УДК 004.056.5

DOI: 10.18372/2073-4751.76.18240

Марковський О.П., к.т.н.,
orcid.org/0000-0003-3483-4233,
Аль-Мріят Гассан Абдель Жаліль,

МЕТОД МУЛЬТИПРОЦЕСОРНОГО МОДУЛЯРНОГО МНОЖЕННЯ З ВИКОРИСТАННЯМ ГРУПОВОЇ РЕДУКЦІЇ МОНТГОМЕРІ

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

markovskyy@i.ua,
grayatjo@gmail.com

Вступ

Динамічне розширення та поглиблення інформаційної інтеграції передбачає адекватне вдосконалення засобів захисту даних та розмежування прав доступу до даних. Переважна більшість існуючого на сьогоднішній день арсеналу таких засобів базується на криптографічних перетвореннях. Значну роль в сучасних технологіях захисту даних відіграють криптографічні алгоритми з “відкритим” ключем, які, в математичному сенсі мають за основу важко розв’язувані задачі теорії чисел [1]. Базовими операціями обчислювальної реалізації переважної більшості криптографічних алгоритмів цього класу є мультиплікативні операції модулярної арифметики і, зокрема модулярне множення над числами, довжина яких значно перевищує розрядність процесора. Зокрема, більшість протоколів інформаційної безпеки передбачає роботу з 2048-розрядними числами, що є для більшості застосувань прийнятним компромісом між рівнем захищеності та швидкодією [1].

Поява та швидке поширення хмарних технологій певною мірою мають наслідком порушення цього компромісу, оскільки вони надають зловмисникам можливості використання значних за обсягом віддалених комп’ютерних потужностей для порушення механізмів криптографічного захисту. Такий стан речей потребує підвищення рівня захищеності, яке в рамках криптографічних алгоритмів на основі важкорозв’язуваних задач теорії чисел може бути досягнуте лише за рахунок збільшення їх розрядності. В свою чергу, збільшення розрядності чисел має наслідком значне уповільнення реалізації функції захисту інформації. Тому важливим напрямком вдосконалення широкого класу механізмів криптографічного захисту даних є пошук можливостей прискорення обчислювальної реалізації їх базової операції – модулярного множення, в тому числі за рахунок розпаралелювання відповідних обчислень.

Таким чином, наукова задача прискорення модулярного множення є актуальною з огляду на сучасний стан та перспективи розвитку технологій захисту інформації.

Оглядовий аналіз методів прискорення модулярного множення

Модулярне множення $A \cdot B \bmod M$ чисел, довжина n яких значно перевищує розрядність r процесора є базовою обчислювальною операцією для широкого кола криптографічних алгоритмів з відкритим ключем [2]. Фактично, вказаний клас алгоритмів реалізує обчислення модулярного експоненціювання $A^E \bmod M$, яке має строго послідовний характер, тобто практично, не може бути розпаралелене [1]. Класичний алгоритм виконання модулярного експоненціювання складається n операцій модулярного піднесення до квадрату і, в середньому, $0.5 \cdot n$ операцій модулярного множення на однакове число А.

Операція модулярного множення фактично має дві складові: обчислення добутку $A \cdot B$ та редукції, тобто віднаходження залишку від ділення добутку на значення модулю M. Відповідно, всі методи модулярного множення можуть бути умовно

розділені на два класи: до першого з яких відносяться методи, в яких обчислення добутку та його редукція здійснюються окремо і послідовно у часі. До другого класу, відповідно, відносяться методи, які суміщають в часі множення і редукцію.

Суттєва перевага алгоритмів модулярного множення, що відносяться до першого класу, полягає в можливості використання вбудованої процесорної операції множення, яка ефективно реалізується апаратними засобами сучасних процесорів. Відповідно, для алгоритмів цього класу характерною рисою є виконання множення $A \cdot B$ з розбиттям чисел A і B на секції, довжина яких дорівнює розрядності процесора r. Кількість s секцій при цьому дорівнює $s=n/r$. Відповідно, множення $A \cdot B$ зводиться до попарного множення секцій та обчислення суми отриманих в результаті добутків. Вагома перевага посекційного множення полягає в можливості використання вбудованої операції процесорного множення, яка ефективно реалізується апаратними засобами. При цьому кількість операцій процесорного множення становить s^2. При виконанні піднесення до квадрату посекційна організація множення дозволяє зменшити кількість процесорних множень до $(s^2+s)/2$.

Крім того, в якості вагомого резерву прискорення операцій множення багаторозрядних чисел виступає можливість застосування алгоритмів швидкого множення, зокрема алгоритму А.Карацуби [3] та алгоритмів на основі швидкого перетворення Фурє [4].

Операція редукції отриманого добутку $A \cdot B$, тобто віднаходження залишку від його ділення на модуль M потребує значно більших ресурсів з огляду на те, що для цього неможливо використання вбудованої операції процесорного ділення. Тому основні зусилля дослідників зосереджуються на прискоренні саме операції редукції. Основний вектор цих досліджень полягає в намаганні перейти від операції ділення, до значно більш простої операції зсуву. Найбільш відомою технологією редукції цього типу є метод Барретта [5], який полягає в визначення найбільшого значення q, для якого $A \cdot B - q \cdot M < M$. Для обчислення потрібного значення q використовується одна операція множення та зсуви, кількість яких перевищує n. Оскільки результат редукції R обчислюється у вигляді: $R = A \cdot B - q \cdot M$, то цілком очевидно, що для виконання редукції потрібно дві операції множення чисел розрядністю n, тобто час редукції добутку $A \cdot B$ мінімум вдвоє перевищує час його обчислення. Таким чином, при використанні прискореної редукції Барретта, час модулярного множення, в першому наближенні, визначається часом виконання $3 \cdot s^2$ операцій процесорного множення.

Інша відома технологія переходу від операції ділення для віднаходження залишку до більш ефективної в обчислювальному плані операції зсуву запропонована П.Монтгомері [6]. Вказана технологія передбачає віднаходження такого u, що сума $A \cdot B + u \cdot M$ є кратною до $Q=2^n$. Значення u обчислюється у вигляді: $u = A \cdot B \cdot \eta \bmod Q$, причому значення η в свою чергу обчислюється як: $\eta = Q - M^{-1} \bmod Q$, де M^{-1} – мультиплікативна інверсія M по модулю Q. В силу того, що η залежить лише від модуля, воно може бути обчислене заздалегідь. З викладеного очевидно, що редукція Монтгомері передбачає дві операції множення. Характерна особливість редукції Монтгомері полягає в тому, що при її застосуванні результат дорівнює $A \cdot B \cdot Q^{-1} \bmod M$, де Q^{-1} – мультиплікативна інверсія числа $Q=2^n \bmod M$, тобто $Q \cdot Q^{-1} \bmod M = 1$.

Ще один підхід до швидкої редукції добутку $A \cdot B$ полягає в використанні передобчислень, які залежать тільки від модуля M. В реальних протоколах криптографічного захисту даних модуль M входить до складу відкритого ключа. Це означає, що на практиці він не змінюється [7]. Відповідно, існує можливість виділити всі операції, які залежать тільки від модуля M і можуть обчислені один раз зі збереженням отриманих результатів в табличній пам'яті. Конкретизація технології використання передобчислень для редукції $2 \cdot n$-

розрядного добутку $A \cdot B$ полягає в тому, що попередньо обчислюються значення: $\forall i \in \{n, n+1, \ldots, 2 \cdot n-1\}$: $g_i = 2^i \bmod M$. Результат редукції розраховується як модулярна сума тих значень g_i, для яких відповідний розряд $A \cdot B$ дорівнює одиниці та коду, утвореного n-1 молодшими розрядами добутку $A \cdot B$ [8].

Найбільш суттєвий недолік алгоритмів модулярного множення з рознесеними складовими обчислення добутку та редукції полягає в складності мультипрограмної реалізації, зумовленою значним обсягом обмінів даними та труднощами синхронізації [9].

Найбільш відомою технологією, в якій перемежаються множення та редукція, є технологія Монтгомері [10]. Ця технологія орієнтована на побітову організацію модулярного множення. Відповідно, вона передбачає виконання цієї операції у вигляді n циклів, в кожному з яких, залежно від поточного біту множника B здійснюється додавання множника до поточного результату, залежно від значення його молодшого біту, додавання модуля, а також завжди виконується зсув. Таким чином, час модулярного множення за описаною технологією, в першому наближенні, визначається часом виконання $2 \cdot n \cdot s$ операцій процесорного додавання.

Проведений аналіз публікацій присвячених проблемі прискорення модулярного множення багаторозрядних чисел показав, що в рамках одного процесора можливості подальшого скорочення часу виконання операції практично вичерпані [11]. Поява та широке впровадження багатоядерних архітектур відкривають нові можливості для прискорення модулярного множення шляхом розпаралелювання відповідного обчислювального процесу. Дослідження можливостей розпаралелювання обчислювального процесу модулярного множення багаторозрядних чисел дозволяє створювати ефективні апаратні засоби підтримки криптографічних обчислень: криптопроцесори та пристрої модулярного експоненціювання на програмованих логічних матрицях.

Існуючі підходи до паралельної реалізації модулярного множення в рамках групи переміжних алгоритмів [12,13] не забезпечують високої ефективності в силу суттєво різного обчислювального завантаження паралельних процесів.

Мета досліджень

Мета досліджень полягає в підвищенні ефективності мультипрограмної реалізації модулярного множення на числами, довжина яких на порядок перевищує розрядність процесора, за рахунок забезпечення високого рівня автономності та рівної завантаженості паралельних обчислювальних процесів.

Організація мультипроцесорної реалізації модулярного множення

Для досягнення поставленої мети пропонується метод прискорення реалізації модулярного множення $A \cdot B \bmod M$ на багатоядерних процесорах. Метод передбачає організацію модулярного множення у вигляді m обчислювальних процесів, які реалізуються на окремих ядрах процесору. Для цього , n-розрядний множник $B = b_0 + b_1 \cdot 2 + b_2 \cdot 4 + \ldots + b_{n-1} \cdot 2^{n-1}$, $\forall l \in \{0,1,\ldots,n-1\}$: $b_l \in \{0,1\}$, розділяється на m часткових множників $B_1, B_2, \ldots, B_m$. Кожний j–тий з цих часткових множник B_j містить в собі підмножину з n_j суміжних двійкових розрядів множника B. Іншими словами, в j–тому частковому множнику B_j молодші $d_j = n_1 + n_2 + \ldots + n_{j-1}$ двійкових розрядів дорівнюють нулю, наступні n_j розрядів співпадають з олнойменними розрядами повного множника B, а старші $n_{j+1} + n_{j+2} + \ldots + n_m$ розрядів також мають нульове значення. Тобто j–тий частковий множник B_j може бути представлений у вигляді:

$$B_j = b_{d_j+1} \cdot 2^{d_j+1} + b_{d_j+2} \cdot 2^{d_j+2} + \ldots + b_{d_j+n_j} \cdot 2^{d_j+n_j}$$

. Очевидно, що $n_1 + n_2 + \ldots + n_m = n$ та $B_1 + B_2 + \ldots + B_m = B$. Наприклад, якщо n=12 і множник B = 1010 1110 1011_2 = 2795_{10}, то при m=4 і значеннях n_1=1, n_2=2, n_3=3, n_4 = 6 він розділяється на 4 часткових множника: B_1=0000 0000 000**1**$_2$=1, B_2 =

0000 0000 0**01**0_2=2_{10}, B_3=0000 00**10** **1**000_2=40_{10} та B_4 = **1010 11**00 0000_2=2752_{10}.

Виходячи з того, що:

$$A \cdot B \bmod M = (\sum_{j=1}^{m} A \cdot B_j \bmod M) \bmod M$$

обчислення модулярного добутку $A \cdot B \bmod M$ пропонується організувати на m процесорних ядрах. При цьому, на кожному j-му з них пропонується здійснювати часткове модулярне множення $A \cdot B_j \bmod M$. Відповідно, кінцевий результат обчислюється як модулярна сума результатів часткових модулярних множень.

Запропонований спосіб формування часткових множників дозволяє рознести значимі розряди множника і, тим самим, забезпечує розпаралелювання виконання множення. Прискорення редукції часткових добутків досягається за рахунок використання групової редукції Монтгомері.

Звичайна редукція Монтгомері виконується побітно: в залежності від значення молодшого біту (одиниця або нуль) проміжного результату R, до нього додається або не додається непарний код модуля M. Після цього код R з нульовим молодшим бітом зсувається на один розряд праворуч.

Відмінність застосованої групової редукції від звичайної полягає в тому, що за відсутністю операцій додавання множимого, редуціювання здійснюється шляхом зсуву не на один, а на v>1 розрядів. Для того, щоб при такому зсуві не втрачалися значущі розряди проміжного результату, до нього попередньо додається такий добуток $\xi \cdot M$, $\xi \in \{0,1,\ldots,2^v-1\}$, що молодші v розрядів суми $R+\xi \cdot M$ дорівнюють нулю.

За запропонованим способом формування часткових множників, кожен j–тий з них складається с нулів в $d_j=n_1+n_2+\ldots+n_{j-1}$ молодших розрядах, n_j значащих розрядів множника B та нулів в старших $n_{j+1}+n_{j+2}+\ldots+n_m$ розрядах. Відповідно, при здісненні часткового модулярного множення $A \cdot B_j \bmod M$ обробка молодших d_j двійкових розрядів множника B_j може бути опущена оскільки додавання множника А не виконується, а редукція нульового коду проміжного результату не змінює цей результат. Обробка групи наступних n_j значащих розрядів множника B_j може виконуватися за технологією множення Монтгомері, тобто шляхом додавання до проміжного результату А, якщо відповідний біт множника B_j дорівнює одиниці, та редукції, яка здійснюється додаванням непарного коду модуля M в разі непарного значення проміжного результату з подальшим зсувом останнього на один розряд праворуч. При обробці старших $n_{j+1}+n_{j+2}+\ldots+n_m$ нульових розрядів множника B_j виконується лише редукція Монтгомері попередньо отриманого проміжного результата, яку пропонується шляхом зсуву праворуч відразу на декілька розрядів (групова редукції). Для цього має здійснюватися дадавання такої лінійної комбінації модуля, щоб відповідна кількість молодших розрядів проміжного результату обнулилась.

Таким чином, можливості для застосування групової редукції, як засобу прискорення модулярного множення, створені за рахунок запропонованого способу формування часткових множників.

Для застосування групової редукції потрібно організувати передобчислення добутків $\xi \cdot M$, де ξ – ціле число, які при додаванні до проміжного результату R забезпечують рівність нулю v молодших розрядів суми $R+\xi \cdot M$. Якщо позначити через $r_{v-1},r_{v-2},\ldots,r_1,r_0$ молодші v двійкових розрядів проміжного результату R, $\forall \kappa \in \{0,1,\ldots,v-1\}$: $r_\kappa \in \{0,1\}$, то значення ξ можна віднайти в результаті виконання наступної процедури:

1. Покласти $\xi=0$, $\kappa=0$ і $e=1$.
2. Якщо $r_0 = 1$, то $R=R+M$, $\xi=\xi+e$. В силу того, що M –непарне, $r_0 =0$.
3. Зсув R на один розряд праворуч; подвоєння значення e: $e=2 \cdot e$; інкремент κ: $\kappa=\kappa+1$. Якщо $\kappa<v$, то перехід на повторне виконання п.2.

4. Кінець процедури, в результаті якої сформовано ξ таке, що молодші v двійкових розрядів суми $R+\xi\cdot M$ дорівнюють нулю.

Наприклад, якщо модуль M = 3431_{10} = 1101 0110 0111_2, значення коду R = 0111 1001 1011_2 = 1947_{10}, а v=4, то виконання описаної процедури дає значення ξ = 3, сума $R+\xi\cdot M$ = 1947+3·3431 = 12240_{10}= 10 1111 1101 0000_2: тобто 4 її молодші двійкові розряди дорівнюють нулю.

З використанням наведеної процедури можна для визначеного зазадалегідь значенния v побудувати таблицю T[$r_{v-1}\cdot 2^{v-1}+\ldots+r_1\cdot 2+r_0$] для всіх 2^v можливих значень числа $\rho = r_{v-1}\cdot 2^{v-1}+\ldots+r_1\cdot 2+r_0$. Оскільки в реальних криптосистемах з відкритим ключем модуль входить до його складу, значення модуля можна вважати практично незмінним і сформувати вказану таблицю передобчислень лише один раз.

Оскільки, як було зазначено вище, обробка молодших нульових розрядів часткових множників не впливає на результат часткового множення, то перед початком виконання цієї операції доцільно зсунути на d_j розрядів праворуч кожен j–тий частковий множник B_j. Після цієї операції кожен j–тий частковий множник B_j' множник може бути представлений у вигляді:

$$B_j' = b_{d_j+1}\cdot 1 + b_{d_j+2}\cdot 2 + \ldots + b_{d_j+n_j}\cdot 2^{n_j-1}.$$

Тоді процес обчислення кожного j-го часткового модулярного добутку можна розділити на дві фази: фазу модулярного множення числа A на n_j молодших розрядів зсунутого часткового множника B_j', які дорівнюють відповідній підмножині розрядів множника B, та фазу виконання залишкової редукції. Перша фаза здійснюється у відповідності з технологією модулярного множення Монтгомері. Фаза залишкової редукції передбачає виконання $\phi_j=n-n_j-d_j$ циклів редукції Монтгомері. Для прискорення модулярного множення фазу залишкової редукції пропонується реалізувати у вигляді групової редукції, тобто суміщення в часі виконання не більш ніж v циклів редукції.

Наприклад, якщо n=12, m=4 і v=4, то для значеннь n_1=1, n_2=2, n_3=3, n_4 = 6, при обчисленні першого часткового добутку фаза залишкової редукції включає $\phi_1=n-n_1$=11 циклів, які можуть бути згруповані в три групи по 4,4 і 3. Відповідно, фаза залишкової редукції реалізується за три цикли групової редукції. Виконання залишкової фази редукції при обчисленні другого часткового добутку включає в себе $\phi_2=n-n_1-n_2$ = 12-1-2 =9 циклів, які можуть бути згруповані в три групи по 4,4 і 1, тобто перших 4 цикли редукції здійснюються одночасно в рамках першої групи. Аналогічно, наступні 4 цикли редукції згруповані в другу групу. Накінець виконується один цикл модулярної редукції. При реалізації обчислення третього часткового модулярного добутку залишкова фаза включає в себе $\phi_3=n-n_1-n_2-n_3$ = 12-1-2-3=6 циклів редукції, які можуть бути реалізовані за два цикли групової редукції: в першому з яких суміщено виконання 4-х циклів редукції, а в другому – 2. Для четвертого часткового модулярного множення ϕ_4=0 і, відповідно, фаза залишкової редукції не виконується.

Для реалізації групової редукції потрібно попередньо створити таблиці передобчислень для можливості одночасної обробки $\omega\le v$, тобто 2-х, 3-х і 4-х циклів редукції.

Нижче, в таблиці 1, в якості прикладу, наведено результати передобчислень T для значення модуля M=3431 та v=4.

Наведена таблиця передобчислень для v=4 може бути використана і для значень ω менших за v=4.

Таблиця 1. Приклад значень ξ та табличних значень T[ρ] = ξ·*M* для всіх можливих значень 4-розрядних чисел (*v*=4) при модулі M=3431

α	ξ	T[ρ] = ξ·*M*	α	ξ	T[ρ] = ξ·*M*
0	0	0	8	8	27448
1	9	30879	9	1	3431
2	2	6862	10	10	34310
3	11	37741	11	3	10293
4	4	13724	12	12	41172
5	13	44603	13	5	17155
6	6	20586	14	14	48034
7	15	51465	15	7	24017

Адресація таблиці передобчислень, тобто визначення номеру α потрібного рядка по заданим значенням ρ та *ω* здіснюється за такою формулою:

$$\alpha = \begin{cases} \rho, \text{ якщо } \omega = 4, \\ 2 \cdot \rho, \text{ якщо } \omega < 4 \ i \ \rho \ mod 2 = 0, \\ 8 + \rho, \text{ якщо } \omega < 4 \ i \ \rho \ mod 2 = 1 \end{cases} \quad (1)$$

Процедура обчислення *j*–того часткового модулярного добутку $A \cdot B_j \bmod M$ реалізується наступною послідовністю дій:

1. Лічильник *c* кількості циклів і поточне значення результату R_j встановлюються в нуль: $c=0$ та $R_j=0$.

2. Якщо молодший розряд часткового множника B_j' дорівнює одиниці: $b'_{j0}=0$, то до поточного результату R_j додається множене A : $R_j=R_j+A$.

3. Якщо молодший розряд поточного редультату R_j дорівнює одиниці: $r'_{j0}=1$, то до поточного результату R_j додається модуль M : $R_j=R_j+M$.

4. Здійснюється зсув праворуч на один розряд кодів поточного результату R_j та часткового множника : $R_j=R_j>>1$ і $B_j'=B_j'>>1$.

5. Виконується інкремент лічильника *c* циклів: $c=c+1$. Якщо $c<d_j$, то здіснюється повереннння на повторне виконання п.2.

6. Якщо значення лічильника *c* циклів дорівнює $n\text{-}d_j$: $c = n\text{-}d_j$, то перехід на п.9, якщо $n\text{-}d_j\text{-}\ c \geq v$, то встановлюється $\omega=v$, інакше $\omega = n\text{-}d_j\text{-}\ c$.

7. Копіюється в ρ значення ω молодших розрядів коду поточного результату R_j: $\rho = r_{j\omega\text{-}1} \cdot 2^{\omega\text{-}1} + \ldots + r_{j1} \cdot 2 + r_{j0}$. Додається до поточного результату R_j код T[ρ] з таблиці передобчислень, який адресується значенням α згідно формули (1) в залежності від значень ρ та α: $R_j=R_j+$T[α].

8. Здійснюється зсув на ω розрядів праворуч результату R_j: $R_j=R_j>>\omega$. До значення лічильника *c* циклів додається ω: $c=c+\omega$. Перехід на п.6.

9. Якщо $R_j \geq M$, то $R_j=R_j\text{-}M$. Результат $R_j = A \cdot B_j \bmod M$.

Робота запропонованої процедури може бути ілюстрована наступним прикладом. Нехай, здійснюється модулярне множення 12-розрядних чисел 2523·2789 mod 3431. Відповідно, $n=12$, множиме $A = 2523_{10} = 1001\ 1101\ 1011_2$, множник $B= 2789_{10} = 1010\ 1110\ 1011_2$, модуль $M = 3431_{10} = 1101\ 0110\ 0111_2$. При значеннях $n_1=1$, $n_2=2$, $n_3=3$, $n_4 = 6$ множник B розділяється на 4 часткових множника: B_1=0000 0000 000**1**$_2$=1, B_2 = 0000 0000 0**10**0$_2$=4, B_3=0000 00**10** **1**000$_2$=40 та B_4 = **1010** **11**00 0000$_2$=2744, для яких відповідно $d_1=0$, $d_2=1$, $d_3=3$, $d_4=6$. Після зсуву праворуч часткові множники мають такий вигляд: B_1'=0000 0000 000**1**$_2$, B_2' = 0000 0000 00**10**$_2$, B_3'=0000 0000 0**101**$_2$ та B_4' = 0000 00**10** **1011**$_2$.

Оскільки всі три часткових добутки обчислюються паралельно за однаковою схемою, нижче наведено детальний опис формування тільки третього часткового добутку $A \cdot B_3 \bmod M = 2523 \cdot 40 \bmod 3431$. Очевидно, що реальний результат цього добутку дорівнює 1421. Проте при

використанні редукції Монтгомері в результаті множення $A \cdot B_3$ mod M отримується результат $A \cdot B_3 \cdot Q^{-1}$ mod M, де Q^{-1} – мультиплікативна інверсія $Q=2^n$. Для n=12 значення $Q=2^{12}$, а його мультиплікативна інверсія Q^{-1}=3787. Відповідно в результаті роботи наведеної процедури значення третього часткого добутку отримується у вигляді $A \cdot B_3 \cdot Q^{-1}$ mod M = 2523·40·3787 mod 3431=1519.

Згідно описаною процедурою процес часткового модулярного множення розпочинається з того, що обнуляються значення лічильника c циклів та проміжного результату R_3.

В рамках першого циклу (c=0), оскільки молодший розряд множника B_3'=5 дорівнює одиниці, здійснюється додавання множимого A до проміжного результату: R_3=0+A=A=2523. Так як молодший біт коду R_3 результату дорівнює одиниці, до R_3 додається модуль M: R_3= R_3+M=2523+3431= 5954 і виконується зсув R_3>>1 = 5954/2 = 2977 та множителя B_3' = B_3' >>1=5/2=2. Інкрементується лічильник c= c+1=1. Так як c<n_3=3, здійснюється повернення на повторне виконання п.2 процедури.

При c=1 аналізується значення молодшого біту поточного коду множителя B_3'=2: так, як цей біт дорівнює нулю, то згідно п.3-4 виконується редукція: оскільки молодший розряд результату R_3=2977 дорівнює одиниці, до проміжного результату додається модуль R_3= R_3+M=2977+3431=6408 після чого здійснюється зсув праворуч: R_3= R_3>>1=6408/2=3204. Зсувається також код часткового множителя: B_3' = B_3' >>1=1. Збільшується на одиницю лічильник циклів: c=c+1=2 і, так як його значення c<n_3=3, здійснюється повернення на повторне виконання п.2 процедури.

В межах третього циклу (c=2) молодший біт множника B_3' дорівнює одиниці, тому до R_3 додається множиме A: R_3= R_3+A=3204 + 2523 = 5727. Для коректної редукції до непарного R_3 додається непарний модуль M: R_3= R_3+M=5727 + 3431 = 9158. Після цього результат R_3 зсувається на один розряд праворуч: R_3= R_3>>1=9158/2=4579. Інкрементується лічильник c= c+1=3. Так як c = n_3=3, і n-d_j - c = 12-3-3=6 ≥ v=4, то встановлюється ω=v=4.

В рамках наступного циклу (c=3) у відповідності з п.7 в ρ копіюється значення ω=4 молодших розрядів коду поточного результату R_3 = 4579_{10}= 1 0001 1110 **0011**, тобто ρ=0011_2=3**.** За формулою (1) обчислюється адреса α таблиці передобчислень. Для ω=4 α=ρ=3. Віповідно, до поточного результату R_3 додається значення T[3] = 37741 з наведеної вище таблиці передобчислень: R_3 = R_3 + T[3] = 4579 + 37741 = 42320_{10}= $1010\ 0101\ 0101\ 0000_2$. Чотири молодші двійкові розряди R_j дорівнюють нулю. Згідно п.8 виконується звув на ω=4 розряди праворуч коду R_3= R_3>>4=42320/16=4645. Значення c збільшується на ω: c=c+ω = 3+4=7.

При виконанні останнього циклу (c=7) в силу того, що n-d_j- c = 12-3-7=2 <v=4, згідно п.6 ω = n-d_j- c = 12-3-7=2. Далі за п.7 в ρ заносяться значення двох (ω=2) молодших розрядів коду проміжного результату R_3 = 2645_{10}=1010 0101 01**01**$_2$ після чого ρ=1. Обчислюється за формулою (1) значення адреси α таблиці передобислень. Оскільки ω=2 і ρ непарне, α = 8+ρ=9. Далі до R_3 додається значення T[9] = 3431 з таблиці передобчислень: R_3 = R_3 + T[9] = 2645 + 2645 + 3431 = 6076_{10}= $0001\ 0111\ 1011\ 1100_2$. У відповідності з п.8 здійснюється зсув на ω=2 розряди праворуч коду R_3= R_3>>2 = 6076/4=1519. Значення c збільшується на ω: c=c+ω = 7+2=9. Оскільки c= n-d_j, виконується перехід на п.9. Так як R_3=1519<M=3431, то результатом третього часткового модулярного множення є число 1519.

Аналогічним чином за запропонованою процедурою можуть бути обчислені результаті всіх трьох інших часткових добутків. Зокрема, в рамках розглянутого прикладу, R_1= 2523·1·3787 mod 3431=2697, R_2= 2523·4·3787 mod 3431=495 та R_4= 2523·2744·3787 mod

3431=1388. Модульна сума обчисленийх часткових добутків становить: (2080+1963+ 1519+891) mod 3431 = 2668, що співпадає з рузультатом модулярного добутку Монтгомері 2523·2789·3787 mod 3431 = 2668.

Таким чином, запропонований метод модулярного множення дозволяє реалізувати розпаралелювання обчислення базової операції криптогріфії з відкритим ключем на багатоядерних процесорах.

Оцінка ефективності

В якості основного показника ефективності запропонованого методу реалізації модулярного множення довгих чисел на багатоядерних процесорах доцільно розглядати прискорення відповідних обчислень за рахунок розпаралелювання. Числовою характеристикою такого прискорення може слугувати коєфіцієнт μ, який визначається співідношенням часу T_1 модулярного множення на одному процесорі до часу T_m здіснення цієї операції за запропонованим методом на m процесорних ядрах:

$$\mu = \frac{T_1}{T_m}. \qquad (2)$$

Модулярне множення багаторозрядних n–розрядних чисел з використанням редукції Монтгомері на одному процесорі зводиться виконання n циклів, в кожному з який зі ймовірністю 0.5 відбувається додавання до проміжного результату множимого, з такою ж ймовірністю відбувається додавання модуля і завжди здійснюється зсув праворуч проміжного результату. Вважаючи, що час t додавання n-розрядних чисел близький до часу зсуву чисел такої ж розрядності, середнє значення часу T_1 становить: $T_1=2\cdot n\cdot t$.

Цілком очевидно, що максимальний ефект прискорення модулярного множення при його реалізації на m процесорах досягається при повній завантаженості кожного з них, тобто за умови, що час виконання операцій на кожному із процесорів приблизно однаковий. Час τ_j обчислення j-го часткового модулярного добутку на j-тому ядрі утворюється двома складовими: часом модулярного множення числа A на n_j-розрядний фрагмент множника B, а також часом виконання n-n_j-d_j циклів редукції отриманого добутку. Перша складова дорівнює $2\cdot n_j\cdot t$. Середній час одного циклу редукції Монтгомері визначається часом додавання модулю до проміжного результату та часом його зсуву. Враховуючи, що операція додавання здійснюється з ймовірністю 0.5, середній час виконання n-n_j-d_j циклів редукції становить $1.5\cdot(n-n_j-d_j)\cdot t$. При використанні групової редукції, в рамках якої суміщається виконання до v окремих циклів, середній час скорочується приблизно в v раз. Відповідно, час τ_j обчислення j-го часткового модулярного добутку на j-тому процесорі визначається формулою:

$$\tau_j = 2\cdot n_j\cdot t + \frac{1.5\cdot(n-n_j-d_j)\cdot t}{v}. \qquad (3)$$

Як зазначалося вище, для досягнення максимального ефекту прискорення від розпаралелювання обчислень, час їх виконання на кожному із процесорів має бути приблизно однаковим, зокрема має виконуватися умова $\tau_j\approx\tau_{j+1}$. З урахуванням того, що $d_{j+1}=d_j+n_j$ та після елементарних скорочень, ця умова може бути представлена у такому вигляді:

$$\begin{matrix} 2\cdot n_j\cdot v - 1.5\cdot n_j - 1.5\cdot d_j = \\ 2\cdot n_{j+1}\cdot v - 1.5\cdot n_{j+1} - 1.5\cdot d_j - 1.5\cdot n_j \end{matrix}. \qquad (4)$$

З формули (4) можна отримати співідношення γ між кількостями значимих розрядів множника B в j–тому B_j та (j+1)–тому B_{j+1} часткових множниках:

$$\gamma = \frac{n_{j1}}{n_{j+1}} = \frac{2\cdot v - 1.5}{2\cdot v}. \qquad (5)$$

Визначення чисельних значень $n_1,n_2,\ldots,n_m$ може бути здійснене на основі наступних міркувань. Якщо вважати, що при обчисленні m-го часткового добутку

B_m відсутня фаза залишкової редукції отриманого добутку, то справедливо наступне:

$$n_m + n_{m-1}\cdot\gamma + n_{m-2}\cdot\gamma^2 + ... + n_1\cdot\gamma^{m-1} = \\ = n_m\cdot\sum_{j=0}^{m-1}\gamma^j = n \quad . \quad (6)$$

Зазначена в формулі (6) сума являє собою сумою кінцевої геометричної прогресії. Відповідно, чисельні значення $n_1,n_2,\ldots,n_m$ визначаються на наступними формулами:

$$\forall j \in \{1,,2,..,m\} : n_j = n\cdot\frac{1-\gamma}{1-\gamma^m}\gamma^{m-j}. \quad (7)$$

Наприклад, якщо модулярне множення 2048-розрядних чисел (n=2048) здійснюється на 4-х процесорних ядрах (m=4) з використанням групової редукції, що дозволяє сумістити в часі максимум до 4-х циклів редукції залишкової фази (v=4), то визначені за формулою (7) значення кількості значимих розрядів по частковим множникам дорівнюють: n_4 = 683, n_3 = 556, n_2= 452 та n_1=367. При цьому час $T_m \approx \tau_1 \approx \tau_2 \approx \ldots \approx \tau_m$ формування кожного із часткових добутків складає приблизно 1361·t. Відповідно, при заданих умовах, використання запропонованої організації модулярного множення дозволяє скоростити час обчислень в $\mu \approx 2\cdot n\cdot t/1361\cdot t$ = 3 рази. При збільшенні в рамках розглянутого прикладу значення v до 8-ми (v=8) відповідним чином зростає коєфіцієнт μ, який дорівнює μ=3.47. Відповідно, якщо v=16 n_4 = 549, n_3 = 524, n_2= 499 та n_1=476. 1100·t. в $\mu \approx 4096/1100$ = 3.72 рази.

Наведений приклад ілюструє загальний принцип визначення кількості значимих розрядів у кожному із часткових множників: їх кількість збільшується з молодших до старших розрядів, оскільки для останніх зменшується час виконання залишкової редукції.

В загальному випадку, за умови вибору чисельних значень $n_1,n_2,\ldots,n_m$ за формулою (7), значення коєфіцієнту μ прискорення обчислення модулярного добутку при використанні запропонованого методу визначається наступною формулою:

$$\mu = \frac{T_1}{\tau_m} = \frac{n}{n_m} = \frac{1-\gamma^m}{1-\gamma}. \quad (8)$$

Значення коєфіцієнту μ прискорення модулярного множення для найбільш реальних кількостей m паралельних процесів та чисел v циклів фази залишкової редукції, виконання яких суміщується у часі, наведено в таблиці 2.

Таблиця 2. Значення коєфіцієнту μ прискорення модулярного множення для різних значень m та v.

m	μ		
	v = 4	v = 8	v = 16
4	3	3.48	3.72
8	4.3	5.8	6.8
16	5.1	8.5	11.4

Аналіз наведених в таблиці 2 даних свідчить про те, що з укрупненням групової редукції, тобто зі збільшенням v, коефіцієнт μ прискорення модулярного множення наближається до кількості m часткових модулярних добутків, що обчислюються паралельно.

Висновки

В результаті проведених досліджень, направлених на прискорення важливої для криптографічних застосувань модулярного множення над числами довжиною значно більшою за розрядність процесора, запропоновано метод реалізації цієї операції у вигляді незалежних обчислювальних процесів.

Запропонований метод базується на технології Монтгомері і відрізняється тим. що множення здійснюється у вигляді m часткових модулярних множень, множниками в яких виступають групи значущих розрядів множника, а редукція отриманих часткових добутків виконується з суміщенням у часі груп окремих циклів редукції з використанням передобчислень, що дозволяє розпаралелити процес множення і організувати групову редукцію, за рахунок чого досягається прискорення обчислення модулярного добутку.

Розроблено методику розрахунку кількості розрядів множника, що розподіляються по кожному із часткових множникам, яка забезпечує однаковий об'єм обчислень в кожному із паралельних процесів з урахуванням редукції.

Теоретично встановлена залежність ефективності запропонованого методу від кількості паралельних процесів обчислення часткових добутків та числа циклів редукції в групі. Доведено, що при збільшенні останнього коєфіцієнт прискорення наближається до кількості процесів, що виконуються паралельно.

Проведені експериментальні дослідження в цілому підтвердили отримані теоретичним шляхом результати.

Запропонований метод орієнтований для прискорення обчислювальної реалізації криптографічних алгоритмів захисту даних відкритим ключем на багатоядерних комп'ютерних платформах.

Література

1. Alfred Menezes, Paul C. van Oorschot, and Scott A. Vanstone. Handbook of Applied Cryptography. CRC Press. 2001. 780 p.

2. Hars Laszlo. Long Modular Multiplication for Cryptographic Applications. Cryptographic Hardware and Embedded Systems (CHES 2004). Lecture Notes in Computer Science. 2004. V. 3156. P. 45–61.

3. Карацуба А.А., Офман Ю.П. Умножение многозначных чисел на автоматах. *ДАН СССР*. 1962. Т. 145. № 2. С. 293–294.

4. Furer M. Faster integer multiplication algorithm. *SIAM Journal on Computing*. 2009. V. 39. No. 3. P. 979–1005.

5. Barrett P. Implementing the Rivest Shamir and Adleman Public Key Encryption Algorithm on a Standard Digital Signal Processor. *Proceedings CRYPTO'86*. P. 311-323.

6. Montgomery P. Modular multiplication without trial division. *Mathematics of Computation*. 1985. V. 44(170). P. 519–521.

7. Bardis N., Markovskyi O. Secure Implementation of Modular Exponentiation on Cloud Computing Resources. *Proceeding of International Conference Applied Mathematics, Computational Science and Systems Engineering* / Athens, Greece, 2017. P. 90–96.

8. Самофалов К.Г., Рамзи Анвар Салиба Сунна, Романовский А.Е. Алгоритм ускоренного модулярного умножения чисел большой разрядности при фиксированном модуле. *Проблеми інформатизації та управління*. 2005. В. 3(14). С. 121–128.

9. Giorgi P., Imbert L., Izard T. Parallel modular multiplication on multi-core processors. *IEEE Symposium on Computer Arithmetic* / Austin, United States, 2013. P. 135–142.

10. Bos J.W., Montgomery P.L., Shumow D., Zaverucha G.M. Montgomery multiplication using vector instructions. *Selected Areas in Cryptography – SAC*. V. 8282. P. 471–489.

11. Che Wun Chion, Ted C. Yang. Parallel modular multiplication with table look-up. *International Journal of Computer Mathematics*. 1998. V. 69. Iss. 1-2. P. 22–23.

12. Tiago Vanderlei de Arruda, Yeda Regina Venturini, Tiemi Christine Sakata. Performance analysis of parallel modular multiplication algorithms for ECC in mobile device. *Revista de Sistemas de Informacao da FSMA*. 2014. No. 13. P. 57–67.

13. Buhrow B., Gilbert B., Haider C. Parallel modular multiplication using 512-bit advanced vector instructions: RSA fault-injection countermeasure via interleaved parallel multiplication. *Journal of Cryptographic Engineering*. 2021. № 2. P. 17–25.

Марковський О.П., Аль-Мріят Гассан Абдель Жаліль

МЕТОД МУЛЬТИПРОЦЕСОРНОГО МОДУЛЯРНОГО МНОЖЕННЯ З ВИКОРИСТАННЯМ ГРУПОВОЇ РЕДУКЦІЇ МОНТГОМЕРІ

В статті запропоновано метод прискорення важливої для криптографічних застосувань операції модулярного множення за рахунок її реалізації у вигляді m автономних обчислювальних процесів. Детально розроблено технологію розділення множника на складові та процедуру виконання часткових модулярних множень з застосуванням групової редукції Монтгомері на базі передобчислень. Виклад проілюстровано числовим прикладом. Теоретично обгрунтовано вибір параметрів організації обчислень, що забезпечує найбільший ефект розпаралелювання модулярного множення. Доведено, що за рахунок розпаралелювання множень та групової редукції Монтгомері досягається прискорення обчислення модулярного добутку в 0.75·m-0.93·m раз.

***Ключові слова**: модулярне множення, модулярна редукція Монтгомері, криптографія з відкритии ключем, паралельна арифметика.*

Markovskyi O.P., Al-Mrayat Ghassan Abdel Jalil Halil

METHOD OF MULTIPROCESSOR MODULAR MULTIPLICATION WITH USING MONTGOMERY GROUP REDUCTION

The article proposes a method of accelerating the operation of modular multiplication important for cryptographic applications due to its implementation in the form of m autonomous computational processes. The technology of allocation the multiplier into components and the procedure for performing partial modular multiplications using Montgomery group reduction based on precalculations have been developed in detail. The statement is illustrated by a numerical example. The choice of parameters of the organization of calculations that provides the greatest effect of parallelization of modular multiplication is theoretically substantiated. It is proved that due to the parallelization of multiplications and the group reduction of Montgomery, the calculation of the modular product is accelerated by 0.75·m-0.93·m times.

***Keywords:** modular multiplication, Montgomary modular reductions, open key cryptography, parallel arithmetic.*