УДК 004.056.55:004.8]

О. О. КУБАЙЧУК, К. С. БУРЯК
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

ПІДХІД ДО КРИПТОАНАЛІЗУ З ВИКОРИСТАННЯМ ЕВОЛЮЦІЙНИХ ОБЧИСЛЕНЬ

Розглянуто можливість застосування еволюційних обчислень у криптоаналізі, зокрема використання генетичних алгоритмів (ГА) для розв'язання задачі факторизації. Актуальність дослідження зумовлена тим, що надійність криптосистеми RSA базується саме на обчислювальній складності задачі факторизації. Оскільки сучасні класичні субекспоненційні алгоритми (наприклад, GNFS, QS) стикаються з критичними труднощами при обробці чисел великої розрядності, виникає потреба в альтернативних евристичних підходах. Головною метою роботи є практична перевірка спроможності ГА вирішувати задачу факторизації.

В основу методології дослідження покладено формалізацію задачі факторизації як задачі комбінаторної оптимізації, де пошук дільників зводиться до мінімізації цільової функції помилки (розглянуто два її варіанти). Вибір ГА як інструменту дослідження зумовлений його математичними властивостями, підтримкою дискретного кодування, а також успішним досвідом застосування до інших задач криптоаналізу. Для підтвердження ефективності методу алгоритм було попередньо протестовано на задачі частотного криптоаналізу шифру простої заміни, де він показав високу точність розшифрування для текстів довжиною понад 200 символів.

Результати практичної факторизації за допомогою ГА подано в порівнянні з класичним методом пробного ділення. Для невеликих чисел (до 10^6) ГА працює стабільно, хоча й поступається у швидкості класичному методу. Для більших чисел (до 10^9) алгоритм демонструє конкурентоспроможність, випереджаючи пробне ділення у 8 % випадків, проте виявляє нестабільність. Для ще більших чисел (до 10^{18}) алгоритм давав хибні результати через обмеження точності стандартного типу float. Використання підвищеної точності (decimal) у середовищі Python усунуло передчасну зупинку алгоритму, але не дозволило ГА знайти розв'язок у межах прийнятного ліміту часу.

Отже, генетичний алгоритм наразі не можна розглядати як повноцінну заміну субекспоненційним алгоритмам для факторизації криптографічно значущих чисел. Проте його застосування як допоміжного евристичного інструменту є цілком доцільним. Попереднє використання ГА дозволяє потенційно звузити простір пошуку перед застосуванням потужних субекспоненційних алгоритмів, таких як GNFS, QS чи ECM.

Ключові слова: *криптоаналіз, еволюційні обчислення, генетичний алгоритм, задача факторизації, шифр простої заміни.*

O. O. KUBAYCHUK, K. S. BURIAK
National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute"

CRYPTANALYSIS APPROACH BASED ON EVOLUTIONARY COMPUTATION

This study explores the feasibility of applying evolutionary computation in cryptanalysis, specifically the use of genetic algorithms (GA) to solve the integer factorization problem. The relevance of this research is driven by the fact that the security of the RSA cryptosystem relies heavily on the computational complexity of the factorization problem. Since modern classical subexponential algorithms (e.g., GNFS, QS) face critical difficulties when processing large-scale numbers, there is a need for alternative heuristic approaches. The primary objective of this work is to practically verify the capability of GA in solving the factorization problem.

The research methodology is based on formalizing the factorization problem as a combinatorial optimization problem, where the search for divisors is reduced to minimizing an objective error function (two variants of which are considered). The selection of GA as the research tool is justified by its mathematical properties, support for discrete representation, and successful prior application to other cryptanalysis tasks. To validate the method's effectiveness, the algorithm was preliminarily tested on the frequency cryptanalysis of a simple substitution cipher, demonstrating high decryption accuracy for ciphertexts exceeding 200 characters in length.

The results of practical factorization using GA are presented and compared with the classical trial division method. For small numbers (up to 10^6), the GA operates stably, although it is slower than the classical method. For larger numbers (up to 10^9), the algorithm demonstrates competitiveness, outperforming trial division in 8 % of cases; however, it exhibits instability. For even larger numbers (up to 10^{18}), the algorithm produced incorrect results due to the precision limitations of the standard float data type. Implementing higher precision (decimal) in the Python environment prevented premature algorithm termination but did not enable the GA to find a solution within an acceptable time limit.

https://doi.org/10.32782/mathematical-modelling/2026-9-1-18 ISSN 2618-0332 (print), 2618-0340 (online)

Consequently, the genetic algorithm cannot currently be considered a full-fledged replacement for subexponential algorithms when factoring cryptographically significant numbers. However, its application as an auxiliary heuristic tool is entirely viable. The preliminary use of GA can potentially narrow the search space prior to the application of powerful subexponential algorithms, such as GNFS, QS, or ECM.

Keywords: *cryptanalysis, evolutionary computation, genetic algorithm, integer factorization problem, simple substitution cipher.*

Постановка проблеми

Евристичні алгоритми, що імітують механізми біологічної еволюції, називаються еволюційними, а їх застосування – еволюційними обчисленнями. Еволюційні алгоритми становлять важливий підклас методів оптимізації, зокрема комбінаторної.

Оптимізаційні задачі природним чином виникають у багатьох галузях інформаційної безпеки, зокрема у криптоаналізі. Криптоаналіз як наука має на меті подолання криптографічних систем захисту інформації. Криптоаналітичні атаки поділяють на аналітичні, що використовують математичні та структурні вразливості алгоритму, і атаки повного перебору. Сучасні криптографічні системи захисту інформації проектуються таким чином, що метод повного перебору стає обчислювально недосяжним через експоненційне зростання розмірності простору пошуку. Застосування ж евристичних методів дозволяє знаходити субоптимальні або точні розв'язки за прийнятний час, трансформуючи криптоаналітичну задачу в задачу оптимізації.

Аналіз останніх досліджень і публікацій

Евристичні методи набули популярності на межі 60–70-х років минулого сторіччя. Хронологічно інтерес до застосування евристик у криптографії мав хвилеподібний характер. Перша хвиля, започаткована А. Кларком, завершилася на початок 2000-х і представлена роботами Р. Метьюза [1], Р. Спіллмана [2; 3], А. Кларка та Е. Доусона [4; 5]. У роботах того періоду досліджується застосування евристик, зокрема генетичного алгоритму, до криптоаналізу класичних шифрів. Наступна хвиля завершилася близько 2015 року і корисні застосування еволюційних обчислень для криптоаналізу як симетричних, так і асиметричних криптосистем можна знайти, наприклад, у [6]. У контексті криптоаналізу еволюційні методи, зокрема генетичні алгоритми (ГА), розглядаються як перспективний інструмент для дослідження величезних просторів пошуку, де традиційні методи стикаються з субекспоненційним або експоненційним зростанням складності.

Остання хвиля триває дотепер і представлена роботами [7–10], що відображають вплив еволюційних обчислень на проблеми криптографії. В роботі [11] досліджується вплив вибору функції пристосованості для деяких задач криптоаналізу у випадку використання ройових алгоритмів оптимізації. Робота [12] є спробою узагальнення досягнень евристичного підходу до розв'язання задач криптографії, зокрема задачі факторизації.

Мета дослідження

Метою дослідження є формалізація задач криптоаналізу як задач оптимізації та застосування еволюційних обчислень, як інструменту їх розв'язання.

Виклад основного матеріалу дослідження

Основні поняття, означення та алгоритми

1. Задача факторизації полягає у зображенні складеного цілого числа добутком менших цілих чисел. За основною теоремою арифметики, будь-яке натуральне число $N > 1$ однозначно (з точністю до порядку множників) можна подати у вигляді добутку:

$$N = p_1^{k_1} p_2^{k_2} \dots p_r^{k_r},$$

де $p_1, p\}, \dots, p_r$ – прості числа, $k_1, k_2, \dots, k_r$ – натуральні числа.

https://doi.org/10.32782/mathematical-modelling/2026-9-1-18 ISSN 2618-0332 (print), 2618-0340 (online)

Для криптографії важливим є окремий випадок. Дано натуральне число N, яке є добутком двох невідомих великих простих чисел $N = pq$. Необхідно знайти множники p і q.

Серед класичних методів (алгоритмів) факторизації виділяють *спеціальні*, які є ефективними лише за певних умов (наприклад, якщо один із дільників малий або множники мають специфічну структуру) та *універсальні*. Для кожного з методів можна говорити про рекомендовані межі (наприклад, розрядність числа чи спеціальні умови на співмножники) їх застосування.

Для запису складності алгоритмів обчислювальної теорії чисел зручно використовувати L-нотацію. Функція Ленстри є залежністю від $\ln N$, тобто залежністю від розрядності (довжини) числа N:

$$L_N[\alpha, c] = \exp\{(c + o(1))(\ln N)^{\alpha}(\ln \ln N)^{1-\alpha}\},$$

де $c > 0$, $0 \le \alpha \le 1$.

Для $\alpha = 0$ функція Ленстри $L_N[0, c] = (\ln N)^{(c + o(1))}$, і є поліномом від $\ln N$.

Для $\alpha = 1$ функція Ленстри $L_N[1, c] = N^{(c + o(1))}$, і є експонентою від $\ln N$.

Для $0 < \alpha < 1$ функція Ленстри є субекспоненціальною, тобто зростає повільніше експоненти з основою більше 1.

Класифікацію та мінімальні рекомендації щодо застосування історичних та сучасних алгоритмів факторизації наведено у таблиці 1.

З таблиці 1, зокрема випливає необхідність пошуку альтернативних шляхів до розв'язання задачі факторизації, які не спираються безпосередньо на математичну структуру числа, а орієнтуються на дослідження простору можливих рішень. Розгляд факторизації як задачі оптимізації відкриває можливість застосування адаптивних і стохастичних методів пошуку, серед яких особливе місце займають еволюційні обчислення.

Застосування класичних методів є обмеженим для чисел дуже великої розрядності. Основною проблемою є стрімке зростання обчислювальної складності зі збільшенням довжини вхідних даних, що робить факторизацію криптографічно значущих чисел практично недосяжною в межах прийнятного часу [13].

Відмітимо, що субекспоненційні алгоритми є потужним знаряддям для зламу RSA-систем, але їх реалізація потребує значних обчислювальних потужностей, зокрема, великих обсягів пам'яті та спеціального програмного забезпечення [14].

Триває пошук альтернативних підходів, орієнтованих на адаптивний та евристичний пошук розв'язків, зокрема, до еволюційних обчислень.

Сформулюємо задачу факторизації, як задачу комбінаторної оптимізації. Процес факторизації числа N, наприклад, можна представляти як пошук оптимальної конфігурації послідовностей бітів двійкових зображень цілих десяткових чисел.

Нехай x та y – множники у двійковому вигляді. Простір пошуку S – множина всіх двійкових рядків довжиною $\left\lfloor \frac{1}{2}\log_2 N \right\rfloor + 1$. Цільова функція: $f(x, y) = |N - xy|$. Оптимальний розв'язок: пара (p, q), така, що $f(p, q) = 0$.

В даній роботі розглядається дещо простіша задача. Ми спробуємо шукати дільники числа N на відрізку $[a, b]$, a і b натуральні числа, $1 < a, b < \sqrt{N}$. Цільовою функцією обираємо $f(x) = \{N/x\}$. Оптимальний розв'язок: число p, таке, що $f(p) = 0$.

Інструментом для розв'язання поставленої задачі обрано класичний евристичний алгоритм комбінаторної оптимізації –генетичний алгоритм (ГА).

2. Частотний криптоаналіз з ГА. Для сучасних шифрів типу AES чи DES застосування ГА є недоцільним, оскільки розробники подібних шифрів дотримувались вимог конфузії та дифузії. За таких умов ГА вироджується у звичайний випадковий перебір.

Класичні шифри, наприклад, Цезаря, Віженера, простої заміни (підстановки) зберігають статистичні властивості вихідної мови, тобто є вразливими для частотного аналізу. Для шифру

https://doi.org/10.32782/mathematical-modelling/2026-9-1-18 ISSN 2618-0332 (print), 2618-0340 (online)

Таблиця 1

Класифікація та мінімальні рекомендації щодо застосування історичних та сучасних алгоритмів факторизації

Метод	Складність	Тип	Оптимальне застосування
Пробне ділення	$L_N[1, 1/2]$	експоненційний	Цілі числа до 60 біт; числа з малими простими множниками, в тому числі для попередньої підготовки до застосування потужних алгоритмів типу GNFS.
Метод Ферма	$L_N[1, c]$	експоненційний	Метод Ферма стає ефективним лише тоді, коли різниця між p та q мала. У цьому випадку він працює за поліноміальний час
-метод Полларда	$L_N[1, 1/4]$	експоненційний	Числа до 70 біт.. Метод базується на «парадоксі днів народження», є універсальним. Використовують перед застосуванням QS та GNFS
$(p-1)$-метод Полларда	$L_N[1, c]$	експоненційний	Залежить не від розміру самого числа, а від властивостей його дільників. Метод є ефективним за умови, що число $(p-1)$ є дуже гладким для дільника p
Метод еліптичних кривих (ECM)	$L_p[1/2, 2]$	субекспоненційний	Швидкість залежить від розміру найменшого простого дільника p. Підходить для пошуку дільників від 70 до 216 біт. Застосовують перед QS та GNFS
Квадратичне решето (QS)	$L_N[1/2, 1]$	субекспоненційний	Універсальний метод, який показує гарні результати для чисел від 160 до 360 біт. Значно легший для практичної реалізації від GNFS. Використовується для зламу слабких ключів RSA, які часто зустрічаються в IoT-пристроях
Загальний метод решета числового поля (GNFS)	$L_N[1/3, \sqrt[3]{64/9}]$	субекспоненційний	Універсальний метод, що має найкращу асимптотичну складність серед відомих методів в загальному випадку. Має складний етап підготовки. Оптимальний діапазон чисел від 360 до 400 біт. Основний інструмент для атак на RSA-768, RSA-829 та RSA-1024

простої заміни існує $\alpha!$ можливих ключів, де α –потужність алфавіту певної мови, тому повний перебір ключів є малоефективним. Змоделюємо частотний криптоаналіз шифру простої заміни за допомогою ГА.

Результати моделювання

1. Моделювання частотного криптоаналізу шифру простої заміни з ГА.

Всі обчислення проводились на ПК з процесором AMD A6-9225 RADEON R4, 5 COMPUTE CORES 2C+3G, 2,6 GHz. Google Colaboratory (Colab) обрано середовищем виконання Python-коду.

Задача. Нехай відкрите повідомлення є повідомленням англійською мовою. Згенерувати секретний ключ і виконати шифрування простої заміни на цьому ключі. Провести частотний криптографічний аналіз шифру простої заміни з використанням генетичного алгоритму за шифротекстом.

Примітка. Хромосомами в задачі є перестановки символів обраного алфавіту. Для оцінки якості розшифрування на згенерованому ГА ключі-кандидаті, обрано бібліотечну функцію calculate_fitness. Для якісного частотного аналізу використано таблиці n-грам, $1 \le n \le 4$. Також використовувалась штрафна таблиця, для швидкого вилучення комбінацій нехарактерних для обраної мови.

https://doi.org/10.32782/mathematical-modelling/2026-9-1-18 ISSN 2618-0332 (print), 2618-0340 (online)

Відкрите повідомлення: “GENETIC ALGORITHMS ARE EFFECTIVE TOOLS IN THE CRYPTANALYSIS OF SIMPLE SUBSTITUTION CIPHERS, AS THEY FIND OPTIMAL KEYS BY SIMULATING NATURAL SELECTION. THEY GENERATE A POPULATION OF POSSIBLE DECRYPTIONS, EVALUATE THEIR FITNESS USING FREQUENCY ANALYSIS, AND APPLY CROSSOVER AND MUTATION OPERATIONS. THIS PROCESS GRADUALLY IMPROVES SOLUTIONS, ENABLING AUTOMATED CIPHER BREAKING EVEN WITHOUT PRIOR KEY KNOWLEDGE.”.

Результат розшифрування: “GENETIC ALGOFITHPS AFE EBBECTIME TOOLS IN THE CFRUTANALRSIS OB SIPULE SYKSTITYTION CIUHEFS, AS THER BIND OUTIPAL VERS KR SIPYLATING NATYFAL SELECTION. THER GENEFATE A UOUYLATION OB UOSSIKLE DECFRUTIONS, EMALYATE THEIF BITNESS YSING BFEJYENCR ANALRSIS, AND AUULR CFOSSOMEF AND PYTATION OUEFATIONS. THIS UFOCESS GFADYALLR IPUFOMES SOLYTIONS, ENAKLING AYTOPATED CIUHEF KFEAVING EMEN WITHOYT UFIOF VER VNOWLEDGE.”.

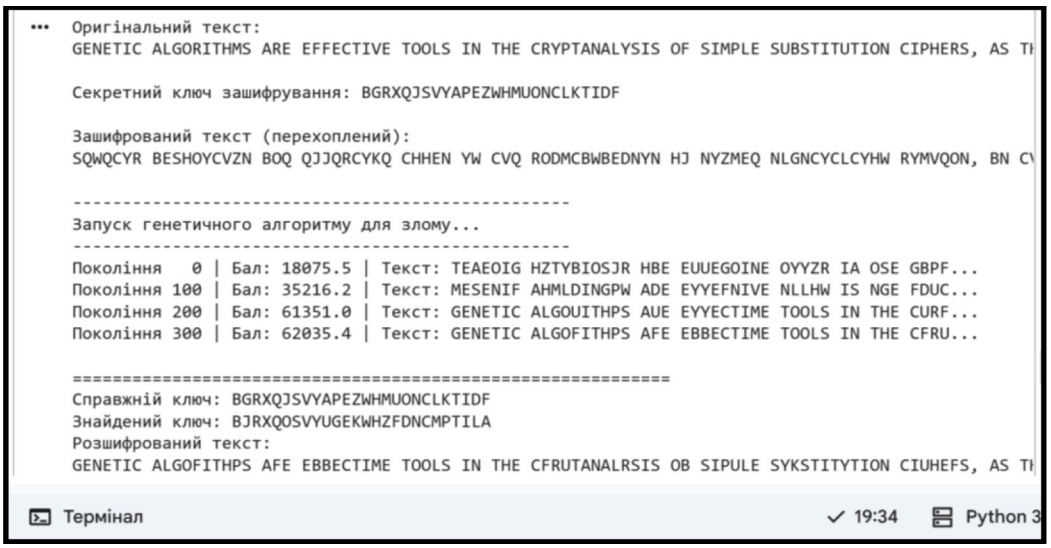

Рис. 1. Результат частотного криптоаналізу з ГА

Результати моделювання. Для прийнятної точності розшифрування потрібен перехоплений текст довжиною більше 200 символів. Генетичний алгоритм показав найкращі результати для таких параметрів: ймовірність мутації 0,2 (рекомендоване значення для ГА), число особин у популяції більше 150, число поколінь близько 300. Застосування тільки моно і біграм є недостатнім для адекватного розшифрування. Врахування нехарактерних для обраної мови комбінацій символів значно покращує результати аналізу (рис. 1).

З врахуванням обмежень, що викладені вище, застосування генетичного алгоритму для частотного криптоаналізу шифру простої заміни є виправданим.

2. Моделювання факторизації з використанням ГА. Всі обчислення проводились на ПК з процесором AMD A6-9225 RADEON R4, 5 COMPUTE CORES 2C+3G, 2,6 GHz. Google Colaboratory (Colab) обрано середовищем виконання Python-коду.

Дослідження проводилося у два етапи. На кожному швидкість роботи ГА порівнювалась зі швидкістю методу пробного ділення, який показує прийнятні результати для цілих чисел довжиною до 60 біт, приблизно до 18 десяткових розрядів.

Етап 1. Розклад на прості множники.

Цільова функція: $f(x, y) = |N - |$. Оптимальний розв’язок: пара (p, q), така, що $f(p, q) = 0$. Досліджувалось одразу дві реалізації ГА: хромосоми –цілі десяткові числа та хромосома –відповідна двійкова послідовність. Для наглядності реалізовувався метод пробного ділення.

https://doi.org/10.32782/mathematical-modelling/2026-9-1-18 ISSN 2618-0332 (print), 2618-0340 (online)

Результати першого етапу.

Для $10^4 < N < 10^6$ всі реалізовані методи працюють досить стабільно. Варіант з бінарними хромосомам працює дещо повільніше.

Для (рис. 2) метод пробного ділення працює стабільно, а от обидві реалізації ГА працюють уже нестабільно, хоча інколи показують кращий час. Бінарний код працює уже дещо краще ніж цілочисельна реалізація ГА

```
--- Факторизація числа N = 112747741 ---

[Пробне ділення] Результат: [10247, 11003], Час: 0.001857 сек
[ГА: Цілі числа] Результат: None, Час: 0.220421 сек
[ГА: Бінарний код] Результат: (11003, 10247), Час: 0.001644 сек
```

Рис. 2. Етап 1, рівень $10^7 < N < 10^9$

Для $10^9 < N < 10^{18}$ метод пробного ділення працює повільно, обидві реалізації ГА не показали позитивного результату.

Висновки щодо параметрів генетичного алгоритму. Збільшення числа особин більше 6-ти не покращує результати обчислень. Оптимальним значенням параметра ймовірності мутації є 0,2. Число допустимої кількості популяцій лежало в межах від 1000 до 100 000.

Етап 2. Пошук дільника на заданому інтервалі.

Цільова функція $f(x) = \{N/x\}$. Необхідно обчислити дільник числа N на відрізку на відрізку $[a, b]$, a і b натуральні числа, $1 < a, b < \sqrt{N}.$,. Оптимальний розв'язок: число p, таке, що $f(p) = 0$. На даному етапі досліджувався бінарний варіант генетичного алгоритму у порівнянні з методом пробного ділення. Параметри генетичного алгоритму були близькими до тих, що використовувались на етапі 1.

Для $10^4 < N < 10^6$ всі реалізовані методи працюють досить стабільно. ГА працює в 2–4 рази повільніше ніж метод пробного ділення.Також відмітимо, що у той час, коли пробне ділення завжди знаходить мінімальний дільник, ГА може знайти більший дільник.

Для $10^7 < N < 10^9$ (рис. 3) метод пробного ділення працює стабільно, а от ГА працює уже нестабільно, хоча приблизно у 8 % випадків показує час, кращий ніж у методі пробного ділення.

```
Тестування для N = 112747741
Інтервал пошуку: [3, 11000]
==========================================
Запуск: Пробне ділення...
Результат: (10247, 11003)
Час: 0.000333 секунд

Запуск: Генетичний алгоритм...
Результат: (10247, 11003)
Час: 0.000287 секунд

Висновок: Пробне ділення швидше у 0.9 разів.
```

Рис. 3. Етап 2, рівень $10^7 < N < 10^9$

Для $10^9 < N < 10^{18}$ (рис. 4) метод пробного ділення працює повільно. Генетичний алгоритм завершується швидше, але результати «майже вірні». ГА розкладає число 717 418 696 464 506 927 на множники

$$504\,474\,874 \times 1\,422\,109\,866 = 717\,418\,695\,464\,506\,884.$$

https://doi.org/10.32782/mathematical-modelling/2026-9-1-18 ISSN 2618-0332 (print), 2618-0340 (online)

Причиною цього є недостатня точність обчислень за замовчуванням *float* у Python. На рівні $N \approx 10^{18}$ генетичний алгоритм уже не відрізняє похибку фітнес функції від нуля.

```
Тестування для N = 717418695464506927
Інтервал пошуку: [3, 850000000]
================================================
Запуск: Пробне ділення...
Результат: (800000011, 896773357)
Час: 77.794240 секунд

Запуск: Генетичний алгоритм...
Результат: (504474874, 1422109866)
Час: 36.824909 секунд

Висновок: Пробне ділення швидше у 0.5 разів.
```

Рис. 4. Етап 2, рівень $10^9 < N < 10^{18}$

Далі Python-код було модифіковано для обчислень з точністю *decimal*.

На швидкість пробного ділення це прогнозовано не вплинуло, але ГА перестав збігатися, принаймні за час до 400 секунд.

Висновок до задачі факторизації. Генетичний алгоритм непогано себе показав для чисел порядку до 10^9. Для діапазону $10^7 < N < 10^9$ цей метод часто мав перевагу перед пробним діленням. Алгоритм недоцільно застосовувати для криптографічно значущих чисел, але його цілком можливо використовувати для пошуку і відсіву малих дільників перед застосуванням субекспоненціальних методів типу QS чи GNFS.

Висновки

У результаті проведеного дослідження було здійснено комплексний аналіз застосування еволюційних обчислень, зокрема генетичних алгоритмів (ГА), для розв'язання задачі факторизації цілих чисел, що є фундаментальною проблемою для сучасної криптографії.

Задача факторизації великих чисел, на якій базується стійкість алгоритму RSA, належить до класу обчислювально важких задач, для яких на сьогодні не відомі поліноміальні алгоритми на класичних ЕОМ. Використання генетичного алгоритму як інструменту комбінаторної оптимізації дозволяє трансформувати пошук дільників у задачу мінімізації цільової функції помилки. У роботі розглянуто два варіанти такої функції.

ГА має низку переваг перед іншими еволюційними стратегіями завдяки природній підтримці дискретного (бінарного) кодування хромосом, що відповідає структурі множників, та гнучкості операторів схрещування й мутації.

Одним з результатів роботи є підтвердження здатності застосування ГА для частотного криптоаналізу шифру простої заміни. Моделювання процесу криптоаналізу в середовищі Python довело, що ГА є цілком спроможним у випадку класичного шифру підстановки. При використанні популяції понад 150 особин, ймовірності мутації 0.2, врахуванні частот 1-4-грам та при застосуванні штрафних таблиць алгоритм продемонстрував високу точність розшифрування текстів довжиною понад 200 символів.

Головною метою роботи була практична перевірка спроможності еволюційних алгоритмів, на прикладі ГА, забезпечити розв'язання задачі факторизації. Моделювання процесу криптоаналізу також відбувалося в середовищі Python, причому обчислення проводилися на фоні методу пробного ділення. Результати експерименту свідчать про вибіркову спроможність ГА, зокрема, залежно від розрядності числа, що факторизується. Наприклад, для чисел до 10^6 класичне пробне ділення залишається швидшим, хоча ГА стабільно знаходить розв'язок.

https://doi.org/10.32782/mathematical-modelling/2026-9-1-18 ISSN 2618-0332 (print), 2618-0340 (online)

У діапазоні $10^7 < N < 10^9$ генетичний алгоритм демонструє конкурентоспроможність, у 8 % випадків випереджаючи метод пробного ділення, проте виявляє меншу стабільність. При переході до чисел порядку 10^{18} алгоритм стикається з проблемою точності обчислень фітнес-функції через обмеження стандартного типу float у Python, що призводить до отримання наближених, але не коректних результатів. Застосування обчислень з точністю decimal вирішує проблему передчасної збіжності, але при цьому виявляє неспроможність ГА знаходит дільник, принаймні за декілька хвилин, у випадку великих чисел.

Отже, експериментально встановлено, що генетичний алгоритм у класичному вигляді не є заміною для потужних субекспоненціальних методів. Проте ГА доцільно використовувати як допоміжний евристичний інструмент для швидкого відсіву малих дільників або для дослідження обмежених інтервалів простору пошуку перед застосуванням алгоритмів типу QS чи GNFS. Таким чином, еволюційні обчислення розширюють арсенал криптоаналітичних засобів, пропонуючи адаптивні підходи до вирішення складних математичних проблем.

Список використаної літератури

1. Matthews R. A. The Use of Genetic Algorithms in Cryptanalysis. *Cryptologia*. 1993. Vol. 17. P. 187–201. DOI: https://doi.org/10.1080/0161-119391867863
2. Spillman R., Janssen M., Nelson B., Kepner M. Use Of A Genetic Algorithm In The Cryptanalysis Of Simple Substitution Ciphers. *Cryptologia*. 1993. Vol. 17, iss. 1. P. 31–44. DOI: https://doi.org/10.1080/0161-119391867746
3. Spillman R. Cryptanalysis Of Knapsack Ciphers Using Genetic Algorithms. *Cryptologia*. 1993. Vol. 17, iss. 4. P. 367–377. DOI: https://doi.org/10.1080/0161-119391867999
4. Clark A., Dawson E. A Parallel Genetic Algorithm For Cryptanalysis Of The Polyalphabetic Substitution Cipher. *Cryptologia*. 1997. Vol. 21, iss. 2. P. 129–138. DOI: https://doi.org/10.1080/0161-119791885850
5. Clark A., Dawson E. Optimisation Heuristics for the Automated Cryptanalysis of Classical Ciphers. *Journal of Combinatorial Mathematics and Combinatorial Computing*. 1998. Vol. 28. P. 63–86.
6. Boryczka U., Dworak K. Genetic Transformation Techniques in Cryptanalysis. *In*: Nguyen N. T., Attachoo B., Trawiński B., Somboonviwat K. (eds). (*Intelligent Information and Database Systems* ACIIDS'2014 : Proceedings 6th Asian Conference, Bangkok, Thailand, April 7–9, 2014). Lecture Notes in Computer Science. 2014. Vol. 8398. P. 147–156. Cham : Springer. https://doi.org/10.1007/978-3-319-05458-2_16
7. Forhad M. S., Hossain M. S., Rahman M. O., Rahaman M., Haque M. M., Patwary M. K. An improved fitness function for automated cryptanalysis using genetic algorithm. *Indonesian Journal of Electrical Engineering and Computer Science*. 2019. Vol. 13. № 2. P. 643–648. DOI: https://doi.org/10.11591/ijeecs.v13.i2.pp643-648
8. Rachmawati D., Tamara H., Sembiring S., Budiman M. RSA public key solving technique by using genetic algorithm. *Journal of Theoretical and Applied Information Technology*. 2020. Vol. 98, no. 15. P. 2990–2999. URL : https://www.jatit.org/volumes/ninetyeight15.php
9. Sabonchi A. K. S., Akay B. Cryptanalysis of polyalphabetic cipher using differential evolution algorithm. *Tehnički vjesnik*. 2020. Vol. 27, no. 4. P. 1101–1107. DOI: https://doi.org/10.17559/TV-20190314095054
10. Mobin M. A., Kamrujjaman M. Cryptanalysis of RSA Cryptosystem: Prime Factorization using Genetic Algorithm. *International Journal of Intelligent Systems and Applications in Engineering*. 2024. Vol. 12, no. 1s. P. 456–468. DOI: https://doi.org/10.48550/arXiv.2407.05944
11. Кубайчук О. О. Особливості застосування алгоритму ACO до деяких задач криптоаналізу. *Наукоємні технології*. 2023. № 2(58). С. 141–148. DOI: https://doi.org/10.18372/2310-5461.58.17650

12. Кубайчук О. О. Огляд застосування метаевристичного підходу в криптоаналізі. *Вісник Херсонського національного технічного університету*. 2023. № 2(85). С. 147–153. DOI: https://doi.org/10.35546/kntu2078-4481.2023.2.20
13. Crandall R., Pomerance C. Prime numbers: A computational perspective. 2nd ed. New York : Springer, 2005. 597 p. DOI: https://doi.org/10.1007/0-387-28979-8
14. Eiben A. E., Smith J. E. Introduction to evolutionary computing. 2nd ed. Berlin : Springer, 2015. 287 p. DOI: https://doi.org/10.1007/978-3-662-44874-8

References

1. Matthews, R. A. (1993). The use of genetic algorithms in cryptanalysis. *Cryptologia*, *17*, 187–201. https://doi.org/10.1080/0161-119391867863 [in English].
2. Spillman, R., Janssen, M., Nelson, B., & Kepner, M. (1993). Use of a genetic algorithm in the cryptanalysis of simple substitution ciphers. *Cryptologia*, *17* (1), 31–44. https://doi.org/10.1080/0161-119391867746 [in English].
3. Spillman, R. (1993). Cryptanalysis of knapsack ciphers using genetic algorithms. *Cryptologia*, *17* (4), 367–377. https://doi.org/10.1080/0161-119391867999 [in English].
4. Clark, A., & Dawson, E. (1997). A parallel genetic algorithm for cryptanalysis of the polyalphabetic substitution cipher. *Cryptologia*, *21* (2), 129–138. https://doi.org/10.1080/0161-119791885850 [in English].
5. Clark, A., & Dawson, E. (1998). Optimisation heuristics for the automated cryptanalysis of classical ciphers. *Journal of Combinatorial Mathematics and Combinatorial Computing*, *28*, 63–86. [in English].
6. Boryczka, U., Dworak, K. (2014). Genetic Transformation Techniques in Cryptanalysis. *In*: Nguyen, N. T., Attachoo, B., Trawiński, B., Somboonviwat, K. (eds). (*Intelligent Information and Database Systems* ACIIDS'2014 : Proceedings 6th Asian Conference, Bangkok, Thailand). *Lecture Notes in Computer Science*. Cham : Springer. https://doi.org/10.1007/978-3-319-05458-2_16 [in English].
7 Forhad, M. S., Hossain, M. S., Rahman, M. O., Rahaman, M., Haque, M. M., & Patwary, M. K. (2019). An improved fitness function for automated cryptanalysis using genetic algorithm. *Indonesian Journal of Electrical Engineering and Computer Science*, *13* (2), 643–648. https://doi.org/10.11591/ijeecs.v13.i2.pp643-648 [in English].
8. Rachmawati, D., Tamara, H., Sembiring, S., & Budiman, M. (2020). RSA public key solving technique by using genetic algorithm. *Journal of Theoretical and Applied Information Technology*, *98* (15), 2990–2999. URL : https://www.jatit.org/volumes/ninetyeight15.php [in English].
9. Sabonchi, A. K. S., & Akay, B. (2020). Cryptanalysis of polyalphabetic cipher using differential evolution algorithm. *Tehnički vjesnik*, *27* (4), 1101–1107. https://doi.org/10.17559/TV-20190314095054 [in English].
10. Mobin, M. A., & Kamrujjaman, M. (2024). Cryptanalysis of RSA cryptosystem: Prime factorization using genetic algorithm. *International Journal of Intelligent Systems and Applications in Engineering*, *12* (1s), 456–468. https://doi.org/10.48550/arXiv.2407.05944 [in English].
11. Kubaichuk, O. O. (2023). Osoblyvosti zastosuvannia alhorytmu ASO do deiakykh zadach kryptoanalizu [Features of applying the ASO algorithm to some cryptanalysis problems]. *Naukoiemni tekhnolohii*, [*Scientific technologies*]. *2* (58), 141–148. https://doi.org/10.18372/2310-5461.58.17650 [in Ukrainian].
12. Kubaichuk, O. O. (2023). Ohliad zastosuvannia metaevrystychnoho pidkhodu v kryptoanalizi [A Review of the Application of Metaheuristic Approach in Cryptanalysis]. *Visnyk Khersonskoho natsionalnoho tekhnichnoho universytetu* [*Bulletin of Kherson National Technical University*], *2* (85), 147–153. https://doi.org/10.35546/kntu2078-4481.2023.2.20 [in Ukrainian].

13. Crandall, R., & Pomerance, C. (2005). *Prime numbers: A computational perspective* (2nd ed.). New York : Springer. https://doi.org/10.1007/0-387-28979-8 [in English].
14. Eiben, A. E., & Smith, J. E. (2015). *Introduction to evolutionary computing* (2nd ed.). Berlin : Springer. https://doi.org/10.1007/978-3-662-44874-8 [in English].

Кубайчук Оксана Олексіївна – к.ф.-м.н., доцент, доцент кафедри математичного аналізу та теорії ймовірностей Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». E-mail: o.kubaychuk@gmail.com, ORCID: 0000-0002-5135-3688.

Буряк Кирил Сергійович – магістрант Спеціальної кафедри № 1 Інституту спеціального зв'язку та захисту інформації Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». E-mail: kirill.buriy123@gmail.com, ORCID: 0009-0001-6010-3197.

Kubaychuk Oksana Oleksiivna – Candidate of Physical and Mathematical Sciences, Associate Professor, Associate Professor at the Department of Mathematical Analysis and Probability Theory of the National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute". E-mail: o.kubaychuk@gmail.com, ORCID: 0000-0002-5135-3688.

Buriak Kyryl Serhiiovych – Master's Student at Special Department № 1 of the Institute of Special Communications and Information Protection of the National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute". E-mail: kirill.buriy123@gmail.com, ORCID: 0009-0001-6010-3197.

Дата першого надходження статті до видання: 27.03.2026
Дата прийняття статті до друку після рецензування: 07.05.2026
Дата публікації (оприлюднення) статті: 01.07.2026