УДК 004.912 doi: 10.15827/0236-235X.142.146-154 2024. Т. 37. № 2. С. 146–154

Глобальная оптимизация на основе гибридизации алгоритмов роя саранчи и колонии пауков

С.И. Родзин [1]✉

[1] Южный федеральный университет, г. Таганрог, 347922, Россия

Ссылка для цитирования
Родзин С.И. Глобальная оптимизация на основе гибридизации алгоритмов роя саранчи и колонии пауков // Программные продукты и системы. 2024. Т. 37. № 2. С. 146–154. doi: 10.15827/0236-235X.142.146-154
Информация о статье
Группа специальностей ВАК: 1.2.1
Поступила в редакцию: 08.01.2024 После доработки: 22.02.2024 Принята к публикации: 06.03.2024

Аннотация. Перспективным решением задач глобальной оптимизации являются метаэвристики, инспирированные природой. Они представляют собой недетерминированные алгоритмы, исследующие пространство поиска решений, обучающиеся в процессе поиска, не привязанные к конкретной задаче, хотя и не гарантирующие точного решения. Целью данного исследования является разработка эффективного алгоритма для решения прикладных проблем глобальной оптимизации многомерных мультиэкстремальных функций, встречающихся в задачах вычислительной филогенетики, при проектировании электрических схем, расчетах инженерной безопасности зданий, калибровке моделей распространения радиосигналов и в других. Для достижения этой цели предложен гибридный алгоритм, моделирующий паттерны поведения роя саранчи и колонии пауков. Основное внимание уделено снижению вероятности преждевременной сходимости гибридного алгоритма, поддержке баланса между скоростью сходимости алгоритма и диверсификацией пространства поиска решений (интенсификация/диверсификация). Приводятся этапы модифицированных алгоритмов колонии пауков и роя саранчи, моделирующих различные паттерны их поведения, что позволяет снизить влияние очень хороших или плохих решений на процесс поиска. Гибридизация алгоритмов осуществляется путем их последовательной комбинации (препроцессор/постпроцессор). Алгоритм экспериментально протестирован на семи известных многомерных функциях. Результаты сопоставлены с конкурирующими алгоритмами роя частиц, дифференциальной эволюции, колонии пчел. Предлагаемый алгоритм обеспечивает лучшие результаты для всех рассмотренных функций. Проверка полученных результатов с использованием Т-критерия суммы рангов Уилкоксона для независимых выборок показала, что результаты по алгоритму являются статистически значимыми. Разработанное программное приложение предназначено для использования в рамках университетского курса по машинному обучению и биоинспирированной оптимизации, а также для решения широкого круга научно-прикладных задач поисковой оптимизации.

Ключевые слова: алгоритм, глобальный оптимум, агент, рой саранчи, колония пауков, паттерн поведения, интенсификация поиска, диверсификация поиска, тестовая функция, критерий Уилкоксона

Благодарности. Исследование выполнено в Южном федеральном университете за счет гранта РНФ № 23-21-00089, https://rscf.ru/project/23-21-00089/

Введение. Эвристические алгоритмы, инспирированные природой, вместе с нейронными сетями и другими методами образуют класс алгоритмов машинного обучения. Для них характерны исследование пространства решений, поиск точек в области определения функции, на которых она достигает минимального или максимального значения, их оценка и селективный отбор. Алгоритм обучается нахождению областей определения функции, которые содержат наилучшие решения. Поиск по БД Web of Science, Scopus, Google Scholar, цифровой библиотеке IEEE Xplore, российским библиотекам eLibrary.Ru и «КиберЛенинка» позволяет говорить о существовании более 400 алгоритмов, инспирированных природой [1]. Их число и показатели цитируемости растут: 85 % статей, в которых предлагаются алгоритмы, инспирированные природой, цитируются в среднем около 20 раз, в то время как среднегодовая норма цитирования статей в целом в области Artificial Intelligence составляет около 5 [2]. С 2010 г. по настоящее время лидерами по количеству публикаций являются роевые алгоритмы. Они применяются практически во всех областях науки, техники и экономики. В этих областях многие задачи оптимизации и проектирования нелинейные, зачастую NP-трудные. Между тем такие алгоритмы, как оптимизация роя частиц (PSO), дифференциальная эволюция (DE), колония муравьев (ACO), пчел (ABC), светлячков (FA), поиска кукушки (CS) и многие другие продемонстрировали большой потенциал при решении сложных задач поисковой оптимизации за разумное время [3].

Чтобы алгоритм, инспирированный природой, был эффективным, он должен обладать некоторыми особыми возможностями. Например, генерировать новые решения, улучшающие существующие путем интенсивного ис-

следования локального пространства вокруг найденного текущего хорошего решения (интенсификация). С другой стороны, алгоритм должен быть способным избегать преждевременной сходимости в локальном оптимуме путем расширения области поиска решений и обнаруживать области поиска, где может находиться глобальный оптимум (диверсификация). Удачная комбинация интенсификации и диверсификации поиска способствует высокой эффективности метаэвристики. Поиск баланса этих двух важных компонент любого алгоритма, инспирированного природой, является открытой исследовательской проблемой. Каждая биоэвристика использует различный баланс между ними, зачастую далекий от оптимального.

В данной работе предлагается гибридный алгоритм глобальной оптимизации, основанный на модифицированных алгоритмах роя саранчи и колонии пауков, а также исследуется его эффективность на ряде известных тестовых задач.

Модифицированный алгоритм роя саранчи

Инспирированные природой алгоритмы, описывающие коллективные разумные паттерны поведения в мире животных и насекомых, привлекают внимание исследователей с конца прошлого века. Стереотипные поведенческие паттерны, наблюдаемые в группах животных и скоплениях насекомых, дают преимущества для выживания. Группа особей может решать достаточно сложные задачи, хотя отдельные особи, составляющие группу, относительно просты, используют ограниченную локальную информацию, допускают специализацию задач.

Саранча, являясь своего рода крупным кузнечиком, представляет пример таких групп насекомых. Саранча демонстрирует как одиночный, так и роевой паттерны поведения с их четкими поведенческими различиями [4]. При одиночном поведении, когда много пищи и места, саранча избегает контакта, поэтому рой исследует пространство, распределившись по всей площади. Паттерн роевого поведения, наоборот, означает стремление саранчи к агрегации вокруг перспективных источников пищи [5]. Использование в алгоритме оптимизации такого рода специализированных паттернов поведения позволяет поддерживать необходимый баланс между интенсификацией и диверсификацией поиска, избегать преждевременной сходимости к неоптимальным решениям.

В алгоритме это достигается путем включения операторов, моделирующих роевые и одиночные паттерны поведения агентов популяции. Речь идет о тех, кто отвечает за изменение позиций особей роя саранчи на текущей итерации алгоритма.

В природе налетевший рой саранчи, механизм образования которого достаточно сложен, способен уничтожить огромную площадь посевов за несколько часов. В работе [6] представлен алгоритм оптимизации, моделирующий поиск пищи роем саранчи, а в [5] – биологическая модель одиночного и роевого поведения саранчи, которая применяется при разработке модифицированного алгоритма роя саранчи.

Рассмотрим паттерн одиночного поведения особи саранчи и связанное с ним изменение ее позиции. Обозначим через $\boldsymbol{x}_i^k$ позицию i-й особи в рое из N особей на k-м шаге алгоритма. Позиция i-й особи на следующем шаге алгоритма x_i^{k+1} определяется как

$$x_i^{k+1} = x_i^k + \Delta x_i,$$

где величина Δx_i отражает изменение позиции i-й саранчи.

Паттерн одиночного поведения предполагает, что пара особей избегает сближения при условии короткой дистанции между ними. Если же дистанция между особями велика, то они начинают сближаться, поддерживая сплоченность роя. Следуя [7], определим силу сближения/удаления саранчи следующим образом:

$$s(r) = kar\, e^{-r/lim} - e^{-r}, \quad (1)$$

где r – дистанция между парой особей; kar – коэффициент сближения/удаления; lim – пороговая дистанция между особями. Сила удаления превышает силу сближения, если $kar < 1$ и $lim > 1$, то есть дистанция между особями небольшая. Иными словами, сила взаимодействия между парой особей зависит от дистанции между ними:

$$s_{ij} = s(r_{ij})\boldsymbol{d}_{ij},$$

где $r_{ij} = |\boldsymbol{x}_j - \boldsymbol{x}_i|$ – дистанция между i-й и j-й особями роя; $\boldsymbol{d}_{ij} = (\boldsymbol{x}_j - \boldsymbol{x}_i)/r_{ij}$ – единичный вектор.

Тогда сумма всех сил взаимодействия с роем для i-й особи дает общую силу сближения/удаления роя и определяет изменение позиции i-й саранчи:

$$S_i = \sum_{i=1,\, j\neq i}^{N} \boldsymbol{s}_{ij}. \quad (2)$$

Триггером перехода от одиночного поведения саранчи к роевому являются изменения

в окружающей среде. Рой в поисках пищи ведет себя очень сплоченно. Характерная черта паттерна роевого поведения заключается в стремлении роя сконцентрироваться вблизи особей, которым удалось найти пищу. Эту особенность роевого поведения можно смоделировать путем введения для каждой особи роя саранчи индикатора пищи f_i ($f_i \in [0, 1]$). После сортировки N особей роя производится отбор b доминирующих особей ($b << N$) с наибольшим значением индикатора f_i, вблизи которых концентрируются остальные особи.

Рой саранчи из N особей образует популяцию решений $\boldsymbol{L}^k(\{\boldsymbol{l}_1^k, \boldsymbol{l}_2^k, ..., \boldsymbol{l}_N{}^k\})$ оптимизационной задачи, где число итераций $k = 0, 1, 2, ..., gen$. Особь саранчи $\boldsymbol{l}_i^k$ ($i = 1..N$) характеризуется через множество $\{l_{i1}{}^k, l_{i2}{}^k, ..., l_{in}{}^k\}$. Элемент этого множества отображает отдельную переменную оптимизационной задачи. Ограничения в задаче образуют множество допустимых решений:

$$\boldsymbol{S} = \{\boldsymbol{l}_i^k \in \boldsymbol{R}^n \mid lb_d \le l_{id}{}^k \le ub_d\},$$

где lb_d – нижняя граница размерности d; ub_d – верхняя граница размерности d. Фитнесс-функция $f_i(\boldsymbol{l}_i^k)$ характеризует уровень индикатора пищи каждой саранчи.

Таким образом, *модифицированный алгоритм роя саранчи* (МАРС) использует оператор $O_{од}$, характеризующий паттерн одиночного поведения саранчи, и $O_{рой}$, характеризующий паттерн роевого поведения саранчи. Оператор $O_{од}$ способствует диверсификации области поиска решений и обнаружению точки глобального оптимума, а оператор $O_{рой}$ – интенсивному исследованию локального пространства вокруг найденного текущего хорошего решения.

С помощью оператора $O_{од}$ позиция $\boldsymbol{l}_i^k$ i-й особи изменяется на величину $\Delta\boldsymbol{l}_i^k$ и становится равной

$$\boldsymbol{p}_i = \boldsymbol{l}_i^k + \Delta\boldsymbol{l}_i^k.$$

При этом учитывается позиция доминирующих особей роя с наибольшим значением фитнесс-функции. Тогда сила сближения/удаления между парой особей (i, j) вычисляется следующим образом:

$$\boldsymbol{s}_{ij}{}^m = \rho(\boldsymbol{l}_i^k, \boldsymbol{l}_j^k)\, s(r_{ij})\boldsymbol{d}_{ij} + rand(1, -1),$$

где $s(r_{ij})$ вычисляется согласно (1); $\boldsymbol{d}_{ij}$ направлен от $\boldsymbol{l}_i^k$ к $\boldsymbol{l}_j^k$; функция $rand(-1, 1)$ генерирует случайное число из интервала (–1, 1); функция $\rho(\boldsymbol{l}_i^k, \boldsymbol{l}_j^k)$ указывает на доминирование между парой особей (i, j).

Особи с наибольшим значением фитнесс-функции присваивается ранг 0, а с наименьшим – ранг N–1. Функция доминирования ρ определяется следующим образом:

$$\rho\left(l_i^k, l_j^k\right) = \begin{cases} e^{-\left(\frac{5rank(l_i^k)}{N}\right)}, \\ \text{если } rank\left(l_i^k\right) < rank\left(l_j^k\right), \\ e^{-\left(\frac{5rank(l_j^k)}{N}\right)}, \\ \text{если } rank\left(l_i^k\right) > rank\left(l_j^k\right), \end{cases} \quad (3)$$

где функция $rank(\alpha)$ указывает на ранг особи.

Согласно (3), функция ρ принимает значения из интервала (0, 1). Значение $\rho = 1$ в случае, если особь имеет наилучшее значение фитнесс-функции в популяции. Значение ρ, близкое к 0, свидетельствует о низком значении фитнесс-функции.

После обновления множества позиций $\{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_N\}$ особей колонии $\boldsymbol{L}^k$ корректируются значения множества фитнесс-функций $\boldsymbol{F}(\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_N)$. Причем результаты поиска должны улучшаться: если $f_i(\boldsymbol{p}_i) > f_i(\boldsymbol{l}_i^k)$, то особь перемещается в новую позицию $\boldsymbol{p}_i$, в противном случае позиция $\boldsymbol{l}_i^k$ остается неизменной.

С помощью оператора $O_{рой}$ производится интенсивное исследование локального пространства вокруг найденного текущего хорошего решения. Это происходит путем упорядочения фитнесс-функций особей по убыванию. Затем результаты упорядочения запоминаются в массиве $\boldsymbol{B} = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_N\}$. Из массива $\boldsymbol{B}$ выбираются g особей с наилучшими значениями фитнесс-функции, которые составляют множество $\boldsymbol{E}$ наиболее перспективных для дальнейшего исследования решений. Исследование производится в окрестности каждой i-й особи из множества $\boldsymbol{E}$. Радиус окрестности вычисляется по формуле

$$e_d = \frac{\sum_{q=1}^{n}(ub_q - lb_q)}{n}\beta,$$

где ub_q и lb_q – верхняя и нижняя границы в q-м измерении соответственно; n – размерность переменных в задаче оптимизации; $\beta \in [0, 1]$ – параметр алгоритма. Верхняя граница $uss^q{}_j$ и нижняя граница $lss^q{}_j$ окрестности C_j в q-м измерении вычисляются по следующим формулам:

$$uss_j^q = b_{j,q} + e_d,$$

$$lss_j^q = b_{j,q} - e_d.$$

В окрестности C_j создаются h ($h < 4$) новых особей. Победителем является особь с наилучшим значением фитнесс-функции.

Таким образом, управляющими параметрами алгоритма МАРС являются *kar* – коэффи-

циент сближения/удаления, *lim* – пороговая дистанция между особями, *g* – число особей с наилучшими фитнесс-функциями, *N* – размер популяции и *gen* – число поколений.

В процессе инициализации алгоритма МАРС устанавливается начальное значение счетчика поколений ($k = 0$) и случайно генерируется начальная популяция $\boldsymbol{L}^0(\{\boldsymbol{l}_1^0, \boldsymbol{l}_2^0, ..., \boldsymbol{l}_N^0\})$. При этом для измерения d предварительно задаются верхняя начальная граница параметра (ub_d) и нижняя начальная граница параметра (lb_d), в которых значения ($\{l_{i1}^0, l_{i2}^0, ..., l_{in}^0\}$) распределены случайно и равномерно:

$$l_{ij}^0 = lb_d + rand(ub_d - lb_d),$$

где $i = 1..N$, $d = 1..n$.

Далее в алгоритме выполняются операторы $О_{од}$, характеризующий паттерн одиночного поведения саранчи, и $О_{рой}$, характеризующий паттерн роевого поведения саранчи для поддержки баланса интенсификации/диверсификации.

Останов алгоритма происходит при достижении числа итераций $k = gen$.

Представим псевдокод МАРС:

1: Ввод: *kar, lim, g, N, gen*
2: Инициализация $\boldsymbol{L}^0$ ($k = 0$)
3: ***until*** ($k = gen$)
4: $\boldsymbol{F} \leftarrow$ оператор $О_{од}$ ($\boldsymbol{L}^k$)
5: $\boldsymbol{L}^{k+1} \leftarrow$ оператор $О_{рой}$ ($\boldsymbol{L}^k$, $\boldsymbol{F}$)
6: $k = k + 1$
7: ***end until***

Модифицированный алгоритм колонии пауков

В работе [8] представлен алгоритм оптимизации, моделирующий кооперативное поведение колонии социальных пауков, а в [9] – биологическая модель их взаимодействия. Используем эту модель для представления паттернов поведения в распределенной самоорганизующейся сети и построения *модифицированного алгоритма колонии пауков* (МАКП).

Основными функциями, характеризующими поведение пауков в колонии, являются строительство сетевой паутины, размножение и охота. При этом инструментом взаимодействия агрегации пауков является сеть. Она представляет собой канал связи через вибрации и постукивания. Вибрации используются пауками для синхронизации и декодирования своих действий, а также для авторизации особи, передающей сообщение. По интенсивности вибраций и постукиваний определяются вес паука и расстояние до него. Каждый шаг паука вызывает вибрацию. Поэтому, например, при охоте колония перестает двигаться, чтобы по сети почувствовать добычу и понять правильное направление движения. Колония использует определенную тактику и командную работу для защиты своих ресурсов и потомства с дифференциацией ролей.

Согласно биологической модели определим паттерны поведения пауков – размножение и кооперация [10]. Альфа-самцы пауков отличаются большим весом в сравнении с остальными самцами. При размножении они стремятся двигаться по сети к ближайшей самке. В отличие от альфа-самцов недоминирующие самцы в основном сосредоточиваются в центре колонии, чтобы воспользоваться ресурсами альфа-самцов.

С учетом этого МАКП включает следующую последовательность шагов.

Шаг 1. Инициализация популяции пауков $S = B \cup M$ размером N. Поскольку, согласно биологической модели, в колонии преобладают самки, то их число N_b определяется по формуле

$$N_b = floor\,[(0{,}9 - rand0{,}25)N],$$

где *rand* – случайное число в интервале [0, 1]; $floor(\cdot)$ – функция преобразования действительных чисел в целые. Общее число самцов $N_m = N - N_b$.

В целом колония социальных пауков S размером N состоит из подмножеств $B = \{b_1, b_2, \ldots, b_{Nb}\}$ и $M = \{m_1, m_2, \ldots, m_{Nm}\}$.

Шаг 2. Случайная инициализация позиций пауков. Позиции самцов b_i и самок m_j генерируются случайно и равномерно. Для этого задаются нижний начальный параметр $q_k^{\text{ниж}}$ и верхний начальный параметр $q_k^{\text{верх}}$:

$$b_{i,k}^0 = q_k^{\text{ниж}} + rand(0,1)(q_k^{\text{верх}} - q_k^{\text{ниж}}),$$
$$m_{j,k}^0 = q_k^{\text{ниж}} + rand(0,1)(q_k^{\text{верх}} - q_k^{\text{ниж}}),$$

где $i = 1\ldots N_b$, $j = 1...N_m$, $k = 1...n$; функция *rand*(0, 1) генерирует случайное число в интервале от 0 до 1.

Шаг 3. Диапазон взаимодействия в колонии социальных пауков определяется размерами пространства поиска по формуле

$$r = \frac{\sum_{k=1}^{n}\left(q_k^{\text{верх}} - q_k^{\text{ниж}}\right)}{2n}.$$

Шаг 4. Вероятность участия в процессе размножения зависит от веса паука и вычисляется по методу рулетки:

$$P_{s_i} = \frac{w_i}{\sum_{k \in T^g} w_k},$$

где T^g – множество пауков в радиусе r. При этом сохраняется первоначальная пропорция между самками и самцами в популяции S.

Шаг 5. Позиция самки b_i на итерации алгоритма $(t + 1)$ изменяется по формулам:

$b_i^{t+1} = b_i^t + \alpha Vib_{i,u}(m_u - b_i^t) + \beta Vib_{i,\max w}(m_{\max w} - b_i^t) + \delta(rand - 1/2)$

с вероятностью PF,

$b_i^{t+1} = b_i^t + \alpha Vib_{i,u}(m_u - b_i^t) + \beta Vib_{i,\max w}(m_{\max w} - b_i^t) + \delta(rand - 1/2)$

с вероятностью $1 - PF$.

Здесь моделируются вибрации паутины. Вибрации зависят от веса и расстояния до паука, который их породил. Иными словами, особи рядом с пауком, создающим вибрации, улавливают их как более сильные, нежели особи, находящиеся вдали. Вибрация, которую i-й паук улавливает от j-го паука, определяется как

$$Vib_{i,j} = w_j e^{-d_{i,j}^2},$$

где $d_{i,j} = \| s_i - s_j \|$ – евклидово расстояние между i-м и j-м пауками.

Модификация алгоритма заключается в использовании в МАКП следующих вариантов вибраций:

- $Vibc_i = w_c e^{-d_{ci}^2}$ – вибрации, улавливаемые i-м пауком от ближайшего к нему паука c, обладающего большим весом ($w_c > w_i$);
- $Vibb_{i,\max w} = w_{\max w} e^{-d_{bi,\max w}^2}$ – вибрации, улавливаемые i-м пауком от паука max$w$ с максимальным весом в колонии;
- $Vibf_i = w_f e^{-d_{fi}^2}$ – вибрации, улавливаемые i-м пауком от ближайшей самки f.

Шаг 6. Выполняется оператор, моделирующий движение самца паука:

$$m_i^{t+1} = \begin{cases} m_i^t + \alpha Vib_{i,b}\left(s_b - m_i^t\right) + \delta\left(rand - \dfrac{1}{2}\right), \\ \text{если } w_{N_{\text{ср}}} > w_i; \\ m_i^t + \alpha\left(\dfrac{\sum_{h=1}^{N_m} m_h^t w_{N_{b+h}}}{\sum_{h=1}^{N_m} w_{N_{b+h}}} - m_i^t\right), \\ \text{если } w_i > w_{N_{\text{ср}}} \end{cases}$$

где s_b – ближайшая к пауку самка, а величина $\left(\dfrac{\sum_{h=1}^{N_m} m_h^t w_{N_{b+h}}}{\sum_{h=1}^{N_m} w_{N_{b+h}}} - m_i^t\right)$ представляет средневзвешенный вес самцов M в колонии.

Оператор определяет различные поведенческие паттерны, что способствует диверсификации поиска оптимума. К тому же уменьшается влияние очень хороших и плохих решений на результаты поиска.

Шаг 7. Выполняется оператор размножения в определенном диапазоне r, который вычисляется как

$$P_{s_i} = \frac{w_i}{\sum_{k \in T^g} w_k},$$

где T^g – множество пауков в радиусе r, которые участвуют в размножении.

Шаг 8. Останов алгоритма происходит при достижении заданного максимального числа итераций. Иначе – возврат к шагу 4 алгоритма.

Гибридизация алгоритмов МАРС и МАКП

Комбинирование нескольких алгоритмов, инспирированных природой, при решении целого класса сложных задач глобальной оптимизации может оказаться более эффективным, нежели использование одного алгоритма. Перспективным направлением здесь является гибридизация различных алгоритмов глобальной оптимизации. Существуют различные способы гибридизации алгоритмов. Одним из них является последовательная комбинация алгоритмов (препроцессор/постпроцессор). Основная идея этого способа гибридизации заключается в том, чтобы на начальных этапах обеспечить широкий обзор всевозможных решений, а на последующих сузить области поиска.

Сценарий предлагаемой гибридизации включает поиск решений с помощью алгоритма МАКП в качестве препроцессора, а затем проверку и выявление лучшего решения с помощью алгоритма МАРС. Триггером перехода с алгоритма препроцессора является отсутствие улучшения глобального оптимума. Алгоритм постпроцессора стартует, используя в основном решения, полученные на заключительном шаге алгоритма препроцессора с исключением решений, близких к области одного и того же экстремума.

Результаты экспериментальных исследований эффективности гибридного алгоритма

Для оценки эффективности предлагаемого гибридного алгоритма глобальной оптимизации, основанного на МАКП и МАРС, были проведены эксперименты с использованием тестовых многомерных функций.

Эксперименты проводились в программной среде на языке программирования C# с использованием среды программирования приложе-

ний Microsoft Visual Studio. Это позволило воспользоваться возможностями, предоставляемыми объектно-ориентированным подходом в разработке ПО, а также максимально использовать фреймворк Windows Forms (зарегистрирована программа для ЭВМ, запись № 2023681642 от 17.10.2023 г.). При отладке и тестировании использован компьютер IBM PC с процессором Core i7 и ОЗУ 8 Гб.

Набор тестовых функций, на которых проводилось программное моделирование, включал как одноэкстремальные, так и мультиэкстремальные функции:

- функция Розенброка

$$f_1(X) = \sum_{i=1}^{n-1}\left[100\left(x_{i+1} - x_i^2\right)^2 + \left(x_i - 1\right)^2\right];$$

$x_i \in [-30, 30]^n$, $x^* = (1, \ldots, 1)$, $f_1(x^*) = 0$;

- сферическая функция

$$f_2(X) = \sum_{i=1}^{n} x_i^2,\ x_i \in [-100, 100]^n,\ x^* = (0, \ldots, 0),$$

$f_2(x^*) = 0$;

- функция Экли

$$f_3(X) = -20\exp\left(\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) -$$

$$-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20;$$

$x_i \in [-32, 32]^n$, $x^* = (0, \ldots, 0)$, $f_3(x^*) = 0$;

- функция Швефеля

$$f_4(X) = \sum_{i=1}^{n}\left(\sum_{j=1}^{i} x_j\right)^2,$$

$x_i \in [-100, 100]^n$, $x^* = (0, \ldots, 0)$, $f_4(x^*) = 0$;

- сумма квадратов

$$f_5(X) = \sum_{i=1}^{n} i x_i^2,\ x_i \in [-10, 10]^n,\ x^* = (0, \ldots, 0),$$

$f_5(x^*) = 0$;

- функция Растригина

$$f_6(X) = \sum_{i=1}^{n}\left[x_i^2 - 10\cos(2\pi x_i) + 10\right],$$

$x_i \in [-5{,}12; 5{,}12]^n$, $x^* = (0, \ldots, 0)$, $f_6(x^*) = 0$;

- функция Соломона

$$f_7(X) = -\cos\left(2\pi\sqrt{\sum_{i=1}^{n} x_i^2}\right) + 0{,}1\sqrt{\sum_{i=1}^{n} x_i^2} + 1,$$

$x_i \in [-100, 100]^n$, $x^* = (0, \ldots, 0)$, $f_7(x^*) = 0$,

где n – размерность функции; x^* – оптимальное решение; $f_i(x^*)$ – минимальное значение функции.

Например, график функции Растригина представляет собой сочетание сферической и синусоидной функций с множеством локальных и одним глобальным минимумом, «банановая» функция Розенброка – большое медленно убывающее плато с глобальным минимумом внутри параболической сильно вытянутой поверхности, функция Швефеля является мультиэкстремальной с «непредсказуемым» глобальным минимумом; функция Соломона – непрерывная, мультиэкстремальная, дифференцируемая. Подробное описание множества тестовых функций содержится в [11].

Результаты тестирования гибридного алгоритма МАКП-МАРС сопоставлялись с алгоритмами пчелиной колонии (Artificial Bee Colony, ABC), дифференциальной эволюции (Differential Evolution, DE), роя частиц (Particle Swarm Optimization, PSO) [12–15], МАРС, МАКП.

В экспериментах использовались следующие настройки параметров алгоритмов: ABC – параметр *limit* = 100; DE − параметр $F \in [0{,}4; 1{,}0]$, параметр $p_m \in [0{,}0; 1{,}0]$; PSO − коэффициенты обучения $c_1 = 2$ и $c_2 = 2$, вес w линейно уменьшается с 0,9 до 0,2 при увеличении числа итераций; МАКП − параметр $PF = 0{,}7$; МАРС − параметры $kar = 0{,}6$, $lim = 1$, $N = 50$, g = 20, $gen = 1000$.

По каждой функции и каждому алгоритму осуществлялось 30 прогонов. Затем полученные результаты усреднялись.

Показателями, по которым проводилось сравнение, являлись среднее значение по лучшим решениям, медианное лучшее решение и стандартное отклонение от лучшего решения [16]. Усредненные результаты по 30 отдельным запускам приведены в таблице.

В каждой клетке таблицы указаны среднее, медианное решение и стандартное отклонение от лучшего решения соответственно.

Лучшие результаты выделены жирным шрифтом. Из таблицы видно, что гибридный алгоритм МАКП-МАРС обеспечивает лучшие результаты, нежели алгоритмы ABC, DE, PSO, МАКП, МАРС, практически для всех указанных функций.

Также оценивалась статистическая значимость полученных результатов. С этой целью применялся Т-критерий суммы рангов Уилкоксона [17] 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 гибридный алгоритм МАКП-МАРС превосходит конкурирующие, а экспериментальные результаты по нему являются статистически значимыми.

Результаты сравнения гибридного алгоритма МАКП-МАРС с алгоритмами ABC, DE, PSO, МАРС, МАКП

Results of comparing MAKP-MARS hybrid algorithm with the ABC, DE, PSO, MARS, MAKP algorithms

Функция	ABC	DE	PSO	МАРС	МАКП	МАКП-МАРС
$f_1(X)$ роз	$9{,}26\cdot10^{-02}$ $3{,}24\cdot10^{-01}$ $1{,}81\ 10^{-01}$	$2{,}27\cdot10^{-02}$ $2{,}23\cdot10^{-02}$ $5{,}03\cdot10^{-03}$	$2{,}73\cdot10^{-02}$ $2{,}61\cdot10^{-02}$ **$5{,}83\cdot10^{-03}$**	$1{,}47\cdot10^{-02}$ $1{,}86\cdot10^{-02}$ $6{,}33\cdot10^{-03}$	$8{,}77\cdot10^{-02}$ $5{,}13\cdot10^{-02}$ $2{,}68\ 10^{-02}$	**$1{,}02\cdot10^{-02}$** **$8{,}15\cdot10^{-03}$** $7{,}17\cdot10^{-03}$
$f_2(X)$ сф	$7{,}08\cdot10^{-03}$ $5{,}16\cdot10^{-03}$ $1{,}86\ 10^{-03}$	$1{,}97\cdot10^{-05}$ $5{,}43\cdot10^{-05}$ $1{,}03\cdot10^{-05}$	$8{,}44\cdot10^{-03}$ $3{,}12\cdot10^{-02}$ $1{,}60\cdot10^{-03}$	$9{,}23\cdot10^{-06}$ **$7{,}95\cdot10^{-06}$** $2{,}17\cdot10^{-06}$	$5{,}12\cdot10^{-05}$ $4{,}12\ 10^{-05}$ $8{,}82\ 10^{-06}$	**$3{,}38\cdot10^{-07}$** $9{,}49\cdot10^{-06}$ **$1{,}12\cdot10^{-06}$**
$f_3(X)$ экл	$6{,}26\cdot10^{-02}$ $5{,}95\cdot10^{-02}$ $1{,}33\ 10^{-03}$	$7{,}01\cdot10^{-04}$ $7{,}20\cdot10^{-04}$ $2{,}21\cdot10^{-04}$	$3{,}57\cdot10^{-02}$ $4{,}82\cdot10^{-02}$ $1{,}15\cdot10^{-03}$	**$3{,}43\cdot10^{-05}$** $2{,}65\cdot10^{-05}$ $3{,}71\cdot10^{-06}$	$4{,}22\cdot10^{-05}$ $1{,}52\cdot10^{-05}$ $8{,}16\cdot10^{-06}$	$4{,}05\cdot10^{-05}$ **$2{,}02\cdot10^{-06}$** **$1{,}15\cdot10^{-06}$**
$f_4(X)$ шв	$2{,}34\cdot10^{-01}$ $6{,}77\cdot10^{-01}$ $2{,}72\ 10^{-01}$	$8{,}26\cdot10^{-01}$ $7{,}35\cdot10^{-01}$ $1{,}66\ 10^{-01}$	$8{,}62\cdot10^{-01}$ $5{,}25\cdot10^{-01}$ $1{,}12\ 10^{-01}$	$7{,}98\cdot10^{-02}$ $6{,}62\cdot10^{-02}$ $2{,}76\ 10^{-02}$	$8{,}77\cdot10^{-02}$ $5{,}13\cdot10^{-02}$ $2{,}68\ 10^{-02}$	**$3{,}20\cdot10^{-03}$** **$7{,}57\cdot10^{-03}$** **$2{,}82\ 10^{-03}$**
$f_5(X)$ ква	$2{,}47\cdot10^{-03}$ $5{,}72\cdot10^{-03}$ $6{,}63\ 10^{-04}$	$2{,}47\cdot10^{-03}$ $5{,}70\cdot10^{-03}$ **$1{,}58\ 10^{-04}$**	$6{,}96\cdot10^{-02}$ $5{,}49\cdot10^{-02}$ $3{,}33\ 10^{-02}$	$9{,}99\cdot10^{-04}$ **$5{,}17\cdot10^{-04}$** $1{,}76\ 10^{-04}$	$8{,}98\cdot10^{-04}$ $6{,}62\cdot10^{-04}$ $1{,}76\ 10^{-04}$	**$1{,}46\cdot10^{-05}$** $7{,}38\cdot10^{-04}$ $1{,}76\ 10^{-04}$
$f_6(X)$ рас	$9{,}58\cdot10^{-01}$ $9{,}18\cdot10^{-01}$ $2{,}19\ 10^{-01}$	$8{,}26\cdot10^{-01}$ $7{,}40\cdot10^{-01}$ $2{,}28\ 10^{-01}$	$8{,}48\cdot10^{-01}$ $8{,}28\cdot10^{-01}$ $1{,}16\ 10^{-01}$	$8{,}14\cdot10^{-02}$ $6{,}96\cdot10^{-02}$ $5{,}39\ 10^{-03}$	$7{,}88\cdot10^{-02}$ $6{,}74\cdot10^{-02}$ $2{,}13\ 10^{-03}$	**$2{,}82\cdot10^{-03}$** **$8{,}44\cdot10^{-03}$** **$1{,}28\ 10^{-03}$**
$f_7(X)$ сал	$4{,}21\cdot10^{-02}$ $5{,}64\cdot10^{-01}$ $3{,}49\ 10^{-02}$	$6{,}52\cdot10^{-01}$ $7{,}46\cdot10^{-01}$ $3{,}35\ 10^{-01}$	$9{,}35\cdot10^{-01}$ $8{,}04\cdot10^{-01}$ $2{,}48\ 10^{-01}$	$7{,}56\cdot10^{-02}$ $6{,}28\cdot10^{-02}$ $2{,}24\ 10^{-03}$	$8{,}24\cdot10^{-02}$ $5{,}79\cdot10^{-03}$ $3{,}20\ 10^{-03}$	**$4{,}81\cdot10^{-03}$** **$3{,}62\cdot10^{-03}$** **$2{,}03\ 10^{-04}$**

Заключение

Результаты, полученные гибридным алгоритмом МАКП-МАРС на одноэкстремальных и многоэкстремальных многомерных тестовых функциях, превосходят результаты конкурирующих алгоритмов и являются статистически значимыми. Это связано с достигаемым балансом между скоростью сходимости алгоритма и диверсификацией пространства поиска решений.

Заслуживает внимания применение гибридного алгоритма МАКП-МАРС для решения задач динамической и стохастической оптимизации, многокритериальной оптимизации, мультимодальной оптимизации, многомерной оптимизации, меметической оптимизации, в которой комбинируется множество поисковых алгоритмов, оптимизации и адаптации настроек параметров метаэвристик для достижения баланса между скоростью сходимости и диверсификацией пространства поиска решений.

Список литературы

1. Родзин С.И. Современное состояние биоэвристик: классификация, бенчмаркинг, области применения // Изв. ЮФУ. Технич. науки. 2023. № 2. С. 280–298. doi: 10.18522/2311-3103-2023-2-280-298.

2. Rajwar K., Deep K., Das S. An exhaustive review of the metaheuristic algorithms for search and optimization: Taxonomy, applications, and open challenges. Artificial Intelligence Review, 2023, vol. 56, pp. 13187–13257. doi: 10.1007/s10462-023-10470-y.

3. Molina D., Poyatos J., Ser J.D. et al. Comprehensive taxonomies of nature- and bio-inspired optimization: Inspiration versus algorithmic behavior, critical analysis recommendations. Cognitive Computation, 2020, vol. 12, pp. 897–939. doi: 10.1007/s12559-020-09730-8.

4. Cuevas E., Fausto F., Gonzalez A. The locust swarm optimization algorithm. In: New Advancements in Swarm Algorithms: Operators and Applications. ISRL, 2019, vol. 160, pp. 139–159. doi: 10.1007/978-3-030-16339-6_5.

5. Camarena O., Cuevas E., Pérez-Cisneros M. et al. Ls-II: An improved locust search algorithm for solving optimization problems. Math. Problems in Eng., 2018, vol. 2018, art. 4148975. doi: 10.1155/2018/4148975.

6. Alekseev V. Application of artificial locust swarm routing algorithm for VFR flights planning. Math. Modeling, 2021, vol. 4, pp. 124–127.

7. Rodzin S., Kuliyev E., Zaporozhets D., Rodzina L., Rodzina O. Locust swarm optimization algorithm: A bio-heuristic for global optimization problem. In: LNNS. Proc. CSOC, 2023, vol. 722, pp. 670–678. doi: 10.1007/978-3-031-35311-6_64.

8. Luque-Chang A., Cuevas E., Fausto F. et al. Social spider optimization algorithm: modifications, applications, and perspectives. Math. Problems in Eng., 2018, vol. 2018, art. 6843923. doi: 10.1155/2018/6843923.

9. Evangeline D., Abirami T. Social spider optimization algorithm: Theory and its applications. IJITEE, 2019, vol. 8, no. 10, pp. 327–332. doi: 10.35940/ijitee.I8261.0881019.

10. Rodzin S., Rodzina L. Spider colony optimization algorithm: A bio-heuristic for global optimization problem. In: LNNS. Proc. CSOC, 2023, vol. 722, pp. 661–669. doi: 10.1007/978-3-031-35311-6_63.

11. Kerschke P., Trautmann H. Automated algorithm selection on continuous black-box problems by combining exploratory landscape analysis and machine learning. Evolutionary Comput., 2019, vol. 27, no. 1, pp. 99–127. doi: 10.1162/evco_a_00236.

12. Long W., Wu T., Liang X., Xu S. Solving high-dimensional global optimization problems using an improved sine cosine algorithm. Expert Systems with Applications, 2019, vol. 123, pp. 108–126. doi: 10.1016/j.eswa.2018.11.032.

13. Dragoi E., Dafinescu V. Review of metaheuristics inspired from the animal kingdom. Math., 2021, vol. 9, no. 18, art. 2335. doi: 10.3390/math9182335.

14. Benítez-Hidalgo A., Nebro A.J., García-Nieto J., Oregi I., Ser J.D. jMetalPy: A Python framework for multi-objective optimization with metaheuristics. Swarm and Evolutionary Computation, 2019, vol. 51, art. 100598. doi: 10.1016/j.swevo.2019.100598.

15. Tian Y., Cheng R., Zhang X., Jin Y. A MATLAB platform for evolutionary multi-objective optimization. IEEE Computational Intelligence Magazine, 2017, vol. 12, no. 4, pp. 73–87. doi: 10.1109/MCI.2017.2742868.

16. Иванов Д.К., Думина Д.С., Семенов Н.А. Определение весовых коэффициентов для аддитивной фитнес-функции генетического алгоритма // Программные продукты и системы. 2020. Т. 33. № 1. С. 47–53. doi: 10.15827/0236-235X.129.047-053.

17. Saha S., Seal D.B., Ghosh A., Dey K.N. A novel gene ranking method using Wilcoxon rank sum test and genetic algorithm. IJBRA, 2016, vol. 12, no. 3, pp. 263–279. doi: 10.1504/IJBRA.2016.078236.

Software & Systems doi: 10.15827/0236-235X.142.146-154 2024, 37(2), pp. 146–154

Global optimization based on hybridization of locust swarm and spider colony algorithm

Sergey I. Rodzin [1] ✉

[1] Southern Federal University, Taganrog, 347922, Russian Federation

For citation

Rodzin, S.I. (2024) 'Global optimization based on hybridization of locust swarm and spider colony algorithm', *Software & Systems*, 37(2), pp. 146–154 (in Russ.). doi: 10.15827/0236-235X.142.146-154

Article info

Received: 08.01.2024 After revision: 22.02.2024 Accepted: 06.03.2024

Abstract. A promising solution to global optimization problems are metaheuristics inspired by nature. They are non-deterministic algorithms that explore solution search space, learn in the search process; they are not tied to a specific task, although they do not guarantee accurate solutions. The purpose of this study is to develop an effective algorithm for solving applied problems of global optimization of multidimensional multi extreme functions in computational phylogenetics problems, in designing electrical circuits, calculations of building engineering safety, calibration of radio propagation models, and others. To achieve this goal, the authors of the paper propose a hybrid algorithm that simulates behavior patterns of a locust swarm and a spider colony. The paper focuses on the issue of reducing the probability of hybrid algorithm premature convergence, maintaining a balance between an algorithm convergence rate and the diversification of a solution search space (intensification/diversification). The paper presents the stages of modified algorithms for a spider colony and a locust swarm that model various patterns of their behavior, which reduces the effect of very good or bad decisions on the search process. The algorithms are hybridized by their sequential combination (preprocessor/postprocessor). The algorithm was tested on seven known multidimensional functions. The results were compared with competing algorithms for a particle swarm, a differential evolution, and a bee colony. The proposed algorithm provides the best results for all considered functions. Verification of the results obtained using the Wilcoxon sum of ranks T-test for independent samples showed that the algorithm results are statistically significant. The developed software application is intended for using in terms of a university course on machine learning and bioinspired optimization, as well as for solving a wide range of scientific and applied problems of search engine optimization.

Keywords: algorithm, global optimum, agent, locust swarm, spider colony, behavior pattern, search intensification, search diversification, test function, Wilcoxon test
Acknowledgements. The study was carried out at the Southern Federal University and supported by the Russian Science Foundation grant no. 23-21-00089, https://rscf.ru/project/23-21-00089/

References

1. Rodzin, S.I. (2023) 'Current state of bio heuristics: Classification, benchmarking, application areas', *Izv. SFedU. Eng. Sci.*, (2), pp. 280–298 (in Russ.). doi: 10.18522/2311-3103-2023-2-280-298.
2. Rajwar, K., Deep, K., Das, S. (2023) 'An exhaustive review of the metaheuristic algorithms for search and optimization: Taxonomy, applications, and open challenges', *Artificial Intelligence Review*, 56, pp. 13187–13257. doi: 10.1007/s10462-023-10470-y.
3. Molina, D., Poyatos, J., Ser, J.D. et al. (2020) 'Comprehensive taxonomies of nature- and bio-inspired optimization: Inspiration versus algorithmic behavior, critical analysis recommendations', *Cognitive Computation*, 12, pp. 897–939. doi: 10.1007/s12559-020-09730-8.
4. Cuevas, E., Fausto, F., Gonzalez, A. (2019) 'The locust swarm optimization algorithm', in *New Advancements in Swarm Algorithms: Operators and Applications. ISRL*, 160, pp. 139–159. doi: 10.1007/978-3-030-16339-6_5.
5. Camarena, O., Cuevas, E., Pérez-Cisneros, M. et al. (2018) 'Ls-II: An improved locust search algorithm for solving optimization problems', *Math. Problems in Eng.*, 2018, art. 4148975. doi: 10.1155/2018/4148975.
6. Alekseev, V. (2021) 'Application of artificial locust swarm routing algorithm for VFR flights planning', *Math. Modeling*, 4, pp. 124–127.
7. Rodzin, S., Kuliyev, E., Zaporozhets, D., Rodzina, L., Rodzina, O. (2023) 'Locust swarm optimization algorithm: A bio-heuristic for global optimization problem', in *LNNS. Proc. CSOC*, 722, pp. 670–678. doi: 10.1007/978-3-031-35311-6_64.
8. Luque-Chang, A., Cuevas, E., Fausto, F. et al. (2018) 'Social spider optimization algorithm: modifications, applications, and perspectives', *Math. Problems in Eng.*, 2018, art. 6843923. doi: 10.1155/2018/6843923.
9. Evangeline, D., Abirami, T. (2019) 'Social spider optimization algorithm: Theory and its applications', *IJITEE*, 8(10), pp. 327–332. doi: 10.35940/ijitee.I8261.0881019.
10. Rodzin, S., Rodzina, L. (2023) 'Spider colony optimization algorithm: A bio-heuristic for global optimization problem', in *LNNS. Proc. CSOC*, 722, pp. 661–669. doi: 10.1007/978-3-031-35311-6_63.
11. Kerschke, P., Trautmann, H. (2019) 'Automated algorithm selection on continuous black-box problems by combining exploratory landscape analysis and machine learning', *Evolutionary Comput.*, 27(1), pp. 99–127. doi: 10.1162/evco_a_00236.
12. Long, W., Wu, T., Liang, X., Xu, S. (2019) 'Solving high-dimensional global optimization problems using an improved sine cosine algorithm', *Expert Systems with Applications*, 123, pp. 108–126. doi: 10.1016/j.eswa.2018.11.032.
13. Dragoi, E., Dafinescu, V. (2021) 'Review of metaheuristics inspired from the animal kingdom', *Math.*, 9(18), art. 2335. doi: 10.3390/math9182335.
14. Benítez-Hidalgo, A., Nebro, A.J., García-Nieto, J., Oregi, I., Ser, J.D. (2019) 'jMetalPy: A Python framework for multi-objective optimization with metaheuristics', *Swarm and Evolutionary Computation*, 51, art. 100598. doi: 10.1016/j.swevo.2019.100598.
15. Tian, Y., Cheng, R., Zhang, X., Jin, Y. (2017) 'A MATLAB platform for evolutionary multi-objective optimization', *IEEE Computational Intelligence Magazine*, 12(4), pp. 73–87. doi: 10.1109/MCI.2017.2742868.
16. Ivanov, D.K., Dumina, D.S., Semenov, N.A. (2020) 'Determination of weight coefficients for additive fitness function of genetic algorithm', *Software & Systems*, 33(1), pp. 47–53 (in Russ.). doi: 10.15827/0236-235X.129.047-053.
17. Saha, S., Seal, D.B., Ghosh, A., Dey, K.N. (2016) 'A novel gene ranking method using Wilcoxon rank sum test and genetic algorithm', *IJBRA*, 12(3), pp. 263–279. doi: 10.1504/IJBRA.2016.078236.

Авторы
Родзин Сергей Иванович [1], к.т.н., доцент, профессор, srodzin@sfedu.ru

[1] Южный федеральный университет, г. Таганрог, 347922, Россия

Authors
Sergey I. Rodzin [1], Cand. of Sci. (Engineering), Associate Professor, Professor, srodzin@sfedu.ru

[1] Southern Federal University, Taganrog, 347922, Russian Federation