

УДК 519.8:519.6:004.8

Doi: 10.31772/2712-8970-2025-26-3-318-333

Для цитирования: Малашин И. П., Тынченко В. С. Применение методов кластерного анализа для динамической коррекции области поиска в генетическом алгоритме // Сибирский аэрокосмический журнал. 2025. Т. 26, № 3. С. 318–333. Doi: 10.31772/2712-8970-2025-26-3-318–333.

For citation: Malashin I. P., Tynchenko V. S. [Application of cluster analysis methods for dynamic search space adjustment in genetic algorithms]. *Siberian Aerospace Journal*. 2025, Vol. 26, No. 3, P. 318–333. Doi: 10.31772/2712-8970-2025-26-3-318–333.

Применение методов кластерного анализа для динамической коррекции области поиска в генетическом алгоритме

И. П. Малашин^{1*}, В. С. Тынченко²

¹Московский государственный технический университет имени Н. Э. Баумана
(национальный исследовательский университет)

Российская Федерация, 105005, г. Москва, ул. 2-я Бауманская, 5, стр. 1

²Сибирский государственный университет науки и технологий имени академика М. Ф. Решетнева
Российская Федерация, 660037, г. Красноярск, просп. им. газ. «Красноярский рабочий», 31

*E-mail: ivan.p.malashin@gmail.com

Аннотация. В статье рассматривается применение методов кластерного анализа для повышения эффективности генетических алгоритмов при решении задач глобальной оптимизации многомерных функций, актуальных в ракетно-космической отрасли. Предмет исследования – динамическая коррекция области поиска генетического алгоритма (ГА) на основе статистической фильтрации кластеров индивидов. Тема исследования включает разработку метода динамической коррекции областей интереса по переменным задачи путём разделения популяции на группы с помощью алгоритмов кластеризации (с заранее определенным количеством, а также динамически определяемым количеством кластеров), вычисления для каждой группы показателей численности и средней функции пригодности и отсеечения кластеров, не вносящих значимого вклада в эволюционный процесс. Целью исследования является повышение скорости сходимости алгоритма без снижения качества результатов при решении задач смешанной оптимизации за счет эффективной адаптации области поиска на каждом шаге алгоритма. Эксперименты на тестовых функциях показали повышение скорости сходимости в среднем на 25–30 % по сравнению со стандартным алгоритмом. Полученные результаты демонстрируют возможность интеграции разработанного подхода в программно-аппаратные комплексы для автоматизированного проектирования ракетно-космических систем, что позволит снизить время расчётов и повысить точность выбора оптимальных параметров. Также предложенный подход может быть использован для выбора эффективных значений гиперпараметров широкого спектра моделей машинного обучения, в частности, архитектурного синтеза искусственных нейронных сетей, различных топологий, включая глубокие нейронные сети и специализированные архитектуры.

Ключевые слова: эволюционные вычисления, глобальная оптимизация, генетические алгоритмы, область поиска, кластерный анализ.

Application of cluster analysis methods for dynamic search space adjustment in genetic algorithms

I. P. Malashin^{1*}, V. S. Tynchenko²

¹Bauman Moscow State Technical University (National Research University)
5, building 1, 2-ya Baumanskaya St., Moscow, 105005, Russian Federation

²Reshetnev Siberian State University of Science and Technology
31, Krasnoyarskii rabochii prospekt, Krasnoyarsk, 660037, Russian Federation

*E-mail: ivan.p.malashin@gmail.com

Abstract. This study investigates the application of cluster analysis techniques to improve the efficiency of genetic algorithms (GAs) in solving multidimensional global optimization problems, particularly those relevant to the aerospace industry. The research focuses on dynamic search space adjustment in GAs through statistical filtering of individual clusters. The proposed methodology involves: (1) developing a dynamic correction approach for variable domains by partitioning the population into clusters using both fixed-number clustering algorithms (k-means, k-medians, agglomerative, and spectral clustering) and density-based methods (DBSCAN); (2) evaluating cluster quality metrics including population size and average fitness; and (3) eliminating clusters that contribute insignificantly to the evolutionary process. The primary objective is to enhance algorithm convergence speed by 25–30 % (as demonstrated in benchmark testing) while maintaining solution quality in mixed optimization problems through effective search space adaptation at each iteration. The three-stage method comprises: (1) current population clustering, (2) elimination of clusters with below-average population size and fitness, and (3) dynamic boundary adjustment for remaining individuals' domains. Experimental results demonstrate the method's potential for integration into aerospace design systems, significantly reducing computation time while improving parameter optimization accuracy. Furthermore, the approach shows promise for hyperparameter optimization in various machine learning models, particularly in neural network architecture synthesis - including deep neural networks and specialized topologies.

Keywords: evolutionary computation, global optimization, genetic algorithms, search space adaptation, cluster analysis.

Введение

Генетические алгоритмы (ГА) зарекомендовали себя как универсальный инструмент для решения сложных задач глобальной оптимизации в различных областях науки и техники. В ракетно-космической отрасли они применяются для проектирования систем управления [1], оптимизации параметров двигательных установок, аэродинамических форм крыльев и обтекателей, траекторий полёта и систем жизнеобеспечения космических аппаратов [2], а также во многих других задачах. Классическая реализация ГА предполагает фиксированные границы областей поиска для каждой переменной, что на ранних этапах может приводить к избыточному исследованию бесперспективных участков n -мерного пространства и замедлять процесс сходимости.

Одним из приоритетов последних модификаций стало сокращение объёма вычислений за счёт более экономного распределения ресурсов и фокусировки на действительно значимых участках пространства решений. Для этого используются различные подходы: адаптация параметров в процессе эволюции, обучение выбору вариантов операторов на данных предыдущих поколений, а также динамическая локализация поиска с учётом текущего распределения решений. Всё это может позволить ускорить сходимость и повысить эффективность поиска без существенного роста вычислительной нагрузки.

В данной работе предложен метод динамической коррекции области поиска генетического алгоритма, отличающийся от известных реализацией процедуры итеративной кластеризации

популяции с последующим удалением неперспективных кластеров, что позволяет эффективно адаптировать область поиска и повысить скорость сходимости к оптимальному решению. Суть метода заключается в том, что после каждого шага селекции популяция разбивается на группы с использованием одного из алгоритмов кластеризации. В настоящей работе исследовано применение 5 подходов: с заранее определенным количеством кластеров – k-средних, k-медианных, агломерированная и спектральная кластеризация, а также подход с динамически определяемым количеством кластеров DBScan. Для каждого кластера вычисляются показатели численности индивидов и их средней пригодности. Кластеры, одновременно не достигшие пороговых значений по критериям возможности коррекции поискового пространства и останова, полностью исключаются из дальнейшего поиска, а области переменных корректируются с учётом распределения оставшихся индивидов.

Основная цель исследования – повысить скорость сходимости без ухудшения качества найденного решения, что важно при сложных инженерных расчётах в области проектирования ракетно-космической систем. В статье приводится математическая постановка задачи, описание процедуры кластеризации и фильтрации, а также результаты вычислительных экспериментов на стандартных тестовых функциях, демонстрирующие 25–30 % экономии вычислительных ресурсов по сравнению с классическим ГА.

Обзор существующих методов

В последние годы появилось немало модификаций для ГА, направленных на повышение их гибкости и исключение преждевременной сходимости к локальным экстремумам. Одна из ключевых тенденций – внедрение автоматической адаптации параметров. Вместо жёстко заданных вероятностей мутации и скрещивания, современные алгоритмы оценивают «разнообразие» популяции (например, через генетическую энтропию или дисперсию) и динамически изменяют эти вероятности. В ряде реализаций сама величина шага мутации эволюционирует вместе с решением (self-adaptive подход), что избавляет от длительной настройки на этапе подготовки и помогает избежать преждевременного сжатия популяции [3].

Вместе с этим всё большее распространение получили многопопуляционные, или «островные», модели ГА. В традиционном ГА существует единственная популяция, но в современных реализациях исследователи стали создавать сразу несколько небольших субпопуляций («островов»), каждая из которых эволюционирует относительно автономно [4]. Периодически между островами происходит миграция индивидов: лучшие особи перетекают в другие популяции, что позволяет сочетать преимущества параллельного исследования различных регионов пространства решений и обмена качественными генами [5].

Немалую роль в развитии ГА сыграла интеграция с локальными методами оптимизации, породив меметические алгоритмы (т. н. memetic algorithms). Идея заключается в том, что генетический оператор глобального поиска дополняется мощными процедурными локальными «доисследованиями», например, градиентными спусками, где это возможно, или алгоритмами прямого локального поиска в случае отсутствия информации о производных. В результате ГА проводит широкое «сканирование» пространства, а локальные алгоритмы обеспечивают улучшение значения пригодности перспективных индивидов [6].

Помимо этого, многокритериальная оптимизация стала важным направлением для развития ГА, так как в реальных инженерных и экономических задачах необходимо учитывать сразу несколько конкурирующих целей. Хотя классические методы NSGA-II [7], MOEA/D [8] и SPEA2 [9] прочно утвердились, в последние годы специалисты предложили новые стратегии для лучшего распределения решений фронта Парето. Например, адаптивное распределение опорных точек (reference points) позволяет более равномерно представлять разные области фронта, а динамическая «ширина окна» (crowding distance) учитывает локальную плотность решений: если решения скапливаются слишком плотно, алгоритм автоматически расширяет расстояние разделения, тем самым стимулируя поиск в менее исследованных областях [10].

Активно развивается направление суррогатного моделирования, где вычисления целевой функции заменяются предиктивными моделями (Gaussian Process, случайный лес, нейросети). В гибридных схемах Surrogate-Assisted GA (SA-GA) сбор данных о целевой функции идёт постепенно: вначале строится простой суррогат, а затем, по мере накопления данных, алгоритм переобучает модель [11].

Наконец, нельзя не упомянуть об интеграции ГА с нейроэволюционными методами, где алгоритм не только оптимизирует веса нейросети, но и её топологию. Изначально предложенный NEAT (NeuroEvolution of Augmenting Topologies [12]) получил новое дыхание в сочетании с современными архитектурами (сверточные сети, трансформеры) [13]. В Deep Neuroevolution GA оператор мутации может добавлять или удалять слои нейросети, а кроссовер «скрещивает» архитектуры, стремясь получить компромисс между точностью и компактностью. Такие подходы активно применяются в AutoML-пайплайнах, где автоматическое проектирование нейросетей позволяет сократить время и усилия инженера на ручную настройку.

Однако вопросам коррекции областей поиска не уделяется достаточного внимания [14], что подтверждает актуальность разработки современных схем повышения скорости сходимости ГА за счет концентрации вычислительных ресурсов в перспективных регионах.

Математическая постановка задачи

Рассматривается задача глобальной оптимизации многомерной функции вещественных переменных на ограниченной области определения. Пусть $\bar{X}$ – вектор переменных, характеризующий пространство поиска:

$$\bar{X} = \{x_i, i = 1, \dots, n\}, \quad (1)$$

где n – размерность задачи (количество переменных целевой функции), x_i – i -я переменная. Для каждой переменной задана начальная область определения:

$$x_i \in D_{x_i} = [x_i^l, x_i^h], \quad (2)$$

где x_i^l и x_i^h – нижняя и верхняя границы области допустимых значений переменной x_i . Эти границы определяют начальную гиперпараллелепипедную область поиска в n -мерном пространстве.

В классических вариантах генетических алгоритмов предполагается [15; 16], что границы D_{x_i} остаются постоянными на протяжении всего процесса эволюции:

$$D_{x_i} - \text{фиксировано, } \forall i = 1, \dots, n. \quad (3)$$

Однако такое допущение может быть неэффективным, поскольку оно не позволяет учитывать изменяющуюся структуру распределения решений в процессе поиска. В настоящей работе предлагается и исследуется метод динамической коррекции области поиска генетического алгоритма на каждом его поколении.

Для формализации предлагаемого подхода введём следующие обозначения:

G – общее количество поколений (итераций) генетического алгоритма;

g – номер текущего поколения, $g = \{1, 2, \dots, G\}$;

M – количество индивидов (решений) в популяции на каждом поколении;

m – индекс конкретного индивида в популяции, $m = \{1, 2, \dots, M\}$;

K – число кластеров, полученных после кластеризации популяции;

k – индекс кластера, $k = \{1, 2, \dots, K\}$;

Ind_m^g – вектор параметров m -го индивида на поколении g ;

$F_{Ind_m^g}$ – значение функции пригодности (fitness) для индивида Ind_m^g ;

$IndSet_k^g$ – множество всех индивидов, принадлежащих кластеру k на поколении g ;

$M_k^g = (IndSet_k^g)$ – количество индивидов в кластере k на поколении g .

Каждому кластеру k на поколении g сопоставляется собственная область определения для каждой переменной x_i (рис. 1), которую обозначим:

$$D_{x_i}^{g,k} = [x_i^{g,k;l}, x_i^{g,k;h}], \quad (4)$$

где $x_i^{g,k;l}$ и $x_i^{g,k;h}$ – текущие нижняя и верхняя границы переменной x_i в кластере k на поколении g .

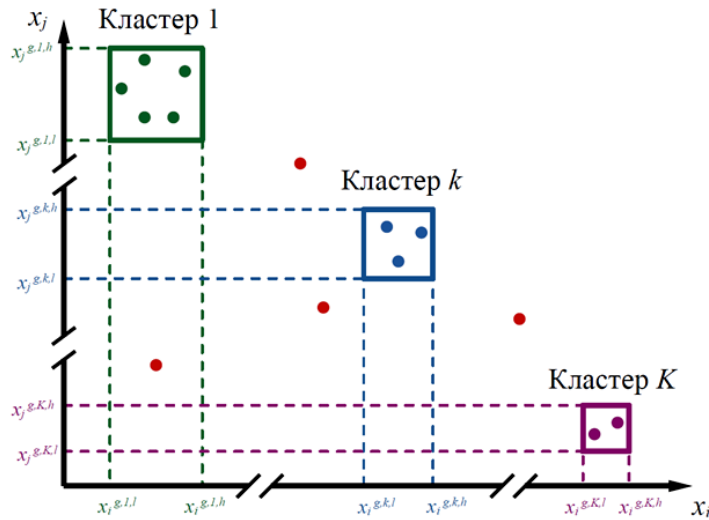


Рис. 1. Схема кластеризации

Fig. 1. Clustering diagram

Полная область поиска по переменной x_i на поколении g определяется как объединение по всем кластерам (5):

$$D_{x_i}^g = \bigcup_{k=1}^K D_{x_i}^{g,k}. \quad (5)$$

Если кластер признаётся неэффективным (по критериям, описанным далее), соответствующая область $D_{x_i}^{g,k}$ исключается из рассмотрения, т. е. полагаем:

$$D_{x_i}^{g,k} = \emptyset. \quad (6)$$

Кластеризация решений

Для повышения эффективности поиска в пространстве решений применяется кластерный анализ, позволяющий выявить перспективные области скопления качественных решений для концентрации вычислительных ресурсов в них. В рамках данной работы используются как методы с заранее фиксированным числом кластеров (K-Means, K-Medoids, агломеративная и спектральная кластеризация), так и плотностной алгоритм, автоматически определяющий количество групп (DBScan).

Методы с фиксированным числом кластеров основаны на минимизации внутрикластерного разброса (K-Means), выборе медоидов для устойчивости к выбросам (K-Medoids), построении иерархической дендрограммы (агломеративная кластеризация) или использовании спектраль-

ных свойств матрицы сходства (спектральная кластеризация). Их основное ограничение – необходимость заранее указать число групп, которое в практических задачах подбирается с помощью «метода локтя» [17] или кросс-валидации по качеству решений.

Плотностные алгоритмы не требуют предварительного задания количества числа кластеров – DBScan определяет кластеры как области с минимальным числом соседей на расстоянии не более ε (параметра, который задает пользователь), автоматически отбрасывая разреженные точки как шум.

Перед кластеризацией необходимо нормализовать данные (привести каждый признак к единому масштабу, например $[0,1]$, и выбрать метрику (евклидова или манхэттенская), поскольку различие диапазонов переменных и способ вычисления расстояний существенно влияет на структуру кластеров.

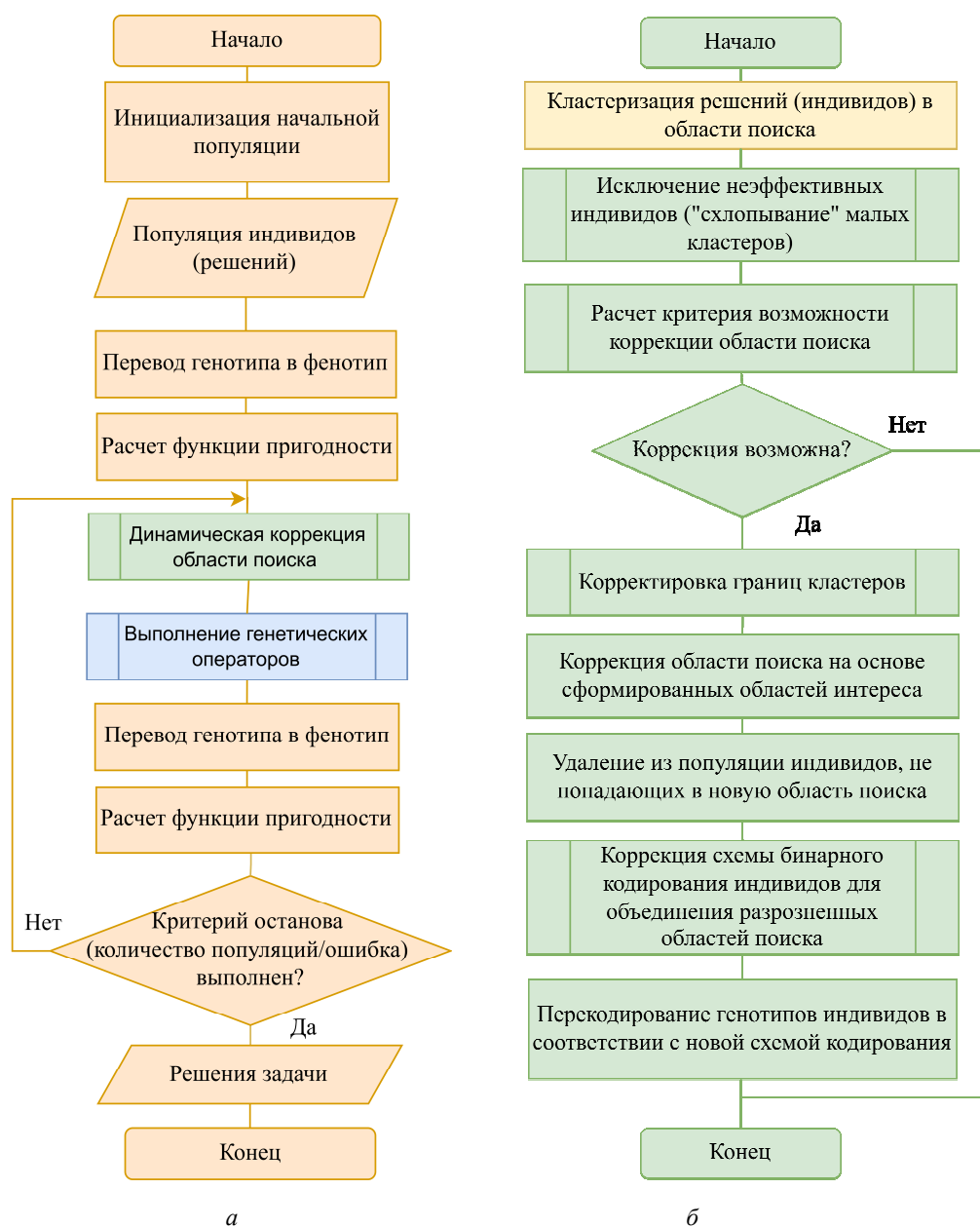


Рис. 2. Блок-схема:

a – ГА с динамической коррекцией области поиска; b – процедуры динамической коррекции

Fig. 2. Block diagram:

a – GA with dynamic correction of search area; b – block diagram of the dynamic correction procedure

В алгоритме оптимизации (рис. 2) результаты кластерного разбиения используются для локальной корректировки границ областей поиска: в каждом кластере вычисляются корректирующие смещения на основе статистик распределения индивидов, что позволяет сузить или расширить область исследования вокруг перспективных решений. Кроме того, внутри кластеров повышается вероятность рекомбинации ближайших представителей, что помогает сохранять локальные особенности найденных оптимумов, а вычислительные ресурсы динамически перераспределяются в пользу кластеров с более высоким качеством решения или скоростью сходимости. Такой подход обеспечивает сбалансированное сочетание исследования и использования пространства решений и способствует более быстрой и надёжной сходимости алгоритма.

Коррекция границ кластеров

После того, как сформированы кластеры, выполняется процедура коррекции границ области поиска в каждом из них, которая производится на основе анализа распределения индивидов внутри кластера. Формально, границы области $D_{x_i}^{g,k}$ обновляются следующим образом:

$$D_{x_i}^{g,k} = \left[x_i^{g,k;l} + \Delta_{x_i^{g,k;l}}, x_i^{g,k;h} + \Delta_{x_i^{g,k;h}} \right], \quad (7)$$

где $\Delta_{x_i^{g,k;l}} \in R$ и $\Delta_{x_i^{g,k;h}} \in R$ – корректирующие значения, на основе которых могут как расширяться, так и сужаться исходные границы. Они могут быть получены на основе стандартного отклонения значений переменной x_i внутри кластера.

Таким образом, на каждом поколении осуществляется локальная адаптация границ области поиска с целью сосредоточения вычислительных ресурсов на перспективных регионах пространства решений.

Реализация и обсуждение

Метод был протестирован на наборе из 4 стандартных тестовых функций различной размерности [18]. Эффективность подхода далее иллюстрируется на примере четырех тестовых функций Экли, Били, Бута и Букина № 6, поскольку они образуют репрезентативную выборку для комплексной оценки эффективности оптимизационных алгоритмов. Ключевое преимущество данного набора функций заключается в том, что они обладают наиболее репрезентативным набором свойств, представляющих наибольшую сложность для алгоритмов оптимизации.

Функция Экли (Ackley) представляет собой сложную многомодальную поверхность с большим количеством локальных минимумов, что позволяет проверить способность алгоритма избегать преждевременной сходимости к субоптимальным решениям:

$$f(x) = -20 \exp \left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2} \right) - \exp \left(\frac{1}{n} \sum_{i=1}^n \cos(2\pi x_i) \right) + 20 + e. \quad (8)$$

Функция Били (Beale) имеет высокую чувствительность к выбору начальной точки:

$$F(x) = \sum_{i=1}^d \left[\left(1.5 - x_{2i-1} + x_{2i-1} x_{2i} \right)^2 + \left(2.25 - x_{2i-1} + x_{2i-1} x_{2i}^2 \right)^2 + \left(2.625 - x_{2i-1} + x_{2i-1} x_{2i}^3 \right)^2 \right],$$

$$x = (x_1, \dots, x_{2d}) \in \mathbb{R}^{2d}. \quad (9)$$

Функция Бута (Booth), обладая относительно простой структурой с единственным ярко выраженным глобальным минимумом, служит тестом на базовую работоспособность оптимизационных методов:

$$f(x, y) = (x + 2y - 7)^2 + (2x + y - 5)^2. \quad (10)$$

Функция Букина № 6 (Bukin № 6), благодаря наличию разрывов и характерным «оврагам», позволяет оценить устойчивость алгоритмов к экстремальным изменениям градиента:

$$f(x, y) = 100\sqrt{|y - 0,01x^2|} + 0,01(x + 10). \quad (11)$$

Изображения изолиний вышеуказанных тестовых функций для случая двух переменных представлены на рис. 3.

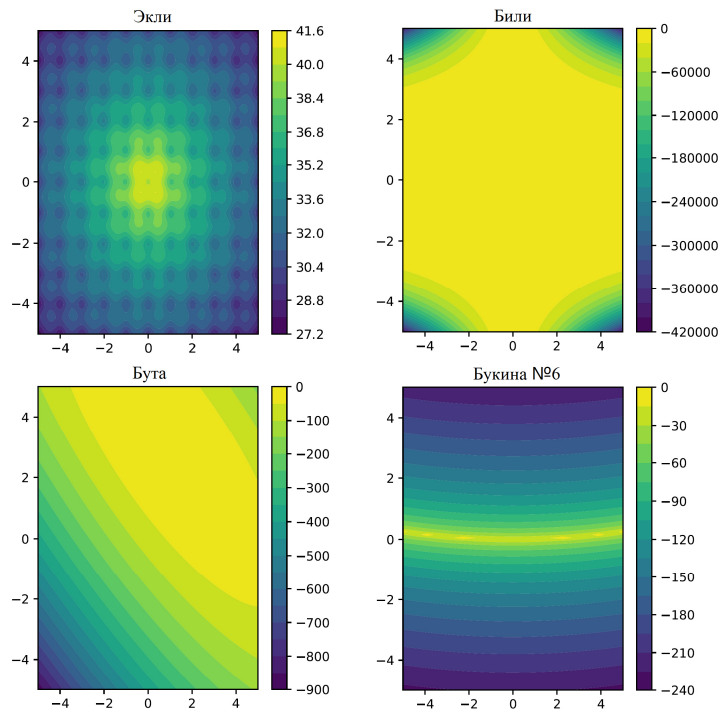


Рис. 3. Изолинии тестовых функций

Fig. 3. Contour plots of test functions

На рис. 4 представлена визуализация оценки эффективности предлагаемого подхода на примере вышеуказанных функций для функций Экли, Били, Бута, Букина № 6 соответственно. Выбор обусловлен различной природой этих тестовых задач: функция Экли традиционно используется как эталон для проверки масштабируемости методов оптимизации в пространстве большой размерности; многомерная модификация функции Били позволяет оценить поведение алгоритма на средних размерностях с выраженной зависимостью от начальных условий; функции Бута и Букина № 6 являются двумерными по определению и применяются для базовой валидации корректности и устойчивости метода в условиях простого (для Бута) и резко изменяющегося (для Букина № 6) рельефа целевой поверхности. На рис. 4 представлены результаты применения пяти методов кластеризации: k-средних (k-means), k-медианных (k-medians), DBSCAN, агломеративной кластеризации и скользящего среднего от верхнего к нижнему ряду соответственно.

Из рис. 4 видно, что для сложных многомерных функций (Ackley, Beale) наилучшие результаты достигаются при использовании 3–6 кластеров, где медианные значения числа итераций ниже и разброс меньше. Для двумерных функций (Booth, Bukin № 6) влияние параметра минимально, поэтому достаточно 1–2 кластеров без заметной потери качества.

На основании прогонов алгоритма на двумерных тестовых функциях, описанных выше, было выявлено, что оптимальный диапазон числа кластеров для большинства задач находится

в пределах 3–6. Для анализа многомерных тестовых функций использовался метод k-means, обеспечивающий стабильное разделение точек и ускорение сходимости алгоритма.

В табл. 1 представлены средние значения числа вычислений целевых функций с указанием стандартного отклонения при различных размерностях и числе кластеров. Функции выбраны из стандартного набора тестовых функций, которые допускают обобщение на пространства большей размерности.

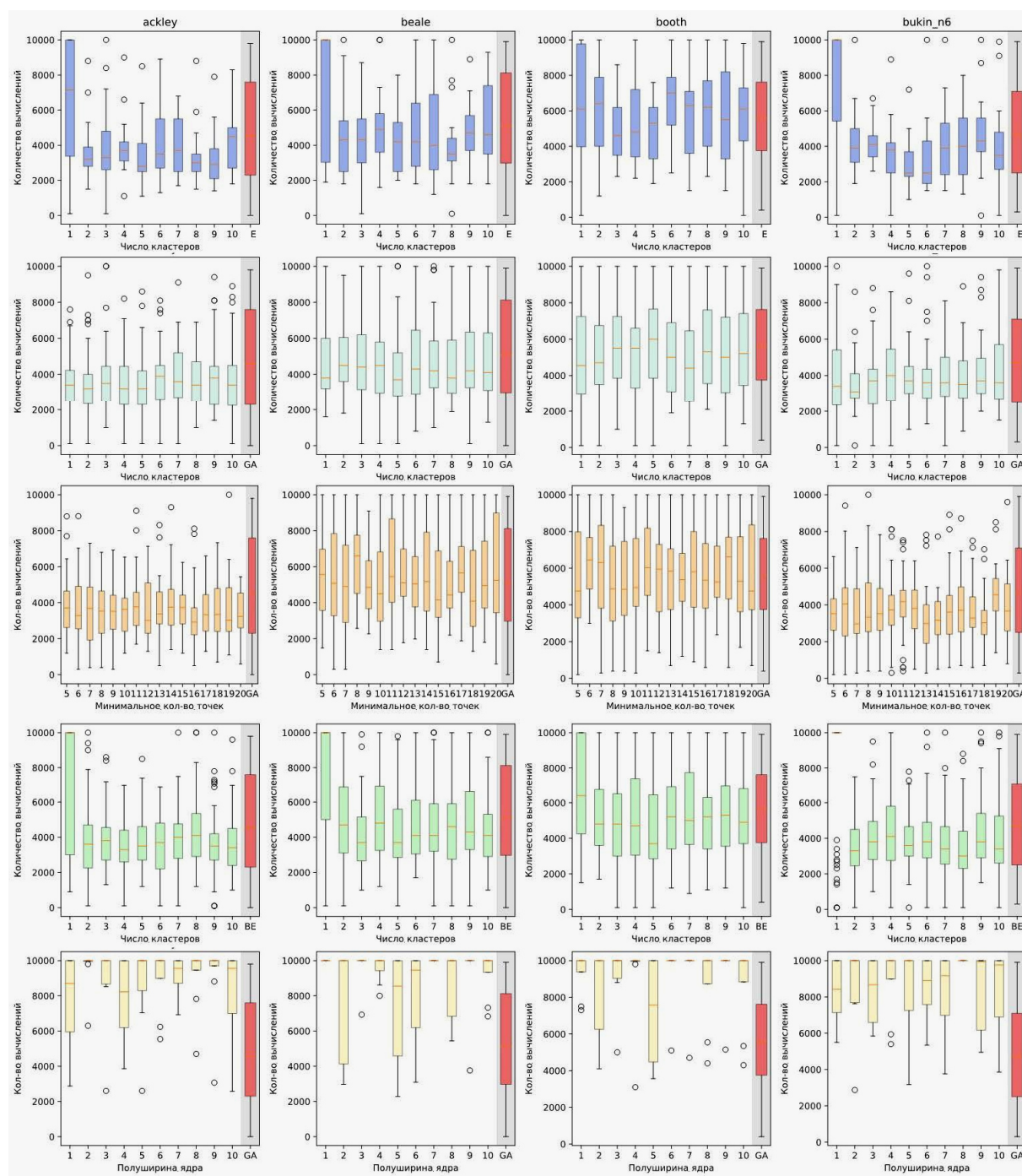


Рис. 4. Диаграммы межквартильного размаха результатов экспериментов. Представлены результаты применения пяти методов кластеризации: k-средних (k-means), k-медианных (k-medians), DBSCAN, агломеративной кластеризации и скользящего среднего (от верхнего к нижнему ряду, соответственно)

Fig. 4. Interquartile (IQR) range diagrams of the experimental results. The figure shows the outcomes of five clustering methods: k-means, k-medians, DBSCAN, agglomerative clustering, and moving average (from top to bottom row, respectively)

Таблица 1

**Среднее количество вычислений целевой функции ($\pm$ стандартное отклонение)
для многомерных тестовых функций при различной размерности и числе кластеров**

Функция	Размерность	3 кластера	4 кластера	5 кластеров	6 кластеров
Сфера	10	1187 $\pm$ 132	2295 $\pm$ 214	2478 $\pm$ 289	2221 $\pm$ 367
	20	2644 $\pm$ 305	5548 $\pm$ 412	4822 $\pm$ 497	4133 $\pm$ 721
	50	4758 $\pm$ 623	10187 $\pm$ 807	11643 $\pm$ 1002	11387 $\pm$ 1154
	100	9442 $\pm$ 1189	8921 $\pm$ 1030	8737 $\pm$ 1655	9845 $\pm$ 2033
Розенброк	10	3368 $\pm$ 247	3124 $\pm$ 301	4192 $\pm$ 382	4087 $\pm$ 472
	20	5278 $\pm$ 493	4842 $\pm$ 615	5217 $\pm$ 786	6462 $\pm$ 912
	50	8956 $\pm$ 1044	8682 $\pm$ 1218	8447 $\pm$ 1659	8973 $\pm$ 1187
	100	10671 $\pm$ 2047	11348 $\pm$ 1965	10582 $\pm$ 2954	12791 $\pm$ 3092
Растригина	10	4142 $\pm$ 198	4084 $\pm$ 153	4226 $\pm$ 433	4211 $\pm$ 498
	20	6637 $\pm$ 527	6438 $\pm$ 642	6582 $\pm$ 804	6276 $\pm$ 986
	50	9672 $\pm$ 1810	9452 $\pm$ 1551	9911 $\pm$ 2792	9374 $\pm$ 2025
	100	16941 $\pm$ 2647	15482 $\pm$ 3082	12653 $\pm$ 3467	13278 $\pm$ 3065
Экли	10	3885 $\pm$ 107	3851 $\pm$ 286	3682 $\pm$ 154	3718 $\pm$ 449
	20	9154 $\pm$ 554	9247 $\pm$ 718	10368 $\pm$ 601	9012 $\pm$ 1802
	50	15934 $\pm$ 1087	17614 $\pm$ 2762	15761 $\pm$ 2634	16286 $\pm$ 2089
	100	22184 $\pm$ 2794	20378 $\pm$ 3358	19756 $\pm$ 3841	20187 $\pm$ 2712
Стыбинского-Танга	10	4082 $\pm$ 103	3738 $\pm$ 214	4187 $\pm$ 478	3724 $\pm$ 251
	20	6326 $\pm$ 389	6843 $\pm$ 518	7624 $\pm$ 967	6557 $\pm$ 842
	50	10762 $\pm$ 895	10574 $\pm$ 1128	11198 $\pm$ 1376	10867 $\pm$ 1582
	100	20468 $\pm$ 1836	21328 $\pm$ 1215	21642 $\pm$ 2048	18124 $\pm$ 3096

Из табл. 1 видно, что в большинстве случаев меньшее число вычислений целевой функции достигается при 3–4 кластерах, особенно для низкой размерности. При высокой размерности влияние числа кластеров менее выражено, однако диапазон 3–6 кластеров остаётся эффективным.

На материале использованных тестовых функций видно, что правильный подбор числа кластеров k может кардинально менять число поколений до сходимости к глобальному оптимально-му решению с предлагаемой модификацией. Во всех случаях ключевыми факторами являются:

- сложность ландшафта целевой функции (наличие локальных минимумов и их плотность, узкие или широкие зоны притяжения);
- адекватность выбора k (малое k не позволяет выделить все «горячие» области, а слишком большое – приводит к перерасходу ресурсов);
- эффективность процедуры обновления границ областей поиска в каждом кластере обеспечивает быстрый «сдвиг» внимания алгоритма к перспективным регионам.

Такой баланс между исследованием пространства решений и использованием результатов исследования обуславливает ускоренную сходимость предложенной модификации по сравнению со стандартным подходом.

Также для сравнения эффективности предложенной в данной статье модификации ГА была проведена серия вычислительных экспериментов на наборе тестовых функций CEC 2017 [19]. В рамках экспериментов рассматривались четыре комбинации алгоритмов:

- стандартный ГА без модификаций;
- стандартный ГА с предложенной модификацией;
- алгоритм SelfCSHAGA (Self-Configuring Success History-based Adaptation Genetic Algorithm) [20];
- алгоритм SelfCSHAGA с предложенной модификацией.

В экспериментальном протоколе использовались тестовые функции из набора CEC'2017. Для каждой функции параметры были выбраны таким образом, чтобы обеспечить максимально возможную размерность задачи при сохранении адекватной вычислительной сложности и устойчивости работы алгоритма. В частности, для большинства функций применялась размерность 50 переменных и размер популяции 200 индивидов. Для более вычислительно слож-

ных функций (например, f_4, f_5, f_7, f_8) размерность была снижена до 20 или 10. Следует отметить, что функция f_9 оказалась чувствительной к размерности: при увеличении размерности до 10 алгоритм SelfCSHAGA работает аналогично стандартному ГА. Однако при размерности 2 алгоритм достаточно быстро находил оптимум.

Количество итераций также адаптировалось в зависимости от сложности функции и размерности области поиска: от 100 итераций (для f_9) до 2000 итераций (для f_2). В табл. 2 приведены параметры экспериментов стандартного ГА и алгоритма SelfCSHAGA на тестовых функциях CEC'2017. Для каждой функции указаны размерность задачи, размер популяции, количество итераций, границы поиска и шаг дискретизации. Следует отметить, что на функциях f_1, f_2 и f_{10} стандартный ГА не справился с поиском оптимума, тогда как SelfCSHAGA продемонстрировал успешную работу.

Таблица 2

Параметры экспериментов для GA и SelfCSHAGA (sSHAGA) на функциях CEC'2017

Функция	Размерность D (GA)	Размерность D (sSHAGA)	Поклоение (GA)	Поклоение (sSHAGA)	Популяция (GA)	Популяция (sSHAGA)	Границы поиска	Оптимальное значение
f_1	–	50	–	1100	–	200	$[-100, 100]$	100
f_2	–	50	–	2000	–	200	$[-100, 100]$	200
f_3	20	50	4500	1000	100	200	$[-100, 100]$	300
f_4	10	20	3500	1000	100	200	$[-100, 100]$	400
f_5	10	20	3500	1300	200	200	$[-100, 100]$	500
f_6	20	50	2500	300	100	200	$[-100, 100]$	600
f_7	10	20	2500	500	100	400	$[-50, 50]$	700
f_8	10	10	3500	1100	300	100	$[-50, 50]$	800
f_9	2	2	100	100	100	50	$[-10, 10]$	900
f_{10}	–	10	–	2500	–	200	$[-100, 100]$	1000

С теми же настройками были протестированы и алгоритмы с модификацией, предложенной в данной статье. Использовался кластеризатор K-means с числом кластеров от 3 до 6. Анализ распределения количества вычислений целевой функции проведён на основе межквартильного размаха. Для стандартного ГА наблюдается вариативность вычислительных затрат между функциями (рис. 5 и 6).

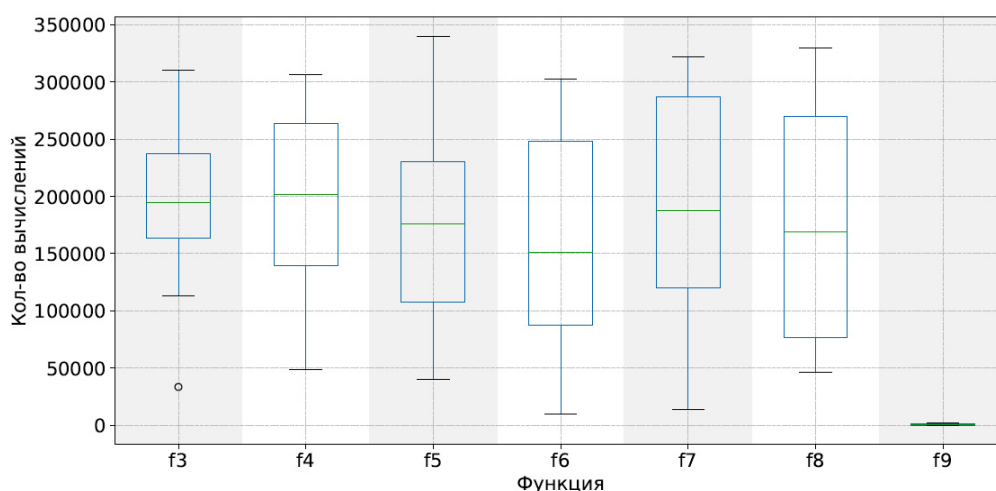


Рис. 5. Диаграммы межквартильного размаха для функций f_3 – f_9 . По оси ординат показано количество вычислений целевой функции. Результаты получены с использованием стандартного генетического алгоритма

Fig. 5. Boxplot diagrams for functions f_3 – f_9 . The y-axis shows the number of objective function evaluations. The results were obtained using the standard genetic algorithm

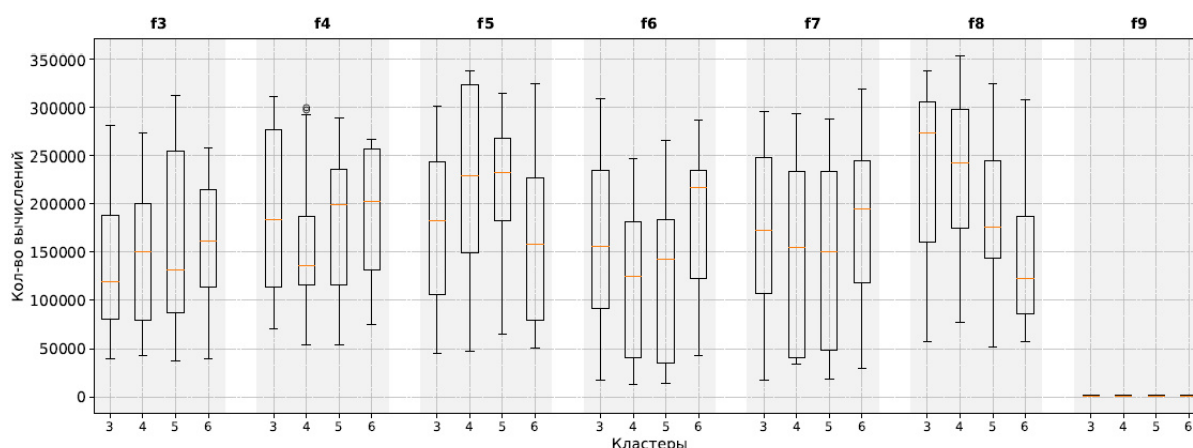


Рис. 6. Диаграммы межквартильного размаха для функций $f3$ – $f9$. Каждый столбик соответствует числу кластеров (3, 4, 5, 6), сгруппированных по функциям. По оси ординат показано количество вычислений целевой функции. Результаты получены с использованием стандартного генетического алгоритма с модификацией, предложенной в настоящей статье

Fig. 6. Boxplot diagrams for functions $f3$ – $f9$. Each bar corresponds to the number of clusters (3, 4, 5, 6), grouped by functions. The y-axis shows the number of objective function evaluations. The results were obtained using the standard genetic algorithm with the modification proposed in this paper

Например, функции $f3$ – $f8$ демонстрируют высокие значения медианы и ширину межквартильного размаха, что указывает на разбросанное потребление вычислительных ресурсов. Вариант алгоритма с модификацией, предложенной в работе, включающей кластеризацию K-means с числом кластеров от 3 до 6, позволил сократить разброс количества вычислений и сместить медиану в сторону меньших значений. Так, генерация индивидов внутри кластеров обеспечивала направленное распределение поиска по пространству решения, что уменьшало количество необходимых вычислений для достижения приближённого оптимума.

Анализ распределения количества вычислений целевой функции для алгоритма SelfCSHAGA (рис. 7 и 8) показывает высокую стабильность и низкую вариативность по сравнению со стандартным ГА.

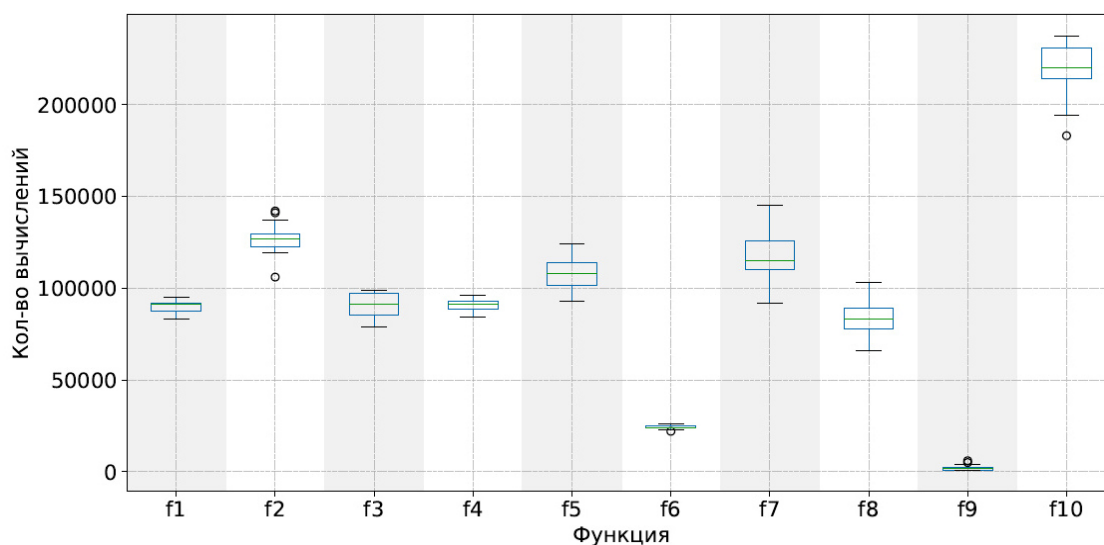


Рис. 7. Диаграммы межквартильного размаха для функций $f1$ – $f10$. По оси ординат показано количество вычислений целевой функции. Результаты получены с использованием алгоритма SelfCSHAGA

Fig. 7. Boxplot diagrams for functions $f1$ – $f10$. The y-axis shows the number of objective function evaluations. The results were obtained using the SelfCSHAGA algorithm

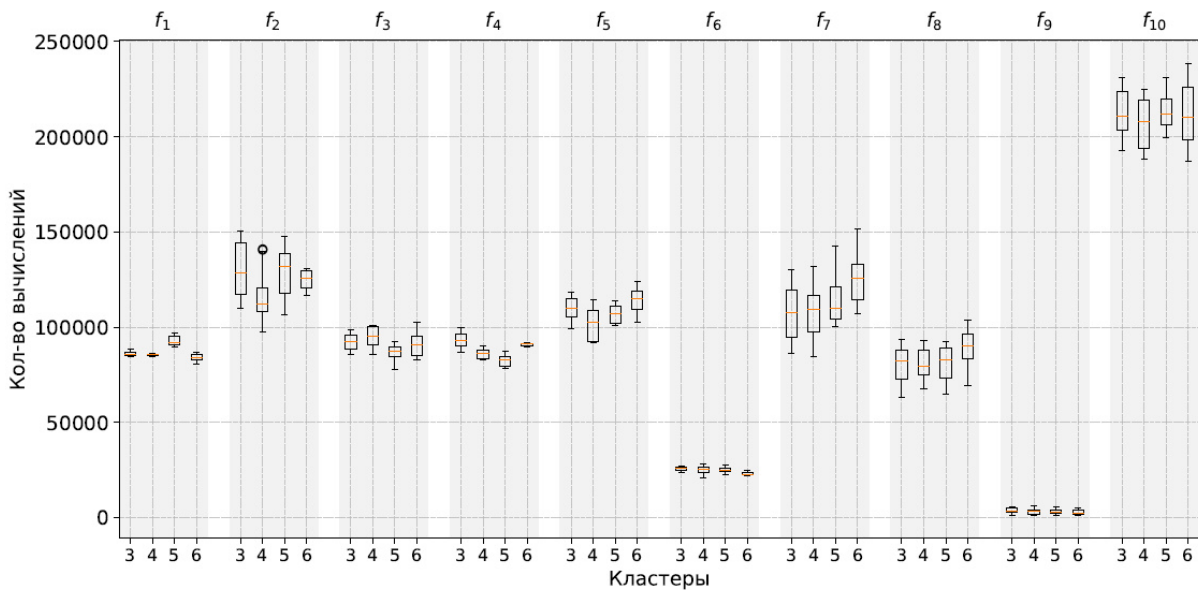


Рис. 8. Диаграммы межквартильного размаха для функций f_1 – f_{10} . Каждый столбик соответствует числу кластеров (3, 4, 5, 6), сгруппированных по функциям. По оси ординат показано количество вычислений целевой функции. Результаты получены с использованием алгоритма SelfCSHAGA с модификацией, предложенной в настоящей статье

Fig. 8. Boxplot diagrams for functions f_1 – f_{10} . Each bar corresponds to the number of clusters (3, 4, 5, 6), grouped by functions. The y-axis shows the number of objective function evaluations. The results were obtained using the SelfCSHAGA algorithm with the modification proposed in this paper

Применение модификации с кластеризацией K-means (число кластеров от 3 до 6) дополнительно снижает разброс вычислений. За счёт распределения популяции по кластерам алгоритм более эффективно исследует пространство решений, особенно на функциях с высокой вычислительной сложностью (f_3 – f_8), уменьшая количество итераций до достижения приближённого оптимума и сужая межквартильный размах. Например, для f_5 и f_7 модифицированный алгоритм потребляет заметно меньше вычислений, чем стандартный SHAGA, при этом IQR сокращается, что отражает более предсказуемую и экономную работу алгоритма.

Заключение

Проведённое исследование предложенного подхода показало, что интеграция метода динамической коррекции области поиска на основе кластеризации в ГА позволяет существенно сократить количество вычислений целевой функции за счёт более точной локализации областей перспективных решений. Коррекция границ областей поиска на основе статистики распределения индивидов в кластерах демонстрирует высокую эффективность в сочетании с динамическим перераспределением вычислительных ресурсов.

Численные эксперименты на репрезентативном множестве тестовых функций подтвердили, что эффективное число кластеров k определяется топологией целевой функции и может варьироваться в широком диапазоне. Небольшие значения k упрощают моделирование ландшафта, но повышают риск пропустить отдельные экстремумы, тогда как избыточно большое k приводит к увеличению количества вычислений. Выбор эффективного значения k (обычно 3–6) обеспечивает наилучший баланс между исследованием пространства решений и использованием результатов исследования, снижая число поколений до сходимости порой в десятки раз по сравнению со стандартным подходом.

Данные результаты подтверждают актуальность и полезность предлагаемого подхода и обосновывают целесообразность дальнейшего его развития. В частности, в дальнейшем целесообразно исследовать возможность автоматической адаптации k в процессе оптимизации,

а также расширить список применяемых методов кластеризации за счёт алгоритмов HDBSCAN [21] и самоорганизующихся карт Кохонена [22]. Кроме того, перспективным направлением развития является комбинирование плотностных и графовых подходов для более гибкого управления поисковыми границами в сложных многомерных задачах.

Библиографические ссылки

1. Semenkin E., Semenkina M. Spacecrafts' control systems effective variants choice with self-configuring genetic algorithm // ICINCO 2012 – Proceedings of the 9th International Conference on Informatics in Control, Automation and Robotics. Rome, 28–31 July 2012. 2012. Vol. 1. P. 84–93.
2. Hidden-variables genetic algorithm for variable-size design space optimal layout problems with application to aerospace vehicles / J. Gamot et al. // Engineering Applications of Artificial Intelligence. 2023. Vol. 121. P. 105941.
3. Srinivas M., Patnaik L. Adaptive probabilities of crossover and mutation in genetic algorithms // IEEE Transactions on Systems, Man, and Cybernetics. 1994. Vol. 24, No. 4. P. 656–667.
4. Liu C., Liu K. An asynchronous island-model genetic algorithm for large-scale global optimization // Applied Soft Computing. 2019. Vol. 76. P. 637–649.
5. Izzo D., Rucinski M., Ampatzis C. Parallel global optimisation meta-heuristics using an asynchronous island-model // Proc. 2009 IEEE Congress on Evolutionary Computation. 2009. P. 2301–2308.
6. Gendreau M., D'Amours G., Rousseau S. Memetic algorithms: Hybrid genetic algorithms and local search methods for combinatorial optimization // Computers & Operations Research. 2020. Vol. 112. P. 104778.
7. Verma S., Pant M., Snasel V. A comprehensive review on NSGA-II for multi-objective combinatorial optimization problems // IEEE access. 2021. Vol. 9. P. 57757–57791.
8. Adaptive replacement strategies for MOEA/D / Wang Z. et al. // IEEE transactions on cybernetics. 2015. Vol. 46, No. 2. P. 474–486.
9. Multiobjective genetic programming: Reducing bloat using SPEA2 / Bleuler S. et al. // Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No. 01TH8546). 2001. Vol. 1. P. 536–543.
10. Jain L., Sinha A. Adaptive reference point selection in NSGA-III for many-objective optimization // Applied Soft Computing. 2021. Vol. 105. P. 107237.
11. Shi Y., Song Y., Sun W. Surrogate-assisted multi-objective genetic algorithm for computationally expensive problems // Applied Soft Computing. 2021. Vol. 111. P. 107637.
12. Galván E., Mooney P. Neuroevolution in deep neural networks: Current trends and future challenges // IEEE Transactions on Artificial Intelligence. 2021. Vol. 2, No. 6. P. 476–493.
13. Papavasileiou E., Cornelis J., Jansen B. A systematic literature review of the successors of “neuroevolution of augmenting topologies” // Evolutionary Computation. 2021. Vol. 29, No. 1. P. 1–73.
14. Katoch S., Chauhan S. S., Kumar V. A review on genetic algorithm: past, present, and future // Multimedia Tools and Applications. 2021. Vol. 80. P. 8091–8126.
15. Alhijawi B., Awajan A. Genetic algorithms: Theory, genetic operators, solutions, and applications // Evolutionary Intelligence. 2024. Vol. 17, No. 3. P. 1245–1256.
16. Sohail A. Genetic algorithms in the fields of artificial intelligence and data sciences // Annals of Data Science. 2023. Vol. 10, No. 4. P. 1007–1018.
17. Schubert E. Stop using the elbow criterion for k-means and how to choose the number of clusters instead // ACM SIGKDD Explorations Newsletter. 2023. Vol. 25, No. 1. P. 36–42.
18. Jamil M., Yang X. S., Zepernick H. J. Test functions for global optimization: a comprehensive survey // Swarm intelligence and Bio-inspired Computation. 2013. P. 193–222.
19. Wu G., Mallipeddi R., Suganthan P. N. Problem definitions and evaluation criteria for the CEC 2017 competition on constrained real-parameter optimization // National University of Defense Tech-

nology, Changsha, Hunan, PR China and Kyungpook National University, Daegu, South Korea and Nanyang Technological University, Singapore, Technical Report. 2017. Vol. 9. P. 2017.

20. Шерстнев П. А., Семенкин Е. С. SelfCSHAGA: Самоконфигурируемый генетический алгоритм оптимизации с адаптацией на основе истории успеха // Вестник МГТУ им. Н. Э. Баумана. Сер. Приборостроение. 2025. № 2 (151). С. 122–139.

21. Stewart G., Al-Khassaweneh M. An implementation of the HDBSCAN* clustering algorithm // Applied Sciences. 2022. Vol. 12, No. 5. P. 2405.

22. Gorokhovatskyi V. et al. Application a committee of Kohonen neural networks to training of image classifier based on description of descriptors set. IEEE Access, 2024.

References

1. Semenkin E., Semenkina M. Spacecrafts' control systems effective variants choice with self-configuring genetic algorithm // ICINCO 2012 – Proceedings of the 9th International Conference on Informatics in Control, Automation and Robotics. Rome, 28–31 July 2012. 2012, Vol. 1, P. 84–93.

2. Gamot J. et al. Hidden-variables genetic algorithm for variable-size design space optimal layout problems with application to aerospace vehicles. Engineering Applications of Artificial Intelligence. 2023, Vol. 121, P. 105941.

3. Srinivas M., Patnaik L. Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*. 1994, Vol. 24, No. 4, P. 656–667.

4. Liu C., Liu K. An asynchronous island-model genetic algorithm for large-scale global optimization. *Applied Soft Computing*, 2019, Vol. 76, P. 637–649.

5. Izzo D., Rucinski M., Ampatzis C. Parallel global optimisation meta-heuristics using an asynchronous island-model. *Proc. 2009 IEEE Congress on Evolutionary Computation*, IEEE, 2009, P. 2301–2308.

6. Gendreau M., D'Amours G., Rousseau S. Memetic algorithms: Hybrid genetic algorithms and local search methods for combinatorial optimization. *Computers & Operations Research*, 2020, Vol. 112, P. 104778.

7. Verma S., Pant M., Snasel V. A comprehensive review on NSGA-II for multi-objective combinatorial optimization problems. *IEEE access*. 2021, Vol. 9, P. 57757–57791.

8. Wang Z. et al. Adaptive replacement strategies for MOEA/D. *IEEE transactions on cybernetics*. 2015, Vol. 46, No. 2, P. 474–486.

9. Bleuler S. et al. Multiobjective genetic programming: Reducing bloat using SPEA2. *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No. 01TH8546)*. IEEE, 2001, Vol. 1, P. 536–543.

10. Jain L., Sinha A. Adaptive reference point selection in NSGA-III for many-objective optimization. *Applied Soft Computing*. 2021, Vol. 105, P. 107237.

11. Shi Y., Song Y., Sun W. Surrogate-assisted multi-objective genetic algorithm for computationally expensive problems. *Applied Soft Computing*. 2021, Vol. 111, P. 107637.

12. Galván E., Mooney P. Neuroevolution in deep neural networks: Current trends and future challenges. *IEEE Transactions on Artificial Intelligence*. 2021, Vol. 2, No. 6, P. 476–493.

13. Papavasileiou E., Cornelis J., Jansen B. A systematic literature review of the successors of “neuroevolution of augmenting topologies”. *Evolutionary Computation*. 2021, Vol. 29, No. 1, P. 1–73.

14. Katoch S., Chauhan S. S., Kumar V. A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*. 2021, Vol. 80, P. 8091–8126.

15. Alhijawi B., Awajan A. Genetic algorithms: Theory, genetic operators, solutions, and applications. *Evolutionary Intelligence*. 2024, Vol. 17, No. 3, P. 1245–1256.

16. Sohail A. Genetic algorithms in the fields of artificial intelligence and data sciences. *Annals of Data Science*. 2023, Vol. 10, No. 4, P. 1007–1018.

17. Schubert E. Stop using the elbow criterion for k-means and how to choose the number of clusters instead. *ACM SIGKDD Explorations Newsletter*. 2023, Vol. 25, No. 1, P. 36–42.

18. Jamil M., Yang X. S., Zepernick H. J. Test functions for global optimization: a comprehensive survey. *Swarm intelligence and Bio-inspired Computation*. 2013, P. 193–222.
19. Wu G., Mallipeddi R., Suganthan P. N. Problem definitions and evaluation criteria for the CEC 2017 competition on constrained real-parameter optimization. *National University of Defense Technology, Changsha, Hunan, PR China and Kyungpook National University, Daegu, South Korea and Nanyang Technological University, Singapore, Technical Report*. 2017, Vol. 9, P. 2017.
20. Sherstnev P. A., Semenko E. S. [SelfCSHAGA: Self-configuring genetic optimization algorithm with adaptation based on success history]. *Vestnik MGTU im. N. E. Baumana. Ser. Priborostroenie*. 2025, No. 2 (151), P. 122–139 (In Russ.).
21. Stewart G., Al-Khassaweneh M. An implementation of the HDBSCAN* clustering algorithm. *Applied Sciences*. 2022, Vol. 12, No. 5, P. 2405.
22. Gorokhovatskyi V. et al. Application a committee of Kohonen neural networks to training of image classifier based on description of descriptors set. *IEEE Access*, 2024.

© Малашин И. П., Тынченко В. С., 2025

Малашин Иван Павлович – инженер-программист; Московский государственный технический университет имени Н. Э. Баумана (национальный исследовательский университет). E-mail: ivan.p.malashin@gmail.ru. <https://orcid.org/0009-0008-8986-402X>.

Тынченко Вадим Сергеевич – доктор технических наук, доцент; Сибирский государственный университет науки и технологий имени академика М. Ф. Решетнева. E-mail: vadimond@mail.ru. <https://orcid.org/0000-0002-3959-2969>.

Malashin Ivan Pavlovich – Software engineer; Bauman Moscow State Technical University (National Research University). E-mail: ivan.p.malashin@gmail.ru. <https://orcid.org/0009-0008-8986-402X>.

Tynchenko Vadim Sergeevich – Dr. Sc., Associate Professor; Reshetnev Siberian State University of Science and Technology. E-mail: vadimond@mail.ru. <https://orcid.org/0000-0002-3959-2969>.

Статья поступила в редакцию 19.06.2025; принята к публикации 12.09.2025; опубликована 13.10.2025
The article was submitted 19.06.2025; accepted for publication 12.09.2025; published 13.10.2025

Статья доступна по лицензии Creative Commons Attribution 4.0
The article can be used under the Creative Commons Attribution 4.0 License