици $A = (a_{ij})_{m\times n}$ и $B = (b_{ij})_{n\times r}$ изхожда директно от дефиницията и се дава от формулата:

$$c_{ij} = \sum_{k=1}^{n} a_{ik}\, b_{kj}$$

В случай на квадратни матрици с размер $n\times n$ времето за пресмятане на резултатната матрица C очевидно е $\Theta(n^3)$: имаме n^2 елемента и пресмятането на всеки от тях отнема време $\Theta(n)$. Тук, разбира се, предполагаме, че събирането и умножението имат константа времева сложност $\Theta(1)$, независеща от n.

Този алгоритъм е толкова прост и очевиден, че дълго време никой не е предполагал, че може да съществува друг и не е опитвал да го търси. В края на 60-те години обаче Щрасен предлага интересно подобрение на алгоритъма, намаляващо сложността до $\Theta(n^{\log 7})$. Откритието му е изненадващо за научния свят и води до отдаване на заслуженото на метода ***разделяй и владей***, който до този момент е недооценяван като ефективна програмистка техника. Огромният ефект на откритието на Щрасен се дължи на факта, че умножението на матрици, като фундаментална операция от алгебрата, е включена в много числени алгоритми. Всяко подобрение тук води до автоматично намаляване на изчислителната сложност на редица други алгоритми.

Да разгледаме матрици A и B с размер 2×2:

$$\begin{pmatrix} c_{11} & c_{12} \\ c_{22} & c_{22} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{22} & a_{22} \end{pmatrix} \times \begin{pmatrix} a_{11} & a_{12} \\ a_{22} & a_{22} \end{pmatrix}$$

Използвайки стандардния алгоритъм, получаваме:

$$C_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j},\ i = 1,2 \text{ и } j = 1,2$$

Вижда се, че са ни необходими 8 умножения и 4 събирания. Щрасен забелязва, че броят на умноженията би могъл да бъде намален до 7 по следния начин (с P_i означаваме помощните променливи):

$$P_1 = (a_{11} + a_{22})(b_{11} + b_{22})$$

$$P_2 = (a_{21} + a_{22})b_{22}$$

$$P_3 = a_{11}(b_{12} - b_{22})$$

$$P_4 = a_{22}(b_{21} - b_{11})$$

$$P_5 = (a_{11} + a_{12})b_{22}$$

$$P_6 = (a_{21} - a_{11})(b_{11} + b_{12})$$

$$P_7 = (a_{12} - a_{22})(b_{21} + b_{22})$$

Сега можем да изразим $c_{i,j}$ през P_k

$$c_{11} = P_1 + P_4 - P_5 + P_7$$

$$c_{12} = P_3 + P_5$$

$$c_{21} = P_2 + P_4$$

$$c_{22} = P_1 + P_3 - P_2 + P_6$$

Ако заместим елементите a_{ij}, b_{ij} и c_{ij} с матрици с размер $n \times n$, получаваме алгоритъм, който умее да умножава матрици с размер $2n$, използвайки 7, вместо 8 умножения на матрици с размер $n \times n$. Умножението на две матрици обаче определено е по-тежка операция от събирането. Наистина, ако при използване на класическите алгоритми от дефинициите за събирането са ни необходими n 2 елементарни събирания, то за умножението са ни нужни n *3* елементарни събирания плюс n 3 елементарни умножения. Т. е. спестяването на едно умножение има сериозен ефект.

В случай да имаме квадратна матрица, при която n не е на втора степен, матриците допълваме с нулеви редове и колоните до най-малкото n, което е на втора степен. Тази техника е известна като *статично допълване*. Това може да доведе най-много до удвояване на размера на матрицата. Извършването на още една итерация, съгласно горната рекурентна зависимост, може да увеличи броя на умноженията най-много 7 пъти.

Друго възможно решение (когато n не е степен на 2) е да отлагаме допълването на матриците с нули, докато това е възможно. Т.е. ако първоначално n е четно, можем да извършим една итерация на алгоритъма. Ако и $n/2$ е четно, можем да извършим още една, а ако е нечетно — ще трябва по необходимост да направим допълване. Извършвайки допълване едва сега обаче, евентуално си спестяваме вмъкването на много допълнителни нули. Тази техника е известна като *динамично допълване*. Въпреки че води до сериозно подобрение, динамичното допълване не е оптималният метод.

Обикновено, когато се прилага алгоритъмът на Щрасен за достатъчно малки подматрици, се използват стандартни методи, които са по-ефективни в този случай. Преходната точка, в която алгоритъмът на Щрасен става по-ефективен, зависи от изпълнението и хардуера. Преди беше предсказано, че алгоритъмът на Щрасен е по-бърз с матрици с размери между 32 и 128 за оптимизирани реализации. Съвсем наскоро обаче се оказа, че точката на преход се увеличава. Изследване, проведено през 2010 г. , показа, че при настоящите архитектури дори една стъпка от алгоритъма на Щрасен не носи предимства в сравнение с оптимизираните традиционни умножения, докато ранговата матрица не надхвърли 1000, и дори тогава, когато рангът е до няколко хиляди, предимството е незначително (в най-добрия случай 10%).

Асимптотична сложност

Стандартното матрично умножение има приблизително $2N^3$ (за $N = 2^n$) аритметични операции; асимптотичната сложност е $\Theta(N^3)$.

Броят на аритметичните операции в алгоритъма на Щрасен може да се изчисли по следния начин: нека $f(n)$ е броят на операциите за матрица $2^n \times 2^n$. След това чрез рекурсивно прилагане на алгоритъма виждаме, че $f(n) = 7f(n-1) + l4^n$ за някаква константа "l", която зависи от броя на добавките в конкретното изпълнение на алгоритъма. Така че $f(n) = (7 + o(1))n$, т.е. асимптотичната сложност за умножението на матрици с размерност $N = 2^n$ с помощта на алгоритъма на Щрасен е:

$$O([7 + o(1)]^n) = O(N^{\log_2 7 + o(1)}) \approx O(N^{2.8074}).$$

Но $log\ 7 \approx 2{,}81 < 3$, т.е. алгоритъмът на Щрасен има по-добра времева сложност от класическия.

Чрез намаляване на броя на аритметичните операции, числената стабилност все още е намалена, също така алгоритъмът изисква значително повече памет в сравнение с наивния алгоритъм. Размерите на двете начални матрици трябва да бъдат разширени до първата следваща степен на две, което може да увеличи броя на елементите до четири пъти и да доведе до седем спомагателни матрици, така че всяка да има една четвърт от елементите от разширената част.

Сравнение между два алгоритма

Параметри	*Класически алгоритъм*	*Алгоритъм на Щрасен*
Времева сложност	$\Theta(N^3)$	$\Theta(N^{2.8074})$
Подход, достъп, път	*Итеративен (повтарящ се)*	*Разделяй и владей*
Прилагане	*Квадратни и неквадратни матрици*	*Подходящ само за квадратни матрици*

Литература :

4. Учебник : Програмиране =++Алгоритми; (Programming =++Algorithms;) Пето преработено издание, Преслав Наков, Панайот Добриков, 2015. (стр. 440-443)

5. Интернет страница:
https://sr.m.wikipedia.org/sr-el/%C5%A0trasenov_algoritam

Изготвил : Стефан Стоилков 123221012

4.Заключение

Редица други опити за намаляване на броя на умноженията и събиранията в класическата схема на Щрасен при $n = 2$, са правени от известни учени (Хъс, Ледерман, Виноград...), но всичките са безуспешни. През 1971 г. Хопкрофт и Кер доказват, че това е невъзможно, ако не се ползва комутативността на умножението (умножението на матрици не е комутативно.). По-късно бива открит начин за умножение на матрици 3×3 с най-много 21 умножения, което дава алгоритъм със сложност $\Theta(n^{log3\ 21})$, т. е. $\Theta(n^{2,771244})$. Следват редица други подобрения при все по-големи размери на матриците и все по-малък шанс за реално практическо приложение.

Литература:

6. Програмиране =++Алгоритми; (Programming =++Algorithms;) Пето преработено издание, Преслав Наков, Панайот Добриков, 2015.
(стр. 443)

Изготвили : Стефан Стоилков 123221012
Кристиян Иванов 123221009