1. *Berezsky O.* Methods and Algorithms of Biomedical Image Transforms in Affine and Topological Spaces // International Journal of Advanced Information Science and Technology (IJAIST) – 2016. – Vol. 10, № 5. – P.1-10.
2. *Berezsky O.* An intelligent system for cytological and histological image analysis / O. Berezsky, G. Melnyk, T. Datsko, S. Verbovy // 13th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM), 2015 – Lviv, 2015. – P.28-31.
3. *Berezsky O.* Fuzzy System of Diagnosing in Oncology Telemedicine / O. Berezsky, S. Verbovyy, L. Dubchak, T. Datsko // Sensors & Transducers – 2017. – № 208. – P.32-38.
4. *Мельник Г.М.* Метод знаходження відповідних точок на контурах мікрооб'єктів біомедичної природи // Вісник Національного університету "Львівська політехніка" "Комп'ютерні науки та інформаційні технології" – 2012. – № 732. – С.343-350.
5. *Мельник Г.М.* Інформаційна технологія аналізу структурних текстур для опрацювання зображень ауто- та ксеногенних тканин //Вісник Хмельницького національного університету – 2014. – № 6 (217). – С.132-141.
6. *Мельник Г.М.* Зменшення простору текстурних ознак гістологічних зображень за допомогою методу головних компонент / Г. М. Мельник // Моделювання та інформаційні технології. Збірник наукових праць. – 2016. – № 77. – С.176-180.
7. *Батько Ю.М.* Аналіз контурних ознак мікрооб'єктів на цифрових кольорових біомедичних зображеннях / Ю.М. Батько // Моделювання та інформаційні технології. – 2016. – Вип. 76. – С.184-190.

Поступила 14.09.2017р.

УДК 511:003.26.09

С.Д. Винничук, В.М. Міський, Київ

ПРИСКОРЕННЯ МЕТОДУ КВАДРАТИЧНОГО РЕШЕТА НА ОСНОВІ ВИЗНАЧЕННЯ ДОСТАТНЬОЇ КІЛЬКОСТІ B-ГЛАДКИХ ЧИСЕЛ

Abstract. The quadratic sieve method is the fastest for integers under 100 decimal digits or so. This paper describes the method which allows to stop sieving with less number of B-smooth than basic Quadratic Sieve method. This fact reduces complexity of sieving part, building and resolving matrix.

Вступ

На сьогоднішній день криптоалгоритм RSA реалізовано у багатьох комерційних системах. Він використовується у web серверах та браузерах для захисту трафіку, у електронній пошті для забезпечення конфіденційності та автентичності, та є ключовою технологією у системах електронних платежів. Найбільш поширена атака на цей криптоалгоритм заснована на факторизації публічного ключа [7,8]. Якщо факторизація успішна, усі повідомлення зашифровані відкритим ключем можуть бути прочитані.

Метод квадратичного решета (QS) займає друге місце у списку найшвидших алгоритмів факторизації [1], поступаючись тільки методу решета числового поля, а для чисел розміром до 100 десяткових знаків і досі є найкращим.

Зниження обчислювальної складності методу Квадратичного решета, надасть змогу покращити процес криптоаналізу алгоритму RSA. Тому дослідження нових способів прискорення методу Квадратичного решета є актуальним.

Постановка задачі

Ідея алгоритму квадратичного решета полягає в тому щоб знайти квадрати чисел, які дорівнюють за модулем *N* (число, яке факторизується), що найчастіше призводить до факторизації *N*.

Алгоритм працює в два етапи: етап збору даних, де він збирає інформацію, яка може привести до рівності квадратів по модулю *N*, та етап обробки даних, де він розміщує всю зібрану інформацію у матрицю та оброблює її для отримання рішення [2].

Розглянемо докладніше перший етап, який включає в себе:

1. Вибір інтервалу просіювання $[-L^b; L^b]$.
2. Побудова факторної бази.
3. Виконання процедури просіювання.

Процес просіювання є найбільш затратною за часом частиною алгоритму квадратичного решета. Для загального випадку (*згідно з* [3]), отримати розмір інтервалу просіювання можна за формулою:

$$M_{\max} = L^b = \left(e^{\sqrt{\ln(n)\ln\ln(n)}}\right)^{3\sqrt{2}/4} = L(n)^{3\sqrt{2}/4} = L^{3\sqrt{2}/4} \quad (1)$$

Під час просіювання шукають пари чисел (A,B) які задовольняють умові $A^2 \equiv B(\mathrm{mod} n)$.

Розмір факторної бази згідно [2, 4, 6] рекомендується вибирати за формулою

$$L^a = \left(e^{\sqrt{\ln(n)\ln\ln(n)}}\right)^{\sqrt{2}/4} = L(n)^{\sqrt{2}/4} = L^{\sqrt{2}/4}. \quad (2)$$

Це один з ключових параметрів, що визначають ефективність алгоритму просіювання. Згідно [3] для випадків, коли розмір факторної бази менший за необхідний, то ми не знайдемо достатню кількість гладких чисел або не знайдемо їх взагалі [3]. Надто великий розмір факторної бази потребує пошуку великої кількості гладких чисел, що збільшує загальний час виконання алгоритму [2, 5, 6]. Задача ж пошуку оптимального розміру факторної бази на даний час не вирішена. Співвідношення (2) – це рекомендована величина, отримана на основі чисельних експериментів. Тоді алгоритм шукає B-гладкі числа у кількості не менше ніж $M = L^a + 2$. Проте можливі випадки, коли задачу факторизації N можна вирішити і при меншому числі B-гладких чисел,

коли для факторної бази, що містить $L^* < L^a$ кількість B-гладких чисел стане рівною $L^* + 2$. Алгоритм пошуку L^* пропонується в даній роботі.

Метод вибору достатньої кількості B-гладких чисел

У рамках цього дослідження, будемо вважати вирішеною задачу пошуку ефективних розмірів інтервалу просіювання L^b та факторної бази L^a.

Нехай також формування матриці виконується під час процедури просіювання аналогічно як у базовому методі QS, тобто для кожного знайденого вектора B-гладкого числа $V_{\exp}$ інформація одразу ж заноситься до загальної матриці M_{gen}.

В пропонованому алгоритмі **LB** методу вибору достатньої кількості B-гладких чисел використовується додатковий вектор $V_{\max}$, розмір якого дорівнює розміру вектору степенів $V_{\exp}$. Визначення початкових значень елементів вектора $V_{\exp}$, їх зміни при отриманні нового B-гладкого числа та умови достатності кількості B-гладких чисел представлені кроками алгоритму **LB** нижче.

1. Довільний елемент вектора $V_{\exp}$ відповідає або знаку B-гладкого числа, або елементу факторної бази. Будемо вважати, що нульовій клітинці вектора $V_{\max}$ відповідає знак B-гладкого числа, а довільній іншій – відповідний порядковий номер елемента факторної бази. Присвоїмо довільному k-у елементу вектора $V_{\max}$ значення k +2.
2. На етапі проріджування при отриманні B-гладкого числа формуємо вектор $V_{\exp}$ та визначаємо номер j останнього ненульового непарного елемента в ньому.
3. Для всіх елементів вектора $V_{\max}$, починаючи з номера j, зменшуємо їх значення на одиницю.
4. Кроки 2 та 3 продовжуємо до тих пір, поки один із елементів вектора $V_{\max}$, наприклад, елемент $V_{\max}[k]$ не стане рівним нулю.
5. Переставляємо рядки матриці так, що перші k +2 рядки містять ненульові значення в стовпчиках від 0 до k включно.
6. Опрацьовуємо виділену частину матриці та виясняємо чи отримане значення кореня не дорівнює N. Якщо ні, то задача факторизації вирішена, а інакше переходимо до кроку 7.
7. Видаляємо рядок, що відповідає отриманому при обробці нульовому вектору, та починаючи з останнього ненульового елемента первинного вектора, що відповідає нульовому, добавляємо одиницю до всіх елементів вектора $V_{\max}$. Переходимо до кроку 2.

Аналіз додаткових векторів дозволяє побудувати матрицю $M_{\min}$ меншого розміру використавши меншу кількість B-гладких чисел.

Застосування методу вибору достатньої кількості B-гладких чисел

Розглянемо на прикладах ефективність запропонованої модифікації.

Приклад 1. Нехай *p=1217* та *q= 1297*. Обрані *p* та *q* створюють число для факторизації *p*q=N=1578449*. Обчислимо за формулою (1) розмір факторної бази: $L^a = 9$. За допомогою формули (2) отримаємо інтервал просіювання: *M= 687*. Зміна значень вектора $V_{\max}$ в ході отримання B-гладких чисел представлено в табл. 1. Згідно отриманих даних, для отримання достатньої кількості B-гладких чисел достатньо скористатися факторною базою, що містить прості числа 2, 5, 11, 17, 19, 41 та 47. Тоді число стовпчиків у матриці $M_{\min} = 8$.

У табл. 2 зображені усі B-гладкі числа, які відповідають новій факторній базі.

Таблиця 1

Вектори $V_{\max}$ та $V_{\exp}$ в ході формування матриці

№ п/п	Вектори	Знак числа	2	5	11	17	19	41	47	59	61	*B*-гладкі
0	$V_{\max}$	2	3	4	5	6	7	8	9	10	11	–
1	$V_{\exp}$	1	0	1	1	1	1	1	0	0	0	-728365
	$V_{\max}$	2	3	4	5	6	7	7	8	9	10	
2	$V_{\exp}$	1	0	3	0	3	0	0	0	0	0	-614125
	$V_{\max}$	2	3	4	5	5	6	6	7	8	9	
3	$V_{\exp}$	1	7	1	0	1	0	0	1	0	0	-511360
	$V_{\max}$	2	3	4	5	5	6	6	6	7	8	
4	$V_{\exp}$	1	4	1	2	0	0	1	0	0	0	-396880
	$V_{\max}$	2	3	4	5	5	6	5	5	6	7	
5	$V_{\exp}$	1	0	1	0	0	1	0	0	1	1	-341905
	$V_{\max}$	2	3	4	5	5	6	5	5	6	6	
6	$V_{\exp}$	1	5	1	0	0	0	1	1	0	0	-308320
	$V_{\max}$	2	3	4	5	5	6	5	4	5	5	
7	$V_{\exp}$	1	7	3	0	1	0	0	0	0	0	-272000
	$V_{\max}$	2	3	4	5	4	5	4	3	4	4	
8	$V_{\exp}$	1	0	2	1	0	1	1	0	0	0	-214225
	$V_{\max}$	2	3	4	5	4	5	3	2	3	3	
9	$V_{\exp}$	1	4	0	0	1	1	1	0	0	0	-211888
	$V_{\max}$	2	3	4	5	4	5	2	1	2	2	
10	$V_{\exp}$	1	8	2	0	0	1	0	0	0	0	-121600
	$V_{\max}$	2	3	4	5	4	4	1	**0**	1	1	

В результаті обробки матриці, представленої в табл. 2, множники числа N будуть знайдені.

Приклад 2. Оберемо *p=179* та *q=19*, отримаємо *p*q=N=3401*. Обчислимо розмір факторної бази та інтервал просіювання *A=5*, *M=79*.

Таблиця 2

Матриця $M_{\min}$ для нової факторної бази

Знак числа	2	5	11	17	19	41	47	*B*-гладкі
1	0	1	1	1	1	1	0	-728365
1	0	1	0	1	0	0	0	-614125
1	1	1	0	1	0	0	1	-511360
1	0	1	0	0	0	1	0	-396880
1	1	1	0	0	0	1	1	-308320
1	1	1	0	1	0	0	0	-272000
1	0	0	1	0	1	1	0	-214225
1	0	0	0	1	1	1	0	-211888
1	0	0	0	0	1	0	0	-121600

В момент додавання B-гладкого числа -800, ми маємо *B*-гладкі числа, які зображені в табл. 3.

Таблиця 3

Вектори $V_{\max}$ та $V_{\exp}$ в ході формування матриці для N = 3401

№ п/п	Вектори	Знак числа	2	5	17	37	41	*B*-гладкі
0	$V_{\max}$	2	3	4	5	6	7	–
1	$V_{\exp}$	1	4	1	0	1	0	-2960
	$V_{\max}$	2	3	4	5	5	6	
2	$V_{\exp}$	1	9	1	0	0	0	-2560
	$V_{\max}$	2	3	3	4	4	5	
3	$V_{\exp}$	1	3	0	2	0	0	-2312
	$V_{\max}$	2	2	2	3	3	4	
4	$V_{\exp}$	1	7	0	1	0	0	-2176
	$V_{\max}$	2	2	2	2	2	3	
5	$V_{\exp}$	1	3	3	0	0	0	-1000
	$V_{\max}$	2	2	1	1	1	2	
6	$V_{\exp}$	1	5	2	0	0	0	-800
	$V_{\max}$	2	1	0	0	0	1	

Згідно отриманих даних, для отримання достатньої кількості B-гладких чисел достатньо скористатися факторною базою, що містить прості числа 2, 5. Тоді число стовпчиків у матриці $M_{\min} = 4$.

У табл. 4 зображені усі B-гладкі числа, які відповідають новій факторній базі.

Таблиця 4

Матриця $M_{\min}$ для нової факторної бази для N = 3401

Знак числа	2	5	*B*-гладкі
1	1	1	-2560
1	1	0	-2312
1	1	1	-1000
1	1	0	-800

Застосування методу дозволяє відкинути B-гладкі числа -2960 та -2176 і зменшити кількість стовпчиків у матриці до $M_{\min} = 3$.

Оцінка складності та часу виконання

Складання матриці для стандартного алгоритму квадратичного решета потребує L^{2a} місця [3]. При застосуванні додаткового вектора $V_{\max}$ необхідний розмір пам'яті збільшується на – L^a та становить $L^{2a} + L^a$. Тобто має місце незначне зростання обсягу необхідної оперативної пам'яті ЕОМ.

Для випадків, коли за допомогою алгоритму **LB** не вдається зменшити число елементів факторної бази, збільшується число операцій віднімання одиниці від елементів вектора $V_{\max}$. Число таких операцій не перевищує $(L^a + 2) \cdot (L^a + 3)/2$, що для вирішуваної задачі є асимптотично малою величиною.

У порівняльних тестах базового алгоритму з прискореним методом для чисел порядку $10^8 - 10^{10}$ у кількості 1000000 було показано, що при використанні базового алгоритму факторизація виконувалась за 211 хвилин. При використанні запропонованого методу час розкладання на множники становив 177 хвилин на тому ж комп'ютері, що на 20 відсотків менше. Результати факторизації наведені у табл. 5.

Таблиця 5

Результати факторизації

	Базовий алгоритм	Модифікований алгоритм
Загальна кількість	1000000	1000000
Успішні розкладання	899326	901078
Не успішні розкладання	100674	98922

Крім того, модифікований алгоритм факторизував на 1752 числа більше ніж базовий алгоритм квадратичного решета. Це мало місце у випадках, коли базовий алгоритм не зміг отримати достатню кількість B-гладких чисел. Приклади таких N наведені у табл. 6.

Таблиця 6

Приклади чисел успішно факторизованих тільки модифікованим алгоритмом

p	q	N	M_{gen} для стандартного методу QS	Зменшена $M_{\min}$
48593	74167	3603997031	21	17
48611	71887	3494498957	21	17
48611	77551	3769831661	21	17
77551	77551	6014157601	21	19
51349	78059	4008251591	21	17
53759	71443	3840704237	21	20
55333	73823	4084848059	21	20

Таблиця 7

Приклади розміру матриці для базового та модифікованого алгоритму квадратичного решета

p	q	N	M_{gen} для стандартного методу QS	Зменшена $M_{\min}$
48593	71867	3492233131	21	17
48593	72893	3542089549	21	18
50069	75169	3763636661	21	16
50513	81197	4101504061	21	15
51361	81019	4161216859	21	18
52747	73459	3874741873	21	18
54011	76801	4148098811	21	18
55291	80789	4466904599	21	20
56527	73291	4142920357	21	17
56527	76001	4296108527	21	16
56957	71347	4063711079	21	17
57331	78779	4516478849	21	20
57751	78511	4534088761	21	17
58049	80141	4652104909	21	18
58391	80153	4680213823	21	19
58458	71249	4165074042	21	17
58963	71263	4201880269	21	17
59051	71917	4246770767	21	17
59029	76157	4495471553	21	17
50929	71399	3636279671	21	20

При загальній кількості тестів 1000000, у 300000 випадках (тобто у 30%) модифікований алгоритм зменшив розмір матриці. Ряд значень числа N, коли при використанні модифікованого алгоритму квадратичного решета зменшувався розмір факторної бази та число стовпчиків матриці, наведені в табл. 7.

Висновки

У результаті чисельних експериментів було показано, що у тридцяти відсотках випадків модифікований алгоритм квадратичного решета зменшує необхідне число елементів факторної бази та розмір матриці. Час виконання для загального випадку також зменшився на двадцять відсотків. Крім того, кількість вдалих факторизацій збільшилась на один відсоток.

Застосування алгоритму **LB** методу вибору достатньої кількості B-гладких чисел не вимагає отримання $M = L^a + 2$ B-гладких чисел і у випадках їх достатньої кількості немає потреби виконувати подальше просіювання на всьому інтервалі. Це не тільки знижує обчислювальну складність етапу проріджування, але й зменшує розмір матриці, за рахунок чого знову ж таки знижується обчислювальна складність процедури обробки матриці.

1. The quadratic sieve factoring algorithm, C. Pomerance, Advances in Cryptology, Proceedings of Eurocrypt 84, Paris, 1984, T. Beth. N. Cot, and I. Ingemarsson, eds., Lecture Notes in Computer Sci. 209 (1985), P.169-182.
2. The Quadratic Sieve Factoring Algorithm Eric Landquist MATH 488: Cryptographic Algorithms December 14, 2001.
3. Analysis and comparison of some integer factoring algorithms, C. Pomerance, Computational Methods in Number Theory, Part I, H.W. Lenstra, Jr. and R. Tijdeman, eds., Math. Centre Tract 154, Amsterdam, 1982, 89-139.
4. *Carl Pomerance*. Smooth numbers and the quadratic sieve. Algorithmic Number Theory: Lattices, Number Fields, Curves and Cryptography, MSRI Publications, 44: 69-81, 2008.
5. *Song Y. Yan.* Cryptanalytic attacks on RSA / Song Y. Yan – Springer Science and Business Media, Inc. 2008. – P.255.
6. Prime Numbers: a computational perspective, second edition, R. E. Crandall and C. Pomerance, Springer, New York, 2005.
7. *Горбенко И.Д.* Анализ каналов уязвимости системы RSA / И.Д. Горбенко, В.И. Долгов, А.В. Потий, В.Н. Федорченко // Безопасность информации. – 1995. – № 2. – С.22-26.
8. *Daniel R.L. Brown.* Breaking RSA May Be As Difficult As Factoring. – [Электронный ресурс]. Режим доступа: http://www.pgpru.com/novosti/2005/1026vzlomrsabezfaktorizaciirealennoneeffektiven – Название с экрана.

Поступила 2.10.2017р.