УДК 512.7+512.9, 688.321

Р. В. Скуратовський, викладач
МАУП, Інститут комп'ютерних та інформаційних технологій, м. Київ

ФАКТОРИЗАЦІЯ ЦІЛОГО ЧИСЛА ВИГЛЯДУ $n = pq$

Запропонований нами метод факторизації, на відміну від більшості різновидів методу *GNFS* [1, 2] окрім *kGNFS*, володіє всіма властивостями для успішного застосування паралельних обчислень.

Ключові слова: *факторизація числа, паралельні обчислення.*

Вступ. Добре відома задача факторизації числа до сьогодні не розв'язується досить ефективно. Безпека криптосистеми Рабіна, як і RSA, обумовлена складністю факторизації великих чисел.

Основні результати. Нехай $n = pq = 2n_1 + 1$ це RSA-модуль, де n_1 — натуральне. Тоді p та q також мають бути непарними, тобто $p = 2k_1 + 1,\ q = 2l_1 + 1$. Запишемо задачу у вигляді рівняння

$$(2k_1 + 1)(2l_1 + 1) = 2n_1 + 1, \qquad (1)$$

яке спростимо до $2k_1l_1 + k_1 + l_1 = n_1$, та яке можна розглядати за $\bmod 2$ тобто $2k_1l_1 + k_1 + l_1 \equiv n_1 \bmod 2$. Для пошуку розв'язків цього рівняння застосуємо метод конгруентного перебору. Тобто, на кожному кроці ми матимемо ланцюжок гіпотез $H_1, H_2, \ldots, H_n$, у якому кожна гіпотеза залежить від попередніх. Якщо гіпотеза H_n, яка іде в ланцюжку останньою, виявиться неправильною (за припущення правильності всіх попередніх гіпотез), ми пропустимо варіант, протилежний H_n, і продовжимо розв'язання. Якщо неправильними виявилися і гіпотеза H_n, і її протилежність, ми робимо висновок про неправильність гіпотези H_{n-1}. У цьому випадку, ми перестаємо розглядати гіпотезу H_n, а останньою в ланцюжку стає H_{n-1}. Тепер ми змінюємо цю гіпотезу на протилежну до неї і продовжимо пошук. На i-ому кроці матимемо рівняння

$$2^i k_i l_i + a_i k_i + b_i l_i = n_i, \qquad (2)$$

де k_i, l_i — цілі невід'ємні числа, які ми шукаємо, a_i та b_i — натуральні непарні коефіцієнти, а n_i — деяке ціле число. На початку маємо $i = 1, a_1 = b_1 = 1$ та $n_1 = (n-1)/2$.

У випадку, якщо n не просте, для переходу до наступної ітерації ми представляємо шукані числа у вигляді $k_i = 2k_{i+1} + r_i$, $l_i = 2l_{i+1} + s_i$,

тобто r_i, s_i це лишки за модулем 2 чисел k_i, l_i, які задовольняють конгруенцію $2^i k_i l_i + a_i k_i + b_i l_i = n_i (\mathrm{mod}\, 2)$. Зрозуміло, що при $i > 1$ вона рівносильна $a_i k_i + b_i l_i = n_i (\mathrm{mod}\, 2)$. Після скорочення на 2 переходимо до рівняння $2^{i+1} k_{i+1} l_{i+1} + (2^i s_i + a_i) k_{i+1} + (2^i r_i + b_i) l_{i+1} = (n_i - a_i r_i - b_i s_i) : 2 - 2^{i-1} r_i s_i$. Якщо n_i — парне, ми маємо перебрати гіпотези $r_i = s_i = 0$ та $r_i = s_i = 1$. Якщо $n_i \equiv 1 (\mathrm{mod}\, 2)$, розглядаємо випадки $r_i = 0, s_i = 1$ та $r_i = 1, s_i = 0$. Кожний з них приводить до рівняння $2^{i+1} k_{i+1} l_{i+1} + a_{i+1} k_{i+1} + b_{i+1} l_{i+1} = n_{i+1}$, де коефіцієнти на новій ітерації обчислюються за формулами перерахунку

$$\begin{cases} a_{i+1} = 2^i s_i + a_i, \quad b_{i+1} = 2^i r_i + b_i \\ n_{i+1} = \left(n_i - a_i r_i - b_i s_i\right) / 2 - 2^{i-1} r_i s_i. \end{cases} \tag{3}$$

Звідси $2^{i+2} k_{i+2} l_{i+2} + a_{i+2} k_{i+2} + b_{i+2} l_{i+2} = n_{i+2}$.

Так діємо до тих пір поки не буде знайдено точний розв'язок рівняння (2), при цьому виконається умова однієї з ознак зупинки.

Лема 1. Коефіцієнти квадратичної форми (2) a_i, b_i задовольняють конгруенцію $a_i \equiv 1 (\mathrm{mod}\, 2)$, $b_i \equiv 1 (\mathrm{mod}\, 2)$.

Доведення випливає з формули (3) рекурсивного обчислення a_i, b_i і їхніх ініціальних значень $a_1 = b_1 = 1$, з якої випливає, що їхнє значення є сумою парного і не парного чисел, отже є непарним.

Нехай отримано точний розв'язок рівняння (2).

Лема 2. Розв'язки рівняння (1) однозначно отримуються з розв'язками рівняння (2) шляхом замін $k_i = 2k_{i+1} + r_i$, $l_i = 2l_{i+1} + s_i$.

Доведення. З того, що на кожному кроці при переході від i-го зведеного рівняння вигляду (3) до $i-1$ ми робимо рівносильні перетворення вигляду $k_{i-1} = 2k_i + r_i$, $l_{i-1} = 2l_i + s_i$, які однозначно визначають k_{i-1} і l_{i-1} за відомими k_i, l_i і r_i, s_i, і в кінці нами отримано точний розв'язок рівняння (2), нехай він отриманий на ітерації $i = k$ слідує, що ми однозначно отримуємо розв'язок рівняння (1).

Тепер можемо повернутися до знайдених розв'язків конгруенції $2^{i-1} k_{i-1} l_{i-1} + a_{i-1} k_{i-1} + b_{i-1} l_{i-1} = n_{i-1} (\mathrm{mod}\, 2)$, що виникла на попередній ітерації і отримати цифри $i-2$-го розряду для p і q відповідно. Аналогічно повертаючись до вже знайдених розв'язків попередніх конгруенцій ми отримуємо всі цифри (знаки) чисел $p = 2k_1 + 1$ і $q = 2l_1 + 1$. Оскільки ми робимо еквівалентні перетворення, то ми отримаємо однозначно розв'язок рівняння (1) і тобто однозначний запис чисел p і q.

Оскільки ми робили рівносильні перетворення, то відповідно і навпаки за розв'язком рівняння (1), яким є $p = 2k_1 + 1, q = 2l_1 + 1$, однозначно визначається розв'язок рівняння (3). Розв'яками рівняння (1) є числа $p = 2k_1 + 1, q = 2l_1 + 1$ які є нетривіальним дільниками числа n, тому лише одна гілка гіпотез дасть розв'язок в натуральних числах рівняння (2). З парності коефіцієнта при $k_i l_i$ маємо наслідок.

Наслідок. Для переходу до наступної ітерації досить розглядати конгруенцію $a_i k_i + b_i l_i = n_i \pmod 2$.

Зауваження. Оскільки в RSA використовують сильно прості числа, тобто $p = 2P + 1$ і $q = 2Q + 1$, де $P, Q \in \mathbb{P}$, то в цьому випадку лишки $r_2, l_2 \equiv 1 \pmod 2$, де $i \geq 1$ в $k_i = 2k_{i+1} + r_i$, $l_i = 2l_{i+1} + s_i$.

Припустимо, що числа p, q мають різну кількість розрядів у двійковому представленні, тоді довжини відповідних їм чисел k, l, що є розв'язками рівняння (2) теж різні і можна застосувати наступну ознаку зупинки. Зауважимо, що **піддерева пошуку** після першої ітерації повністю симетричні бо рівняння $2k_2 l_2 + k_2 + l_2 = n_2$ є симетричним відносно змінних, тому можна обробляти тільки одне. Тому складність обчислень одразу можна поділити на 2.

Ознака зупинки 1. Якщо в процесі ітеративного розв'язання у рівнянні $2^i k_i l_i + a_i k_i + b_i l_i = n_i$ виникла рівність $a_i = n_i$ чи $b_i = n_i$, то знайдено всі знаки обох чисел p і q.

Доведення. Оскільки зазначена умова є рівністю а не конгруенцією, то на цій ітерації нами знайдено точний розв'язок рівняння $2^i k_i l_i + a_i k_i + b_i l_i = n_i$ і ним є $k_i = 1, l_i = 0$, якщо $a_i = n_i$ чи навпаки $k_i = 0, l_i = 1$ при $b_i = n_i$. Звідси і з того, що перетворення $k_{i-1} = 2k_i + r_i, l_{i-1} = 2l_i + s_i$ однозначно визначають k_{i-1} і l_{i-1} за відомими k_i, l_i і r_i, s_i, знайдемо цифри $i-1$-го розряду на чисел p і q відповідно. Тепер можемо повернутися до знайдених розв'язків конгруенції $2^{i-1} k_{i-1} l_{i-1} + a_{i-1} k_{i-1} + b_{i-1} l_{i-1} = n_{i-1} \pmod 2$, що виникла на попередній ітерації і отримати цифри $i-2$-го розряду для p і q відповідно. Аналогічно повертаючись до вже знайдених розв'язків попередніх конгруенцій ми отримуємо всі цифри (знаки) чисел $p = 2k_1 + 1$ і $q = 2l_1 + 1$. Оскільки ми робимо еквівалентні перетворення, то ми отримаємо однозначно розв'язок рівняння (1) і тобто однозначний запис чисел p і q. Якщо б числа не були розв'язками

рівняння $2^i k_i l_i + a_i k_i + b_i l_i = n_i$, то вони не були б і розв'язками (1). Правильно і навпаки: розв'язками рівняння (2) є розв'язки рівняння (1).

Для визначення довжини гілки графа, де ще може бути розв'язок, слід врахувати добре відомі межі величин збалансованих чисел, бо такі використовуються в криптосистемах RSA і Рабіна.

Прості числа p та q, для яких $n = pq$, будемо називати **збалансованими** [1], якщо $4 < \frac{1}{2}\sqrt{n} < p < \sqrt{n} < q < 2\sqrt{n}$. Зрозуміло, що для практичних застосувань використовують лише збалансовані числа $p < q < 2p$. Тому кількість паралельних кроків алгоритму не більше ніж $\left[\log_2 2\sqrt{n}\right] + 1 = \left[\frac{1}{2}\log_2 \sqrt{2}\, n\right] + 1 = \left[\frac{1}{4} + \frac{1}{2}\log_2 n\right] + 1$.

Наслідок 2. Якщо в рівнянні $2^i k_i l_i + a_i k_i + b_i l_i = n_i$ виконується співвідношення $K_i = (n_i - b_i L_i) : (2^i L_i + a_i) \notin \mathbb{N},\ \forall L_i, K_i \in \mathbb{N}$, де $L_i, K_i < n_i$, то обрана гілка розв'язку вже є хибною і не потребує подальшого розгляду.

Доведення випливає з еквівалентності відсутності розв'язку рівняння (2) при виконанні $K_i = (n_i - b_i L_i) : (2^i L_i + a_i) \in \mathrm{N},\ \forall L_i, K_i \in \mathrm{N}$, де $L_i, K_i < n_i$. Таким чином, завдяки ознакам подільності в двійковій системі числення легко відсікати хибні гілки.

Твердження. Якщо виконується нерівність $2^i k_i l_i > n_i$ починаючи з $i+1$ ітерації, то одна з послідовностей знаків розв'язків $\{k_j\}_{i+1}^n$ $\{l_j\}_{i+1}^n$ складається лише з нулів.

Доведення. Оскільки, найменше значення числа k_i, яке задовольняє ознаку зупинки, що може бути на *i*-тій ітерації має вигляд $k_i = 1\underbrace{0...0}_{i-1}$ (насправді в конкретному випадку маємо точне значення для k_j, $j = 1,...,i$), то для того, щоб виконувалася нерівність $2^i k_i l_i < n_i$ (без виконання якої неможливе виконання ознаки зупинки) слід брати послідовність з нулів, починаючи з *i*+1 розряду у числі k_i чи l_i, тому надалі маємо лише дві можливі послідовності або $\{k_j\}_{i+1}^n$ це послідовність з нулів а її **копослідовність** $\{l_j\}_{i+1}^n$ це префікс довжини *n-i* початку коду числа q. Або навпаки $\{l_j\}_{i+1}^n$ це послідовність з нулів а $\{k_j\}_{i+1}^n$ це пре-

фікс довжини *n-i* з молодших цифр числа p і $2^i k_i l_i > n_i$, тоді розв'язок рівняння $2^i k_i l_i + a_i k_i + b_i l_i = n_i$ вже не належатиме до $\mathbb{N}$.

При цьому зупинка відбулася за $m = \min\{\log p, \log q\}$ кроків, де використано двійкове представлення цих чисел.

Ознака зупинки 2. Якщо у рівнянні $2^i k_i l_i + a_i k_i + b_i l_i = n_i$ при $k_i = l_i = 1$ виконується співвідношення $2^i + a_i + b_i = n_i$, тобто $a_i + b_i + c_i = n_i$, то вже знайдено всі знаки обох чисел p і q.

Доведення. Оскільки у цьому випадку розв'язки p і q мають однакову розрядність, то цифра i-го розряду який є старшим розрядом у їх запису це 1. Тому в силу $a_i, b_i, c_i > 0$ на останній ітерації підстановка $k_i = l_i = 1$ утворює у лівій частині суму $a_i + b_i + c_i = n_i$. Звідси і з $k_i = 2k_{i+1} + r_i$, $l_i = 2l_{i+1} + s_i$, знайдемо цифри i-го розряду на чисел p і q відповідно. Тепер можемо повернутися до знайдених розв'язків конгруенції $2^{i-1} k_{i-1} l_{i-1} + a_{i-1} k_{i-1} + b_{i-1} l_{i-1} = n_{i-1} (\mathrm{mod}\, 2)$, що виникла на попередній ітерації і отримати цифри $i-1$-го розряду для p і q відповідно. Аналогічно діємо повертаючись до вже знайдених розв'язків попередніх рівнянь.

Приклад 1. Маємо $(10k_1 + 1)(10l_1 + 1) = 100011$, що перетворюється на $100k_1 l_1 + k_1 + l_1 = 10001$, застосуємо наслідок 2 для відсікання хибної гілки перевіривши умову $10001 - b_i l_i / (100 l_i + a_i) = 10000 / (101) \notin \mathbb{N}$, де $i = 1$, що можна перевірити без ділення. Тобто умова не виконана, при значеннях $r_2 = 0, s_2 = 1$ з формул заміни $k_1 = 2k_2 + r_1$, $l_1 = 2l_2 + s_1$. Тому залишилась лише підстановка $k_1 = 10k_2 + 1$, $l_1 = 10l_2$, де $r_2 = 1, s_2 = 0$. Маємо $100k_2 l_2 + k_2 + 11l_2 = 1000$ для якої виконується ознака зупинки 2 при $k_2 = 1, l_2 = 1$.

Наслідок 3. Для кожного $n = pq$ застосовна ознака 1 чи 2.

Доведення. Якщо виконується $\log p = \log q$, то застосовна ознака 2, якщо ж $\log p \neq \log q$, то застосовна ознака 1, яка теж є швидко обчислювальною, бо містить лише зсуви і додавання.

Приклад 2. $(10k_1 + 1)(10l_1 + 1) = 100011 = 35$, $100k_1 l_1 + 10k_1 + 10l_1 + 1 = 100011$, звідки $1000k_1 l_1 + 10k_1 + 10l_1 = 100010$, скоротивши на 10 маємо $100k_1 l_1 + k_1 + l_1 = 10001$ нехай $k_1 = 10, k_2 + 1, l_1 = 10l_2$, тоді

$1000k_2l_2 + 10k_2 + 110l_2 + 1 = 10001$ звідси $100k_2l_2 + k_2 + 11l_2 = 1000$. Ознака 2 виконується при $k_3 = 1$, $l_3 = 1$ перетворює рівняння в правильну рівність. Отримуємо розв'язок а саме $p = 111 = 7$, $q = 101 = 5$.

Висновки. З критеріїв зупинки слідує, що при $q < p$ зупинка буде як тільки буде обчислено останній біт числа q, тобто за $\lceil \log_2 q \rceil$ кроків. Отже, висота бінарного дерева вибору потрібних підстановок не більша за $m = \min\{\lceil \log_2 q \rceil, \lceil \log_2 p \rceil\}$. На кожній ітерації відбувається як максимум 4 додавання і 3 зсуви у двійковому представленні числа. Зсув числа робиться всього за 1 такт. Але множення на 2 чи ділення на 2 у двійковій системі числення рівносильне зсуву, що виконується за 1 такт. Позначимо додавання як *D*, зсув як *Z*, тоді маємо $4D + 3Z$ операцій. Це досить швидкі обчислення бо для виконання *D* потрібно лише 2 такти. В кожній вершині бінарного дерева треба зберігати лишок $n_i \equiv x_i (\mathrm{mod}\, 2)$ це 1 біт і $r_i, s_i (\mathrm{mod}\, 2)$, разом це 3 біти. Кожне ядро процесора кластера SunWay MPP [6] має 30 Мб кеш пам'яті (по 2Гб на процесор), тому може зберігати ці лишки і потрібні вказівники. Він має 2^{24} ядер, тому може обчислити в паралельному режимі піддерево, що має на нижньому рівні 2^{23} вершин, як наслідок кількість вершин на верхніх рівнях рівна $2^{23} - 1$. Обчисливши всі лишки в вершинах першого піддерева, яке має ширину 2^{23}, якщо там не знайдено розв'язок, ці лишки витираються і починається обхід інших піддерева, тобто використовуються відкладені обчислення. Окрім того обсяг ОЗУ кластера перевищує витрати пам'яті на зберігання усіх необхідних лишків для відновлення роз'вязку з такого піддерева. Обробити все дерево розв'язків для ключа з 1024 біт можна не більше ніж за $(1024 : 2) : 23$ проходів, тут ділимо на 2, бо у піддеревах з коренями у вершинах v_{11} і у v_{12}. Перший індекс це номер рівня, а другий — номер вершини (нумерація рівнів з 0) множини лишків симетричні, кожен з яких вимагає не більше ніж $512 \cdot 11$ обчислювальних тактів процесора архітектури RISC. Частота ядра кластера 1,45 Ггц, тобто за 1 с $1{,}45 \cdot 10^9$ тактів. Наш алгоритм зовсім не потребує витрат часу на збір розв'язків окремих обчислювальних вузлів у спільний розв'язок всієї задачі, бо потрібна гілка дає розв'язок окремої підзадачі, який є розв'язком усієї задачі.

Список використаних джерел:

1. Орлов В. А., Медведев Н. В., Шимко Н. А., Домрачева А. Б. Теория чисел в криптографии. Издательство МГТУ им. Н. Э. Баумана. 222 с.

2. R Elkenbracht-Huizing «An implementation of the number field sieve» 1996. [citeseer.nj.nec.com/elkenbrach-thuizing96implementa tion.html]
3. Lupu Costică. Methods of solving Diophantine equations in secondary education in Romania. *Science Journal of Education.* 2014. 2(1). P. 22–32.
4. Скуратовський Р. В. Модернізований алгоритм Поліга-Хелмана, Шенкса. *Вісник КНУ імені Тараса Шевченка.* 2015. Том 2. С 63.
5. Николайчук Я. Теоретичні основи виконання модулярних операцій множення в базисі Крестенсона-Радемахера. *Інформатика та математичні методи в моделюванні.* 2011. № 2. С. 123–130.
6. Режим доступу: [http://www.nsccwx.cn/] (це оглядова стаття про 500 кращих кластерів світу за 2016 р.).

Problem of factorization is well known and it still has not solving. All known methods that has subexponential complexity are not destined for parallel implementation. For instance not all variants of *GNFS* [1] can be developed in parallel form. Only *kGNFS* admits parallel implementation. Method of factorization proposed by us has all properties for parallel implementation.

Key words: *factorization of integer number, parallel calculus.*

Одержано 24.02.2017

УДК 519

О. В. Славік, аспірант

Українська інженерно-педагогічна академія, м. Харків

НАБЛИЖЕННЯ ФУНКЦІЙ ДВОХ ЗМІННИХ ЗА ДОПОМОГОЮ ЇХ СЛІДІВ НА СИСТЕМІ НЕПЕРЕТИННИХ СМУГ З КРИВОЛІНІЙНИМИ ГРАНИЦЯМИ

В роботі проведено огляд існуючих методів відновлення пошкоджених цифрових зображень. Запропоновано узагальнений метод інтерстріпації для відновлення зображення поверхні за неповною інформацією про неї у випадку, якщо границі пошкоджених (невідомих) ділянок зображення є криволінійними смугами.

Ключові слова: *зображення, відновлення зображень, інтерстріпація, інтерлінація.*

Вступ. Інколи у файлах, які містять графічну інформацію виявляються дефекти. Оцінка значень втрачених пікселів, у яких відсутня інформація про зображення, необхідна в більшості задач цифрової обробки зображень або, наприклад, у задачах оборки архівних документів у вигляді зображень, що мають різноманітні спотворення (подряпини, плями, пил, непотрібні написи, лінії згину тощо).

У роботі [1] запропоновано метод інтерстріпації для відновлення функції двох змінних у точках між смугами за допомогою інформації