https://doi.org/10.31891/2307-5732-2026-363-71
УДК 004.8:519.854

РОМАНЧУК ОЛЕКСАНДР
Український державний університет науки і технологій
https://orcid.org/0000-0003-2623-350X
e-mail: o.o.romanchuk@ust.edu.ua
ЛЕВЧУК ІГОР
Український державний університет науки і технологій
https://orcid.org/0000-0002-8983-0558
e-mail: i.l.levchuk@ust.edu.ua
ГУЗЬ ГАННА
Український державний університет науки і технологій
https://orcid.org/0009-0002-2908-8985
e-mail: h.m.huz@ust.edu.ua
ДУБОВИК ТЕТЯНА
Український державний університет науки і технологій
https://orcid.org/0009-0003-0614-8700
e-mail: t.m.dubovik@ust.edu.ua

РОЗРОБКА ГЕНЕТИЧНОГО АЛГОРИТМУ ДЛЯ ЗНАХОДЖЕННЯ НАБЛИЖЕНИХ РОЗВ'ЯЗКІВ ЗАДАЧІ КОНТАКТНОГО ЧИСЛА

Задача контактного числа є однією з актуальних проблем комбінаторної оптимізації та дискретної геометрії, що має важливе теоретично-прикладне значення. У статті визначено актуальність цієї задачі та обґрунтовано необхідність застосування наближених обчислювальних методів для її дослідження в евклідових просторах довільної вимірності. Крім того, геометричне формулювання задачі тісно пов'язане з аналізом взаємного розташування векторів і кутових обмежень між ними, а тому у статті проаналізовано подання задачі контактного числа через α-допустимі системи векторів та встановлено зв'язок між цими системами і конфігураціями дотичних куль.

Обмеженість класичних (традиційних) аналітичних методів зумовлює пошук альтернативних підходів щодо розв'язання обумовленої задачі. У статті доведено можливість та ефективність використання ε-близьких систем векторів із цілочисловими координатами для знаходження наближених розв'язків. Сучасні поширені еволюційні методи дозволяють ефективно досліджувати великі простори допустимих конфігурацій. У статті запропоновано генетичний алгоритм, у якому критерієм оптимальності є мінімізація максимального косинуса кута між векторами системи. У даному випадку, чималого значення набуває обчислювальна реалізація у практичному застосуванні запропонованого підходу. До того ж, у статті досліджено програмну реалізацію алгоритму та основні механізми еволюції популяції, зокрема репродукцію, мутацію, вимирання та еволюційні стрибки.

Причому, ефективність методу підтверджується результатами чисельних експериментів. У даній статті визначено результати обчислювальних експериментів для просторів малої та середньої вимірності та показано можливість узагальнення підходу для широкого класу комбінаторних задач оптимізації. Видається, що подальший розвиток тематики пов'язаний з підвищенням продуктивності та аналітичних можливостей алгоритму. Через це, у статті окреслено перспективи подальших досліджень, зокрема використання паралельних обчислень, аналізу структури популяцій та засобів візуалізації результатів.

Ключові слова: *генетичний алгоритм; задача контактного числа; комбінаторна оптимізація; евклідовий простір; еволюційні обчислення.*

OLEKSANDR, IHOR LEVCHUK, HANNA HUZ, TATYANA DUBOVYK
Ukrainian State University of Science and Technologies

DEVELOPMENT OF A GENETIC ALGORITHM FOR FINDING APPROXIMATE SOLUTIONS TO THE CONTACT NUMBER PROBLEM

The contact number problem is one of the pressing issues in combinatorial optimisation and discrete geometry, which has important theoretical and practical significance. The article determines the relevance of this problem and justifies the need to apply approximate computational methods for its study in Euclidean spaces of arbitrary dimension. In addition, the geometric formulation of the problem is closely related to the analysis of the mutual arrangement of vectors and angular constraints between them, and therefore the article analyses the representation of the contact number problem through α-admissible systems of vectors and establishes a connection between these systems and configurations of tangent spheres.

The limitations of classical (traditional) analytical methods necessitate the search for alternative approaches to solving the given problem. The article proves the possibility and effectiveness of using ε-close systems of vectors with integer coordinates to find approximate solutions. Modern widespread evolutionary methods allow for the effective exploration of large spaces of admissible configurations. The article proposes a genetic algorithm in which the criterion of optimality is the minimisation of the maximum cosine of the angle between the vectors of the system. In this case, the computational implementation of the proposed approach in practical application is of considerable importance. In addition, the article investigates the software implementation of the algorithm and the basic mechanisms of population evolution, in particular reproduction, mutation, extinction, and evolutionary leaps.

Moreover, the effectiveness of the method is confirmed by the results of numerous experiments. This article presents the results of computational experiments for low- and medium-dimensional spaces and shows the possibility of generalising the approach for a wide class of combinatorial optimisation problems. It seems that further development of the topic is related to improving the performance and analytical capabilities of the algorithm. Therefore, the article outlines prospects for further research, in particular the use of parallel computing, analysis of population structure, and means of visualising results.

Keywords: *genetic algorithm; contact number problem; combinatorial optimisation; Euclidean space; evolutionary computation.*

Стаття надійшла до редакції / Received 18.01.2026
Прийнята до друку / Accepted 11.02.2026
Опубліковано / Published 26.03.2026

© Романчук Олександр, Левчук Ігор, Гузь Ганна, Дубовик Тетяна

Постановка проблеми у загальному вигляді та ії зв'язок із важливими науковими чи практичними завданнями

Задача контактного числа належить до класу складних задач комбінаторної оптимізації та залишається нерозв'язаною у загальному випадку для евклідових просторів довільної вимірності. В свою чергу, класичні (традиційні) аналітичні методи дають лише окремі точні значення або грубі оцінки, що суттєво нівелює відповідні можливості дослідження цієї задачі у просторах вищих вимірів. Видається, що зазначені фактори каталізують необхідність розробки обчислювальних та евристичних підходів, здатних знаходити якісні наближені розв'язки. Крім того, особливої актуальності у цьому контексті набуває використання генетичних алгоритмів, які дозволяють ефективно досліджувати великі простори допустимих конфігурацій. Отримані результати у даному дослідженні можуть бути використані як для поглиблення теоретичних досліджень у дискретній геометрії, а також у напрямку розв'язання прикладних задач оптимального розміщення та пакування в багатовимірних просторах.

Аналіз останніх досліджень та публікацій

Проблематика розробки генетичного алгоритму для знаходження наближених розв'язків задачі контактного числа, а також суміжні питання стало предметом наукових розвідок значної кількості науковців. Тим не менш, особливу увагу належить віднести наступним: А. Чопра, Б. ван Гімен, Г. Лю, Д. Фесті, І. Гіль Фернандес, К. Ма, Л. Персоа, М. Резенде, О. Мусін, Я. Кім та інші. Зокрема, у працях [1–4] проводилися дослідження, спрямовані на вивчення задачі контактного числа та пов'язаних із нею геометричних і алгоритмічних підходів. Натомість, у роботах [5–7] розглядалися загальні методи оптимізації та еволюційні алгоритми, а також класичні результати, що визначають фундаментальні властивості задачі контактного числа. Тим не менш, навіть враховуючи такі ґрунтовні доктринальні напрацювання пропонується провести дане дослідження крізь призму практично-прикладного характеру.

Формулювання цілей статті

Метою статті є розробка та реалізація генетичного алгоритму для висвітлення наближених розв'язків задачі контактного числа в евклідових просторах довільної вимірності. Крім того, у даному дослідженні метою є й здійснення експериментальної перевірки ефективності запропонованого підходу та аналіз отриманих геометричних конфігурацій крізь призму просторів малої та середньої вимірності.

Виклад основного матеріалу

При розв'язанні деяких задач виникає необхідність зі скінченної чи зліченної множини певних об'єктів обрати скінченну підмножину, яка задовольняє деяким наперед визначеним умовам. Такі проблеми отримали назву задач комбінаторної оптимізації. До них відносяться зокрема задача комівояжера, задачі цілочислового програмування, задача про призначення, задача пошуку мінімального кістякового дерева у графі тощо [1,2]. Прикладом подібної задачі є і досі нерозв'язана у загальному випадку задача контактного числа: вимагається у багатовимірному евклідовому просторі побудувати найбільшу кількість куль, які дотикаються до кулі радіусу 1, та мають пустий перетин чи дотик між собою. На даний момент відомі значення контактних чисел для просторів виміру 1, 2, 3, 4, 8, 24. Також відомі оцінки значень для інших просторів, зокрема $40 \leq N(5) \leq 46$. У даній роботі цю задачу було розглянуто під іншим кутом зору та побудовано інструмент, що дозволяє досліджувати різноманітні аспекти даної проблеми. Причому розроблений алгоритм дозволяє проводити дослідження для евклідових просторів будь-яких вимірів і легко може бути розширений для аналізу подібних проблем у загальному випадку.

Введемо деякі означення та розглянемо допоміжні твердження необхідні для розуміння роботи алгоритму. Назвемо систему векторів:

$a_i = \left(x_1^i, x_2^i, \dots, x_n^i\right)$ $i = \overline{1,k}$ n-вимірного простору *α-допустимою*, якщо при $i \neq j$ $\frac{a_i \cdot a_j}{\|a_i\| \cdot \|a_j\|} \leq \cos \alpha$.

Теорема 1: Якщо система векторів $a_i = \left(x_1^i, x_2^i, \dots, x_n^i\right)$, $i = \overline{1,k}$ $\frac{\pi}{3}$-допустима, то кулі одиничного радіусу B_i з центрами у точках

$$x_i = \left(\frac{2x_1^i}{\|a_i\|}, \frac{2x_2^i}{\|a_i\|}, \dots, \frac{2x_n^i}{\|a_i\|}\right) \; i = \overline{1,k}$$

будуть дотикатися до кулі одиничного радіусу з центром в початку координат, а між собою або дотикатися, або мати пустий перетин.

Доведення

Нехай $\rho(a_i, a_i)$ – відстань між центрами двох різних куль, тоді

$$\rho(a_i, a_i) = \sqrt{\left(\frac{2x_1^i}{\|a_i\|} - \frac{2x_1^j}{\|a_j\|}\right)^2 + \left(\frac{2x_2^i}{\|a_i\|} - \frac{2x_2^j}{\|a_j\|}\right)^2 + \cdots + \left(\frac{2x_n^i}{\|a_i\|} - \frac{2x_n^j}{\|a_j\|}\right)^2} =$$

$$2\sqrt{\left(\frac{x_1^i}{\|a_i\|} - \frac{x_1^j}{\|a_j\|}\right)^2 + \left(\frac{x_2^i}{\|a_i\|} - \frac{x_2^j}{\|a_j\|}\right)^2 + \cdots + \left(\frac{x_n^i}{\|a_i\|} - \frac{x_n^j}{\|a_j\|}\right)^2} =$$

$$2\sqrt{\left(\frac{x_1^i}{\|a_i\|}\right)^2 + \cdots + \left(\frac{x_n^i}{\|a_i\|}\right)^2 + \left(\frac{x_1^j}{\|a_j\|}\right)^2 + \cdots + \left(\frac{x_n^j}{\|a_j\|}\right)^2 - 2\left(\frac{x_1^i x_1^j + \cdots + x_n^i x_n^j}{\|a_i\| \cdot \|a_j\|}\right)} = 2\sqrt{1 + 1 - 2\frac{a_i \cdot a_j}{\|a_i\| \cdot \|a_j\|}} .$$

Оскільки система векторів $\frac{\pi}{3}$-допустима, то $\frac{a_i \cdot a_j}{\|a_i\| \cdot \|a_j\|} \leq 0{,}5$ і матимемо $\rho(a_i, a_i) \geq 2$. Отже, цим самим доведено, що кулі не будуть перетинатися, адже відстань між їх центрами більша, ніж сума радіусів. Оскільки $\|x_i\| = 2,\ i = \overline{1,k}$, то очевидно, що кулі одиничного радіусу з центрами в точках x_i дотикатимуться до кулі одиничного радіусу з центром в початку координат. Теорема доведена.

Отриманий результат дозволяє стверджувати, що ми розв'яжемо задачу контактного числа, якщо знайдемо максимальну $\frac{\pi}{3}$-допустиму систему векторів або навчимось розв'язувати більш загальну задачу: в n-вимірному просторі знаходити максимальні α -допустимі системи для будь-якого параметра α (використовуючи інтуїтивні геометричні поняття, можна сказати, що задача зводиться до знаходження найбільшого числа радіус-векторів, кут між будь-якими двома з них, більше або дорівнює заданому куту α) [3,4]. В даному дослідженні погляд на проблему відрізнявся: ми пропонуємо відмовитись від пошуку максимальної α-допустимої системи, а надамо програмі певну кількість векторів і критерій оптимізації, - найбільший кут між ними має бути найменшим, і подивимось, які кути «допускатимуться» по вищеозначеному критерію для цієї кількості векторів.

Передусім, слід зазначити, що у разі коли ми маємо деяку α-допустиму систему векторів, то поворот простору (ортогональне перетворення) навколо початку координат, очевидно, відобразить цю систему в α-допустиму, а оскільки таких ортогональних перетворень існує нескінченно багато, то одна α-допустима система породжуватиме нескінченну кількість ідентичних їй, отриманих поворотами вихідної навколо початку координат. Це дозволяє зробити евристичне припущення, що достатньо велика кількість максимальних α-допустимих систем складається з систем «схожих» між собою. Уточнимо сказане та назвемо дві системи:

$A = \left\{a_i = \left(x_1^i, x_2^i, \ldots, x_n^i\right) \middle| i = \overline{1,k}\right\}$ та $B = \left\{b_i = \left(y_1^i, y_2^i, \ldots, y_n^i\right) \middle|\ i = \overline{1,k}\right\}$ *ε-близькими*, якщо між ними існує взаємно-однозначна відповідність $F \mid A \to B$, така що $\frac{a \cdot F(a)}{\|a\| \cdot \|F(a)\|} < \varepsilon$ для всіх $a \in A$. Далі, нехай X – деяка максимальна α-допустима система в E^n. Оскільки множина векторів з раціональними координатами всюди щільна в просторі E^n, то для будь-якого $\varepsilon > 0$ можна знайти ε-близьку систему X' до X, яка складається з векторів з раціональними координатами. Нехай $m = \text{НСК}(q_1, q_2, \ldots)$, де q_i – знаменники дробів, з яких складаються координати векторів системи X'. Помноживши кожен вектор із X' на m отримаємо ε-близьку до X систему векторів, координати яких вже будуть цілими числами.

Теорема 2. Нехай X – деяка максимальна α-допустима система в E^n. Для будь-якого $\varepsilon > 0$ можна побудувати таку ε-близьку систему X' до X, яка буде складатися лише з векторів, координати яких є цілими числами. Доведена теорема дозволяє розглядати вектори з цілочисловими координатами при знаходженні наближених розв'язків (ε-близьких) задачі контактного числа. Перейдемо до опису алгоритму. Розробка велася на Java. В основі лежать класи Individual, Population, Evolution. Нижче наведено UML-діаграму зв'язків між класами Individual інкапсулює інформацію про ε-близький розв'язок - набір векторів (поле individual – двовимірний масив int [] []) з цілочисловими координатами, які формуються випадковим чином, а також, так звану, «генетичну ознаку» - дійсне число, що дорівнює $\max\limits_{i,j} \frac{a_i \cdot a_j}{\|a_i\| \cdot \|a_j\|}$. Геометрично генетична ознака означає найбільший косинус кута між векторами об'єкта Individual (далі - особина), а оскільки косинус найбільший, то кут буде найменший. Видається, що цей параметр має бути мінімальним, щоб отриманий розв'язок був оптимальним. Кількість та розмірність векторів передаються параметрами конструктору [5,6]. У разі, якщо б вдалося побудувати особину з полем individual, що налічує 41 п'яти-вимірний вектор та з генетичною ознакою меншою ніж 0.5, то можна було б стверджувати, що $41 \leq N(5)$. При створенні екземпляру Individual полю, що відповідає генетичній ознаці присвоюється значення за замовчуванням –1. При роботі програми воно перераховується методом createGeneticTrait() на основі елементів, з яких складається масив individual. Також у цьому класі є дуже важливий метод evalLeap() (так званий, «еволюційний стрибок») з параметром типу int. В цьому методі всі вектори з масиву individual множаться на параметр, що передається йому. Значення цього методу пояснимо пізніше. Клас Individual реалізує інтерфейси Cloneable (метод clone() забезпечує глибоке клонування і використовується в класі Population, де «нащадки» створюються змінами у клонах, а не в оригіналах; якщо оригінал краще, то він залишиться в популяції), Comparable (об'єкти типу Individual порівнюються по генетичній ознаці $\max\limits_{i,j} \frac{a_i \cdot a_j}{\|a_i\| \cdot \|a_j\|}$, ті особини, у яких вона найбільша з часом «вимирають»), Serializable (реалізація цього інтерфейсу потрібна, щоб мати можливість

зупинити програму, зберегти об'єкт популяції у файлі, і потім продовжити «еволюцію» з цією популяцією). Клас Population моделює популяцію – одиницю еволюції. Він містить декілька статичних методів найважливіші з них – reproduction() та extinction(). Пропонуємо коротко дослідити ці методи. Метод extinction() приймає наступні параметри: об'єкт популяції, в якій відбуватиметься «вимирання», ціле число, що відповідає кількості найкращих особин, які залишаться, а також назву файлу, в який буде виведено особини популяції після вимирання. Методу reproduction() передається параметр популяції, де відбуватиметься репродукція, ціле число sizeOfPopulation, назва файлу (для передачі в extinction()), ціле число paramEx (для передачі в extinction()) і цілочисловий масив mutations. В свою чергу, перший оператор у методі здійснює перевірку, чи розмір популяції не перевищує параметр sizeOfPopulation, якщо так, то викликається extinction(), у противному випадку особини клонуються і до клонів застосовується мутація – до кожної координати клона додається випадковим чином обране число з масиву mutations. Фактично, популяція збільшується вдвічі, а тому, ми можемо регулювати наступні параметри: розмірність векторів та їх кількість в об'єкті Individual, множину цілих чисел, з яких випадково обиратимуться координати об'єктів Individual, максимальну кількість особин в популяції, по досягненні якої починається вимирання (у коді - видалення з відсортованого динамічного масиву останніх елементів), кількість особин, які залишаться після вимирання та вектор мутацій [7]. Все це відкриває широкі можливості для подальшого дослідження.

Точка входу в програму знаходиться у класі Evolution. У методі main() створюється масив fromWhichElements, в якому вказано елементи, з яких випадково обиратимуться числа для побудови особин першої популяції, також ініціалізується масив mutations та деякі інші параметри необхідні для роботи методів. Назву файлу, розмір популяції, по досягненню якого буде викликано метод extinction(), кількість особин, які залишаться в популяції, після того як метод extinction() поверне керування, а також флагову змінну, яка означає, чи починати роботу алгоритму спочатку, чи десеріалізувати популяцію з файлу, де зберігся стан програми після попередніх запусків, - пропонується ввести користувачу з консолі. Далі, починається робота алгоритму: у циклі, умовою виходу з якого, є досягнення певного моменту часу (через годину, дві тощо, - можна регулювати, але з перекомпіляцією вихідного коду), викликається метод reproduction(), після цього викликається метод evalLeap() для кожної особини з популяції та знову запускається цикл і так далі. З'ясуємо тепер на простому прикладі з тривимірного простору, яким цілям слугує метод evalLeap(). Нехай на певному етапі роботи алгоритму ми отримали найкращу популяцію, в якій особина – екземпляр Individual – містить два вектора a = [1, 0, 1], b = [1, 0, 0] і нехай вектор mutations = [1, 0, -1]. Видається, що додаючи випадково обрані числа з вектору mutations до координат векторів a, b ми значно змінюватимемо кут між ними і просто «ходитимемо по колу», не покращуючи популяцію. Тим не менш, якщо ми збільшимо довжини векторів в декілька разів, наприклад, a = [10, 0, 10], b = [10, 0, 0], то тепер мутації незначно змінюватимуть кут і є ймовірність покращення особини, до якої входили дані вектори.

Зробимо зведення результатів отриманих на даний момент. Дані виведені в такому форматі: перший рядок – дата та час, коли популяцію було збережено у файл, потім – вектори, з яких складаються особини популяції, після них – генетична ознака особини. Надалі, пропонуємо декілька популяцій з двовимірного простору, а також графічна ілюстрація розташування куль для найкращої особини з популяції:

```
2025-04-03T22:05:24.606682800
[[-3, -1], [1, -1], [0, 1], [4, 1], [-1, -3], [-5, 3]]
0.6507913734559684

2025-04-03T22:05:24.778504400
[[7, -3], [4, 3], [-4, -2], [0, 3], [0, -4], [-7, 5]]
0.6
```

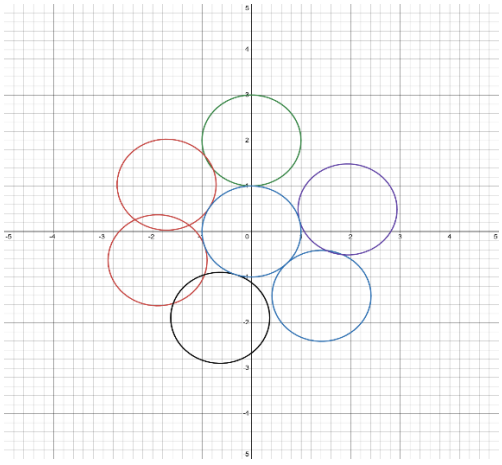

Рис. 1. Оптимальне розташування куль у популяції 1

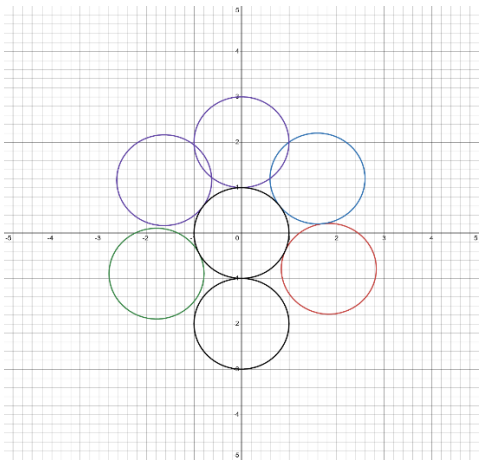

Рис. 2. Оптимальне розташування куль у популяції 2

```
2025-04-03T22:05:25.044044
[[12, -4], [12, 10], [-4, -4], [-1, 7], [2, -9], [-10, 4]]
0.5368754921931592
2025-04-03T22:05:25.278343200
[[11, -3], [14, 14], [-5, -5], [-2, 8], [2, -8], [-12, 3]]
0.5144957554275266
```

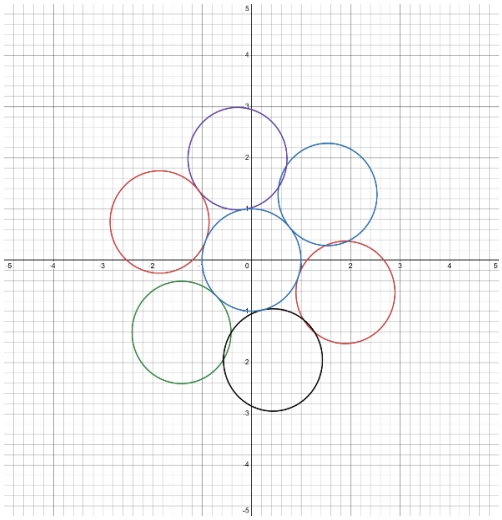

Рис. 3. Оптимальне розташування куль у популяції 3

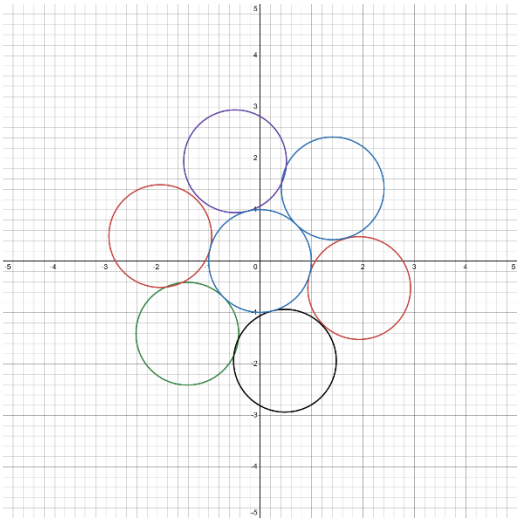

Рис. 4. Оптимальне розташування куль у популяції 4

```
2025-04-03T22:05:51.222069100
[[11, -3], [14, 14], [-7, -7], [-4, 15], [4, -15], [-11, 3]]
0.5028168340384036
```

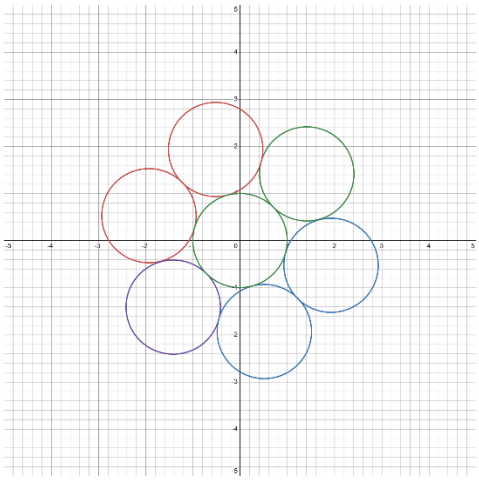

Рис 1. Розташування куль для найкращої особини з вищевказаної популяції

Як бачимо, гарне наближення було отримане менше, ніж за хвилину. На ілюстраціях чітко простежується, як покращується результат. Ось декілька популяцій для тривимірного простору (графічна ілюстрація не наводиться):

```
2025-04-04T12:26:35.997052400
[[-4, -10, -4], [16, 0, 2], [-10, 8, 10], [4, 14, 5], [-6, -7, 9], [-
11, -1, -3], [-5, 1, -18], [8, -17, 6], [7, 8, -8], [-6, 11, -5], [5, 2, 16],
[8, -7, -10]]
0.5190408978964643
```

Близько хвилини є достатнім для отримання гарного наближення. Зауважимо, що вагома частина експериментів проводилася для п’яти-вимірного простору. В загальному довелося витратити близько 50 год, щоб отримати наступний результат, в якому вказно набір векторів, що дають оптимальне знайдене на даний момент наближення:

```
2025-04-03T10:55:18.057602400
[[3323, -8618, -1505, 7521, -14212], [1225, -19929, 3670, -2420, -4225],
[-20456, 11309, -924, 12674, 9527], [4460, -4779, -10144, -5500, 48], [0, 11576,
2611, 4134, 1528], [-7593, -7668, -4080, 5092, 14910], [2845, 9689, -1405,
18655, -13017], [1053, -9331, -992, -19636, -7510], [14604, 7695, -9237, -3653,
3856], [10999, -8843, -11996, 8065, 11729], [2890, -8370, 6172, 3866, 7094], [-
10988, -8077, -2776, -5408, -2333], [3839, 772, -12913, 5962, -4348], [7817,
5493, 3817, -11692, -1395], [516, -9476, 16387, -12474, 4046], [10283, -6626,
660, -5529, 5167], [-10696, -1405, 12991, 972, 11331], [-9262, -6838, 11157, -
2111, -12319], [-10548, -900, -5045, 8356, -7369], [23991, -7002, -3057, -9032,
-17415], [-10809, 1159, -17442, -1213, 1168], [-644, 8216, 16438, -1345, -
4943], [-4088, 3168, 12082, 14607, 531], [-10729, 10545, 2922, 572, -8050],
[11726, -1088, 3465, 12049, 2324], [-1260, 8876, -10618, -15076, 1371], [836,
13774, -17019, 9251, 13565], [1782, 13974, -8022, -734, -8607], [1757, 4618,
4096, 8141, 13682], [-11067, 5505, 9242, -17395, -3574], [12377, 2865, 14565,
-1052, 7834], [24649, 18447, 8306, 4301, -11287], [-2391, -1923, -5142, 18414,
4285], [1673, 6380, 4724, -8466, -19373], [-919, 10226, 6521, -7459, 9751], [-
4250, -14783, -12227, 6227, -1461], [3165, 353, -6524, -6882, 14744], [6674, -
6357, 13264, 4487, -6429], [-4016, -3692, -12867, -5854, -16398], [-19176, 3181,
-3916, -12347, 14797], [-16081, -18098, 8075, 14182, 768]]
0.5463694212288149
```

Таким чином, наразі можна стверджувати, що виникає можливість в п'яти-вимірному просторі розташувати 41 вектор так, що кут між будь-якими двома з них буде не менше, ніж $\arccos(0.5464) \approx 56.8^{\circ}$. Зауважимо, що всі результати були отримані на машині з невисокими технічними характеристиками: двох'ядерний процесор з тактовою частотою 1.10 GHz, об'єм оперативної пам'яті - 3.83 GB.

Висновки з даного дослідження і перспективи подальших розвідок у даному напрямі

В даному дослідженні було розроблено та запропоновано еволюційний алгоритм для знаходження наближених розв'язків задачі контактного числа. Даний алгоритм може бути узагальнений для аналізу будь-яких комбінаторних задач оптимізації. Наразі розробляється програма PopulationAnalyzer, котра дозволятиме виявляти закономірності в популяціях, сформованих алгоритмом, а також проводяться експерименти з різними векторами мутацій та векторами, з яких формується початкова популяція. Розробляється багатопоточний варіант програми: планується, що популяція розбиватиметься на частини і оброблятиметься в паралельних потоках з використанням інфраструктури Fork Join Framework, а також графічний інтерфейс до програми. Видається, що перспективним напрямом подальших досліджень є підвищення обчислювальної ефективності алгоритму шляхом паралелізації обчислень. А тому, актуальним є розроблення багатопоточного варіанту програми, у якому популяція розбиватиметься на частини та оброблятиметься в паралельних потоках з використанням інфраструктури Fork Join Framework, а також передбачається створення графічного інтерфейсу для візуалізації отриманих результатів.

Література

1. Gil Fernández, I., Kim, J., Liu, H., Pikhurko, O. New lower bounds on kissing numbers and spherical codes in high dimensions / I. Gil Fernández, J. Kim, H. Liu, O. Pikhurko // *American Journal of Mathematics*. – Johns Hopkins University Press. – 2025. – Vol. 147, No. 4. – P. 901–925. http://doi.org/10.1353/ajm.2025.a966288
2. Festi, D., Van Geemen, B. A Calabi–Yau threefold coming from two black holes / D. Festi, B. Van Geemen // *Journal of Geometry and Physics*. – 2023. – Vol. 186, April. – Article 104753. http://doi.org/10.1016/j.geomphys.2023.104753
3. Ma, C., Tao Zhaowei, T., Li, P., Liu, M., Chen, H., Mao, Z., Cheng, Y., Qi, Y., Yang, Y. Finding kissing numbers with game-theoretic reinforcement learning / C. Ma, T. Tao Zhaowei, P. Li, M. Liu, H. Chen, Z. Mao, Y. Cheng, Y. Qi, Y. Yang // *arXiv preprint*. – 2025. http://arxiv.org/abs/2511.13391
4. Al-Zewairi, L. Genetic Algorithm: A study survey / L. Al-Zewairi // *Iraqi Journal of Science*. – 2022. – Vol. 63, No. 3. – P. 1251–1231. http://doi.org/10.24996/ijs.2022.63.3.27
5. Londe, M. A., Pessoa, L. S., Andrade, C. E., Resende, M. G. C. Biased random-key genetic algorithms: a review / M. A. Londe, L. S. Pessoa, C. E. Andrade, M. G. C. Resende // *arXiv preprint*. – 2023. http://arxiv.org/abs/2312.00961
6. Kumar, K. E. S., Supreeth, B. S., Dalvi, R., Mittal, A., Akhtar, A., Don Bosco, F., Lineswala, R., Chopra, A. Benchmarking of GPU-optimized quantum-inspired evolutionary optimization algorithm using functional analysis / K. E. S. Kumar, B. S. Supreeth, R. Dalvi, A. Mittal, A. Akhtar, F. Don Bosco, R. Lineswala, A. Chopra // *arXiv preprint*. – 2024. http://arxiv.org/abs/2412.08992
7. Bachoc, C., Vallentin, F. New upper bounds for kissing numbers from semidefinite programming / C. Bachoc, F. Vallentin // *Journal of the American Mathematical Society*. – 2020. – Vol. 21, No. 3. – P. 909–924. http://doi.org/10.1090/S0894-0347-08-00606-3

References

1. Gil Fernández, I., Kim, J., Liu, H., & Pikhurko, O. (2025). New lower bounds on kissing numbers and spherical codes in high dimensions. *American Journal of Mathematics*, 147(4), 901–925.https://doi.org/10.1353/ajm.2025.a966288
2. Festi, D., & Van Geemen, B. (2023). A Calabi–Yau threefold coming from two black holes. *Journal of Geometry and Physics*, 186, 104753. https://doi.org/10.1016/j.geomphys.2023.104753
3. Ma, C., Tao Zhaowei, T., Li, P., Liu, M., Chen, H., Mao, Z., Cheng, Y., Qi, Y., & Yang, Y. (2025). Finding kissing numbers with game-theoretic reinforcement learning. *arXiv preprint*, arXiv:2511.13391.https://arxiv.org/abs/2511.13391
4. Al-Zewairi, L. (2022). Genetic algorithm: A study survey. *Iraqi Journal of Science*, 63(3), 1251–1231.https://doi.org/10.24996/ijs.2022.63.3.27
5. Londe, M. A., Pessoa, L. S., Andrade, C. E., & Resende, M. G. C. (2023). Biased random-key genetic algorithms: A review. *arXiv preprint*, arXiv:2312.00961. https://arxiv.org/abs/2312.00961
6. Kumar, K. E. S., Supreeth, B. S., Dalvi, R., Mittal, A., Akhtar, A., Don Bosco, F., Lineswala, R., & Chopra, A. (2024). Benchmarking of GPU-optimized quantum-inspired evolutionary optimization algorithm using functional analysis. *arXiv preprint*, arXiv:2412.08992. https://arxiv.org/abs/2412.08992
7. Musin, O. R. (2006). The kissing problem in three dimensions. *Discrete & Computational Geometry*, 35(3), 375–384. https://doi.org/10.48550/arXiv.math/0410324