УДК 519.6
Doi: 10.31772/2712-8970-2025-26-1-60-70

Для цитирования: Шерстнев П. А., Семенкин Е. С. Самоконфигурируемые алгоритмы генетического программирования с адаптацией на основе истории успеха // Сибирский аэрокосмический журнал. 2025. Т. 26, № 1. С. 60–70. Doi: 10.31772/2712-8970-2025-26-1-60-70.

For citation: Sherstnev P. A., Semenkin E. S. [Self-Configuring Genetic Programming Algorithms with Success History-Based Adaptation]. *Siberian Aerospace Journal.* 2025, Vol. 26, No. 1, P. 60–70. Doi: 10.31772/2712-8970-2025-26-1-60-70.

Самоконфигурируемые алгоритмы генетического программирования с адаптацией на основе истории успеха

П. А. Шерстнев[1*], Е. С. Семенкин[2]

[1]Сибирский федеральный университет
Российская Федерация, 660041, г. Красноярск, просп. Свободный, 79
[2]Сибирский государственный университет науки и технологий имени академика М. Ф. Решетнева
Российская Федерация, 660037, г. Красноярск, просп. им. газ. «Красноярский рабочий», 31
[*]E-mail: sherstpasha99@gmail.com

Аннотация. В данной работе представлен новый метод самонастройки алгоритмов генетического программирования (ГП), который базируется на идеях метода Success History based Parameter Adaptation (SHA), изначально разработанного для алгоритма дифференциальной эволюции (ДЭ). Основная идея метода заключается в динамическом анализе истории успешных решений для адаптации параметров алгоритма в процессе поиска решения. Для реализации этой концепции схема работы классического ГП была модифицирована таким образом, чтобы имитировать схему ДЭ, что позволило интегрировать механизм SHA в ГП. Полученный алгоритм, обозначенный как SHAGP (Success-History based Adaptive Genetic Programming), демонстрирует новые возможности для адаптации параметров, таких как вероятность скрещивания и мутации. В работе также проведён обзор существующих методов самонастройки алгоритмов ГП, что позволило выявить их ключевые преимущества и ограничения и использовать эти знания при разработке SHAGP. Дополнительно предложены новые операторы скрещивания, позволяющие динамически настраивать вероятность скрещивания, учитывать селективное давление на данном этапе, а также реализующие многородительское скрещивание. Такая модификация позволяет более гибко управлять процессом рекомбинации генотипов, улучшая адаптивность алгоритма к решаемой задаче. Для настройки вероятностей применения различных операторов (селекции, скрещивания, мутации) используются методы самоконфигурирования эволюционных алгоритмов, в частности, Self-Configuring Evolutionary Algorithm и Population-Level Dynamic Probabilities Evolutionary Algorithm. В рамках работы было реализовано два варианта алгоритма – SelfCSHAGP и PDPSHAGP. Эффективность предложенных алгоритмов была проверена на наборах задач из Feynman Symbolic Regression Database. Каждый алгоритм запускался многократно на каждой задаче для получения достоверной статистической выборки, а результаты сравнивались с использованием статистического критерия Манна – Уитни. Экспериментальные данные показали, что предложенные алгоритмы достигают более высокого показателя надёжности по сравнению с существующими методами самонастройки ГП, причём метод PDPSHAGP демонстрирует наилучшую эффективность более чем в 90 % случаев. Такой универсальный механизм самонастройки может найти применение в широком наборе областей, таких как автоматизация машинного обучения, обработка больших данных, инженерный дизайн, медицина, а также в космических приложениях, например, при проектировании навигационных систем для космических аппаратов и разработке систем управления летательными аппаратами. В этих сферах критически важны высокая надёжность алгоритмов и их способность находить оптимальные решения в сложных многомерных пространствах.

Ключевые слова: самонастройка, генетическое программирование, адаптация, самоконфигурирование, скрещивание, регрессия.

Self-Configuring Genetic Programming Algorithms with Success History-Based Adaptation

P. A. Sherstnev[1*], E. S. Semenkin[2]

[1]Siberian Federal University
79, Svobodny Av., Krasnoyarsk, 660041, Russian Federation
[2]Reshetnev Siberian State University of Science and Technology
31, Krasnoyarskii rabochii prospekt, Krasnoyarsk, 660037, Russian Federation
[*]E-mail: sherstpasha99@gmail.com

Abstract. In this work, a novel method for self-tuning genetic programming (GP) algorithms is presented, based on the ideas of the Success History based Parameter Adaptation (SHA) method, originally developed for the Differential Evolution (DE) algorithm. The main idea of the method is to perform a dynamic analysis of the history of successful solutions to adapt the algorithm's parameters during the search process. To implement this concept, the operation scheme of classical GP was modified to mimic the DE scheme, allowing the integration of the success history mechanism into GP. The resulting algorithm, denoted as SHAGP (Success-History based Adaptive Genetic Programming), demonstrates new capabilities for parameter adaptation, such as the adjustment of crossover and mutation probabilities. The work also includes a detailed review of existing self-tuning methods for GP algorithms, which allowed for the identification of their key advantages and limitations and the application of this knowledge in the development of SHAGP. Additionally, new crossover operators are proposed that enable dynamic adjustment of the crossover probability, account for the selective pressure at the current stage, and implement a multi-parent approach. This modification allows for more flexible control over the process of genotype recombination, thereby enhancing the algorithm's adaptability to the problem at hand. To adjust the probabilities of applying various operators (selection, crossover, mutation), self-configuring evolutionary algorithm methods are employed, in particular, the Self-Configuring Evolutionary Algorithm and the Population-Level Dynamic Probabilities Evolutionary Algorithm. Within the framework of this work, two variants of the algorithm were implemented – SelfCSHAGP and PDPSHAGP. The efficiency of the proposed algorithms was tested on problem sets from the Feynman Symbolic Regression Database. Each algorithm was run multiple times on each problem to obtain a reliable statistical sample, and the results were compared using the Mann–Whitney statistical test. The experimental data showed that the proposed algorithms achieve a higher reliability metric compared to existing GP self-tuning methods, with the PDPSHAGP method demonstrating the best efficiency in more than 90 % of the cases. Such a universal self-tuning mechanism can find applications in a wide range of fields, such as automated machine learning, big data processing, engineering design, and medicine, as well as in space applications – for example, in the design of navigation systems for spacecraft and the development of control systems for aerial vehicles. In these areas, the high reliability of algorithms and their ability to find optimal solutions in complex multidimensional spaces are critically important.

Keywords: self-tuning, genetic programming, adaptation, self-configuration, crossover, regression.

Введение

Область исследований, связанная с самонастройкой, является одним из самых актуальных направлений в развитии эволюционных алгоритмов (ЭА). Методы самонастройки ЭА стали неотъемлемой частью алгоритмов, представляемых на IEEE Congress on Evolutionary Computation – одном из ведущих международных форумов по эволюционным вычислениям и вычислительному интеллекту [1]. Это связано с тем, что эффективность оптимизации с использованием ЭА напрямую зависит от выбора конфигурации и числовых параметров, при этом заранее определить их оптимальные значения для конкретной задачи невозможно. Методы самонастройки принято делить на два класса: самоконфигурируемые, которые настраивают конфигурацию алгоритма (варианты операторов селекции, скрещивания и мутации), и адаптивные,

регулирующие числовые параметры алгоритма (вероятности скрещивания и мутации, размер популяции). По мере усложнения задач оптимизации возрастает потребность в более гибких и адаптивных ЭА. Особенно это касается генетического программирования (ГП), которое находит применение в таких областях, как автоматизация машинного обучения, обработка больших данных, инженерный дизайн и медицина, где критически важны высокая надёжность алгоритмов и их способность находить оптимальные решения в сложных многомерных пространствах. Аналогичные требования к адаптивности и точности управления наблюдаются и в ряде технических приложений, что также касается некоторых аспектов ракетно-космических исследований.

Алгоритм генетического программирования

Алгоритм ГП – это семейство алгоритмов оптимизации, эволюционирующие программы, представленные в виде древовидных структур, каждый внутренний узел в которых является операцией, а конечный узел – операндом [2; 3]. Благодаря гибкости такого способа кодирования, с помощью ГП могут решаться задачи, где структура решения заранее неизвестна или сложна для аналитического описания. Наиболее часто ГП используется для решения задач символьной регрессии [4; 5], классификации [6–8], формирования моделей машинного обучения и алгоритмов оптимизации [9–11], синтеза программ и оптимизации сложных систем [12; 13]. Этапы работы ГП обычно аналогичны этапам большинства ЭА и включают следующие шаги: инициализацию начальной популяции (полный метод, метод выращивания, комбинированный метод); оценку индивидов (вычисление значений функции пригодности (ФП) каждого индивида популяции); отбор индивидов, которые будут формировать новое поколение с помощью генетического оператора селекции (пропорциональная, ранговая или турнирная селекция); рекомбинация выбранных индивидов для создания потомков путем применения генетического оператора скрещивания (одноточечное, стандартное или равномерное скрещивание); мутация индивидов-потомков путем применения генетического оператора мутации (точечная, выращиванием, обмен или сжатие); замещение предыдущего поколения потомками. Затем происходит переход на этап оценки индивидов и цикл повторяется [3].

Обзор методов самонастройки алгоритма генетического программирования

Для ГП было разработано и исследовано множество различных методов самонастройки. Так, в работе [14] был предложен один из первых методов самоконфигурирования ГП, упоминаемый как Population-Level Dynamic Probabilities (PDP), в котором при создании индивида генетические операторы выбираются случайно из заданного множества вариантов. При этом вероятность выбора оператора динамически корректируется в процессе поиска решения так, что успешные индивиды получают больше шансов быть выбранными в дальнейшем. Успешность оператора определяется как достижение созданным им потомком лучшего значения ФП, чем было у родителя. Одной из проблем данного метода является неопределённость в выборе родителя для сравнения с потомком. Как правило, выбор производится случайным образом. Несмотря на это, PDP успешно используется для самонастройки не только ГП, но и других ЭА [15; 16]. Другой эффективный метод самонастройки был предложен в [17]. Метод называется SelfCEA (Self-Configuring Evolutionary Algorithm) и во многом схож с PDP – в нем также динамически меняются вероятности применения генетических операторов, но на основе среднего значения ФП, достигнутого оператором. Вероятность применения оператора, при использовании которого в среднем получаются индивиды с более высоким значением ФП, увеличивается [5; 18]. В другом методе, предложенном в работе [19], каждому индивиду назначаются свои собственные вероятности применения каждого типа оператора, а затем, на основе обратной связи от качества создаваемых решений, вероятности увеличиваются или уменьшаются на заранее заданное значение. Метод показал значительное увеличение надежности ГП по сравнению с фиксированными вероятностями для геометрически семантического ГП, но его эффективность для стандартного древовидного ГП (Tree-based GP) не доказана. Кроме методов само-

конфигурирования ГП были предложены и методы адаптации числовых параметров ЭА. Так, в [20] описывается алгоритм SAGP, настраивающий вероятности скрещивания и мутации на основе средних значений размеров деревьев в предыдущем и текущем поколении. Это позволяет не допускать разрастания деревьев и получать интерпретируемые зависимости, но может приводить к чрезмерному сокращению сложности создаваемых функций. Авторы статьи [21] предложили алгоритм CF-GP (Adaptive Crossover + Adaptive Function List), в котором сочетаются адаптивное управление вероятностями скрещивания и динамическое удаление неэффективных функций из функционального множества. Однако в работе отсутствует детальный статистический анализ результатов, что затрудняет оценку его эффективности.

Предлагаемый подход

Схема адаптации, основанная на истории успешных применений (SHA), зарекомендовала себя как высокоэффективный метод настройки вероятностей скрещивания и мутации, что подтверждено успешными экспериментальными результатами [22; 23] и её регулярным применением в различных алгоритмах, включая генетические алгоритмы (ГА). Например, в работе [24] применение SHA к ГА позволило создать SHAGA, демонстрирующий более высокую надёжность по сравнению с SelfCGA на задачах вещественной и псевдобулевой оптимизации. Достигнутые результаты дают основания считать, что применение SHA в ГП приведет к аналогичным улучшениям. Для этого потребуется изменить схему работы ГП. На каждом поколении для каждого *i*-го индивида из популяции последовательно применяются генетические операторы. Сначала посредством селекции отбираются родители – поскольку *i*-й индивид уже служит первым родителем, выбирается на одного меньше, чем в стандартной схеме. Затем *i*-й индивид скрещивается с выбранными родителями, после чего к полученному потомку применяется оператор мутации. Это изменение схемы алгоритма введено для интеграции метода SHA, который адаптирует вероятности мутации и скрещивания на основе критерия: если значение ФП потомка выше, чем у *i*-го решения, текущая настройка параметров считается успешной.

Кроме того, требуется изменить работу оператора скрещивания. В стандартной схеме ГП оператор скрещивания определяет, будет ли создан потомок, и если нет, то скрещивание не происходит. В методе SHA для каждого бита с вероятностью *CR* выбирается, передавать ли его от родителя или мутанта, что напоминает оператор равномерного скрещивания, но с динамически изменяемой вероятностью, отличной от фиксированной и равной 0,5. Дополнительно, при модификации оператора скрещивания в ГП необходимо обеспечить возможность селективного давления на данном этапе и выбора более чем двух родителей [18]. Процесс скрещивания организован в два этапа: сначала для каждого гена с вероятностью *CR* определяется, будет ли он унаследован от первого родителя (текущего решения) или других родителей. Если ген выбирается от первого родителя либо родителей всего два, алгоритм переходит к следующему гену. В противном случае на втором этапе происходит выбор среди оставшихся родителей с учетом их значений ФП, что соответствует подходу, описанному в [18].

Предлагаемая модификация оператора скрещивания в SHAGP позволяет реализовать многородительское скрещивание с возможностью регулирования его интенсивности посредством параметра *CR* и учетом селективного давления на этапе скрещивания. Кроме того, допускается использование классических операторов (одноточечного и стандартного), где процедура выполняется без описанных ранее изменений, но инициируется с вероятностью *CR*; если скрещивание не происходит, оператор возвращает первого родителя (текущее решение). Согласно [18], при использовании многородительского скрещивания, оптимальным числом родителей для большинства операторов является 2 и 7, а для турнирного – 3 и 7. Однако в данном алгоритме на этапе скрещивания применяется дополнительное селективное давление посредством оператора селекции на втором этапе, поэтому общее число родителей увеличивается на 1 по сравнению с оригинальной реализацией.

Данная модификация позволяет использовать различные варианты операторов скрещивания: одноточечное, стандартное, равномерное равновероятное с двумя родителями, равномерное равновероятное с тремя родителями, равномерное равновероятное с восемью родителями, равномерное пропорциональное с тремя родителями, равномерное пропорциональное с восемью родителями, равномерное ранговое с тремя родителями, равномерное ранговое с восемью родителями, равномерное турнирное с тремя родителями, равномерное турнирное с восемью родителями. Оператор селекции при этом может быть любым. В данном исследовании используются: пропорциональная, ранговая, турнирная с размером турнира, равным 3, 5 и 7 индивидов. При этом применяются следующие операторы мутации: точечная, выращиванием, обмена, сжатия.

Поскольку алгоритм обладает 160 возможными конфигурациями, возникает проблема определения оптимальной настройки для каждой решаемой задачи. В этом случае целесообразно использовать методы самоконфигурирования ГП, которые динамически настраивают параметры в процессе работы, обеспечивая большую надежность, чем при случайном выборе.

Объединив все описанные модификации (изменение схемы работы ГП, модифицированный оператор скрещивания, адаптацию на основе истории успешных применений и методы самоконфигурирования), получается единый алгоритм – самоконфигурируемый алгоритм ГП с адаптацией на основе истории успешных применений (Self-Configuring SHAGP). Псевдокод предлагаемого алгоритма представлен ниже:

1. Инициализация.
- *1.1. Сгенерировать начальную популяцию бинарных деревьев случайным образом.*
- *1.2. Вычислить значение ФП для каждого индивида.*
- *1.3. Инициализировать историю параметров:*
  - *1.3.1. Массив H_MR (для вероятности мутации) заполнить значениями 0,1.*
  - *1.3.2. Массив H_CR (для вероятности скрещивания) заполнить значениями 0,9.*
  - *1.3.3. Установить индекс истории k = 0.*
- *1.4. Инициализировать вероятности применения операторов для каждого типа:*
  - *1.4.1. P_sel (операторы селекции) – равновероятно по всем вариантам.*
  - *1.4.2. P_cross (операторы скрещивания) – равновероятно по всем вариантам.*
  - *1.4.3. P_mut (операторы мутации) – равновероятно по всем вариантам.*

2. Основной цикл (для каждого поколения):
- *2.1. Для каждого индивида i:*
  - *2.1.1. Случайно выбрать индекс r из диапазона [0, H_size].*
  - *2.1.2. Задать MR_i, используя распределение Коши с центром H_MR[r] и масштабом 0,1.*
  - *2.1.3. Задать CR_i, используя нормальное распределение с центром H_CR[r] и со стандартным отклонением 0,1.*
  - *2.1.4. Выбрать вариант оператора селекции с помощью распределения вероятностей P_sel.*
  - *2.1.5. Выбрать вариант оператора скрещивания с помощью распределения вероятностей P_cross.*
  - *2.1.6. Выбрать вариант оператора мутации с помощью распределения вероятностей P_mut.*
  - *2.1.7. Применить выбранный оператор селекции для отбора родителей.*
  - *2.1.8. Применить выбранный оператор скрещивания к i-му индивиду (первому родителю) и другим родителям, формируя потомка с вероятностью CR_i.*
  - *2.1.9. Применить выбранный оператор мутации к полученному потомку с вероятностью MR_i.*
  - *2.1.10. Вычислить значение ФП потомка.*
- *2.2. Замещение:*
  - *2.2.1. Для каждого индивида i: если значение ФП потомка лучше, чем i-го индивида, заменить i-го индивида потомком.*

2.3. Обновление истории параметров:
2.3.1. Для всех индивидов, у которых произошла замена, собрать использованные значения MR и CR, а также величины улучшения значений ФП.
2.3.2. Обновить H_MR[k] и H_CR[k] с использованием взвешенного среднего успешных значений.
2.3.3. Положить k = k+1 или 0, если k > H_size.
2.4. Обновление вероятностей применения операторов:
2.4.1. Обновить значения вероятностей применения генетических операторов P_sel, P_cross и P_mut, используя метод самоконфигурирования.
2.5. Обновить глобально лучшего индивида.
2.6. Если критерий остановки не выполнен, то перейти к 2,1.
3. Завершение:
3.1. Вернуть лучшего найденного индивида и статистику работы алгоритма.

Рассмотрим ход работы Self-Configuring SHAGP. Инициализация. Алгоритм стартует с генерации случайной популяции бинарных деревьев и вычисления их значений ФП. Начальные параметры фиксируются: массив H_MR заполняется значением 0,1, массив H_CR – значением 0,9, а вероятности применения операторов (P_sel, P_cross, P_mut) задаются равными, как в оригинальной реализации методов SelfCGP и PDPGP. Формирование нового поколения. Перед созданием потомка для каждого индивида случайно выбирается индекс r из истории параметров. На его основе генерируются значения MR и CR: MR определяется с помощью распределения Коши с центром H_MR[r] (0,1) и масштабом 0,1, CR – посредством нормального распределения с центром H_CR[r] (0,9) и стандартным отклонением 0,1. Операторы селекции, скрещивания и мутации выбираются на основе значений вероятностей их применения, и с их помощью формируется новый потомок. Адаптация параметров. При замещении индивидов успешные значения MR и CR сохраняются для последующей адаптации. Обновление параметров производится по взвешенному среднему успешных значений. Самоконфигурирование операторов. После формирования нового поколения обновляются значения вероятностей применения генетических операторов с использованием выбранного метода самоконфигурирования.

Исследование эффективности самоконфигурируемых алгоритмов генетического программирования

Для апробации предложенного метода использовался набор Feynman Symbolic Regression Database [25], содержащий 120 уравнений различной сложности с количеством неизвестных от 1 до 9. Эти уравнения охватывают широкий спектр физических явлений, включая механические, электромагнитные, квантовые и термодинамические процессы. Каждый из тестируемых самонастраиваемых алгоритмов обладал одинаковым набором типов генетических операторов и функциональным множеством. Всем алгоритмам было задано одинаковое количество поколений (1000), в течение которых они работали, и размер популяции (100).

В исследовании участвовали следующие алгоритмы: SelfCGP – версия ГП на основе SelfCEA с расширенным набором операторов, включающих селективное давление; PDPGP – алгоритм, использующий механизм PDP для настройки операторов с селективным давлением; PDPSHAGP – PDP-модификация алгоритма SHAGP, реализующая динамическую адаптацию вероятностей скрещивания и мутации; SelfCSHAGP – версия SHAGP, основанная на SelfCEA.

Для каждого из 120 уравнений использовалась выборка из 1000 точек, распределённых случайным образом в пространстве. Подробности формирования выборки описаны в [25]. После этого данные были разделены на обучающую (750 точек) и тестовую (250 точек) выборки. Чтобы учесть стохастическую природу эволюционных алгоритмов, проводилось по 100 запусков для каждой задачи, при этом при каждом запуске сохранялось наилучшее значение метрики R^2 [26]. Для подтверждения статистической значимости различий результатов алгоритмов применялся статистический критерий Манна – Уитни с уровнем значимости 0,05.

При сравнении результатов решения задач регрессии с использованием большого количества задач возникает проблема интерпретации значения коэффициента детерминации R^2, которое может принимать отрицательные значения и тем самым смещать средние показатели и искажать оценку методов. Часто для решения этой проблемы используют показатель надежности – долю успешно найденных решений, где успех определяется достижением заранее установленного порога ошибки. Однако такой подход может приводить к потере информации, поскольку результат сильно зависит от выбранного порога. Более информативным является вычисление надежности при различных пороговых значениях. На рис. 1 представлен график значений усредненной по 120 уравнениям надежности при различных значениях порога (от 0 до 1 с шагом 0,01) для каждого из тестируемых методов.

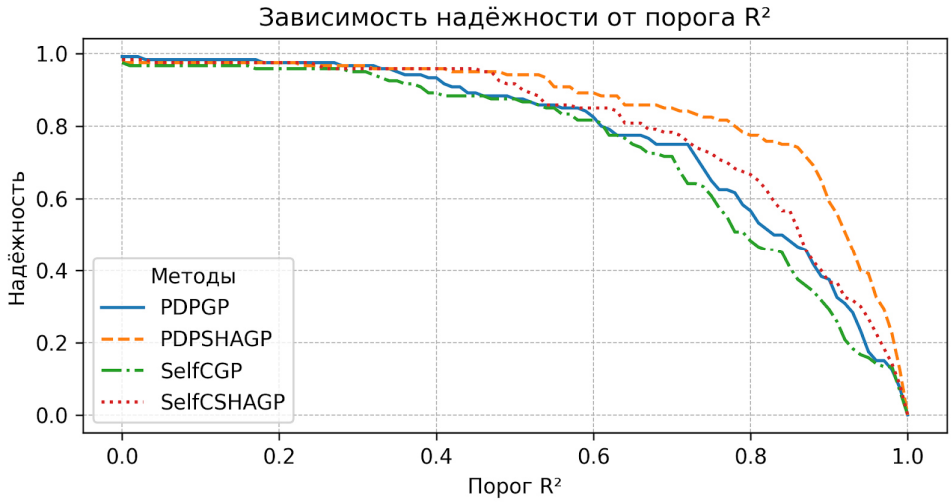

Рис. 1. Зависимость надежности от порогового значения коэффициента детерминации

Fig. 1. Dependence of reliability on the threshold value of the coefficient of determination

График (рис. 1) показывает, как изменяется надёжность различных методов при увеличении порога. PDPSHAGP (оранжевая пунктирная линия) отражает лучшие результаты, оставаясь выше остальных по всему диапазону. SelfCSHAGP (красная точечная линия) тоже выше остальных, но кривая спадает быстрее. PDPGP (синяя сплошная линия) и SelfCGP (зелёная штрихпунктирная линия) заметно уступают, особенно при высоких значениях порога. Усреднённые значения надежности, рассчитанные по всем порогам и задачам, равны: SelfCGP – 0,742; PDPGP – 0,773; SelfCSHAGP – 0,797; PDPSHAGP – 0,848.

На рис. 2 приведены круговые диаграммы, построенные на основе результатов статистического теста, выполненного для сравнения алгоритмов SelfCSHAGP и PDPSHAGP с другими самонастраивающимися алгоритмами. Диаграммы разделены на три категории: «превосходит» (зелёный сектор) – количество функций, где первый алгоритм показал лучшие результаты; «без различий» (серый сектор) – статистически незначимые различия; «уступает» (красный сектор) – случаи, когда второй алгоритм продемонстрировал лучшие показатели.

Из представленных диаграмм видно, что оба алгоритма с использованием SHA (SelfCSHAGP и PDPSHAGP) превосходят конкурентов в большинстве тестовых функций. SelfCSHAGP уверенно опережает SelfCGP (80 против 2) и заметно выигрывает у PDPGP (41 против 3), хотя доля задач, в которых различия оказались статистически незначимыми, достаточно велика (38 и 76 соответственно). PDPSHAGP же демонстрирует ещё более высокие результаты: алгоритм опережает SelfCGP (109 против 4) и PDPGP (92 против 2) с незначительным числом «ничейных» исходов, что указывает на его лидерство среди сравниваемых алгоритмов.

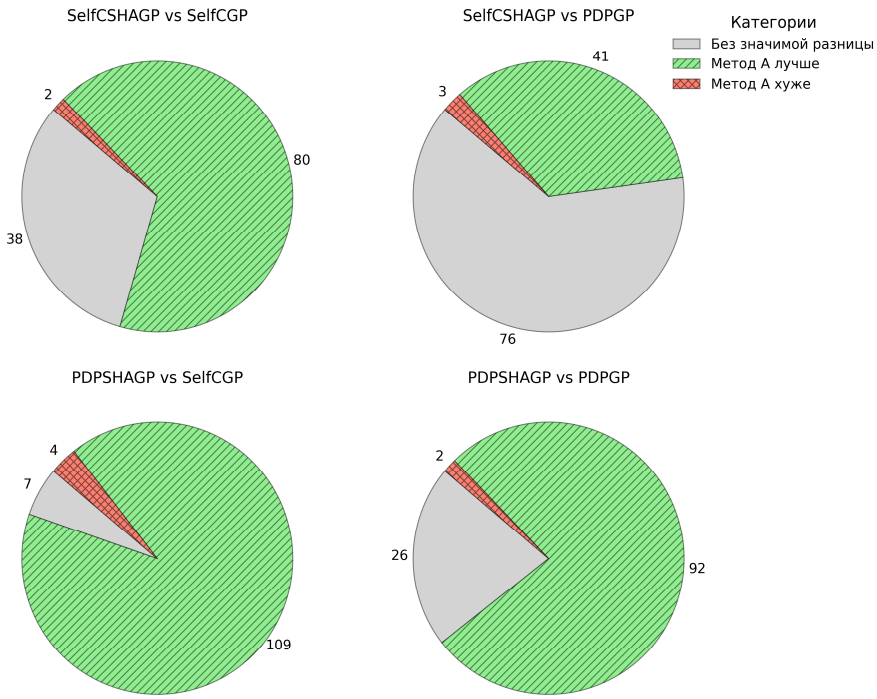

Рис. 2. Результаты сравнения методов самонастройки с использованием статистического теста

Fig. 2. Results of comparing self-tuning methods using a statistical test

Заключение

В данной работе представлен и исследован самоконфигурируемый алгоритм ГП с адаптацией параметров на основе истории успешных применений. Алгоритм позволяет настраивать как параметры вероятностей скрещивания и мутации, так и варианты генетических операторов. Особое внимание уделено модифицированному оператору скрещивания, который отличается возможностью адаптации интенсивности скрещивания за счёт настройки вероятности его применения, применения селективного давления на данном этапе и использования многородительского скрещивания. В рамках исследования алгоритм реализован в двух вариантах, отличающихся методом самоконфигурирования: SelfCSHAGP и PDPSHAGP. Результаты сравнительных экспериментов на задачах регрессии показали, что предложенные алгоритмы превосходят ранние подходы на большинстве тестовых задач, а в оставшихся случаях демонстрируют сопоставимые показатели. Наиболее эффективной оказалась реализация с использованием метода PDPEA для настройки операторов.

Полученные результаты подтверждают перспективность подхода и позволяют наметить дальнейшие направления его развития: анализ эффективности алгоритма при решении задач других классов (например, при формировании моделей машинного обучения) и интеграция дополнительных методов самонастройки, включая настройку размера популяции.

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

Acknowledgment. This research was funded by the State Assignment project № FEFE-2023-0004.

Библиографические ссылки

1. IEEE Congress on Evolutionary Computation [Электронный ресурс]. 2025. URL: https://www.cec2025.org/ (дата обращения: 08.01.2025).

2. Koza J. R. Genetic programming: on the programming of computers by means of natural selection. Sixth printing, 1998, Massachusetts Institute of Technology, 609 p.

3. Kuranga C., Pillay N. A Comparative Study of Genetic Programming Variants // Artificial Intelligence and Soft Computing. ICAISC 2022. Lecture Notes in Computer Science. 2023. Vol. 13588, P. 377–386. DOI: 10.1007/978-3-031-23492-7_32.

4. Genetic Programming-based Feature Selection for Symbolic Regression on Incomplete Data. Evolutionary Computation / B. Al-Helali et al. 2024. P. 1–27.

5. Karaseva T. S., Mitrofanov S. A. Self-configuring genetic programming algorithm for solving symbolic regression problems // IOP Conference Series: Materials Science and Engineering. Krasnoyarsk, 16–18 April 2020. 2020. Vol. 862. P. 52–69. DOI: 10.1088/1757-899X/862/5/052069.

6. Traffic Classification in Software-Defined Networking Using Genetic Programming Tools / S. Margariti, I. Tsoulos, E. Kiousi, E. Stergiou // Future Internet. 2024. Vol. 16. P. 338. DOI: 10.3390/fi16090338.

7. Maurya P., Kushwaha A., Prakash O. Medical Data Classification Using Genetic Programming: A Systematic Literature Review // Expert Systems. 2025. Vol. 42, No. 3. DOI: 10.1111/exsy.70007.

8. A Genetic Programming Approach to Binary Classification Problem / L. Santoso, B. Singh, S. Rajest et al. EAI Endorsed Transactions on Energy Web, 2020. DOI: 10.4108/eai.13-7-2018.165523.

9. A hyper-heuristic approach to automated generation of mutation operators for evolutionary programming / L. Hong, J. Drake, J. Woodward, E. Özcan // Applied Soft Computing. 2018. Vol. 62. DOI: 10.1016/j.asoc.2017.10.002.

10. Scott E. 'Siggy, Bassett J. Learning Genetic Representations for Classes of Real-Valued Optimization Problems. 2015. DOI: 10.1145/2739482.2768460.

11. Hyper-heuristic approach: automatically designing adaptive mutation operators for evolutionary programming / L. Hong, J. R. Woodward, E. Özcan et al. // Complex Intell. Syst. 2021. Vol. 7. P. 3135–3163. DOI: 10.1007/s40747-021-00507-6.

12. Trajectory optimization method for spacecraft orbit transfer with finite thrust. Xinan Jiaotong Daxue Xuebao / C. Wang, Y. Qu, Z. Lu et al. // Journal of Southwest Jiaotong University. 2013. Vol. 48. P. 390–394. DOI: 10.3969/j.issn.0258-2724.2013.02.030.

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

14. Niehaus J., Banzhaf W. Adaption of Operator Probabilities in Genetic Programming // Genetic Programming. EuroGP 2001. Lecture Notes in Computer Science. 2001. Vol. 2038. P. 325–336. DOI: 10.1007/3-540-45355-5_26.

15. 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. 2016. Т. 17, № 3. С. 579–586.

16. Митрофанов С. А., Семенкин Е. С. Дифференциальная эволюция в алгоритме обучения деревьев принятия решений // Сибирский журнал науки и технологий. 2019. Т. 20, № 3. С. 312–319. DOI: 10.31772/2587-6066-2019-20-3-312-319.

17. Semenkin E. S., Semenkina M. E. Self-configuring Genetic Algorithm with Modified Uniform Crossover Operator // LNCS. 2012. Vol. 7331. P. 414–421.

18. Semenkin E., Semenkina M. Self-configuring genetic programming algorithm with modified uniform crossover // Evolutionary Computation (CEC), 2012 IEEE Congress on Evolutionary Computation, June 2012. P. 1–6. DOI: 10.1109/CEC.2012.6256587.

19. Self-tuning geometric semantic Genetic Programming / M. Castelli, L. Manzoni, L. Vanneschi et al. // Genetic Programming and Evolvable Machines. 2016. Vol. 17, No. 1. DOI: 10.1007/s10710-015-9251-7.

20. Oh S., Suh W.-H., Ahn C.-W. Self-Adaptive Genetic Programming for Manufacturing Big Data Analysis // Symmetry. 2021. Vol. 13, No. 4. P. 709. DOI: 10.3390/sym13040709.

21. Al-Madi N., Ludwig S. Adaptive Genetic Programming applied to Classification in Data Mining. Fourth World Congress on Nature and Biologically Inspired Computing (IEEE NaBIC'12), Mexico City, Mexico, November 2012. DOI: 10.1109/NaBIC.2012.6402243.

22. Tanabe R., Fukunaga A. Success-history based parameter adaptation for Differential Evolution // 2013 IEEE Congress on Evolutionary Computation, Cancun, Mexico, 2013, P. 71–78. DOI: 10.1109/CEC.2013.6557555.

23. Renkavieski C., Parpinelli R. L-SHADE with Alternative Population Size Reduction for Unconstrained Continuous Optimization. Computer on the Beach, September 2020, P. 351–358. DOI: 10.14210/cotb.v11n1.p351-358.

24. Stanovov V., Akhmedova S., Semenkin E. Genetic algorithm with success history based parameter adaptation // IJCCI 2019 – Proceedings of the 11th International Joint Conference on Computational Intelligence: 11, Vienna, 17–19 September 2019. Vienna : 2019. P. 180–187. DOI: 10.5220/0008071201800187.

25. Marian S., Tegmark M. E. AI Feynman: A physics-inspired method for symbolic regression // Science Advances. 2020, Vol. 6, No. 16. DOI: 10.1126/sciadv.aay2631.

26. Rights M. D., Sterba S. K. A framework for effect size measures in multilevel models: A review and recommendations // Psychological Methods. 2019. Vol. 24, No. 3. P. 289–315.

References

1. IEEE Congress on Evolutionary Computation (CEC) [Electronic resource]. 2025. Available at: https://www.cec2025.org/ (accessed: 08.01.2025).

2. Koza J. R. Genetic programming: on the programming of computers by means of natural selection. Sixth printing, 1998, Massachusetts Institute of Technology, 609 p.

3. Kuranga C., Pillay N. A Comparative Study of Genetic Programming Variants. *Artificial Intelligence and Soft Computing*. ICAISC 2022. Lecture Notes in Computer Science. 2023. Vol. 13588, P. 377–386. DOI: 10.1007/978-3-031-23492-7_32.

4. Al-Helali B. et al. Genetic Programming-based Feature Selection for Symbolic Regression on Incomplete Data. Evolutionary Computation, 2024, P. 1–27.

5. Karaseva T. S., Mitrofanov S. A. Self-configuring genetic programming algorithm for solving symbolic regression problems. I*OP Conference Series: Materials Science and Engineering*. Krasnoyarsk, 16–18 April 2020. 2020, Vol. 862, P. 52069. DOI: 10.1088/1757-899X/862/5/052069.

6. Margariti S., Tsoulos I., Kiousi E., Stergiou E. Traffic Classification in Software-Defined Networking Using Genetic Programming Tools. *Future Internet.* 2024, Vol. 16, P. 338. DOI: 10.3390/fi16090338.

7. Maurya P., Kushwaha A., Prakash O. Medical Data Classification Using Genetic Programming: A Systematic Literature Review. *Expert Systems*. 2025, Vol. 42, No. 3. DOI: 10.1111/exsy.70007.

8. Santoso L., Singh B., Rajest S., Rajan R., Kadhim K. A Genetic Programming Approach to Binary Classification Problem. EAI Endorsed Transactions on Energy Web, 2020. DOI: 10.4108/eai.13-7-2018.165523.

9. Hong L., Drake J., Woodward J., Özcan E. A hyper-heuristic approach to automated generation of mutation operators for evolutionary programming. Applied Soft Computing. 2018, Vol. 62. DOI: 10.1016/j.asoc.2017.10.002.

10. Scott E. 'Siggy, Bassett J. Learning Genetic Representations for Classes of Real-Valued Optimization Problems. 2015. DOI: 10.1145/2739482.2768460.

11. Hong L., Woodward J. R., Özcan E. et al. Hyper-heuristic approach: automatically designing adaptive mutation operators for evolutionary programming. *Complex Intell. Syst.* 2021, Vol. 7, P. 3135–3163. DOI: 10.1007/s40747-021-00507-6.

12. Wang C., Qu Y., Lu Z., An H., Xia H., Ma G. Trajectory optimization method for spacecraft orbit transfer with finite thrust. Xinan Jiaotong Daxue Xuebao. *Journal of Southwest Jiaotong University*. 2013, Vol. 48, P. 390–394. DOI: 10.3969/j.issn.0258-2724.2013.02.030.

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

14. Niehaus J., Banzhaf W. Adaption of Operator Probabilities in Genetic Programming. *Genetic Programming*. EuroGP 2001. Lecture Notes in Computer Science. 2001, Vol. 2038, P. 325–336. DOI: 10.1007/3-540-45355-5_26.

15. Lipinskiy L. V., Kushnareva T. V. [A study of models and procedures of self-configuring genetic programming for forming decision trees in data mining tasks]. *Vestnik SibSAU.* 2016, Vol. 17, No. 3, P. 579–586 (In Russ.).

16. Mitrofanov S. A., Semenkin E. S. [Differential evolution in the decision tree learning algorithm]. *Siberian Journal of Science and Technology*. 2019, Vol. 20, No. 3, P. 312–319. DOI: 10.31772/2587-6066-2019-20-3-312-319 (In Russ.).

17. Semenkin E. S., Semenkina M. E. Self-configuring Genetic Algorithm with Modified Uniform Crossover Operator. *LNCS*. 2012, Vol. 7331, P. 414–421.

18. Semenkin E., Semenkina M. Self-configuring genetic programming algorithm with modified uniform crossover. *Evolutionary Computation (CEC), 2012 IEEE Congress on Evolutionary Computation*, June 2012, P. 1–6. DOI: 10.1109/CEC.2012.6256587.

19. Castelli M., Manzoni L., Vanneschi L., Popovič A. et al. Self-tuning geometric semantic Genetic Programming. *Genetic Programming and Evolvable Machines*. 2016, Vol. 17, No. 1. DOI: 10.1007/s10710-015-9251-7.

20. Oh S., Suh W.-H., Ahn C.-W. Self-Adaptive Genetic Programming for Manufacturing Big Data Analysis. *Symmetry*. 2021, Vol. 13, No. 4, P. 709. Available at: DOI: 10.3390/sym13040709.

21. Al-Madi N., Ludwig S. Adaptive Genetic Programming applied to Classification in Data Mining. *Fourth World Congress on Nature and Biologically Inspired Computing (IEEE NaBIC'12)*. Mexico City, Mexico, November 2012. Available at: https://doi.org/10.1109/NaBIC.2012.6402243.

22. Tanabe R., Fukunaga A. Success-history based parameter adaptation for Differential Evolution. *2013 IEEE Congress on Evolutionary Computation*, Cancun, Mexico, 2013, P. 71–78. DOI: 10.1109/CEC.2013.6557555.

23. Renkavieski C., Parpinelli R. L-SHADE with Alternative Population Size Reduction for Unconstrained Continuous Optimization. *Computer on the Beach*, September 2020, P. 351–358. DOI: 10.14210/cotb.v11n1.p351-358.

24. Stanovov V., Akhmedova S., Semenkin E. Genetic algorithm with success history based parameter adaptation. *IJCCI 2019 – Proceedings of the 11th International Joint Conference on Computational Intelligence*: 11, Vienna, 17–19 September 2019. Vienna, 2019, P. 180–187. DOI: 10.5220/0008071201800187.

25. Marian S., Tegmark M. E. AI Feynman: A physics-inspired method for symbolic regression. *Science Advances*. 2020, Vol. 6, No. 16, P. 2631. DOI: 10.1126/sciadv.aay2631.

26. Rights M. D., Sterba S. K. A framework for effect size measures in multilevel models: A review and recommendations. *Psychological Methods*. 2019, Vol. 24, No. 3, P. 289–315.

Шерстнев Павел Александрович – аспирант кафедры программной инженерии, инженер-исследователь Центра искусственного интеллекта; Сибирский федеральный университет. E-mail: sherstpasha99@gmail.com.

Семенкин Евгений Станиславович – доктор технических наук, профессор; кафедра системного анализа и исследования операций, Сибирский государственный университет науки и технологий имени академика М. Ф. Решетнева. E-mail: eugenesemenkin@yandex.ru. https://orcid.org/0000-0002-3776-5707

Sherstnev Pavel Aleksandrovich – graduate student, Research Engineer; Artificial Intelligence Center, Siberian Federal University. E-mail: sherstpasha99@gmail.com.

Semenkin Evgeniy Stanislavovich – Dr. Sc., Professor, Department of Systems Analysis and Operations Research; Siberian State University of Science and Technology. E-mail: eugenesemenkin@yandex.ru. https://orcid.org/0000-0002-3776-5707

Статья поступила в редакцию 24.02.2025; принята к публикации 04.03.2025; опубликована 11.04.2025
The article was submitted 24.02.2025; accepted for publication 04.03.2025; published 11.04.2025