Раздел I. Системы управления и моделирование

УДК 004.832.23 DOI 10.18522/2311-3103-2024-2-6-16

С.И. Родзин

ВЫЧИСЛИТЕЛЬНАЯ МОДЕЛЬ КОЛЛЕКТИВНОГО ПОВЕДЕНИЯ ГРУППЫ ЖИВОТНЫХ: ЭФФЕКТИВНАЯ БИОЭВРИСТИКА ДЛЯ РЕШЕНИЯ ПРИКЛАДНЫХ ЗАДАЧ ГЛОБАЛЬНОЙ ОПТИМИЗАЦИИ*

Перспективным решением задач глобальной оптимизации являются метаэвристики, инспирированные природой, представляющие собой недетерминированные алгоритмы, исследующие пространство поиска, решений, обучающиеся в процессе поиска, не привязанные к конкретной задаче, хотя и не гарантирующие точное решений. Целью данного исследования является разработка эффективного алгоритма для решения прикладных проблем глобальной оптимизации многомерных одномодальных и мультимодальных функций, встречающихся в задачах инженерного проектирования, обработки изображений и компьютерного зрения, энергетики и энергоменеджмента, анализа данных и машинного обучения, робототехники. Для достижения этой цели в статье предлагается вычислительная модель коллективного поведения группы животных и эффективный алгоритм дифференциально-векторного движения. Модель включает разнообразные паттерны поведения в группе животных: удерживать текущую позицию; двигаться в направлении к ближайшим соседям или, наоборот, от ближайших соседей; двигаться случайным образом; конкурировать за позицию. В коллективной памяти хранится информация о местоположении доминирующих особей группы и направлении движения группы, лучшие позиции агентов с учетом механизмов конкуренции и доминирования в группе. Алгоритм был экспериментально протестирован на семи известных многомерных одномодальных и мультимодальных функциях. Результаты были сопоставлены с генетическим алгоритмом, алгоритмом роя частиц, гравитационного поиска дифференциальной эволюции. Предлагаемый алгоритм показал лучшие результаты, нежели конкурирующие алгоритмы, на всех тестовых функциях. Это объясняется лучшим балансом нового алгоритма между скоростью сходимости и диверсификацией пространства поиска решений. Проверка полученных результатов с использованием Т-критерия суммы рангов Уилкоксона для независимых выборок показала, что результаты по алгоритму являются статистически значимыми. Также проводилось сравнение с одним из наиболее эффективных алгоритмов непрерывной оптимизации BFGS - квазиньютоновским итерационным алгоритмом численной оптимизации, предназначенным для нахождения локального экстремума одномодальных функций. Результаты оказались сопоставимы для многомерных функций. Алгоритм также сравнивался с методом мультистарта в задаче глобальной оптимизации мультиэкстремальных функций и доказал свое преимущество по времени и точности найденных решений.

Биоэвристика; глобальная оптимизация; дифференциально-векторное движение; память; многомерная функция; агент; оператор; популяция; критерий Уилкоксона.

S.I. Rodzin

A COMPUTATIONAL MODEL OF THE COLLECTIVE BEHAVIOR OF A GROUP OF ANIMALS: EFFECTIVE BIO HEURISTICS FOR SOLVING APPLIED GLOBAL OPTIMIZATION PROBLEMS

A promising solution to global optimization problems are metaheuristics inspired by nature, which are non-deterministic algorithms that explore the search space, solutions, learning in the search process, not tied to a specific task, although they do not guarantee accurate solutions. The purpose of this study is

* Исследование выполнено за счет гранта Российского научного фонда № 23-21-00089, https://rscf.ru/project/23-21-00089/ в Южном федеральном университете.

to develop an effective algorithm for solving applied problems of global optimization of multidimensional single-modal and multimodal functions found in engineering design, image processing and computer vision, energy and energy management, data analysis and machine learning, robotics. To achieve this goal, the article proposes a computational model of the collective behavior of a group of animals and an effective algorithm for differential vector motion. The model includes various patterns of behavior in a group of animals: to hold the current position; to move towards the nearest neighbors or, conversely, from the nearest neighbors; to move randomly; to compete for a position. The collective memory stores information about the location of the dominant individuals of the group and the direction of movement of the group, the best positions of agents, taking into account the mechanisms of competition and dominance in the group. The algorithm was experimentally tested on seven known multidimensional single-modal and multimodal functions. The results were compared with a genetic algorithm, a particle swarm algorithm, and a gravitational search for differential evolution. The proposed algorithm showed better results than competing algorithms on all test functions. This is due to the better balance of the new algorithm between the rate of convergence and the diversification of the solution search space. Verification of the results obtained using the Wilcoxon sum of ranks T-test for independent samples showed that the results of the algorithm are statistically significant. A comparison was also made with one of the most effective continuous optimization algorithms of BFGS - a quasi-Newtonian iterative numerical optimization algorithm designed to find the local extremum of single-modal functions. The results were comparable for multidimensional functions. The algorithm was also compared with the multistart method in the problem of global optimization of multi-extreme functions and proved its advantage in terms of time and accuracy of the solutions found.

Bioheuristics; global optimization; differential vector motion; memory; multidimensional function; agent; operator; population; Wilcoxon criterion.

Введение. Найти произвольный локальный оптимум относительно просто, используя классические методы оптимизации. Найти глобальный оптимум функции гораздо труднее. Детерминированные методы такие как ветвей и границ, численные методы, методы алгебраической геометрии часто неприменимы, дают лишь теоретическую гарантию нахождения локального оптимума. Перспективным решением задачи глобальной оптимизации являются метаэвристики, представляющие собой недетерминированные алгоритмы, исследующие пространство поиска, решений, обучающиеся в процессе поиска, не привязанные к конкретной задаче, хотя и не гарантирующие точное решений. Активной областью исследований является разработка метаэвристик, инспирированных природой. Эта область включает эволюционные и роевые биоэвристики [1], биоэвристики, основанные на физических и химических процессах, на особенностях многоклеточных организмов, способных к фотосинтезу, а также биоэвристики, инспирированные когнитивными процессами и деятельностью человека [2]. Здесь природа выступает источником концепций, механизмов и принципов проектирования искусственных вычислительных систем для решения сложных вычислительных задач глобальной оптимизации, является примером адаптивного решения задачи оптимизации.

В настоящее время известно свыше 500 метаэвристик [3]. Наиболее цитируемым является роевой алгоритм (*Particle Swarm Optimization*, *PSO*) [4]. По результатам решения трех инженерных тестовых задач (растяжение/сжатие пружины), проектирование сосуда высокого давления), сварка балки) были определены четыре лучших биоэвристики: алгоритм императорских пингвинов (*Emperor Penguin Optimization*, *EPO*), алгоритм охоты орлов (*Aquila Optimizer*, *AO*), алгоритм хамелеона (*Chameleon Swarm Algorithm*, *ChSA*) и алгоритм африканских стервятников (*African Vulture Optimization Algorithm*, *AVOA*) [5].

Известными примерами приложений глобальной оптимизации является вычислительная филогенетика, проблема коммивояжера и проектирование электрических схем, инженерная безопасность зданий, гипотеза Кеплера, задача упаковки, калибровка моделей распространения радиосигналов [6].

Около половины существующих метаэвристик инспирированы феноменами поведения животных [3, 7]:

1. Птиц:

- гнездовой паразитизм кукушек, *Cuckoo Search, CS*;
- стая кур в поисках пищи, *Chicken Swarm Optimization, CSO*;
- поиск пищи, ее хранение и воровство вороной, *Crow Search Algorithm, CSA*;

- поиск пищи совой, *Owl Search Algorithm, OSA*;
- поиск пищи колибри, *Hummingbird's Optimization Algorithm, HOA*;
- кооперативное охотничье поведение ястребов, *Harris Hawks Optimization, HHO*;
- охота беркута, *Golden Eagle Optimizer, GEO*;
- полет голубей, *Pigeon Optimization Algorithm, POA*;
- миграция чаек, *Seagull Optimization Algorithm, SeOA*;
- выживание в суровых условиях полярных регионов императорских пингвинов, *Emperor Penguin Optimizer*, *EPO*;
- полет африканских стервятников в поисках пищи, *AVOA*;

2. Млекопитающих:

- летучих мышей, Bat Algorithm, BA;
- дельфинов, Dolphin Echolocation, DEO;
- кашалотов, Sperm Whale Algorithm, SWA;
- горбатых китов, Whale Optimization Algorithm, WOA;
- серых волков, Grey Wolf Optimizer, GWO;
- паукообразных обезьян, Spider Monkey Optimization, SMO;
- горилл, Gorilla Troops Optimizer, GTO;
- пятнистых гиен, Spotted Hyena Optimizer, SHO;
- белок-летяг, Squirrel Search Algorithm, SSA;
- львов, Lion's Algorithm, LA;
- слонов, Elephant Search Algorithm, ESA;

3. Насекомых:

- муравьиной колонии, Ant Colony Optimization, ACO;
- пчелиной колонии, Artificial Bee Colony, ABC;
- мухи дрозофилы, Drosophila Food Search Optimization, DFO;
- самок комаров, Mosquito Host Seeking Algorithm, MHSA;
- бабочек-монархов, Monarch Butterfly Optimization, MBO;
- мотыльков, Moth Flame Optimization, MFO;
- саранчи, Locust Swarm, LS;
- кузнечиков, Grasshopper Optimization Algorithm, GOA;
- стрекоз, Dragonfly Algorithm, DA [8];
- жуков короедов, Pity Beetle Algorithm, PBA [9].

В большинстве из упомянутых алгоритмов для моделирования коллективного поведения животных используется концепция их индивидуального поведения. Ключевой принцип индивидуального поведения состоит в том, что простые повторяющиеся взаимодействия между агентами порождают сложные поведенческие паттерны на уровне колонии или стаи. Примерами являются сеть муравьиных феромонных следов, танцы пчел-разведчиков, миграция рыбных косяков.

Однако исследования в [10] показали существование коллективной памяти в группах животных. Наличие такой памяти свидетельствует о том, что предыстория групповой структуры влияет на коллективное поведение на последующих этапах. С учетом этого появляется возможность моделировать коллективное поведение группы с помощью простых индивидуальных правил и настройки общей памяти.

В статье рассматривается биологическая и вычислительная модель коллективной памяти стаи животных в поисках пищи, а также предлагается эффективная биоэвристика для решения прикладных задач глобальной оптимизации.

В алгоритме поисковые агенты моделируют группу животных, взаимодействующих друг с другом на основе простых поведенческих правил, реализуемых в виде математических операторов. Операторы применяются к каждому агенту с учетом того, что вся группа имеет общую память, в которой хранятся их собственные лучшие позиции. Подход сопоставляется с известными метаэвристиками, а получаемые результаты свидетельствуют об эффективности алгоритма.

Биологическая модель. Паттерны коллективного поведения таких организмов, как муравьи, рыбы, птицы уже давно привлекали внимание натуралистов и ученых. Однако несмотря на долгую историю научных исследований, связь между индивидуальными и групповыми паттернами поведения начала устанавливаться не так давно [11].

Например, агенты в группе часто вынуждены быстро принимать решение о направлении движения в неопределенной и опасной среде. Группа может состоять из агентов, имеющих различный статус и не осведомленных о состоянии других индивидов или об угрозе. В соответствии с принципом конкуренции и доминирования группа может иметь иерархическую структуру. В [12] установлено, что такая структура приводит к более стабильным группам с лучшими свойствами сплоченности среди агентов. Повторяющиеся взаимодействия в группе масштабируются до коллективного поведения и принятия решений в широком диапазоне типов групп животных от насекомых до птиц. Даже среди людей в определенных обстоятельствах имеют место быть сходные паттерны поведения. При этом коллективное принятие решений имеет существенную черту – общую память.

Несмотря на разнообразие передвижений группы животных, многие из коллективных паттернов порождаются простыми правилами, которым следуют отдельные агенты группы: (1) удерживать текущую позицию; (2) двигаться в направлении к ближайшим соседям или, наоборот, от ближайших соседей; (3) двигаться случайным образом; (4) конкурировать за определенную позицию. Выбор правила определяется в соответствии с внутренней мотивацией индивида.

Однако существование коллективной памяти в группах животных влияет на коллективное поведение. Так в коллективной памяти может храниться информация о местоположении доминирующих особей группы или направлении движения группы. Это открывает возможности для моделирования сложного коллективного поведения группы, используя простые индивидуальные правила и общую память. В этой работе при построении алгоритма коллективного поведения животных используется поведенческая модель группы животных, что позволяет определить новые операторы поиска оптимального решения. Также используется коллективная память для хранения лучших позиций агентов (лучших решений) с учетом механизмов конкуренции и доминирования в группе.

Алгоритм коллективного поведения животных (АКПЖ). Алгоритм АКПЖ предполагает наличие набора операторов, реализующих правила взаимодействия и коллективного поведения животных. Каждое решение в пространстве поиска оптимума представляет собой позицию животного. В общей памяти хранятся наилучшие решения в каждом поколении ($\boldsymbol{M}_g$) и наилучшие решения в течение всех предыдущих поколений ($\boldsymbol{M}_h$).

Алгоритм АКПЖ является итерационным, начинается со случайной инициализации популяции решений и определения наилучшей позиции в группе. Затем применяются операторы алгоритма пока не будет выполнен критерий останова. Критерием завершения алгоритма является заранее заданное число итераций *NI*. АКПЖ относится к классу алгоритмов дифференциально-векторного движения, характерной чертой которых является использование механизмов рекомбинации имеющихся репрезентативных решений для создания новых решений [5]. Операторы алгоритма реализуют передвижения относительно ближайших соседей, случайные передвижения, а также обновление общей памяти.

Рассмотрим шаги алгоритма подробнее.

Инициализация популяции.

Алгоритм начинается с инициализации позиций популяции животных (агентов) $\boldsymbol{A} = \{\boldsymbol{a}_1, \boldsymbol{a}_2, \dots, \boldsymbol{a}_{N_p}\}$, где N_p – размер популяции. Каждая позиция агента $\boldsymbol{a}_l$ представляет собой D-мерный вектор, содержащий значения параметров, подлежащих оптимизации. Эти значения случайным образом и равномерно распределяются между заранее заданной начальной нижней и верхней границей j-го параметра a_j^{low} и a_j^{high} соответственно:

$$a_{i,j} = a_j^{low} + rand(0,1) \cdot \left(a_j^{high} - a_j^{low}\right), j = 1,2, \dots, D; i = 1,2, \dots, N_p, \qquad (1)$$

Здесь $a_{i,j}$ – это j-й параметр i-го агента.

Все начальные позиции $\boldsymbol{A}$ сортируются в соответствии с фитнесс функцией приспособленности (доминирования) для формирования новой популяции $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_{N_p}\}$ с последующим сохранением в памяти наилучших решений в каждом поколении ($\boldsymbol{M}_g$) и наилучших решений в течение всех предыдущих поколений ($\boldsymbol{M}_h$).

Определение наилучшей позиции в популяции.

В процессе определения наилучшей позиции генерируются первые B $\{\boldsymbol{a}_1, \boldsymbol{a}_2, \dots, \boldsymbol{a}_B\}$ новых позиций агентов популяции $\boldsymbol{A}$. Эти позиции вычисляются с учетом значений, содержащихся в памяти всех предыдущих поколений $\boldsymbol{M}_h$, с небольшим случайным возмущением вокруг них:

$$\boldsymbol{a}_l = \boldsymbol{m}_h^l + \boldsymbol{v}, \tag{2}$$

где $l \in \{1, 2, \dots, B\}$, $\boldsymbol{m}_h^l$ – l-й элемент памяти $\boldsymbol{M}_h$, $\boldsymbol{v}$ – случайный вектор малой длины.

Оператор передвижения относительно ближайших соседей.

Согласно биологической метафоре, животные в соответствии с внутренней мотивацией случайно сближаются или отдаляются от ближайших соседей. Необходимы новые операторы, моделирующие такой биологический паттерн. С этой целью генерируется случайное число r в диапазоне [0, 1]. Если r не больше некоторого порогового значения H, то индивидуальная позиция агента перемещается (сближается или отдаляется) относительно ближайшей наилучшей позиции из $\boldsymbol{M}_h$, иначе – перемещается к ближайшей наилучшей позиции из $\boldsymbol{M}_g$:

$$\boldsymbol{a}_i = \begin{cases} \boldsymbol{x}_i \pm r \cdot \left(\boldsymbol{m}_h^{\text{ближ}} - \boldsymbol{x}_i\right), \text{если } r \le H \\ \boldsymbol{x}_i \pm r \cdot \left(\boldsymbol{m}_g^{\text{ближ}} - \boldsymbol{x}_i\right), \text{если } r > H \end{cases}, \tag{3}$$

где $i \in \{B+1, B+2, \dots, N_p\}$, $\boldsymbol{m}_h^{\text{ближ}}$ и $\boldsymbol{m}_g^{\text{ближ}}$ – ближайшие наилучшие позиции из $\boldsymbol{M}_h$ и $\boldsymbol{M}_g$ для $\boldsymbol{x}_i$.

Оператор случайного передвижения.

Следуя биологической модели, один агент с некоторой вероятностью P случайным образом изменяет свою позицию. Это поведенческое правило реализуется с учетом следующего выражения:

$$\boldsymbol{a}_i = \begin{cases} \boldsymbol{r} \text{ с вероятностью } P \\ \boldsymbol{x}_i \text{ с вероятностью } (1 - P) \end{cases}, \tag{4}$$

где $i \in \{B+1, B+2, \dots, N_p\}$, $\boldsymbol{r}$ – случайный вектор, определяемый в пространстве поиска. Этот оператор аналогичен повторной инициализации агента в случайной позиции, как это указано в (1).

Обновление общей памяти.

После выполнения операторов определения наилучшей позиции в популяции, передвижения относительно ближайших соседей и случайного передвижения для всех N_p агентов, необходимо обновить память $\boldsymbol{M}_h$.

Чтобы актуализировать $\boldsymbol{M}_h$, используется механизм доминирования. Животные, взаимодействующие внутри группы, поддерживают между собой минимальную дистанцию ρ. Эта дистанция зависит от того, насколько агрессивно ведет себя животное. Следовательно, когда два животных противостоят друг другу на таком расстоянии, преобладает наиболее доминирующая особь тем временем другие удаляются.

В алгоритме АКПЖ память $\boldsymbol{M}_h$ обновляется согласно следующей процедуры:

1. Элементы $\boldsymbol{M}_h$ и $\boldsymbol{M}_g$ объединяются: $\boldsymbol{M}_U = \boldsymbol{M}_h \cup \boldsymbol{M}_g$.

2. Каждый элемент $\boldsymbol{m}_U^i \in \boldsymbol{M_U}$ сравнивается попарно с оставшимися элементами памяти $\{\boldsymbol{m}_U^1, \boldsymbol{m}_U^2, \dots, \boldsymbol{m}_U^{2B-1}\}$. Если расстояние между сравниваемыми элементами меньше ρ, то преимущество получает элемент с лучшим значением фитнес-функции, а другой удаляется.

3. Из полученных на предыдущем шаге элементов $\boldsymbol{M}_U$ выбирается наилучшее значение B для построения нового $\boldsymbol{M}_h$.

Значение ρ влияет на скорость сходимости алгоритма и время вычислений, поэтому параметр ρ вычисляется с учетом следующего уравнения:

$$\rho = \frac{\prod_{j=1}^{D}(a_j^{high} - a_j^{low})}{10 \cdot D}, \quad (5)$$

где a_j^{high} и a_j^{low} представляют собой предварительно заданные нижнюю и верхнюю границы j-го параметра соответственно в D-мерном пространстве.

Алгоритм АКПЖ включает следующие шаги.

Шаг 1. Установка параметров N_p, B, H, P и NI.

Шаг 2. Случайная инициализация позиций популяции агентов $\boldsymbol{A} = \{\boldsymbol{a}_1, \boldsymbol{a}_2, \dots, \boldsymbol{a}_{N_p}\}$ согласно (1).

Шаг 3. Сортировка позиций $\boldsymbol{A}$ в соответствии с фитнесс функцией для формирования новой популяции $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_{N_p}\}$ с последующим сохранением в памяти наилучших решений в каждом поколении $\boldsymbol{M}_g$ и наилучших решений в течение всех предыдущих поколений $\boldsymbol{M}_h$.

Шаг 4. Выбор первых B позиций из $\boldsymbol{A}$ и сохранение их в памяти $\boldsymbol{M}_g$.

Шаг 5. Обновление $\boldsymbol{M}_h$ (на первой итерации $\boldsymbol{M}_h = \boldsymbol{M}_g$).

Шаг 6. Вычисление первых позиций B новой популяции решений $\boldsymbol{A} = \{\boldsymbol{a}_1, \boldsymbol{a}_2, \dots, \boldsymbol{a}_B\}$ согласно (2).

Шаг 7. Вычисление остальных элементов, используя операторы передвижения относительно ближайших соседей и случайного передвижения:

for $i=B+1$: N_p
if ($r_1<P$) *then*
оператор передвижения относительно ближайших соседей
{***if*** ($r_2<H$) *then*
$\boldsymbol{a}_i = \boldsymbol{x}_i \pm r \cdot (\boldsymbol{m}_h^{\text{ближайший}} - \boldsymbol{x}_i)$
else if
$\boldsymbol{a}_i = \boldsymbol{x}_i \pm r \cdot (\boldsymbol{m}_g^{\text{ближайший}} - \boldsymbol{x}_i)$
}
else if
оператор случайного передвижения
{
$\boldsymbol{a}_i = \mathbf{r}$
}
end *for* (r_1, $r_2 \in$ *rand*(0,1))

Шаг 8. Если условие останова NI выполнено, то алгоритм завершается; иначе – возврат к шагу 3.

Наилучшее значение в $\boldsymbol{M}_h$ представляет глобальное решение задачи оптимизации.

Результаты экспериментов. Для оценки эффективности представленного алгоритма использовался набор из 7 тестовых функций, представленных в [13], где приводится описание нескольких десятков унимодальных и мультиэкстремальных функций, а также результаты их тестирования различными метаэвристиками для многомерных задач глобальной оптимизации:

- $f_1(\boldsymbol{X}) = \sum_{i=1}^{n} x_i^2, x_i \in [-100,100]^n, f_{opt} = 0;$
- $f_2(\boldsymbol{X}) = \sum_{i=1}^{n} |x_i| + \prod_{i=1}^{n} |x_i|, x_i \in [-10,10]^n, f_{opt} = 0;$
- $f_3(\boldsymbol{X}) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2, x_i \in [-100,100]^n, f_{opt} = 0;$
- $f_4(\boldsymbol{X}) = \sum_{i=1}^{n-1} [100(x_{i-1} - x_i^2)^2 + (x_i - 1)^2], x_i \in [-30,30]^n, f_{opt} = 0;$
- $f_5(\boldsymbol{X}) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{|x_i|}\right), x_i \in [-500,500]^n, f_{opt} = -418{,}98 * n;$
- $f_6(\boldsymbol{X}) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10], x_i \in [-5{,}12; 5{,}12]^n, f_{opt} = 0;$
- $f_7(\boldsymbol{X}) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right), x_i \in [-600,600]^n, f_{opt} = 0.$

Здесь n – размерность функции (n устанавливалась равной 30), f_{opt} – минимальное значение функции.

Чтобы продемонстрировать эффективность, а также вычислительные характеристики предложенного алгоритма создана программная среда на языке программирования *C#*. Тестирование проводилось на персональном компьютере с процессором *Intel Core i7* с ОЗУ-8 Гб в среде *Windows* 10.

Вначале для исследования влияния параметров P и H на результативность алгоритма были протестированы четыре функции: f_1, f_3, f_4, f_7. При этом максимальное число итераций $NI = 1000$, размер популяции $N_p = 50$, $B = 10$. Поиск глобального минимума для каждой из функций проводился 30 раз.

В табл. 1 представлены результаты экспериментов по алгоритму АКПЖ: наилучшие значения для каждой функции μ и дисперсия σ^2, усредненные за 30 прогонов при фиксированном значении параметра $H = 0{,}8$ и изменении P от 0,5 до 0,9 с шагом 0,1.

Таблица 1

Результаты АКПЖ с различными значениями параметра *P* для функций f_1, f_3, f_4, f_7 при $H = 0{,}8$

Функция	f_1	f_3	f_4	f_7
P=0,5; μ (σ^2)	$2{,}63\cdot10^{-11}$ ($2{,}13\cdot10^{-12}$)	$5{,}71\cdot10^{-13}$ ($1{,}11\cdot10^{-14}$)	$5{,}68\cdot10^{-11}$ ($2{,}21\cdot10^{-12}$)	$1{,}57\cdot10^{-2}$ ($1{,}25\cdot10^{-3}$)
P=0,6; μ (σ^2)	$1{,}98\cdot10^{-17}$ ($6{,}51\cdot10^{-18}$)	$7{,}78\cdot10^{-19}$ ($1{,}52\cdot10^{-20}$)	$1{,}54\cdot10^{-17}$ ($1{,}68\cdot10^{-18}$)	$1{,}14\cdot10^{-6}$ ($3{,}71\cdot10^{-7}$)
P=0,7; μ (σ^2)	$1{,}28\cdot10^{-23}$ ($3{,}54\cdot10^{-24}$)	$4{,}47\cdot10^{-27}$ ($3{,}60\cdot10^{-28}$)	$5{,}11\cdot10^{-22}$ ($4{,}42\cdot10^{-23}$)	$2{,}81\cdot10^{-8}$ ($5{,}21\cdot10^{-9}$)
P=0,8; μ (σ^2)	**$2{,}33\cdot10^{-29}$ ($4{,}41\cdot10^{-30}$)**	**$7{,}62\cdot10^{-31}$ ($4{,}23\cdot10^{-32}$)**	**$9{,}02\cdot10^{-28}$ ($6{,}77\cdot10^{-29}$)**	**$4{,}21\cdot10^{-10}$ ($4{,}87\cdot10^{-11}$)**
P=0,9; μ (σ^2)	$4{,}53\cdot10^{-23}$ ($5{,}12\cdot10^{-24}$)	$3{,}42\cdot10^{-26}$ ($3{,}54\cdot10^{-27}$)	$4{,}77\cdot10^{-20}$ ($1{,}94\cdot10^{-21}$)	$4{,}58\cdot10^{-4}$ ($6{,}92\cdot10^{-5}$)

В табл. 2 представлены результаты экспериментов по алгоритму АКПЖ: наилучшие значения для каждой функции μ и дисперсия σ^2, усредненные за 30 прогонов при фиксированном значении параметра $P = 0{,}8$ и изменении H от 0,5 до 0,9 с шагом 0,1.

Для наглядности наилучшие результаты выделены шрифтом. Результаты свидетельствуют о том, что правильное сочетание различных значений параметров повышает точность решения.

Сравнение производительности АКПЖ на 7 тестовых функциях производилось с результатами, полученными с помощью генетического алгоритма (*GA*) [14], алгоритма роя частиц (*PSO*) [15], алгоритм гравитационного поиска (*GSA*) [16] и алгоритма дифференциальной эволюции (*DE*) [17]. Для корректного сравнения устанавливались одни и те же общие параметры управления: размер популяции $N_p = 50$, максимальное число итераций $NI = 1000$.

Таблица 2

Результаты АКПЖ с различными значениями параметра *H* для функций f_1, f_3, f_4, f_7 при $P = 0{,}8$

Функция	f_1	f_3	f_4	f_7
H=0,5; μ (σ^2)	$2{,}23\cdot10^{-10}$ ($8{,}92\cdot10^{-11}$)	$5{,}71\cdot10^{-10}$ ($5{,}12\cdot10^{-11}$)	$8{,}80\cdot10^{-9}$ ($5{,}55\cdot10^{-10}$)	$1{,}81\cdot10^{-4}$ ($2{,}16\cdot10^{-5}$)
H=0,6; μ (σ^2)	$3{,}35\cdot10^{-18}$ ($3{,}21\cdot10^{-19}$)	$3{,}24\cdot10^{-18}$ ($1{,}32\cdot10^{-19}$)	$6{,}72\cdot10^{-21}$ ($1{,}11\cdot10^{-22}$)	$2{,}89\cdot10^{-6}$ ($6{,}43\cdot10^{-7}$)
H=0,7; μ (σ^2)	$3{,}85\cdot10^{-22}$ ($6{,}78\cdot10^{-23}$)	$6{,}29\cdot10^{-27}$ ($8{,}26\cdot10^{-28}$)	$1{,}69\cdot10^{-23}$ ($1{,}34\cdot10^{-24}$)	$2{,}36\cdot10^{-7}$ ($3{,}75\cdot10^{-8}$)
H=0,8; μ (σ^2)	**$2{,}33\cdot10^{-29}$ ($4{,}41\cdot10^{-30}$)**	**$7{,}62\cdot10^{-31}$ ($4{,}23\cdot10^{-32}$)**	**$9{,}02\cdot10^{-28}$ ($6{,}77\cdot10^{-29}$)**	**$4{,}21\cdot10^{-10}$ ($4{,}87\cdot10^{-11}$)**
H=0,9; μ (σ^2)	$4{,}72\cdot10^{-21}$ ($6{,}29\cdot10^{-22}$)	$5{,}41\cdot10^{-22}$ ($5{,}28\cdot10^{-23}$)	$7{,}39\cdot10^{-21}$ ($4{,}41\cdot10^{-22}$)	$3{,}02\cdot10^{-4}$ ($4{,}37\cdot10^{-6}$)

Настройки параметров для каждого из конкурирующих алгоритмов соответствовали оригинальным работам:

♦ согласно [14] *GA* использует арифметический кроссовер, гауссовскую мутацию и селекцию колеса рулетки. Вероятности операторов кроссинговера и мутации были установлены на 0,3 и 0,1 соответственно;

♦ в алгоритме *PSO* устанавливались параметры $c_1 = c_2 = 2$, а коэффициент инерции (ω) линейно уменьшается от 0,9 до 0,2;

♦ в алгоритме *GSA* параметры $G_0 = 100$, $\alpha = 20$; $T = 1000$. В [16] эти значения были установлены в качестве наилучших;

♦ в алгоритме *DE* согласно [17] вероятность кроссинговера равна 0,9, а весовой коэффициент F равен 0,8.

Результаты сравнения алгоритма АКПЖ с конкурирующими алгоритмами *GA, PSO, GSA, DE* на тестовых функциях $f_1, f_2, f_3, f_4, f_5, f_6, f_7$ приведены в табл. 3.

Сравнение проводилось по показателям среднее наилучшее значение функции (*av*) и медиана наилучшего значения функции на последней итерации (*med*). Наилучший результат для каждой функции выделен шрифтом.

Согласно табл. 3 АКПЖ показал лучшие результаты, нежели конкурирующие алгоритмы *GA, PSO,GSA* и *DE* на всех семи тестовых функциях. Это объясняется лучшим балансом АКПЖ между скоростью сходимости и диверсификацией пространства поиска решений.

Таблица 3

Результаты АКПЖ в сравнении с алгоритмами *GA, PSO, GSA, DE*

Функция		f_1	f_2	f_3	f_4	f_5	f_6	f_7
GA	*av*	23,13	1,07	$5{,}6\cdot10^{3}$	$1{,}1\cdot10^{3}$	**$-1{,}2\cdot10^{4}$**	5,90	1,16
	med	21,87	1,13	$5{,}6\cdot10^{3}$	$1{,}0\cdot10^{3}$	**$-1{,}2\cdot10^{4}$**	5,71	1,14
PSO	*av*	$1{,}8\cdot10^{-3}$	2,0	$4{,}1\cdot10^{-3}$	$3{,}6\cdot10^{4}$	$-9{,}8\cdot10^{3}$	55,1	0,01
	med	$1{,}2\cdot10^{-3}$	$1{,}9\cdot10^{-3}$	$2{,}2\cdot10^{-3}$	$1{,}7\cdot10^{3}$	$-9{,}8\cdot10^{3}$	56,6	0,008
GSA	*av*	$7\cdot10^{-11}$	$4\cdot10^{-5}$	$0{,}16\cdot10^{3}$	25,16	$-2{,}8\cdot10^{3}$	15,32	0,29
	med	$7\cdot10^{-11}$	$4\cdot10^{-5}$	$0{,}15\cdot10^{3}$	25,18	$-2{,}6\cdot10^{3}$	14,42	0,04
DE	*av*	11,21	0,95	0,12	0,25	$-4{,}1\cdot10^{3}$	30,12	$1\cdot10^{-3}$
	med	13,21	1,05	0,09	0,31	$-4{,}1\cdot10^{3}$	31,43	$1\cdot10^{-3}$
АКПЖ	*av*	**$2\cdot10^{-29}$**	**$5\cdot10^{-20}$**	**$8\cdot10^{-31}$**	**$9\cdot10^{-28}$**	**$-1{,}2\cdot10^{4}$**	**$1\cdot10^{-3}$**	**$1\cdot10^{-13}$**
	med	**$1\cdot10^{-20}$**	**$3\cdot10^{-11}$**	**$1\cdot10^{-19}$**	**$3\cdot10^{-18}$**	**$-1{,}2\cdot10^{4}$**	**$7\cdot10^{-4}$**	**$1\cdot10^{-13}$**

Также оценивалась статистическая значимость полученных результатов. С этой целью применялся T-критерий суммы рангов Уилкоксона [18] для независимых выборок, найденных каждым из сравниваемых алгоритмов на 30 тестовых запусках, при уровне значимости 5%. Значение $T < 0{,}05$ рассматривалось как адекватное доказательство против нулевой гипотезы, которая отвергается. Предложенный алгоритм АКПЖ превосходит конкурирующие, а экспериментальные результаты по алгоритму являются статистически значимыми.

Также проводилось сравнение с одним из наиболее эффективных алгоритмов непрерывной оптимизации *BFGS* – квазиньютоновским итерационным алгоритмом численной оптимизации, предназначенным для нахождения локального экстремума нелинейного функционала без ограничений [19]. Эксперименты проводились в *MatLab*.

В первом эксперименте сравнивалась производительность алгоритмов *BFGS* и АКПЖ на унимодальной функции. В унимодальных функциях глобальный минимум совпадает с локальным минимумом. *BFGS* имеет высокую скорость локальной сходимости, которая зависит от размерности задачи. В качестве эталона для сравнения была выбрана гладкая и дифференцируемая функция Розенброка (f_5).

В эксперименте оба алгоритма *BFGS* и АКПЖ решали задачу минимизации функции f_5 при различной размерности функции. Для реализации *BFGS* в качестве исходной матрицы принималась $\boldsymbol{B}_0 = \boldsymbol{I}$ (первый шаг эквивалентен градиентному спуску). В качестве показателей эффективности сравнения рассматривались время решения и число итераций для достижения минимума. Для *BFGS* в качестве условия останова алгоритма принималось $\|g_5(\boldsymbol{X})\| \le 1 \times 10^{-6}$, где $g_5(\boldsymbol{X})$ является градиентом $f_5(\boldsymbol{X})$. Для АКПЖ критерием останова является условие, когда больше не происходят изменения в памяти $\boldsymbol{M}_h$.

В табл. 4 представлены результаты обоих алгоритмов с учетом следующих размерностей функции f_5: $n \in \{2, 10, 30, 50, 70, 100, 120\}$.

Для обеспечения согласованности результатов учитывались два показателя: среднее затраченное время (t_{cp}) и среднее число итераций (IN_{cp}) после 30 прогонов алгоритмов.

Таблица 4

Результаты сравнения алгоритма *BFGS* и алгоритма АКПЖ на многомерной функции Розенброка $f_5(\boldsymbol{X})$

f_5 / n	t_{cp} (с)		IN_{cp}	
	BFGS	АКПЖ	*BFGS*	АКПЖ
2	0,15	4,21	6	89
10	0,55	5,28	22	98
30	1,35	5,44	41	108
50	2,77	5,88	68	112
70	4,23	6,11	93	115
100	5,55	6,22	105	121
120	6,64	6,71	125	129

Из табл. 4 видно, что алгоритм *BFGS* превосходит АКПЖ по показателям t_{cp} и IN_{cp} при небольших значениях размерности. Однако при размерности функции $f_5(\boldsymbol{X})$, начиная с $n \geq 70$, результаты по алгоритму АКПЖ сопоставимы с результатами *BFGS*. Тот факт, что алгоритм *BFGS* превосходит предлагаемый гибридный алгоритм, нельзя рассматривать как недостаток АКПЖ, учитывая ограничения, накладываемые на функции методом *BFGS*.

В случае решения задачи глобальной оптимизации мультиэкстремальной функции алгоритм *BFGS* попадает в ловушку локальных оптимумов. Это ограничивает его применение для решения задач глобальной оптимизации. Одним из наиболее широко используемых для решения задач глобальной оптимизации является метод мультистарта (*MS*) [20]. В *MS* в качестве исходного решения случайным образом выбирается точка из допустимой области, а затем из нее запускается алгоритм непрерывной оптимизации. Затем процесс повторяется до тех пор, пока не будет достигнуто значение, близкое к глобальному оптимуму.

Во втором эксперименте сравнивалась производительность *MS* и АКПЖ на мультимодальных функциях $f_6(\boldsymbol{X})$ и $f_7(\boldsymbol{X})$. Сравнение проводилось по показателям среднее затраченное время (t_{cp}), среднее число итераций (IN_{cp}) и среднее наилучшее значение функции (av) после 30 прогонов алгоритмов. Результаты сравнения приведены в табл. 5.

Таблица 5

Результаты сравнения алгоритма *MS* и алгоритма АКПЖ на мультимодальных функциях $f_6(\boldsymbol{X})$ и $f_7(\boldsymbol{X})$ при $n = 30$

Функция	*MS*			АКПЖ		
	t_{cp}	IN_{cp}	av	t_{cp}	IN_{cp}	av
$f_6(\boldsymbol{X})$	45,4	23,3	$1{,}2 \cdot 10^{-2}$	10,2	633	$1{,}0 \cdot 10^{-3}$
$f_7(\boldsymbol{X})$	72,1	102,3	$4{,}51 \cdot 10^{-10}$	15,8	884	$1{,}14 \cdot 10^{-13}$

Из табл. 5 видно преимущество АКПЖ по времени и точности найденных решений.

Заключение. Представлена биоэвристика для решения прикладных задач глобальной оптимизации, основанная на вычислительной модели коллективного поведения группы животных, взаимодействующих между собой на основе простых поведенческих правил, реализуемых в виде математических операторов и имеющих общую память.

Алгоритм был экспериментально протестирован на множестве из семи известных одномодальных и мультимодальных функций. Его производительность сравнивалась с генетическим алгоритмом, алгоритмом роя частиц, алгоритмом гравитационного поиска и алгоритмом дифференциальной эволюции. Эксперименты показали, что предлагаемый алгоритм в целом превосходит конкурирующие алгоритмы по всем тестовым функциям,

а полученные результаты являются статистически значимыми. Сравнение предлагаемой биоэвристики с одним из наиболее эффективных алгоритмов непрерывной оптимизации *BFGS* на унимодальной функции с размерностью от $n = 2$ до $n = 120$ показало, что *BFGS* превосходит АКПЖ при небольших значениях размерности, однако, начиная с $n = 70$, их результаты становятся сопоставимыми при том, что на функции, оптимизируемые алгоритмом АКПЖ, не накладывается ограничений, в отличие от *BFGS*. Сравнение АКПЖ с методом мультистарта на мультимодальных функциях также показывает его преимущество по времени и точности найденных решений.

Результаты, в основном, обусловлены двумя причинами: (1) операторы, моделирующие биологические паттерны поведения позволяют лучше исследовать пространство поиска оптимума; (2) мультимодальной оптимизации не только сохраняется разнообразие решений, содержащихся в памяти $\boldsymbol{M}_h$, но даже увеличивается за счет реализации принципа конкуренции.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Курейчик В.В., Родзин С.И.* Вычислительные модели эволюционных и роевых биоэвристик (обзор) // Информационные технологии. – 2021. – Т. 27, № 10. – С. 507-520.
2. *Курейчик В.В., Родзин С.И.* Вычислительные модели биоэвристик, основанных на физических и когнитивных процессах (обзор) // Информационные технологии. – 2021. – Т. 27, № 11. – С. 563-574.
3. *Dragoi E.N., Dafinescu V.* Review of Metaheuristics Inspired from the Animal Kingdom // Mathematics. – 2021. – No. 9. – P. 2335.
4. *Bonyadi M., Michalewicz Z*. Particle swarm optimization for single objective continuous space problems: a review // Evolutionary Computation. – 2017. – Vol. 25 (1). – P. 1-54.
5. *Родзин С.И.* Современное состояние биоэвристик: классификация, бенчмаркинг, области применения // Известия ЮФУ. Технические науки. – 2023. – № 2. – С. 280–298.
6. Глобальная оптимизация: Википедия. Свободная энциклопедия. – URL: https://ru.wikipedia.org/wiki/Глобальная оптимизация (дата обращения: 20.01.2024).
7. *Курейчик В.В., Родзин С.И.* Биоэвристики, инспирированные фауной (обзор) // Информационные технологии. – 2023. – Т. 29, № 11. – С. 559-573.
8. *Mirjalili S*. Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi objective problems // Neural Comput. Appl. – 2015. – Vol. 27. – P. 1053-1073.
9. *Kallioras N.A., et. al*. Pity beetle algorithm – A new metaheuristic inspired by the behavior of bark beetles // Adv. Eng. Softw. – 2018. – Vol. 121. – P. 147-166.
10. *Couzin I.D*. Collective minds // Nature. – 2007. – Vol. 445. – P. 715-728.
11. *Bode N., Wood A., Franks D*. The impact of social networks on animal collective motion // Anim. Behav. – 2011. – Vol. 82 (1). – P. 29-38.
12. *Broom M., Koenig A., Borries C*. Variation in dominance hierarchies among group-living animals: modeling stability and the likelihood of coalitions // Behav. Ecol. – 2009. – Vol. 20. – P. 844-855.
13. *Long W., et. al*. Solving high-dimensional global optimization problems using an improved sine cosine algorithm // Expert systems with applications. – 2019. – Vol. 123. – P. 108-126.
14. *Hamzaçebi C*. Improving genetic algorithms' performance by local search for optimization // Appl. Math. Comput. – 2008. – Vol. 196 (1). – P. 309-317.
15. *Kennedy J., Eberhart R.C*. Particle swarm optimization // Proc. IEEE Inter. Conf. on Neural Networks. – 1995. – Vol. 4. – P. 1942-1948.
16. *Rashedi E., Nezamabadi-pour H., Saryazdi S*. GSA: a gravitational search algorithm // Inf. Sci. – 2009. – Vol. 179. – P. 2232-2248.
17. *Storn R., Price K*. Differential evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces // Technical report TR-95–012. ICSI. Berkeley. CA. – 1995.
18. *Wilcoxon F.* Individual comparisons by ranking methods Frank Wilcoxon // Bio-metrics Bulletin. – 2006. – Vol. 1 (6). – P. 80-83.
19. *Al-Baali M*. On the behavior of bombined extra-updating/self-scaling BFGS method // Jour. Comput. Appl. Math. – 2001. – Vol. 134. – P. 269-281.
20. *art i ., et al*. Intelligent multi-start methods // Handbook of Metaheuristics. – Cham: Springer, 2019. – P. 221-243.

REFERENCES

1. *Kureychik V.V., Rodzin S.I.* Vychislitel'nye modeli evolyutsionnykh i roevykh bioevristik (obzor) [Computational models of evolutionary and swarm bioheuristics (review)], *Informatsionnye tekhnologii* [Information Technology], 2021, Vol. 27, No. 10, pp. 507-520.
2. *Kureychik V.V., Rodzin S.I.* Vychislitel'nye modeli bioevristik, osnovannykh na fizicheskikh i kognitivnykh protsessakh (obzor) [Computational models and bio heuristics based on physical and cognitive processes (review], *Informatsionnye tekhnologii* [Information Technology], 2021, Vol. 27, No. 11, pp. 563-574.
3. *Dragoi E.N., Dafinescu V.* Review of Metaheuristics Inspired from the Animal Kingdom, *Mathematics,* 2021, No. 9, pp. 2335.
4. *Dragoi E.N., Dafinescu V.* Review of Metaheuristics Inspired from the Animal Kingdom, *Mathematics,* 2021, No. 9, pp. 2335.
5. *Rodzin S.I.* Sovremennoe sostoyanie bioevristik: klassifikatsiya, benchmarking, oblasti primeneniya [Computational models of bioeuristics based on physical and cognitive processes (review)], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2023, No. 2, pp. 280-298.
6. Global'naya optimizatsiya: Vikipediya. Svobodnaya entsiklopediya [Global optimization: Wikipedia. The free encyclopedia]. Available at: https://ru.wikipedia.org/wiki/Global'naya optimizatsiya (accessed 20 January 2024).
7. *Kureychik V.V., Rodzin S.I.* Bioevristiki, inspirirovannye faunoy (obzor) [Bio heuristics inspired by fauna (review)], *Informatsionnye tekhnologii* [Information Technology], 2023, Vol. 29, No. 11, pp. 559-573.
8. *Mirjalili S*. Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi objective problems, *Neural Comput. Appl*., 2015, Vol. 27, pp. 1053-1073.
9. *Kallioras N.A., et. al*. Pity beetle algorithm – A new metaheuristic inspired by the behavior of bark beetles, *Adv. Eng. Softw*., 2018, Vol. 121, pp. 147-166.
10. *Couzin I.D*. Collective minds, *Nature,* 2007, Vol. 445, pp. 715-728.
11. *Bode N., Wood A., Franks D*. The impact of social networks on animal collective motion, *Anim. Behav*., 2011, Vol. 82 (1), pp. 29-38.
12. *Broom M., Koenig A., Borries C*. Variation in dominance hierarchies among group-living animals: modeling stability and the likelihood of coalitions, *Behav. Ecol*., 2009, Vol. 20, pp. 844-855.
13. *Long W., et. al*. Solving high-dimensional global optimization problems using an improved sine cosine algorithm, *Expert systems with applications,* 2019, Vol. 123, pp. 108-126.
14. *Hamzaçebi C*. Improving genetic algorithms' performance by local search for optimization, *Appl. Math. Comput*., 2008, Vol. 196 (1), pp. 309-317.
15. *Kennedy J., Eberhart R.C*. Particle swarm optimization, *Proc. IEEE Inter. Conf. on Neural Networks,* 1995, Vol. 4, pp. 1942-1948.
16. *Rashedi E., Nezamabadi-pour H., Saryazdi S*. GSA: a gravitational search algorithm, *Inf. Sci.,* 2009, Vol. 179, pp. 2232-2248.
17. *Storn R., Price K*. Differential evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces, Technical report TR-95–012. ICSI. Berkeley. CA, 1995.
18. *Wilcoxon F.* Individual comparisons by ranking methods Frank Wilcoxon, *Bio-metrics Bulletin,* 2006, Vol. 1 (6), pp. 80-83.
19. *Al-Baali M*. On the behavior of bombined extra-updating/self-scaling BFGS method, *Jour. Comput. Appl. Math*., 2001, Vol. 134, pp. 269-281.
20. *art i ., et al*. Intelligent multi-start methods, *Handbook of Metaheuristics*. Cham: Springer, 2019, pp. 221-243.

Статью рекомендовала к опубликованию д.т.н., профессор Л.С. Лисицына.

Родзин Сергей Иванович – Южный федеральный университет; e-mail: srodzin@sfedu.ru; г. Таганрог, Россия; кафедра математического обеспечения и применения ЭВМ; профессор.

Rodzin Sergey Ivanovich – Southern Federal University; e-mail: srodzin@sfedu.ru; Taganrog, Russia; phone: +78634371673; the department of software engineering; professor.