КОМПЬЮТЕРНАЯ МАТЕМАТИКА

Оптимизация вычислений

УДК 519.6

А.М. ТЕРЕЩЕНКО

АНАЛІЗ СКЛАДНОСТІ БАГАТОРОЗРЯДНОГО МНОЖЕННЯ НА ОСНОВІ МЕТОДУ КАРАЦУБИ – ОФМАНА В ПАРАЛЕЛЬНІЙ МОДЕЛІ ОБЧИСЛЕНЬ

Аналізується складність за кількістю операцій одного паралельного процесора при реалізації операції багаторозрядного множення чисел у паралельній моделі обчислень. Аналізується складність при необмеженій кількості доступних процесорів, та вивчається, як змінюється кількість операцій, коли кількість доступних процесорів обмежується. Дано аналіз складності багаторозрядного множення на основі методу Карацуби – Офмана у паралельній моделі обчислень. Наведена таблиця прискорення часу виконання операції множення на основі методу Карацуби – Офмана.

Вступ. На даний час мікропроцесорна техніка розвивається надзвичайно швидко [1 – 4]. Кластери з кількістю процесорів більше ніж 4000 можливі на персональних комп'ютерах на основі графічних прискорювачів. Поряд з подальшим розвитком алгоритмів у послідовній моделі обчислень, необхідних для енергоекономних нешвидких пристроїв, таких як смарткарти, є потреба в розробці нових методів ефективних у паралельній моделі обчислень. При розробці та вдосконаленні алгоритмів, виконуваних у паралельній моделі обчислень, дуже важливо мати методи оцінки складності алгоритмів для порівняння з існуючими та знаходження ефективних діапазонів їх використання. Наступні пункти мають значний вплив (змінюють час виконання в рази) на загальну складність:

1) кількість доступних паралельних процесорів;
2) кількість операцій (ітерацій), виконуваних кожним з паралельних процесорів;
3) тип операцій (цілі або комплексні числа, числа з плаваючою комою і т. д.);
4) врахування знака переносу;
5) кількість тісно пов'язаних кроків;
6) обсяг використовуваної пам'яті.

Даний перелік не є вичерпним, у цій роботі основна увага приділяється пп. 1), 2).

Розглянемо множення 4-розрядних цілих чисел стандартним методом «у стовпчик» у паралельній моделі обчислень. Нехай X_4,

Y_4 – такі 4-розрядні числа: $X_4 = \sum_{i=0}^{3} x_i 2^{\omega i}$, $Y_4 = \sum_{i=0}^{3} y_i 2^{\omega i}$ та ω – довжина слова у бітах, тоді $R_8 = \sum_{i=0}^{7} r_i 2^{\omega i}$ буде результатом множення. Нехай 2-розрядне число $h_{i,j} l_{i,j}$ позначає тимчасовий результат множення $x_i \cdot y_j = h_{i,j} \cdot 2^{\mu} + l_{i,j}$, де $l_{i,j} = L(x_i \cdot y_j)$, $h_{i,j} = H(x_i \cdot y_j)$ (l – Low, h – $High$). Старші $h_{i,j}$ та молодші $l_{i,j}$ розряди тимчасових результатів множення можна розмістити у матриці $T_{7\times 8}$ розмірністю 7×8, де сума по кожній колонці дає розряд r_i результату множення. Нехай $t_{i,j}(h_{a,b})$, позначає, що значення $h_{a,b}$ знаходиться в комірці $t_{i,j}$. Суму в кожній колонці $T_{7\times 8}$ можна обчислити одночасно процесорами P_i, $i = \overline{0,7}$ (рис. 1).

P_7	P_6	P_5	P_4	P_3	P_2	P_1	P_0
			$t_{4,6}(h_{3,0})$	$t_{3,6}(h_{2,0})$			
			$t_{4,5}(h_{2,1})$	$t_{3,5}(h_{1,1})$			
		$t_{5,4}(h_{3,1})$	$t_{4,4}(h_{1,2})$	$t_{3,4}(h_{0,2})$	$t_{2,4}(h_{1,0})$		
		$t_{5,3}(h_{2,2})$	$t_{4,3}(h_{0,3})$	$t_{3,3}(l_{3,0})$	$t_{2,3}(h_{0,1})$		
	$t_{6,2}(h_{3,2})$	$t_{5,2}(h_{1,3})$	$t_{4,2}(l_{3,1})$	$t_{3,2}(l_{2,1})$	$t_{2,2}(l_{2,0})$	$t_{1,2}(h_{0,0})$	
	$t_{6,1}(h_{2,3})$	$t_{5,1}(l_{3,2})$	$t_{4,1}(l_{2,2})$	$t_{3,1}(l_{1,2})$	$t_{2,1}(l_{1,1})$	$t_{1,1}(l_{1,0})$	
$t_{7,0}(h_{3,3})$	$t_{6,0}(l_{3,3})$	$t_{5,0}(l_{2,3})$	$t_{4,0}(l_{1,3})$	$t_{3,0}(l_{0,3})$	$t_{2,0}(l_{0,2})$	$t_{1,0}(l_{0,1})$	$t_{0,0}(l_{0,0})$
r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0

РИС. 1. Обчислення тимчасових результатів

Знаходження суми кожної колонки може бути виконано за три ітерації при задіянні 12 паралельних процесорів, як показано на рис. 2.

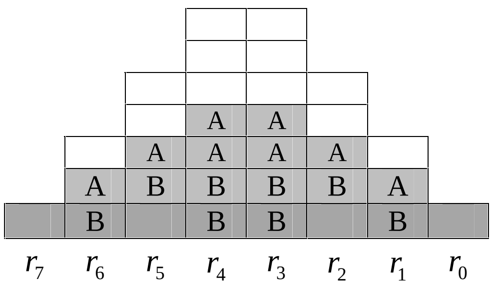


РИС. 2. Ітераційне знаходження суми по кожній колонці при множенні двох 4-разрядних чисел

На першій ітерації значення комірок, позначених білим кольором, додаються до комірок світло сірого кольору. На другій ітерації необхідно задіяти вісім процесорів для одночасного додавання значень комірок, позначених літерами «А», до комірок позначених літерами «В». На останній третій ітерації достатньо чотирьох процесорів, щоб додати значення комірок світло сірого кольору, позначених літерами «В», до комірок темно сірого кольору.

Алгоритм 1. Множення N-розрядних чисел $R_{2N} = X_N \cdot Y_N$ методом «у стовпчик» у паралельній моделі обчислень (без врахування знаків переносу).

Вхід: X_N, Y_N − N-розрядні числа, P − кількість процесорів.

Вихід: R_{2N} − результат множення двох N-розрядних чисел.

1. $h_{i,j} \leftarrow H(x_i \cdot y_j)$, $l_{i,j} \leftarrow L(x_i \cdot y_j)$, $i, j = \overline{0, N-1}$. // Поелементне множення.
2. $r_0 \leftarrow l_{0,0}$, $r_{2N-1} \leftarrow h_{N-1,N-1}$. // Паралельне виконання двох потоків.
3. Для i від 1 до $N-1$. // Індексація.
4. Для j від 0 до i. // Паралельне виконання $i+1$ потоків.
5. $t_{i,j} \leftarrow l_{j,i-j}$; $t_{2N-1-i,i+j} \leftarrow h_{N-1-i+j,N-1-j}$.
6. Кінець по j.
7. Для j від 0 до $i-1$. // Паралельне виконання i потоків.
8. $t_{i,i+1+j} \leftarrow h_{j,i-1-j}$; $t_{2N-1-i,j} \leftarrow l_{N-i+j,N-1-j}$.
9. Кінець по j.
10. Кінець по i.
11. $n \leftarrow \lceil \log_2(2N-1) \rceil$.
12. Для j від 1 до n. // Ітерацій
13. Для i від 1 до $N-1$.
14. $K \leftarrow \left\lfloor \frac{D}{P} \right\rfloor$, $D \leftarrow \left\lceil \frac{2i-1}{2^{j-1}} \right\rceil - M$, $M \leftarrow \left\lceil \frac{2i-1}{2^j} \right\rceil$, $I \leftarrow 2N-1-i$.
15. Для k від 0 до K.
16. $R \leftarrow \begin{cases} k = K, & D-kP-1 \\ k < K, & P-1 \end{cases}$.
17. Для m від 0 до R. // Паралельне виконання $R+1$ потоків.
18. $t_{i,kP+m} \leftarrow t_{i,kP+m} + t_{i,M+kP+m}$, $t_{I,kP+m} \leftarrow t_{I,kP+m} + t_{I,M+kP+m}$.
19. Кінець по m.
20. Кінець по k.
21. Кінець по i.
22. Кінець по j.

Лема 1. При виконанні Алгоритму 1 при задіянні P паралельних процесорів, кожний з процесорів виконає операцій з цілими числами не більше ніж

$$O^{*,+}(N,P)=O^{*}(N,P)+O^{+}(N,P), \qquad (1)$$

де $O^{*}(N,P)=\left\lceil \frac{N^2}{P} \right\rceil$, $O^{+}(N,P)=\sum_{j=0}^{\lfloor \log_2(2N-1)\rfloor}\left\lceil 2\sum_{i=0}^{N-1}\left(\left\lceil \frac{2i+1}{2^j}\right\rceil-\left\lceil \frac{2i+1}{2^{j+1}}\right\rceil\right)\Big/P\right\rceil$.

Доведення для додавань базуються на тому, що кількість значень у колонці i (див. рис. 2) для опрацювання зменшується на кожній ітерації $j=0,1,\ldots,$ майже у два рази $\left\lceil \frac{2i+1}{2^j}\right\rceil-\left\lceil \frac{2i+1}{2^{j+1}}\right\rceil$. Кількість процесорів, задіяних на ітерації j у припущенні, що кількість процесорів достатня, буде наступною:

$$P(N)_j=2\sum_{i=0}^{N-1}\left(\left\lceil \frac{2i+1}{2^j}\right\rceil-\left\lceil \frac{2i+1}{2^{j+1}}\right\rceil\right). \qquad (2)$$

Для випадку $N=4$ (рис. 2)

$$\sum_{j=0}^{\lfloor \log_2 7\rfloor}\left\lceil 2\sum_{i=0}^{3}\left(\left\lceil \frac{2i+1}{2^j}\right\rceil-\left\lceil \frac{2i+1}{2^{j+1}}\right\rceil\right)\Big/12\right\rceil=3,\ O^{+}(N=4,P=12)=3.$$

Тобто кожний з 12 паралельних процесорів виконає не більше трьох операцій додавання згідно (2).

Аналізуючи табл. 1 можна стверджувати наступне.

Лема 2. При виконанні Алгоритму 1 для будь-якого N при достатній кількості паралельних процесорів справедливе наступне співвідношення:

$$\frac{P(N)_{j+1}}{P(N)_j}\geq 2,\ 1<j\leq\lceil \log_2 N\rceil,$$

де $P(N)_j$ – кількість задіяних процесорів на ітерації j для N з табл. 1.

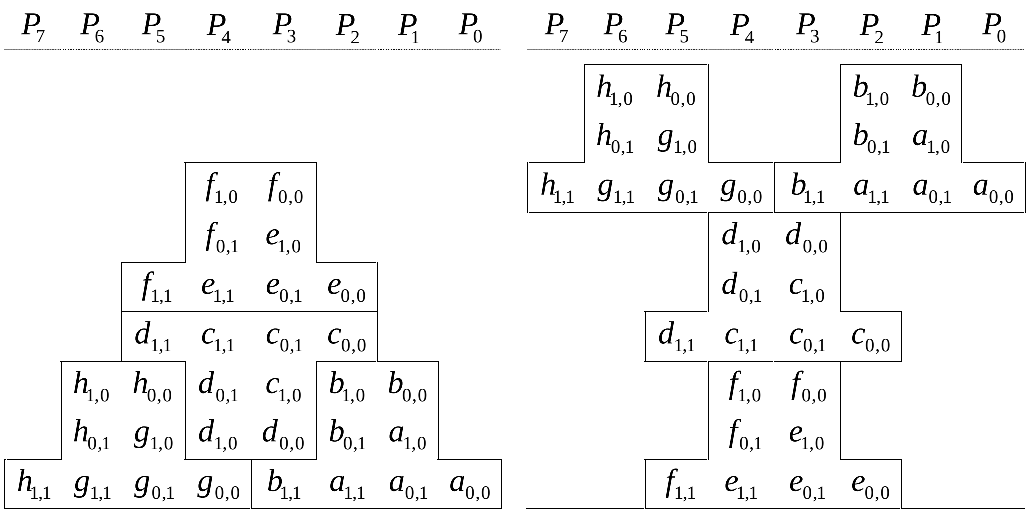
ТАБЛИЦЯ 1. Кількість задіяних процесорів (2) на ітерації j при множенні двох чисел розрядністю $N=(\overline{1,17}),24,32,48,64$ в паралельній моделі обчислень

	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$	$j=6$	$j=7$	$j=8$
$N=1$	0							
$N=2$	2	2	0					
$N=3$	6	4	2					
$N=4$	12	8	4	0				
$N=5$	20	12	6	2	0			
$N=6$	30	18	8	4	0			
$N=7$	42	24	12	6	0			

Закінчення табл. 1

	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$	$j=6$	$j=7$	$j=8$
$N=8$	56	32	16	8	0			
$N=9$	72	40	20	10	2	0		
$N=10$	90	50	24	12	4	0		
$N=11$	110	60	30	14	6	0		
$N=12$	132	72	36	16	8	0		
$N=13$	156	84	42	20	10	0		
$N=14$	182	98	48	24	12	0		
$N=15$	210	112	56	28	14	0		
$N=16$	240	128	64	32	16	0		
$N=17$	272	144	72	36	18	2	0	
$N=24$	552	288	144	72	32	16	0	
$N=32$	992	512	256	128	64	32	0	
$N=48$	2256	1152	576	288	144	64	32	0
$N=64$	4032	2048	1024	512	256	128	64	0

Якщо $L(x_i \cdot y_j)$, $H(x_i \cdot y_j)$ − молодший та старший (***L**ow*, ***H**igh*) розряди результату множення однорозрядних чисел x_i та y_j, то використаємо позначення: $a_{i,j} = L(x_i \cdot y_j)$, $b_{i,j} = H(x_i \cdot y_j)$, $c_{i,j} = L(x_{i+2} \cdot y_j)$, $d_{i,j} = H(x_{i+2} \cdot y_j)$, $e_{i,j} = L(x_i \cdot y_{j+2})$, $f_{i,j} = H(x_i \cdot y_{j+2})$, $g_{i,j} = L(x_{i+2} \cdot y_{j+2})$, $h_{i,j} = H(x_{i+2} \cdot y_{j+2})$, $i, j = \overline{0,1}$, для представлення $T_{7\times 8}$ (див. рис. 1) у вигляді секцій (див. рис. 3, 4).



r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0

РИС. 3. Множення стандартним методом

r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0

РИС. 4. Множення зі секціонуванням

На рис. 4 показано, що множення двох 4-розрядних чисел можна представити у вигляді чотирьох однакових за формою секцій. На рис. 5 показано, що при задіянні 16 процесорів всі секції можуть бути обчислені одночасно, з отриманнямпроміжкових результатів, для додавання яких необхідно задіяти 4 процесори, як показано на рис. 6.

P_{15}	P_{14}	P_{13}	P_{12}	P_{11}	P_{10}	P_9	P_8	P_7	P_6	P_5	P_4	P_3	P_2	P_1	P_0
	$h_{1,0}$	$h_{0,0}$			$f_{1,0}$	$f_{0,0}$			$d_{1,0}$	$d_{0,0}$			$b_{1,0}$	$b_{0,0}$	
	$h_{0,1}$	$g_{1,0}$			$f_{0,1}$	$e_{1,0}$			$d_{0,1}$	$c_{1,0}$			$b_{0,1}$	$a_{1,0}$	
$h_{1,1}$	$g_{1,1}$	$g_{0,1}$	$g_{0,0}$	$f_{1,1}$	$e_{1,1}$	$e_{0,1}$	$e_{0,0}$	$d_{1,1}$	$c_{1,1}$	$c_{0,1}$	$c_{0,0}$	$b_{1,1}$	$a_{1,1}$	$a_{0,1}$	$a_{0,0}$
r_7	r_6	r_5'''	r_4'''	r_5''	r_4''	r_3'''	r_2'''	r_5'	r_4'	r_3''	r_2''	r_3'	r_2'	r_1	r_0

РИС. 5. Одночасне обчислення чотирьох секцій

P_3	P_2	P_1	P_0
r_5'	r_4'	r_3'	r_2'
r_5''	r_4''	r_3''	r_2''
r_5'''	r_4'''	r_3'''	r_2'''
r_5	r_4	r_3	r_2

РИС. 6. Додавання проміжкових результатів

Алгоритм 2. Множення двох MN-розрядних чисел методом розбиття на числа меншої розрядності.

Вхід: X_{MN}, Y_{MN} – $MN = 2^n$-розрядні числа $(1 \le M \le N)$, $n > 0$, n – ціле число.

Вихід: R_{2MN} – результат множення двох MN-розрядних чисел.

1. $r_k \leftarrow 0$, $k = \overline{0, 2MN-1}$.
2. Для i від 0 до $\log_2 M$
3. $\quad V_N \leftarrow \{x_{iN+k}, k = \overline{0, N-1}\}$.
4. $\quad$ Для j від 0 до $\log_2 M$
5. $\quad\quad W_N \leftarrow \{y_{jN+k}, k = \overline{0, N-1}\}$.
6. $\quad\quad Z_{2N} \leftarrow$ Алгоритм_1(V_N, W_N).

7. $r_{(i\cdot+j)\cdot N+k} \leftarrow r_{(i\cdot+\cdot j)\cdot N+k} + z_k$, $k = \overline{0,2N-1}$.

8. Кінець по j.

9. Кінець по i.

Лема 3. При виконанні Алгоритму 2 при задіянні P паралельних процесорів кожний з процесорів виконає не більше ніж

$$O^{*,+}(MN,P) = O^{*}(MN,P) + O^{+}(MN,P), \quad O^{*}(MN,P) = \left\lceil \frac{M^2 \cdot N^2}{P} \right\rceil, \tag{3}$$

$$O^{+}(MN,P) = \sum_{i=0}^{\lfloor \log_2(2M-1) \rfloor} \left\lceil 2N \sum_{j=0}^{M-1} \left(\left\lceil \frac{2j+1}{2^i} \right\rceil - \left\lceil \frac{2j+1}{2^{i+1}} \right\rceil \right) \Big/ P \right\rceil +$$
$$+ \sum_{i=0}^{\lfloor \log_2(2N-1) \rfloor} \left\lceil 2M^2 \sum_{j=0}^{N-1} \left(\left\lceil \frac{2j+1}{2^i} \right\rceil - \left\lceil \frac{2j+1}{2^{i+1}} \right\rceil \right) \Big/ P \right\rceil \tag{4}$$

операцій з цілими числами.

Доведення. Розіб’ємо MN-розрядні числа на M чисел розрядністю N. Згідно Леми 1 при множенні двох N-розрядних чисел кожний з задіяних процесорів виконає не більше (2) операцій додавання над цілими числами. Таких множень необхідно виконати M^2, що відповідає другому доданку в (4).

При розбитті MN-розрядних чисел на M чисел розрядністю N виконуються операції множення N-розрядних чисел. Для отримання остаточного результату необхідно додати не більше M проміжкових результатів. З урахуванням того, що молодша та старша частини проміжкових результатів мають розрядність N, кількість додавань збільшується в N разів, що відповідає першому доданку в (4). Лема доведена.

Зауваження 1. Лема враховує, що кількість процесорів, задіяних на кожній ітерації, зменшується (див. табл. 1).

Програма *paraNP2ALL* для розрахунку кількості операцій над цілими числами $O^{*,+}(MN,P)$, розроблена на мові програмування APL, показала, що Алгоритм 2 дає виграш в одну (або декілька) операцію у порівнянні з Алгоритмом 1 для деяких MN та P, що говорить про недоцільність реалізації Алгоритму 2, враховуючи його більшу складність.

Метод Карацуби – Офмана. Одна з процедур, що дозволяє зменшити загальну кількість однослівних множень, запропонована Карацубою та Офманом у роботі [4]. Ідея її полягає в наступному.

$$U_{2N} \cdot V_{2N} = (H(U_{2N}) \cdot \Omega + L(U_{2N})) \cdot (H(V_{2N}) \cdot \Omega + L(V_{2N})) =$$
$$= H(U_{2N}) \cdot H(V_{2N}) \cdot \Omega^2 +$$
$$+ \begin{pmatrix} ((H(U_{2N}) + L(U_{2N})) \cdot (H(V_{2N}) + L(V_{2N})) - \\ -H(U_{2N}) \cdot H(V_{2N}) - L(U_{2N}) \cdot L(V_{2N}) \end{pmatrix} \cdot \Omega + L(U_{2N}) \cdot L(V_{2N}),$$

де $\Omega = 2^{\omega N}$, ω – довжина слова у бітах.

Такі співвідношення дають можливість звести задачу множення двох $2N$-розрядних чисел до трьох множень N-розрядних чисел. Даний метод можна використовувати рекурсивно, що значно зменшує кількість однослівних операцій множення та задіяних процесорів у паралельній моделі обчислень.

Алгоритм 3. Множення двох MN-розрядних чисел на основі методу Карацуби-Офмана у паралельній моделі обчислення.

Вхід: X_{MN}, Y_{MN} − $MN = 2^n$-розрядні числа $(1 \le M \le N)$, $0 < n$, n – ціле число.

Вихід: R_{2MN} − результат множення двох MN-розрядних чисел.

1. $D \leftarrow T,\ t \leftarrow T,\ T \leftarrow MN.$
2. $v_k \leftarrow 0,\ w_k \leftarrow 0,\ r_{2k} \leftarrow 0,\ r_{2k+1} \leftarrow 0,\ k = \overline{0, K-1},\ K \leftarrow 2T\left(3/2\right)^{\log_2 M}.$
3. $v_k \leftarrow x_k, w_k \leftarrow y_k, k = \overline{0, T-1}.$ // Завантаження.
4. Для i від 1 до $\log_2 M$ // Знаходження тимчасових сум.
5. $\quad D \leftarrow D/2,\ s \leftarrow 0.$
6. $\quad$ Для j від 1 до 3^{i-1}.
7. $\quad\quad v_{t+k} \leftarrow v_{s+k} + v_{s+D+k},\ k = \overline{0, D-1}.$
8. $\quad\quad w_{t+k} \leftarrow w_{s+k} + w_{s+D+k},\ k = \overline{0, D-1}.$
9. $\quad\quad s \leftarrow s + D,\ t \leftarrow t + D.$
10. $\quad$ Кінець по j.
11. Кінець по i.
12. Для i від 0 до $3^{\log_2 M} - 1.$ // Перемноження
13. $\quad S_N \leftarrow \{v_{iD+k}, k = \overline{0, D-1}\},\ T_N \leftarrow \{w_{iD+k}, k = \overline{0, D-1}\}.$
14. $\quad Z_{2D} \leftarrow$ Алгоритм_1$(S_D \cdot T_D)$.
15. $\quad r_{i2D+k} \leftarrow z_{i2D+k},\ k = \overline{0, 2D-1}.$
16. Кінець по i.
17. Для i від 1 до $\log_2 M$. // Корегування методом Карацуби
18. $\quad s \leftarrow 0,\ t \leftarrow K \cdot \left(2/3\right)^i,\ D \leftarrow N \cdot 2^{i-1}.$
19. $\quad$ Для j від 1 до 3^{i-1}
20. $\quad\quad S_{2D} \leftarrow \{r_{t+k}, k = \overline{0, 2D-1}\};\ L_{2D} \leftarrow \{r_{s+k}, k = \overline{0, 2D-1}\}.$
21. $\quad\quad H_{2D} \leftarrow \{r_{s+2D+k}, k = \overline{0, 2D-1}\};\ Z_{2D} \leftarrow S_{2D} - H_{2D} - L_{2D}.$
22. $\quad\quad r_{s+D+k} \leftarrow z_k,\ k = \overline{0, 2D-1}.$
23. $\quad\quad s \leftarrow s + 4D,\ t \leftarrow t + 2D.$
24. $\quad$ Кінець по j.
25. Кінець по i.

Лема 4. При виконанні Алгоритму 3 при задіянні P паралельних процесорів кожний з процесорів виконає не більше операцій з цілими числами ніж

$$O^{*,+}(MN,P) = O^{*}(MN,P) + O^{+}(MN,P),\ \ O^{*}(MN,P) = \left\lceil \frac{N^2 \cdot 3^{\log_2 M}}{P} \right\rceil, \qquad (5)$$

$$O^{+}(MN,P)=\sum_{i=0}^{\log_2 M}\left\lceil \frac{(3/2)^i\cdot M\cdot N}{P}\right\rceil+\sum_{i=0}^{\lfloor \log_2(2N-1)\rfloor}\left\lceil 3^{\log_2 M}\sum_{j=0}^{N-1}\left(\left\lceil\frac{2j+1}{2^i}\right\rceil-\left\lceil\frac{2j+1}{2^{i+1}}\right\rceil\right)\Big/P\right\rceil+$$

$$+\sum_{i=0}^{\log_2 M}\left\lceil\sum_{j=1}^{2}\left(\frac{2j\cdot M\cdot N\cdot(3/2)^i}{P}\right)\right\rceil. \quad (6)$$

Алгоритм 3 дає прискорення у швидкодії майже на 70 % у порівнянні з Алгоритмом 1, що наведено у табл. 2.

ТАБЛИЦЯ 2. Кількість операцій над цілими числами при множенні двох чисел розрядністю $N = 4, 8, 16, 32, 48, 64, 96, 112, 128$ при задіянні процесорів $P = 4, 8, 16, 32, 64, 128$ та розбитті на $M = 1, 2, 4, 8, 16, 32$ багаторозрядних чисел на основі методу Карацуби – Офмана у паралельній моделі обчислень

P	NM	$M=1$	$M=2$	$M=4$	$M=8$	$M=16$	$M=32$	$MIN(P(NM))/P(NM)$
4	8	44	35	32				72,73%
4	16	184	142	115	103			55,98%
8	16	92	59	54				58,70%
8	32	376	286	223	184	166		44,15%
8	48	852	645	498	399	341		40,02%
8	64	1520	1148	878	689	569	510	33,55%
16	16	46	37	32	31			67,39%
16	32	188	143	113	95	87		46,28%
16	48	426	324	253	203	176		41,31%
16	64	760	574	439	346	287	259	34,08%
32	16	24	21	20	19			79,17%
32	32	94	73	59	52	49		52,13%
32	64	380	287	221	176	148	135	35,53%
64	32	48	39	33	30	28		58,33%
64	64	190	145	113	92	80	73	38,42%
64	96	430	314	253				58,83%
128	32	25	22	21	19	19		76,00%
128	64	96	75	60	50	44	40	41,67%
128	96	217	165	131	102	84	73	33,64%
128	128	382	289	221	173	141	119	31,15%

Висновок. У даній роботі вивчається складність за кількістю однорозрядних операцій додавання та множення над цілими числами одним з паралельних процесорів при виконанні багаторозрядного множення стандартним методом «у стовпчик» в паралельній моделі обчислень. В роботі розглянуто два алгоритми «Розбиття на числа меншої розрядності» та «Розбиття на основі методу Карацуби – Офмана» в паралельній моделі обчислень, в яких досліджена кількість операцій за умови, коли кількість доступних процесорів необмежена та коли

кількість процесорів обмежена і кратна N. Показано, що реалізація алгоритму «Розбиття на числа меншої розрядності» не є ефективним у порівнянні зі стандартним методом. Алгоритм «Розбиття на основі методу Карацуби-Офмана» дає прискорення до 70 %. В роботі наведено 4 леми, надані таблиці. Розроблена програма *paraNP2ALL* та *paraNP3ALL* на мові програмування *APL*, які підтверджують теоретичні результати отримані для оцінки складності при використанні одного процесора.

А.Н. Терещенко

АНАЛИЗ СЛОЖНОСТИ МНОГОРАЗРЯДНОГО УМНОЖЕНИЯ НА ОСНОВЕ МЕТОДА КАРАЦУБЫ-ОФМАНА ПРИ РАЗНОМ КОЛИЧЕСТВЕ ПАРАЛЛЕЛЬНЫХ ПРОЦЕССОРОВ

Анализируется сложность по количеству операций одного процессора в параллельной модели вычислений. Дано анализ сложности многоразрядного умножения на основе метода Карацубы – Офмана в параллельной модели вычислений. Приведена таблица ускорения времени на основе данного метода.

A.N. Tereshchenko

COMPLEXITY ANALYSIS OF MULTI-DIGIT MULTIPLICATION OPERATION BASED ON KARATSUBA METHOD USING DIFFERENT NUMBER OF PARALLEL PROCESSORS

The complexity of number operation of a processor is analyzed in parallel computational model. Complexity analysis of multi-digit multiplication based on Karatsuba method is given. The table of performance increase of multiplication based on Karatsuba method is provided.

1. *Задірака В., Олексюк О.* Комп'ютерна арифметика багаторозрядних чисел. – К.: Наук. думка, 2003. – 263 с.
2. *Березовский А.И., Задирака В.К., Шевчук Л.Б.* О тестировании быстродействия алгоритмов и программ выполнения основных операций для асимметричной криптографии // Кибернетика и системный анализ. – 1999. – № 5. – С. 61 – 68.
3. *Вельшенбах М.* Криптография на Си С++ в действии: Учебное пособие. – М.: Изд-во Триумф, 2004. – 464 с.
4. *Карацуба А.А., Офман Ю.П.* Умножение многоразрядных чисел на автоматах // ДАН СССР, 1962. – Т. 145. – С. 293 – 294.

Одержано 20.03.2015

Про автора:

Терещенко Андрій Миколайович,
кандидат фізико-математичних наук,
старший інженер-програміст ТОВ «СімКорп Україна».
E-mail: *teramidi@ukr.net*