Инженерный вестник Дона, №8 (2025)
ivdon.ru/ru/magazine/archive/n8y2025/10315

Обучение с подкреплением в адаптивном управлении параметрами генетического алгоритма

К.С. Привалов

Финансовый университет при правительстве РФ

Аннотация: В статье представлен новый подход адаптивного управления параметрами генетического алгоритма, основанный на методах обучения с подкреплением. Использование алгоритма Q-обучения позволяет динамически изменять вероятности мутации и кроссовера в зависимости от текущего состояния популяции и прогресса эволюционного процесса. Экспериментально показано, что данный подход обеспечивает более эффективное решение задач оптимизации по сравнению с классическим генетическим алгоритмом и предыдущими подходами с использованием искусственных нейронных сетей. Проведены тестирования на функциях Растригина и Шаффера, подтверждающие преимущества нового метода в задачах с большим числом локальных экстремумов и высокой размерностью.
Ключевые слова: генетический алгоритм, обучение с подкреплением, адаптивное управление, Q-обучение, глобальная оптимизация, функция Растригина, функция Шаффера.

Введение

Проблема эффективного поиска глобального экстремума сложных многомерных функций занимает ключевое место в современной вычислительной математике и прикладной оптимизации. В последние десятилетия особое внимание уделяется стохастическим эволюционным методам, среди которых генетические алгоритмы (ГА) получили наибольшее распространение благодаря универсальности [1, 2] и способности решать задачи с многочисленными локальными минимумами [3]. В то же время, согласно теореме о «бесплатном обеде», не существует универсального оптимизационного метода, который был бы одинаково эффективен для всех классов задач [4]. Несмотря на популярность ГА, классическая версия метода часто сталкивается с эффектом преждевременной сходимости, когда популяция особей быстро теряет разнообразие и сосредотачивается вокруг неглобального экстремума. Такая проблема особенно ярко проявляется при увеличении размерности задачи или в присутствии так называемых

«обманных» функций с множеством ложных минимумов, к числу которых относится функция Растригина [5].

Для повышения эффективности поиска и предотвращения преждевременной сходимости в литературе рассматриваются различные способы адаптации управляющих параметров ГА. Наиболее изученными являются методы с автоматической настройкой вероятностей мутации и кроссовера, а также введение специальных операторов для поддержания разнообразия популяции [6]. В недавних работах, включая предыдущую публикацию автора [7], а также исследования по применению нейронных сетей [8, 9] и динамическому управлению параметрами ГА [10], была предложена интеграция искусственных нейронных сетей (ИНС) в структуру эволюционного поиска. Это позволяет в реальном времени корректировать параметры эволюционного процесса, опираясь на статистику состояния текущей популяции. Однако, нейросетевые подходы требуют подбора архитектуры, предобучения на специально сформированных данных, а также могут быть чувствительны к выбору гиперпараметров.

Вместе с тем, последние успехи методов обучения с подкреплением (Reinforcement Learning, RL) в задачах управления и оптимизации открывают новые перспективы для адаптивного регулирования эволюционного поиска [9, 10]. В отличие от классических и даже нейросетевых схем, агент RL способен не только реагировать на текущие характеристики поиска, но и обучаться долгосрочной стратегии взаимодействия с популяцией особей на основе награды за улучшение результата. Такое обучение без учителя формирует у агента «ощущение» состояния процесса оптимизации и позволяет более гибко реагировать на изменение ситуации в пространстве решений.

Целью настоящего исследования является разработка и экспериментальная проверка гибридного алгоритма, в котором обучение с

подкреплением интегрировано в процесс управления параметрами генетического алгоритма. В работе подробно рассматриваются теоретические основы и архитектура предлагаемого метода, приводятся результаты сравнения с классическим ГА и ранее реализованным подходом с использованием нейронной сети, анализируется динамика эволюционного процесса на тестовых функциях Растригина и Шаффера. Полученные результаты позволяют обосновать преимущества применения методов RL для повышения адаптивности и эффективности стохастической оптимизации.

Теоретические основы используемых методов

Эволюционные методы оптимизации, и в частности генетический алгоритм, основаны на идее естественного отбора и наследования признаков. Популяция решений на каждом этапе подвергается отбору, рекомбинации и случайным мутациям, что позволяет осуществлять стохастический поиск в сложном пространстве параметров [1, 2]. Оператор селекции выделяет наиболее приспособленных особей для продолжения рода; кроссовер (рекомбинация) создает новых особей за счёт обмена частями генома между родителями, а мутация обеспечивает поступление новых генетических вариаций в популяцию [3].

Функция приспособленности:

Пусть $x = (x_1, x_2, \dots, x_n)$ — вектор параметров решения, тогда целевая функция для задачи оптимизации запишется как:

$$f(x) = 10n + \sum_{i=1}^{n} \left(x_i^2 - 10cos(2\pi x_i)\right),$$

Это — функция Растригина [5], классический тест для оценки качества стохастических алгоритмов. Также рассмотрим и функцию Шаффера:

$$f(x, y) = 0.5 + \frac{sin^2(x^2 - y^2) - 0.5}{(1 + 0.001(x^2 + y^2))^2}$$

Операторы алгоритма:

Селекция: обычно используется турнирная селекция или рулетка, где вероятность выбрать особь i пропорциональна её приспособленности.

Кроссовер: двухточечный оператор:

$$child_1, child_2 = Crossover(parent_1, parent_2, p_c),$$

где p_c – вероятность применения кроссовера.

Мутация: гауссовская мутация каждого гена:

$$x_j' = x_j + \xi, \xi \sim \mathcal{N}(0, \sigma^2), \text{ с вероятностью } p_m,$$

где p_m - вероятность мутации каждого гена.

Классический ГА использует фиксированные значения p_c и p_m на всём протяжении поиска. Это делает алгоритм подверженным как застыванию на одном решении (слишком малые вероятности мутации), так и избыточному «шуму» (слишком большие значения p_m), что снижает его эффективность.

Обучение с подкреплением: Q-обучение

Обучение с подкреплением (RL) — это класс алгоритмов, где агент взаимодействует с средой, последовательно наблюдая состояния, выбирая действия и получая награду, стремясь максимизировать сумму будущих вознаграждений [9].

Алгоритм Q-обучения формально описывается следующим образом. Пусть s_t — состояние среды на шаге t, a_t — выбранное агентом действие, r_{t+1} — награда, s_{t+1} — следующее состояние. Q-функция $Q(s, a)$ аппроксимирует ценность действия a в состоянии s. Алгоритм обновляет значения Q по формуле:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_{t+1} + \gamma * \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t)),$$

где α – скорость обучения, γ - коэффициент дисконтирования будущей награды [9, 10].

В нашей постановке средой для агента является сам эволюционный процесс: состояние среды описывается статистикой популяции $(f_{avg}, f_{min}, \sigma_f, D)$, а действия – это выбор/изменение значений p_m и p_c для следующего поколения.

Агент получает награду, пропорциональную улучшению среднего (или лучшего) значения функции приспособленности:

$$r_{t+1} = f_{avg}^{(t)} - f_{avg}^{(t+1)},$$

(или аналогично для f_{min}), что стимулирует не только быстрое начальное улучшение, но и поиск новых, потенциально более выгодных стратегий на поздних этапах эволюции.

Такой подход делает возможным обучение эффективной политики управления параметрами генетического алгоритма без явного подбора вручную или необходимости внешнего обучения, как в случае с ИНС.

Описание предлагаемого гибридного подхода

Интеграция методов обучения с подкреплением в классическую схему генетического алгоритма направлена на повышение его адаптивности и эффективности при поиске глобального экстремума в задачах высокой сложности. Принципиальное отличие гибридного подхода заключается в динамическом управлении вероятностями мутации p_m и кроссовера p_c с помощью специально обучаемого RL-агента, способного анализировать состояние популяции и выбирать стратегию настройки, максимально способствующую прогрессу поиска.

В разрабатываемой системе каждое поколение алгоритма оптимизации строится по следующей схеме:

1. Оценка состояния популяции.

Перед началом итерации вычисляются признаки текущего состояния популяции:

- Среднее значение функции приспособленности f_{avg};
- Лучшее найденное значение f_{min};
- Стандартное отклонение приспособленности σ_f;
- Разнообразие популяции D;
- (Дополнительно: размер популяции, количество поколений без улучшения и др.).

2. Выбор действия RL-агентом.

   Состояние популяции подаётся на вход агенту обучения с подкреплением (алгоритм Q-обучения). Агент выбирает действие из дискретного множества — увеличить, уменьшить или оставить неизменными вероятности мутации и кроссовера:

$$a_t \in \{\text{уменьшить } p_m, \text{увеличить } p_m, \text{уменьшить } p_c, \text{увеличить } p_c, \text{ничего не менять}\}$$

3. Применение выбранных параметров к ГА.

   Генетический алгоритм выполняет очередную итерацию (поколение), используя выбранные агентом значения p_m, p_c для операторов мутации и кроссовера.

4. Получение награды и обновление Q-функции.

   По завершении поколения агент получает награду r_{t+1}, равную разности между средним (или минимальным) значением функции приспособленности до и после применения действия:

$$r_{t+1} = f_{avg}^{(t)} - f_{avg}^{(t+1)}$$

   Если прогресс достигнут (приспособленность улучшилась), награда положительна; если стагнация или ухудшение — нулевая или

отрицательная. Q-таблица обновляется по стандартной формуле (см. предыдущий раздел).

5. Повторение цикла для следующего поколения.

Таким образом, агент постепенно обучается находить наиболее выгодные стратегии изменения параметров, исходя из динамики поиска.

Особенности реализации

В отличие от классического подхода (где параметры ГА фиксированы) и предыдущего нейросетевого (где параметры вычисляются регрессией по состоянию популяции), здесь RL-агент осваивает долгосрочную стратегию, позволяющую избежать не только локальных ловушек, но и избыточной «размытой» эволюции. По сути, агент выступает в роли «мета-алгоритма», способного подстраиваться под изменяющийся характер пространства решений.

Для программной реализации используется Python с применением библиотек DEAP (для эволюционного алгоритма) и PyTorch (если агент реализуется как небольшая нейросеть Q-value или Q-таблица для малых пространств состояний и действий). В качестве состояния выбирается конкатенация нормированных признаков популяции; в качестве действий — дискретное изменение p_m, p_c на заранее выбранный шаг (например, ± 0.05), ограниченных допустимым диапазоном $[0.05, 0.8]$.

Пространство состояний кодируется как биннинг признаков (например, дискретизация среднего значения функции и разнообразия популяции). Такая дискретизация обеспечивает небольшие размеры Q-таблицы для задач средней сложности, а для задач высокой размерности возможно применение аппроксиматоров (MLP, RBF).

В предыдущей работе [7] вероятности мутации и кроссовера вычислялись с помощью небольшой полносвязной нейросети, обучаемой на разнице средней приспособленности между поколениями. Признаковое описание и архитектура были схожи: сеть брала на вход вектор характеристик популяции и выдавала два выхода, соответствующих новым параметрам. Теперь же RL-агент не просто регрессирует параметры, а строит последовательную стратегию: накапливает опыт успешных и неудачных изменений, что даёт возможность реализовать сложные сценарии (например, кратковременное увеличение мутации после застоя или удержание низких значений для финального «доисследования»).

Важным достоинством предложенного метода является его способность к самообучению и выработке политики адаптации, оптимальной именно для конкретной задачи и траектории эволюционного поиска. Это особенно заметно в задачах, где локальные минимумы могут временно «перехватывать» всю популяцию — агент RL сам находит моменты для резкого усиления мутаций или, напротив, консервативной селекции.

Подобная стратегия выгодно отличается как от жёстких (фиксированных) настроек классических ГА, так и от регрессионных моделей на основе нейросетей, где нет накопления долгосрочного опыта успешных (или неудачных) изменений параметров [9, 10].

Программная реализация

Ключевые параметры эксперимента:

- Размер популяции $N = 30$;
- Число поколений: 50;

- Размерность задачи $n = 5$ (для Растригина); $n = 2$ (для Шаффера);
- Начальная вероятность $p_m = 0.2, p_c = 0.5$;
- Начальная вероятность кроссовера 0.5;
- Параметры RL: $\alpha = 0.1, \gamma = 0.9, \varepsilon$ - жадная политика выбора.

Сравниваемые алгоритмы:

- Классический ГА;
- ГА с нейросетевым адаптером (ИНС);
- ГА с агентом обучения с подкреплением.

Анализ результатов

Экспериментальные результаты, полученные на функциях Растригина и Шаффера, позволяют провести сравнительный анализ эффективности трёх рассмотренных подходов: классического генетического алгоритма, ГА с нейросетевым регулятором и ГА с управлением параметрами на основе обучения с подкреплением (RL).

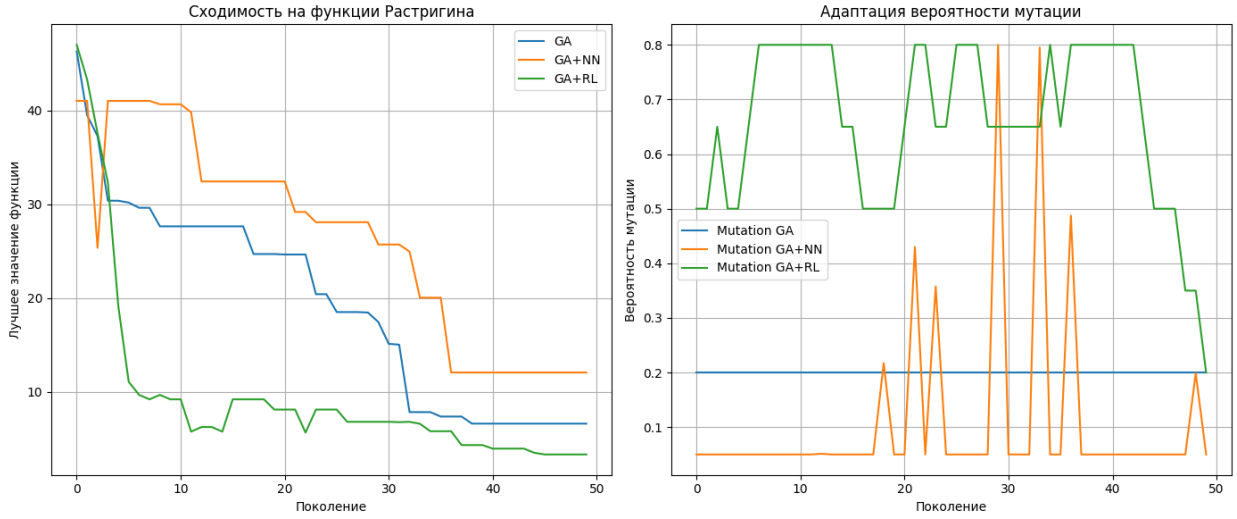

Рис. 1. Динамика лучшего значения функции Растригина и вероятности мутации при различных подходах

Для функции Растригина (рис. 1) гибридный алгоритм с агентом RL показал наилучшие результаты как по скорости сходимости, так и по качеству найденного решения. Это объясняется способностью RL-агента динамически регулировать параметры эволюционного процесса, своевременно увеличивая вероятность мутации при стагнации и снижая её по мере приближения к экстремуму. Благодаря этому гибридный подход не только избегал преждевременной сходимости, но и существенно превосходил как классический ГА, так и ГА+ИНС по стабильности траектории поиска и итоговым значениям целевой функции.

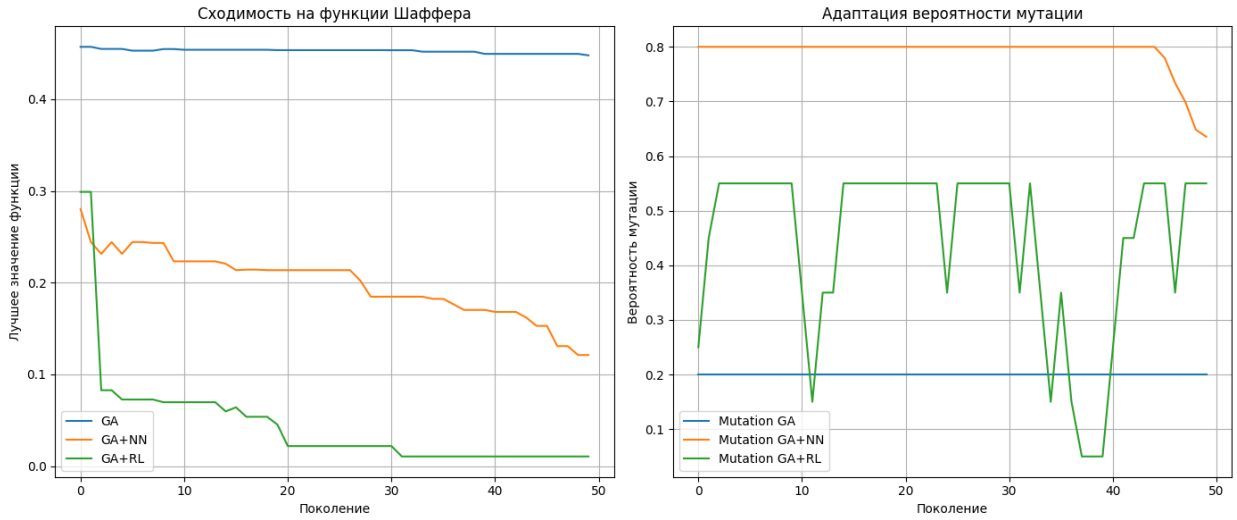

Рис. 2. Сходимость на функции Шаффера и динамика вероятности мутации для различных подходов

Аналогичная тенденция наблюдается и на функции Шаффера (рис. 2). После дополнительной настройки параметров агента RL стало заметно, что гибридный подход способен обеспечить не только быстрое достижение глобального минимума, но и устойчивое удержание решения без потери разнообразия популяции. Классический ГА в ряде случаев останавливается на локальном экстремуме, а нейросетевой регулятор склонен к чрезмерной адаптации, что иногда приводит к деградации результата.

Отдельно стоит отметить характерную для RL-подхода динамику вероятности мутации (рис. 1 и 2, справа): вероятность активно изменяется, что способствует поиску новых решений при возникновении застоя и позволяет избежать «залипания» популяции в локальных минимумах. В то же время RL-агент учится удерживать вероятность мутации в оптимальном диапазоне для этапа уточнения решения.

Результаты экспериментов убедительно показывают универсальность и перспективность гибридных методов с элементами обучения с подкреплением для оптимизации сложных многомодальных функций. Такой подход хорошо адаптируется к структуре задачи, позволяет избежать как преждевременной сходимости, так и чрезмерного шума, а также демонстрирует устойчивые преимущества как в задачах средней размерности (Шаффер), так и в многомерных задачах (Растригин).

Заключение

В данной работе предложен и реализован гибридный эволюционный алгоритм, в котором управление ключевыми параметрами — вероятностями мутации и кроссовера — осуществляется агентом обучения с подкреплением. Проведённое сравнение с классическим ГА и с ГА на базе искусственной нейронной сети показало, что интеграция RL-агента существенно повышает эффективность и устойчивость поиска в задачах с множеством локальных экстремумов. Полученные результаты свидетельствуют о перспективности дальнейшего развития этого направления, в частности, для оптимизации высокоразмерных и сложноструктурированных задач.

В перспективе целесообразно исследовать более сложные архитектуры агентов, а также расширить тестирование на реальные инженерные и прикладные задачи. Дополнительным направлением может стать интеграция

методов глубокого обучения для построения аппроксиматоров Q-функции в задачах с большим числом признаков состояния.

Литература

1. Голдберг Д. Э. Генетические алгоритмы в поиске, оптимизации и машинном обучении. М.: Вильямс, 2001. 416 с.
2. Holland J. H. Adaptation in Natural and Artificial Systems. Moscow: Mir, 1988. 304 p.
3. He X. A Hybrid Genetic Algorithm with an Adaptive Local Search Operator // Computational Intelligence. 2019. Vol. 35, no. 2. P. 547–562.
4. Wolpert D. H., Macready W. G. No Free Lunch Theorems for Optimization // IEEE Transactions on Evolutionary Computation. 1997. Vol. 1, no. 1. P. 67–82.
5. Растригин И. Ф. О сходимости экстремального метода при наличии большого числа переменных // Вопросы вычислительной математики и математической физики. 1963. Т. 3, № 6. С. 128–137.
6. Zhang J., Liao S., Lee S. The Use of Neural Networks for Adaptive Optimization Problems in Large Dimensions // Applied Soft Computing. 2020. Vol. 90. P. 106187.
7. Привалов К. С. Гибридные методы оптимизации: адаптивное управление эволюционным процессом с использованием искусственных нейронных сетей // Инженерный вестник Дона, 2025, № 3. URL: ivdon.ru/ru/magazine/archive/n3y2025/9910
8. Fogel D. B. Evolutionary Computation: Toward a New Philosophy of Machine Intelligence. 3rd ed. New York: IEEE Press, 2006. 365 p.
9. McKinley P. A., McBride K. Hybrid Methods for Solving Complex Optimization Problems // International Journal of Computational Science and Engineering. 2017. Vol. 10, no. 3. pp. 210–223.

10. Darnell H., Parham A. Neural Network-Based Dynamic Mutation Control in Genetic Algorithms // Journal of Computational Intelligence and Applications. 2021. Vol. 23, no. 1. pp. 45–58.

References

1. Goldberg D.E. Geneticheskie algoritmy v poiske, optimizatsii i mashinnom obuchenii [Genetic Algorithms in Search, Optimization, and Machine Learning]. Moskva: Williams, 2001. 416 p.
2. Holland J.H. Adaptation in natural and artificial systems. Moscow: Mir, 1988. 304 p.
3. He X. Computational Intelligence. 2019;35(2):547–562.
4. Wolpert D.H., Macready W.G. IEEE Trans Evol Comput. 1997;1(1):67–82.
5. Rastrigin I.F. Voprosy vychislitelnoi matematiki i matematicheskoi fiziki. 1963;3(6):128–137.
6. Zhang J, Liao S, Lee S. Applied Soft Computing. 2020; 90:106187.
7. Privalov K.S. Inzhenernyj vestnik Dona, 2025, № 3. URL: ivdon.ru/ru/magazine/archive/n3y2025/9910
8. Fogel D.B. Evolutionary computation: toward a new philosophy of machine intelligence. 3rd ed. New York: IEEE Press, 2006. 365 p.
9. McKinley P.A., McBride K. International Journal of Computational Science and Engineering. 2017;10(3):210–223.
10. Darnell H, Parham A. Journal of Computational Intelligence and Applications. 2021;23(1):45–58.

Дата поступления: 13.06.2025

Дата публикации: 26.07.2025