

SelfCSHAGA: САМОКОНФИГУРИРУЕМЫЙ ГЕНЕТИЧЕСКИЙ АЛГОРИТМ ОПТИМИЗАЦИИ С АДАПТАЦИЕЙ НА ОСНОВЕ ИСТОРИИ УСПЕХА

П.А. Шерстнев¹

psherstnev@sfu-kras.ru

Е.С. Семенкин²

eugenesemenkin@yandex.ru

¹ СФУ, г. Красноярск, Российская Федерация

² СибГУ им. М.Ф. Решетнева, г. Красноярск, Российская Федерация

Аннотация

Рассмотрена проблема повышения эффективности генетических алгоритмов оптимизации за счет применения методов самонастройки, которые изменяют параметры и поведение алгоритма в процессе поиска решения. Приведен обзор актуальных и наиболее эффективных методов самонастройки и адаптации, в рамках которого выделены их преимущества и недостатки. Предложен новый алгоритм, позволяющий объединить лучшие стороны отдельных методов. Он представляет собой расширенную версию SHAGA с усовершенствованной процедурой скрещивания, которая позволяет адаптировать его интенсивность, применять селективное давление на данном этапе и использовать многокритериальное скрещивание. Предложены различные варианты оператора скрещивания и метод самоконфигурирования генетических операторов на основе SelfCEA, который динамически корректирует вероятности их применения в зависимости от успешности. Предложенный алгоритм протестирован с применением статистических критериев для проверки значимости различий результатов и сравнения с другими подходами в задачах оптимизации с вещественными и булевыми переменными. В результате тестирования новый генетический алгоритм продемонстрировал более высокую эффективность и значительное улучшение надежности в большинстве тестовых задач

Ключевые слова

Генетические алгоритмы, оптимизация, самоконфигурирование, самонастройка, генетические операторы

Поступила 29.01.2025

Принята 10.03.2025

© Автор(ы), 2025

Работа выполнена при поддержке Минобрнауки России в рамках государственного задания в сфере науки (проект № FEFE-2023-0004)

Введение. Эволюционные алгоритмы (ЭА) — мощные инструменты оптимизации в сложных системах, их эффективность зависит от выбора генетических операторов и параметров [1]. Поскольку структура функций неизвестна, заранее определить оптимальные настройки невозможно [2]. На разных стадиях поиска могут быть эффективны различные комбинации операторов и параметры: на начальных этапах важен глобальный охват, а на поздних — уточнение найденных решений. Для решения этих проблем разрабатываются самонастраивающиеся алгоритмы, которые динамически подстраиваются под задачу и стадию поиска [3, 4]. Они делятся на две категории: самоконфигурируемые, изменяющие виды генетических операторов, и адаптивные, корректирующие численные параметры в процессе поиска. Самонастройка является важным направлением в исследованиях ЭА. Алгоритмы с ее использованием выступают ключевым инструментом на конференциях CEC¹ (*Computational Intelligence Conference on Evolutionary Computation*), где рассматриваются методы оптимизации для задач с минимальной информацией о системе. Значительное внимание этой теме уделяется также на конференциях GECCO² (*Genetic and Evolutionary Computation Conference*), где рассматриваются передовые разработки ЭА.

Постановка задачи оптимизации. Рассмотрим задачу однокритериальной безусловной глобальной оптимизации типа «черный ящик», где целевая функция не имеет явного аналитического выражения и ее свойства заранее неизвестны. Формально задача заключается в определении вектора параметров x^* , минимизирующего (или максимизирующего) целевую функцию:

$$x^* = \arg \min_{x \in \Omega} f(x), \quad (1)$$

где Ω — множество допустимых решений. В реальных сценариях невозможно точно определить глобальный минимум из-за численных ошибок и неопределенностей, поэтому критерий оптимальности ослабляется:

$$f(x^*) - \varepsilon \leq \min_{x \in \Omega} f(x), \quad (2)$$

¹ Computational Intelligence Conference on Evolutionary Computation (CEC). URL: <https://www.cec2025.org> (дата обращения: 08.01.2025).

² Genetic and Evolutionary Computation Conference (GECCO). URL: <https://gecco-2025.sigevo.org> (дата обращения: 08.01.2025).

где ε — параметр, определяющий допустимый уровень погрешности при поиске оптимального решения. Для решения подобных задач традиционные методы могут быть неэффективны, поскольку они зависят от свойств и структуры функции. В таких случаях ЭА, включая генетические алгоритмы, предоставляют более гибкий подход.

Генетический алгоритм. Генетический алгоритм (ГА) — семейство ЭА, решающих задачи оптимизации с вещественными, целочисленными, комбинаторными и разношкальными переменными. С помощью бинарного кодирования ГА ищет точку в гиперкубе, которая обеспечивает экстремум функции пригодности (ФП). Первая популяция в ГА инициализируется случайно по всему пространству поиска, после чего в цикле выполняются три этапа [5–7].

Этап 1. Селекция: отбираются решения-родители для следующих этапов эволюции с использованием операторов селекции. Основных операторов три:

- 1) пропорциональная селекция — вероятность выбора индивида пропорциональна его значению ФП;
- 2) ранговая селекция пропорциональна рангу по значению ФП;
- 3) турнирная селекция — выбирается индивид с наилучшим значением ФП из случайной подгруппы определенного размера.

Этап 2. Скрещивание: отобранные решения комбинируются для передачи информации потомкам. Выделяют несколько видов скрещивания:

- одноточечное, хромосомы родителей разделяются в одной случайной точке, части меняются местами;
- двухточечное, разделение в двух случайных точках, части меняются местами;
- равномерное, каждый ген потомка выбирается случайно (равновероятно) из генов родителей.

Этап 3. Мутация: вносятся небольшие случайные изменения в решения, поддерживается разнообразие популяции. В ГА с бинарным кодированием оператор мутации инвертирует биты с определенной вероятностью. Виды мутации различаются по вероятности изменения, например: средняя, слабая и сильная.

Для повышения эффективности поиска предложены модифицированные операторы, включая равномерное скрещивание с селективным давлением [8–10]. Его принцип действия заключается в наследовании битов от родителей с вероятностью, зависящей от значений их ФП, что дает преимущество более приспособленным особям. Предлагаются три варианта равномерного скрещивания с селективным давлением:

1) пропорциональное равномерное — ген потомка выбирается от родителя с вероятностью, пропорциональной значению ФП;

2) ранговое равномерное — с вероятностью, пропорциональной рангу по его значению ФП;

3) турнирное равномерное — ген передает родитель с наивысшим значением ФП из случайной группы заданного размера. В этом случае классическое равномерное скрещивание называется равновероятным равномерным. Турнирное равномерное скрещивание используется, когда родителей больше двух, так как при двух родителях потомок становится копией более пригодного (клоном).

Методы самонастройки ГА. Рассмотрены три популярных и эффективных метода самонастройки ГА, доказавшие результативность в исследованиях.

Алгоритм SelfCGA (Self-Configuring Genetic Algorithm) — в этом ГА при создании нового потомка вариант генетического оператора выбирается случайным образом из заданного набора. Сначала вероятности выбора операторов равны, но в процессе поиска решения они корректируются в пользу более эффективных по формуле из [8–10]:

$$p_i = \frac{(z-1)K}{zN}, \quad i = 1, 2, \dots, z, \quad (3)$$

где p_i — вероятность выбора каждого варианта оператора; z — число вариантов; K — постоянная, контролирующая скорость изменения вероятности; N — число поколений.

Алгоритм PDPGA (Population-Level Dynamic Probabilities Genetic Algorithm) — ГА самоконфигурирования, в котором вероятность применения генетических операторов настраивается на основе их успешности. Успешным считается оператор, если потомок, созданный с его помощью, превосходит родителя по значению ФП. Для каждого варианта оператора рассчитываются значения по формуле из [11–13]:

$$r_i = \frac{success_i^2}{used_i}, \quad (4)$$

где $success_i$ — число успешных применений этого варианта оператора; $used_i$ — число использований данного варианта оператора. Затем для каждого варианта оператора вычисляется вероятность его применения:

$$p_i = p_{all} + \left[r_i \frac{1 - np_{all}}{scale} \right], \quad scale = \sum_{j=1}^n r_j. \quad (5)$$

Алгоритм SHAGA (Success History-Based Adaptive Genetic Algorithm) адаптирует вероятности мутации M_r и скрещивания C_r на основе истории успешных применений. Первоначально SHA (Success History-Based Parameter Adaptation) был предложен для дифференциальной эволюции (ДЭ) [14, 15], а позднее успешно адаптирован для ГА [16]. В SHAGA текущий индивид фиксируется как первый родитель, второй выбирается с помощью турнирной селекции (размер турнира равен 2). Оператор скрещивания применяется к каждому гену отдельно, определяя, наследуется ли он от второго родителя с вероятностью, вычисляемой как

$$C_r = randn(MC_r, 0, 1), \quad (6)$$

где $randn$ — случайное число с нормальным распределением; MC_r — запоминаемое значение вероятности скрещивания, обновляемое на основе успешности применения на предыдущих поколениях. Оператор мутации инвертирует биты строки с вероятностью M_r , которая динамически обновляется как

$$M_r = randc(MM_r, 0, 1). \quad (7)$$

Здесь $randc$ — генератор случайных чисел по распределению Коши; MM_r — запоминаемое значение вероятности мутации. Эти параметры адаптируются в ходе работы алгоритма: если применение оператора улучшает решение, то соответствующие значения запоминаются и используются для генерации последующих. Такой механизм позволяет SHAGA гибко настраивать интенсивности мутации и скрещивания в зависимости от задачи оптимизации.

Предлагаемый ГА. В оригинальной версии алгоритма SHAGA используется только одна конфигурация генетических операторов [16]. Эффективность других операторов, таких как ранговая или пропорциональная селекция и равномерное скрещивание с селективным давлением [8], остается неисследованной.

Для использования различных операторов селекции в SHAGA не требуется их изменять, но необходимо выбирать на одного родителя меньше, чем в классическом ГА. В исследовании применяются следующие операторы: пропорциональная селекция, ранговая селекция, турнирные селекции с размером турниров 3, 5, 7.

В SHAGA вероятность скрещивания определяет интенсивность изменений: высокие значения усиливают вариативность и способствуют глобальному поиску, низкие — делают поиск локальным. В операторах

равномерного скрещивания с селективным давлением [8] вероятность применяется иначе: каждый ген наследуется случайно от одного родителя, причем шанс зависит от значений ФП. Оператор скрещивания в SHAGA можно модифицировать для учета параметра C_r и значений ФП родителей, регулируя интенсивность изменений и добавляя селективное давление. Наследование гена происходит в два этапа. На первом этапе с вероятностью C_r определяется, будет ли ген унаследован от первого родителя (текущего решения) или от других. Если ген наследуется от первого родителя или родителей всего два, то переходят к следующему гену. Иначе выполняется второй этап — выбор среди оставшихся родителей, где вероятность определяется значениями ФП [8]. Такая модификация оператора скрещивания для SHAGA позволяет использовать многородительское скрещивание, настраивать интенсивность через параметр C_r и учитывать селективное давление. При участии двух родителей оператор работает как в оригинальном SHAGA. Доступны классические варианты скрещивания — одноточечное и двухточечное, которые выполняются с вероятностью C_r . Если скрещивание не происходит, то возвращается первый родитель (текущее решение).

Авторы [8] определили, что оптимальное число родителей для большинства операторов скрещивания составляет 2 и 7, а для турнирного скрещивания — 3 и 7. Однако в этом алгоритме на этапе скрещивания используется дополнительное селективное давление с помощью оператора селекции на этапе 2, что увеличивает общее число родителей на одного по сравнению с оригинальной реализацией. В настоящем исследовании используются следующие операторы скрещивания:

- равномерное, число родителей 2 (оператор из оригинального алгоритма SHAGA);
- одноточечное скрещивание;
- двухточечное скрещивание;
- равномерное равновероятное, число родителей 3 или 8;
- равномерное пропорциональное, число родителей 3 или 8;
- равномерное ранговое, число родителей 3 или 8;
- равномерное турнирное, число родителей 4 или 8 (размер турниров 2).

Таким образом, алгоритм учитывает как классические варианты операторов скрещивания, так и их модификации. Генетический алгоритм с адаптацией параметров на основе истории успешных применений, использующий разные реализации операторов скрещивания и селекции, будет далее называться конфигурируемым SHAGA (Configuring SHAGA — CSHAGA).

Множество доступных генетических операторов для алгоритма CSHAGA создает сложность в выборе оптимальных настроек для каждой новой задачи оптимизации. Алгоритм SelfCSHAGA — предлагаемый гибридный подход, который сочетает идеи самоконфигурирования операторов из SelfCGA и адаптивную настройку параметров из CSHAGA, обеспечивая полную самонастройку ГА.

Рассмотрим работу алгоритма SelfCSHAGA.

1. Инициализация. Алгоритм запускается с фиксированными начальными значениями параметров мутации и скрещивания, аналогично оригинальной реализации. Эти значения сохраняются в памяти и имеют распределение, соответствующее начальной версии SHAGA. Вероятности применения операторов задаются равными, как в SelfCGA.

2. Формирование нового поколения. Перед созданием потомка генерируются значения C_r и M_r по схеме из [16], а операторы выбираются на основе их вероятностей применения. С использованием этих параметров формируется новый потомок.

3. Адаптация параметров. Успешные значения C_r и M_r сохраняются для последующей адаптации.

4. Самоконфигурирование операторов. После формирования нового поколения значения вероятностей применения вариантов генетических операторов обновляются.

Исследование эффективности ГА. Для исследования CSHAGA и SelfCSHAGA и сравнения с другими алгоритмами сформировано два набора задач: 1) набор сложных тестовых функций из CEC2005³, CEC2013 и CEC2014, содержащий задачи вещественной оптимизации ($f(X): \mathbb{R}^n \rightarrow \mathbb{R}^1$) различной размерности от 2 до 30 переменных (в табл. 1 приведены функции, их характеристики (размерность пространства, границы переменных) и параметры алгоритма (число итераций и размер популяции для оптимизации целевой функции); 2) набор сложных задач глобальной псевдобулевой оптимизации ($f(X): \mathbb{B}^n \rightarrow \mathbb{R}^1$) из [17], содержащий функции F11–F16, которые характеризуются множеством локальных оптимумов (в табл. 2 приведены характеристики функций и параметры генетического алгоритма).

³ Suganthan, P.N., Hansen, N., Liang, J.J., et al. Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. Technical Report no. 2005005. Singapore, Nanyang Technological University, 2005.

Таблица 1

Функции и параметры для набора задач вещественной оптимизации

Функция	Размерность	Граница		Число итераций	Размер популяции
		левая	правая		
ShiftedRosenbrock	2	-100	100	270	270
ShiftedRotatedGriewank		-1000	1000	650	650
ShiftedExpandedGriewankRosenbrock		-3	1	110	110
RotatedVersionHybridCompositionFunction1		-5	5	130	130
RotatedVersionHybridCompositionFunction1 Noise				130	130
RotatedHybridCompositionFunction				250	250
HybridCompositionFunction3				477	477
HybridCompositionFunction3H				477	477
NonContinuousHybridCompositionFunction3				477	477
HybridCompositionFunction4				799	799
HybridCompositionFunction4withoutbounds		-10	10	1000	1000
Rosenbrock		-2,048	2,048	100	100
ExpandedScaffers_F6	-100	100	100	100	
Weierstrass	-1	1	1157	1157	
ShiftedSphere	10	-100	100	125	
ShiftedSchwefe1_2			100	728	
ShiftedSchwefe1_2WithNoise			100	757	
ShiftedRastrigin		5	470		
ShiftedRotatedRastrigin		5	463		
HybridCompositionFunction1		5	799		

Окончание табл. 1

Функция	Размерность	Граница		Число итераций	Размер популяции
		левая	правая		
Sphere	30	-5,12	5,12	210	210
HighConditionedElliptic		-100	100	355	355
Griewank		-600	600	600	600
Ackley		-32,768	32,768	247	247
Rastrigin		-5,12	5,12	799	799

Функции и параметры для набора задач псевдобулевой оптимизации

Функция	Размерность	Число итераций	Размер популяции
OneMax	100	200	100
F11	30	200	250
F12	30	200	250
F13	24	200	250
F14	30	200	250
F15	30	200	250
F16	40	200	250

Эволюционные алгоритмы являются стохастическими, и результаты варьируются при каждом запуске, поэтому для их оценки использовались результаты множества запусков. Каждая конфигурация тестировалась 100 раз для каждой функции. Полагалось, что ГА нашел решение задачи вещественной оптимизации, если точность по положению точки составляла 0,01, а для псевдобулевых задач — если бинарная строка полностью совпадала с оптимальным решением. Метрикой для сравнения алгоритмов выступала надежность — доля найденных с заданной точностью решений от общего числа запусков. Объем ресурсов для работы ГА выбирался для каждой функции так, чтобы надежность алгоритма не достигала нуля или единицы, иначе сравнение алгоритмов становилось невозможным. Для псевдобулевых задач из [17] применялся объем ресурсов, указанный авторами.

Различия в метриках проверялись на статистическую значимость с помощью U-критерия Манна — Уитни, который подходит для малых выборок и данных, не соответствующих нормальному распределению [18, 19]. Уровень значимости составлял 0,05. Для каждого ГА надежность рассчитывалась 10 раз, выборка состояла из 10 измерений, полученных из 100 независимых запусков.

В исследовании использовались следующие ГА.

1. Genetic Algorithm — ГА с операторами скрещивания с селективным давлением. Всего 165 конфигураций.
2. SelfCGA — самоконфигурируемая версия ГА на основе SelfCEA с расширенным набором операторов, включая селективное давление.
3. PDPGA — PDP-версия ГА с операторами с селективным давлением [11].

4. SHAGA — реализация алгоритма SHAGA из оригинальной статьи [16].

5. CSHAGA — предлагаемая расширенная версия SHAGA с разными конфигурациями генетических операторов. Всего 60 конфигураций.

6. SelfCSHAGA — предлагаемая самоконфигурируемая версия CSHAGA на основе SelfCEA.

Усредненные значения надежности шести разных вариантов ГА, полученные на основе задач вещественной оптимизации с переменными из табл. 1, приведены на рис. 1. Если алгоритм не является самонастраиваемым (например, GA или CSHAGA), то для него по всем конфигурациям усредняются значения надежности, а также фиксируется максимальная надежность, достигнутая на одной (лучшей) конфигурации.

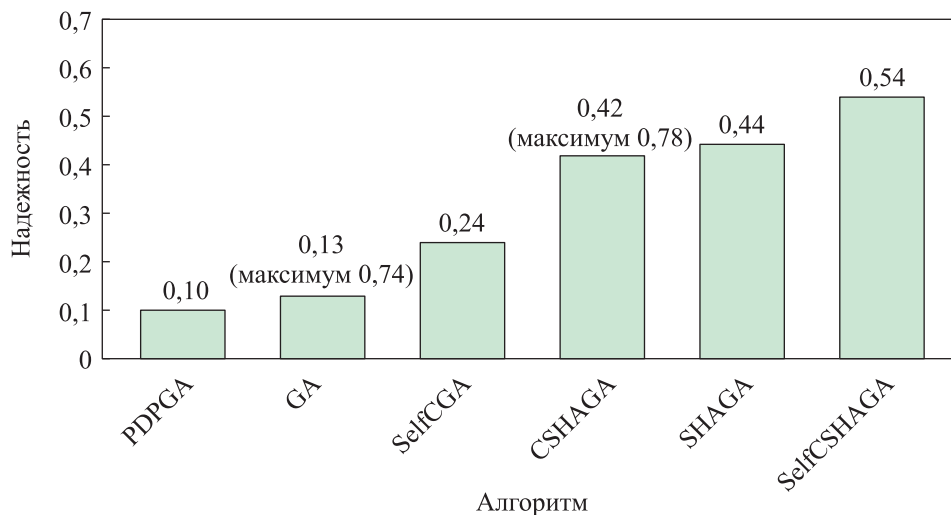


Рис. 1. Усредненные значения надежности ГА

Согласно рис. 1, алгоритм SelfCSHAGA достиг наивысшей средней надежности среди всех исследуемых алгоритмов. Алгоритм CSHAGA показал лучшие результаты, чем классический Genetic Algorithm, но его средняя надежность оказалась ниже, чем у SelfCSHAGA и SHAGA. Таким образом, при отсутствии информации о свойствах оптимизируемой функции предпочтительнее использовать алгоритмы SelfCSHAGA или SHAGA вместо CSHAGA с произвольной конфигурацией.

Результаты статистического теста для сравнения алгоритма SelfCSHAGA с другими ГА самонастройки приведены на рис. 2. Диаграммы сравнения ГА SelfCSHAGA с SelfCGA, PDPGA и SHAGA выполнены на основе результатов решений задач вещественной оптимизации. Результаты

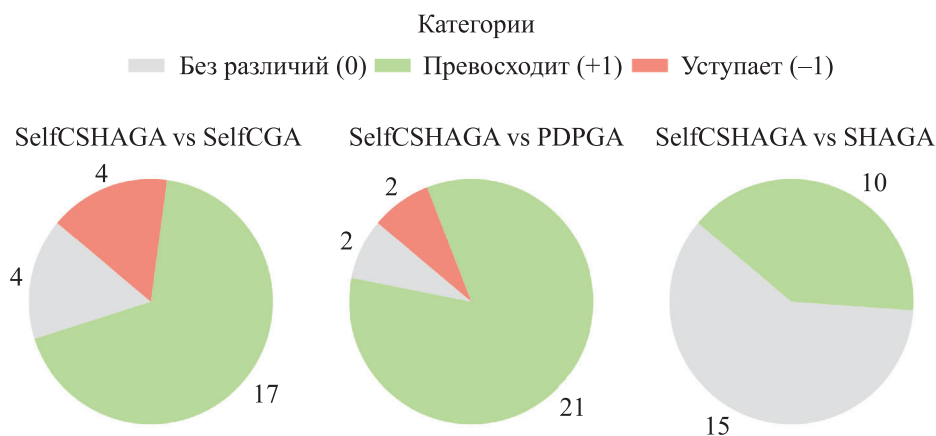


Рис. 2. Результаты сравнения SelfCSHAGA с другими алгоритмами, полученные на основе решений задач вещественной оптимизации с использованием статистического критерия

разделены на три категории: 1) «превосходит» (зеленый сектор) — число функций, где алгоритм SelfCSHAGA был лучше; 2) «без различий» (серый сектор) — различия статистически незначимы; 3) «уступает» (красный сектор) — другой алгоритм показал лучшие результаты.

Согласно рис. 2, предложенный алгоритм SelfCSHAGA либо превосходит, либо демонстрирует сопоставимую эффективность по сравнению с существующими самонастраиваемыми ГА в большинстве задач вещественной оптимизации. Тем не менее для задач с вещественным представлением решений более подходящими являются адаптивные алгоритмы дифференциальной эволюции — победители мировых соревнований по эволюционной оптимизации [20]. Основное преимущество ГА заключается в их универсальности и возможности решать задачи псевдобулевой, дискретной и разношкальной оптимизации. В связи с этим проведено дополнительное исследование на задачах псевдобулевой оптимизации (см. табл. 2). Показатели надежности и результаты статистического сравнения алгоритмов приведены на рис. 3 и 4.

Предложенный алгоритм SelfCSHAGA достиг наивысшего среднего значения надежности по всему набору задач, что также подтверждается решениями задач вещественной оптимизации. В то же время среднее значение надежности алгоритма CSHAGA оказалось ниже, хотя максимальный результат надежности у двух конфигурируемых достиг единицы. Статистическое сравнение показало, что в большинстве случаев SelfCSHAGA не уступает другим алгоритмам, а для некоторых задач превосходит их.

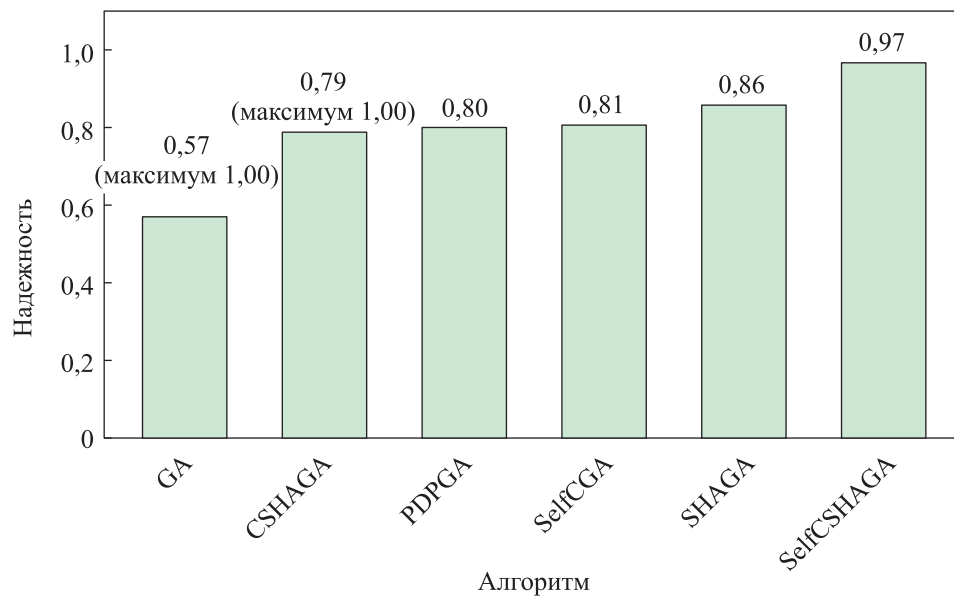


Рис. 3. Показатели надежности алгоритмов
(по результатам решений задач псевдоболевой оптимизации)

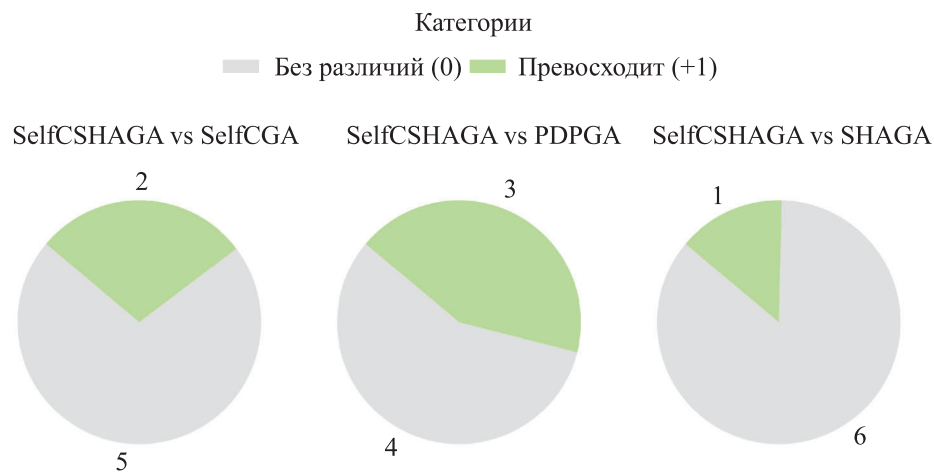


Рис. 4. Сравнение алгоритма SelfCSHAGA с другими
(по результатам решений задач псевдоболевой оптимизации
с использованием статистического критерия)

Закключение. Предложен и исследован новый самоконфигурируемый адаптивный ГА на основе истории успешных применений (CSHAGA), отличающийся расширенной процедурой скрещивания. Эта процедура обеспечивает не только настройку интенсивности скрещивания, но и возможность применения селективного давления на этапе скрещивания и многократного скрещивания, что обеспечивает большую гибкость

алгоритма и его адаптацию к специфике задачи оптимизации. Предложен самоконфигурируемый алгоритм CSHAGA (SelfCSHAGA), который адаптирует вещественные параметры на основе истории успешных применений (SHA) и выбирает наиболее эффективные генетические операторы с помощью SelfCEA. По результатам тестирования алгоритм SelfCSHAGA продемонстрировал наивысшую среднюю надежность при решении тестовых задач вещественной и псевдобулевой оптимизации по сравнению с аналогами.

Результаты исследования позволяют сделать вывод о перспективности предложенного алгоритма и наметить цели его дальнейшего развития: анализ эффективности при решении задач большей размерности, включая задачи WCCI CEC; модификация алгоритма для решения задач условной оптимизации; решение реальных практических задач. Наиболее важным направлением развития алгоритма представляется исследование эффективности предложенных алгоритмов для задач разношкальной оптимизации.

ЛИТЕРАТУРА

- [1] Wang Z., Pei Y., Li J. A survey on search strategy of evolutionary multi-objective optimization algorithms. *Appl. Sc.*, 2023, vol. 13, no. 7, art. 4643. DOI: <https://doi.org/10.3390/app13074643>
- [2] Wolpert D.H., Macready W.G. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.*, 1997, vol. 1, no. 1, pp. 67–82. DOI: <https://doi.org/10.1109/4235.585893>
- [3] Kramer O. Evolutionary self-adaptation: a survey of operators and strategy parameters. *Evol. Intel.*, 2010, vol. 3, no. 2, pp. 51–65. DOI: <https://doi.org/10.1007/s12065-010-0035-y>
- [4] Meyer-Nieberg S., Beyer H.G. Self-adaptation in evolutionary algorithms. In: *Parameter setting in evolutionary algorithms*. Berlin, Heidelberg, Springer, 2007, pp. 47–75. DOI: https://doi.org/10.1007/978-3-540-69432-8_3
- [5] Holland J.H. Genetic algorithms. *Sc. Am.*, 1992, vol. 267, no. 1, pp. 66–72.
- [6] Vie A., Kleinnijenhuis A.M., Farmer D.J. Qualities, challenges and future of genetic algorithms: a literature review. arXiv:2011.05277. DOI: <https://doi.org/10.48550/arXiv.2011.05277>
- [7] Alam T., Qamar S., Dixit A., et al. Genetic algorithm: reviews, implementations, and applications. *ijEP*, 2020, vol. 10, no. 6, pp. 57–76. DOI: <https://doi.org/10.36227/techrxiv.12657173>
- [8] Semenkin E.S., Semenkina M.E. Self-configuring genetic algorithm with modified uniform crossover operator. In: *Lecture Notes in Computer Science*, Cham, Springer Nature Switzerland, 2012, pp. 414–421. DOI: <https://doi.org/10.1007/978-3-642-30976-2>
- [9] Semenkin E., Semenkina M. Self-configuring genetic programming algorithm with modified uniform crossover. *IEEE CEC*, 2012. DOI: <https://doi.org/10.1109/CEC.2012.6256587>

- [10] Семенкина М.Е. Самоадаптивные эволюционные алгоритмы проектирования информационных технологий интеллектуального анализа данных. *Искусственный интеллект и принятие решений*, 2013, № 1, с. 13–24. EDN: QBWXYD
- [11] Niehaus J., Banzhaf W. Adaption of operator probabilities in genetic programming. In: *Genetic Programming*. Berlin, Springer-Verlag, 2001, pp. 325–336. DOI: https://doi.org/10.1007/3-540-45355-5_26
- [12] Липинский Л.В., Кушнарева Т.В. Исследование моделей и процедур самоконфигурации генетического программирования для формирования деревьев принятия решений в задачах интеллектуального анализа данных. *Вестник СибГАУ им. академика М.Ф. Решетнева*, 2016, т. 17, № 3, с. 579–586. EDN: WVPTZH
- [13] Sopov A., Karaseva T. The comparison of different PDP-type self-adaptive schemes for the cooperation of GA, DE, and PSO algorithms. *ITM Web Conf.*, 2024, vol. 59, art. 04013. DOI: <https://doi.org/10.1051/itmconf/20245904013>
- [14] Tanabe R., Fukunaga A. Success-history based parameter adaptation for differential evolution. *IEEE CEC*, 2013, pp. 71–78. DOI: <https://doi.org/10.1109/CEC.2013.6557555>
- [15] Rivera-Lopez R., Mezura-Montes E., Canul-Reich J., et al. An experimental comparison of self-adaptive differential evolution algorithms to induce oblique decision trees. *Math. Comput. Appl.*, 2024, vol. 29, no. 6, p. 103. DOI: <https://doi.org/10.3390/mca29060103>
- [16] Stanovov V., Akhmedova S., Semenkin E. Genetic algorithm with success history based parameter adaptation. *IJCCI 2019*, 2019, pp. 180–187. DOI: <https://doi.org/10.5220/0008071201800187>
- [17] Yu E.L., Suganthan P.N. Ensemble of niching algorithms. *Inform. Sc.*, 2010, vol. 180, no. 15, pp. 2815–2833. DOI: <https://doi.org/10.1016/j.ins.2010.04.008>
- [18] Mann H.B., Whitney D.R. On a test of whether one of two random variables is stochastically larger than the other. *Ann. Math. Statist.*, 1947, vol. 18, no. 1, pp. 50–60. DOI: <https://doi.org/10.1214/aoms/1177730491>
- [19] Zhu X. Sample size calculation for Mann — Whitney U test with five methods. *Int. J. Clin. Trials*, 2021, vol. 8, no. 3, pp. 184–190. DOI: <https://doi.org/10.18203/2349-3259.ijct20212840>
- [20] Stanovov V., Semenkin E. Success rate-based adaptive differential evolution L-SRTDE for CEC 2024 Competition. *IEEE CEC*, 2024, pp. 1–8. DOI: <https://doi.org/10.1109/CEC60901.2024.10611907>]

Шерстнев Павел Александрович — аспирант кафедры программной инженерии СФУ, инженер-исследователь Центра искусственного интеллекта СФУ (Российская Федерация, 660041, г. Красноярск, Свободный пр-т, д. 79).

Семенкин Евгений Станиславович — д-р техн. наук, профессор кафедры системного анализа и исследования операций СибГУ им. М.Ф. Решетнева (Российская Федерация, 660037, г. Красноярск, пр-т имени газеты «Красноярский рабочий», д. 31).

Просьба ссылаться на эту статью следующим образом:

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. EDN: TSKBOX

**SelfCSHAGA: SELF-CONFIGURING GENETIC
OPTIMIZATION ALGORITHM
WITH THE SUCCESS-HISTORY BASED ADAPTATION**

P.A. Sherstnev¹

psherstnev@sfu-kras.ru

E.S. Semenkin²

eugenesemenkin@yandex.ru

¹ Siberian Federal University, Krasnoyarsk, Russian Federation

² Reshetnev Siberian State University of Science and Technology,
Krasnoyarsk, Russian Federation

Abstract

The paper considers a problem of increasing the genetic optimization algorithm efficiency by using the self-tuning methods that change the algorithm parameters and behavior in finding a solution. It presents an overview of the current and most efficient self-tuning and adaptation methods, their advantages and disadvantages are highlighted. The paper proposes a new algorithm that allows combining best aspects of the separate methods, and is an extended version of the SHAGA method with the improved crossover procedure making it possible to adapt its intensity, apply selective pressure at this stage and use the multi-parent crossover. Within the algorithm framework, the paper proposes various options of the crossover operator and a self-configuration method for the genetic operators based on the SelfCEA approach, which dynamically adjusts probabilities of application depending on their success. The proposed algorithm is tested using the statistical criteria to verify significance of differences in the results and compare with other approaches in the optimization problems with the real and Boolean variables. The testing results of the new genetic algorithm are demonstrating higher efficiency and significant reliability improvement in most test problems

Keywords

Genetic algorithms, optimization, self-configuration, self-tuning, genetic operators

Received 29.01.2025

Accepted 10.03.2025

© Author(s), 2025

The work was performed with support by the Ministry of Science and Higher Education of the Russian Federation within the framework of the State Task (project no. FSFN-2023-0004)

REFERENCES

- [1] Wang Z., Pei Y., Li J. A survey on search strategy of evolutionary multi-objective optimization algorithms. *Appl. Sc.*, 2023, vol. 13, no. 7, art. 4643.
DOI: <https://doi.org/10.3390/app13074643>
- [2] Wolpert D.H., Macready W.G. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.*, 1997, vol. 1, no. 1, pp. 67–82.
DOI: <https://doi.org/10.1109/4235.585893>
- [3] Kramer O. Evolutionary self-adaptation: a survey of operators and strategy parameters. *Evol. Intel.*, 2010, vol. 3, no. 2, pp. 51–65.
DOI: <https://doi.org/10.1007/s12065-010-0035-y>
- [4] Meyer-Nieberg S., Beyer H.G. Self-adaptation in evolutionary algorithms. In: *Parameter setting in evolutionary algorithms*. Berlin, Heidelberg, Springer, 2007, pp. 47–75.
DOI: https://doi.org/10.1007/978-3-540-69432-8_3
- [5] Holland J.H. Genetic algorithms. *Sc. Am.*, 1992, vol. 267, no. 1, pp. 66–72.
- [6] Vie A., Kleinnijenhuis A.M., Farmer D.J. Qualities, challenges and future of genetic algorithms: a literature review. arXiv:2011.05277.
DOI: <https://doi.org/10.48550/arXiv.2011.05277>
- [7] Alam T., Qamar S., Dixit A., et al. Genetic algorithm: reviews, implementations, and applications. *ijEP*, 2020, vol. 10, no. 6, pp. 57–76.
DOI: <https://doi.org/10.36227/techrxiv.12657173>
- [8] Semenkin E.S., Semenkina M.E. Self-configuring genetic algorithm with modified uniform crossover operator. In: *Lecture notes in computer science*, Cham, Springer Nature Switzerland, 2012, pp. 414–421. DOI: <https://doi.org/10.1007/978-3-642-30976-2>
- [9] Semenkin E., Semenkina M. Self-configuring genetic programming algorithm with modified uniform crossover. *IEEE CEC*, 2012.
DOI: <https://doi.org/10.1109/CEC.2012.6256587>
- [10] Semenkina M.E. Effectiveness investigation of adaptive evolutionary algorithms for data mining information technology design. *Iskusstvennyy intellekt i prinyatie resheniy* [Artificial Intelligence and Decision Making], 2013, no. 1, pp. 13–24 (in Russ.).
EDN: QBWXYD
- [11] Niehaus J., Banzhaf W. Adaption of operator probabilities in genetic programming. In: *Genetic programming*. Berlin, Springer-Verlag, 2001, pp. 325–336.
DOI: https://doi.org/10.1007/3-540-45355-5_26
- [12] Lipinskiy L.V., Kushnareva T.V. Research of self-configuring models and procedures of genetic programming for formation of decision trees in problems of the intelligent data analysis. *Vestnik SibGAU im. akademika M.F. Reshetneva* [Vestnik SibSAU. Aerospace Technologies and Control Systems], 2016, vol. 17, no. 3, pp. 579–586 (in Russ.). EDN: WVPTZH
- [13] Sopov A., Karaseva T. The comparison of different PDP-type self-adaptive schemes for the cooperation of GA, DE, and PSO algorithms. *ITM Web Conf.*, 2024, vol. 59, art. 04013. DOI: <https://doi.org/10.1051/itmconf/20245904013>

- [14] Tanabe R., Fukunaga A. Success-history based parameter adaptation for differential evolution. *IEEE CEC*, 2013, pp. 71–78.
DOI: <https://doi.org/10.1109/CEC.2013.6557555>
- [15] Rivera-Lopez R., Mezura-Montes E., Canul-Reich J., et al. An experimental comparison of self-adaptive differential evolution algorithms to induce oblique decision trees. *Math. Comput. Appl.*, 2024, vol. 29, no. 6, p. 103.
DOI: <https://doi.org/10.3390/mca29060103>
- [16] Stanovov V., Akhmedova S., Semenkin E. Genetic algorithm with success history based parameter adaptation. *IJCCI 2019*, 2019, pp. 180–187.
DOI: <https://doi.org/10.5220/0008071201800187>
- [17] Yu E.L., Suganthan P.N. Ensemble of niching algorithms. *Inform. Sc.*, 2010, vol. 180, no. 15, pp. 2815–2833. DOI: <https://doi.org/10.1016/j.ins.2010.04.008>
- [18] Mann H.B., Whitney D.R. On a test of whether one of two random variables is stochastically larger than the other. *Ann. Math. Statist.*, 1947, vol. 18, no. 1, pp. 50–60.
DOI: <https://doi.org/10.1214/aoms/1177730491>
- [19] Zhu X. Sample size calculation for Mann — Whitney U test with five methods. *Int. J. Clin. Trials*, 2021, vol. 8, no. 3, pp. 184–190.
DOI: <https://doi.org/10.18203/2349-3259.ijct20212840>
- [20] Stanovov V., Semenkin E. Success rate-based adaptive differential evolution L-SRTDE for CEC 2024 competition. *IEEE CEC*, 2024, pp. 1–8.
DOI: <https://doi.org/10.1109/CEC60901.2024.10611907>

Sherstnev P.A. — Post-Graduate Student, Department of Software Engineering, Siberian Federal University, Research Engineer, Artificial Intelligence Center, Siberian Federal University (Svobodniy prospekt 79, Krasnoyarsk, 660041 Russian Federation).

Semenkin E.S. — Dr. Sc. (Eng.), Professor, Department of System Analysis and Operations Research, Reshetnev Siberian State University of Science and Technology (Imeni gazety “Krasnoyarskiy rabochiy” prospekt 31, Krasnoyarsk, 660037 Russian Federation).

Please cite this article in English as:

Sherstnev P.A., Semenkin E.S. SelfCSHAGA: self-configuring genetic optimization algorithm with the success-history based adaptation. *Herald of the Bauman Moscow State Technical University, Series Instrument Engineering*, 2025, no. 2 (151), pp. 122–139 (in Russ.). EDN: TSKBOX