УДК 004.021 DOI: 10.26102/2310-6018/2021.34.3.014

Гибридная нейроэволюция как способ обучения нейронных сетей на примере решения задачи поиска пути в лабиринте

В.А. Березина, О.С. Мезенцева, К.Ю. Ганьшин
Северо-Кавказский федеральный университет,
Ставрополь, Российская Федерация

Резюме. В данном исследовании нейронная сеть, обученная гибридной нейроэволюцией, решает задачу поиска пути в лабиринте. Гибридная нейроэволюция сочетает в себе дифференциальную эволюцию с поиском новинок. Поиск новинок – это относительно недавний подход к обучению нейронных сетей, который фокусируется на поиске нового поведения, а не лучшего, с точки зрения целевой функции. Новизна недолговечна в том смысле, что ничто не остается новым бесконечно. Алгоритмы, которые сохраняют лучшие решения, сталкиваются с проблемой, заключающейся в том, что оценки новизны этих архивных решений не будут меняться от поколения к поколению. Данное исследование направлено на решение этой проблемы, предлагая два метода корректировки оценок новизны решений: деструкция новизны и актуализация оценок новизны. Деструкция новизны позволяет уменьшать новизну со временем, тем самым помогая алгоритму поиска развиваться, в то время как актуализация оценок новизны обновляет новизну этих решений в каждом поколении. При тестировании на проблеме навигации в лабиринте было замечено, что деструкция новизны и обновление оценок новизны сходятся быстрее, чем только стандартный поиск по целевой функции и поиск новинок.

Ключевые слова: нейроэволюция, нейронные сети, лабиринт, поиск новинок, дифференциальная эволюция.

Для цитирования: Березина В.А., Мезенцева О.С., Ганьшин К.Ю. Гибридная нейроэволюция как способ обучения нейронных сетей на примере решения задачи поиска пути в лабиринте. *Моделирование, оптимизация и информационные технологии.* 2021;9(3). Доступно по: https://moitvivt.ru/ru/journal/pdf?id=1012 DOI: 10.26102/2310-6018/2021.34.3.014

Hybrid neuroevolution as a way to train neural networks by the example to solve the maze problem

V.A. Berezina, O.S. Mezentseva, K.Y. Ganshin
North-Caucasus Federal University,
Stavropol, Russian Federation

Abstract: In this research, a neural network trained by hybrid neuroevolution solves the maze problem. Hybrid neuroevolution combines differential evolution with the novelty search. The novelty search is a relatively recent approach to training neural networks that focuses on finding new behaviors rather than on the best solutions of fitness function. Novelty is short-lived in the sense that nothing remains new indefinitely. Algorithms that preserve the best solutions face the problem of novelty estimates for these archival solutions not changing from generation to generation. This article aims to address this issue by proposing two methods for adjusting estimates of the solutions novelty: novelty destruction and actualization of novelty rates. The novelty destruction allows novelty to diminish over time, thereby allowing the search algorithm to evolve, while the actualization of novelty rates updates the novelty of these solutions in each generation. When testing by the problem of navigation in the maze, it was noticed that the novelty destruction and the actualization of novelty rates converged faster than just the standard search by objective function and novelty search.

Keywords: neuroevolution, neural networks, maze, novelty search, differential evolution

For citation: Berezina V.A., Mezentseva O.S., Ganshin K.Y. Hybrid neuroevolution as a way to train neural networks by the example to solve the maze problem. *Modeling, Optimization and Information Technology*. 2021;9(3). Available from: https://moitvivt.ru/ru/journal/pdf?id=1012 DOI: 10.26102/2310-6018/2021.34.3.014 (In Russ).

Введение (Introduction)

Идея развития нейронной сети путем отбора решений по новизне их поведения привлекла много внимания с момента ее первого предложения [1]. Причина успеха поиска новинок заключается в том, что игнорирование цели в процессе поиска решений позволяет исследовать более широкий диапазон вариантов поведения. Процесс поиска различного поведения дает эволюционному алгоритму больше возможностей для исследования.

Данное исследование фокусируется на адаптации поиска новинок к эволюционным алгоритмам, которые сохраняют лучшие решения для дальнейшего формирования новых поколений. Типичным примером могут быть эволюционные алгоритмы, использующие элитарность, при которой лучшие решения передаются следующему поколению без изменений. Такой подход оказался весьма эффективен для стандартного поиска [2].

Если решение считается лучшим решением, с точки зрения новизны, то оно, вероятно, будет использоваться для создания новых решений. Эти решения, скорее всего, будут обладать тем же поведением, что и решение, признанное новым ранее. Но, таким образом, решение, которое ранее было признано новым, таковым более не является. Стоит отметить, что эта проблема актуальна только для алгоритмов, которые сохраняют лучшие решения для дальнейшего формирования решений. Например, элитарность, используемая в генетических алгоритмах и генетическом программировании [2], лучшие решения записываются и используются для обновления частиц при оптимизации роя частиц [3], а агенты в дифференциальной эволюции обновляют свое решение только в том случае, если полученное новое решение лучше, чем предыдущее [4].

В данном исследовании предлагаются два метода решения данной проблемы: 1) позволить оценке новизны уменьшаться в каждом поколении; 2) пересчитывать новизну лучших решений в каждом поколении. Первый метод обусловливается самой концепцией понятия новизны. Новизна – это временное явление, поэтому из поколения в поколение она будет уменьшаться. Второй метод более затратный, с точки зрения, вычислений, но будет иметь эффект точного обновления новизны в каждом поколении.

Материалы и методы (Materials and Methods)

Традиционно решение оценивается по его приспособленности, где приспособленность представляет собой соответствие выбранной цели. С этой точки зрения, используя поиск новинок, целью становятся новые, уникальные решения. Поиск новинок был впервые предложен Леманом и Стэнли в 2011 году [1]. С тех пор было много успешных приложений данного метода, включая такие, как управление подводными мягкими роботами [5], дизайн аэродинамического профиля [6], классификацию [7] и компьютерные игры [8].

Новизну можно определить по мере того, насколько поведение отличается от другого поведения. Уравнение 1 показывает, как вычисляется новизна при поиске новинок, где nov – оценка новизны для текущего поведения x, k – количество ближайших соседей, а μ_i – поведение, наблюдаемое i-м ближайшим соседом.

$$nov(x) = \frac{1}{x}\sum_{i=0}^{k} dist(x, \mu_i). \quad (1)$$

Для задачи поиска пути в лабиринте, рассматриваемой в этом исследовании, мерой расстояния между двумя поведениями является евклидово расстояние между конечными положениями агента для каждого поведения, так как это классический выбор для решения задачи прохождения лабиринта.

В данном исследовании будет реализована нормализованная оценка новизны, предложенная Кукку и Гомесом, поскольку является более общим показателем новизны для каждого поколения [9]. Эта нормализованная оценка новизны рассчитывается с использованием уравнения 2.

$$\overline{nov}(x) = \frac{nov(x) - nov_{min}}{nov_{max} - nov_{min}}. \quad (2)$$

Одним из методов обучения нейронной сети является нейроэволюция. Это и является простым определением нейроэволюции, а именно, использование эволюционных алгоритмов для поиска оптимальной конфигурации нейронной сети для аппроксимации некоторой функции [10]. Алгоритмы нейроэволюции имеют широкий спектр приложений, включая игры Atari, распознавание образов и робототехнику.

Одним из видов нейроэволюции является дифференциальная эволюция. Алгоритм дифференциальной эволюции (DE) – это алгоритм эволюционной оптимизации [4]. Дифференциальная эволюция успешно применялась для развития нейронных сетей в процессе решения многих различных задач, в том числе: управление, прогнозирование контакта шин, выработка электроэнергии, прогнозирование нагрузки кондиционирования воздуха, прогнозирование загрузки CPU и прогнозирование энергии ветра. Алгоритм использует принципы эволюционных вычислений для решения глобальных задач оптимизации с действительными значениями.

В данном исследовании будет применен поиск новинок к дифференциальной эволюции. Поиск новинок уже успешно применялся во многих других алгоритмах, таких как оптимизация роя частиц и грамматическая эволюция. Предполагается, что дифференцированная эволюция также может быть улучшена путем сочетания ее с поиском новинок.

Фундаментальная движущая сила поиска новинок состоит в том, что выживают и переходят к следующему поколению новые поведения, а не лучшие (наиболее близкие к цели). Было показано, что это эффективная стратегия [1].

Данное исследование основывается на этом принципе, вводя идею деструкции новизны, при котором оценка новизны снижается по мере смены поколений. Деструкция новизны объясняет тот факт, что само понятие новизны – временное явление. То, что сейчас «новое», не останется новым бесконечно. Многие алгоритмы сохраняют лучшие решения для дальнейшего их использования в поиске решений. Например, генетические алгоритмы используют элитарность, а дифференциальная эволюция использует позицию агента. Для поиска новинок лучшие решения – это те, которые имеют наивысшую оценку новизны.

Поиск новинок обычно применяется к алгоритму NEAT [11]. Однако поиск новинок с уменьшением новизны может не подходить для алгоритма NEAT. Это связано с тем, что деструкция новизны основана на алгоритме, который учитывает лучшие (самые новые) решения. Первоначальная реализация NEAT не использует элитарность, поскольку вся популяция заменяется в следующем поколении. Дифференциальная эволюция же заменяет текущие решения только в том случае, если они улучшаются.

При каждом поколении алгоритма DE каждый агент генерирует новое решение. Если это решение более новое, чем текущее решение агента, текущее решение

заменяется. Однако рассмотрим ситуацию, когда агент обнаруживает совершенно новое (но все же неоптимальное) поведение. Это решение будет иметь очень высокую оценку новизны. Это приведет к тому, что алгоритм будет формировать новые решения на основе этого решения. Однако эта высокая оценка новизны создает проблему при вычислении с использованием уравнения 1. Если исходное поведение имеет исключительно высокую оценку новизны, новые решения в последующих поколениях с поведением, которое имеет высокую новизну, но не такие новые, как исходное поведение, будут проигнорированы по этой причине. Таким образом, алгоритм не будет учитывать такое новое поведение. Это можно, в некоторой степени, исправить, используя оценку новизны в уравнении 2, что позволит установить верхнюю границу оценки новизны, но саму проблему не устранит полностью. Именно поэтому и предлагается использовать деструкцию новизны. Применительно к дифференциальной эволюции деструкция новизны будет описываться формулой 3, где δ – значение, которое применяется к оценке предыдущего поколения.

$$\overline{nov}(x)_i = \overline{nov}(x)_{i-1} \times \delta. \quad (3)$$

Позволяя оценке новизны снижаться со временем, алгоритм дифференциальной эволюции признает, что поведение, которое было новым в предыдущих поколениях, не обязательно будет новым в будущем. Это значение новизны δ предотвращает застревание алгоритма на предыдущих новых решениях, которые были новыми для поколения, в котором они появились, но не для последующих поколений. Преимущество деструкции новизны состоит в том, что это нересурсозатратно, с вычислительной точки зрения, и легко реализуется.

Второй метод, который будет оцениваться как средство решения проблемы – метод актуализации оценок новизны (ARN). Этот метод заключается в обновлении оценок новизны всех архивных моделей поведения для каждого поколения, чтобы проверить, являются ли они все еще новыми. Недостатком данного подхода является то, что он требует больших вычислительных затрат. В случае развития нейронной сети с дифференцированной эволюцией это приведет к удвоению количества вычислений оценки новизны.

Поиск новинок был впервые применен в решении задачи поиска пути в лабиринте [1,8]. Эволюционная нейронная сеть должна научиться перемещаться по лабиринту к целевому месту за конечное количество шагов. Понятие лабиринта также позволяет рассчитывать новизну поведения, то есть координаты агента в лабиринте. По этим причинам лабиринт является естественным выбором для проверки предлагаемого распада новизны. Тестовый лабиринт представлен на Рисунке 1. Выход обозначен буквой G (Goal) и помечен красным цветом. Входы обозначены цифрами (1-5) и помечены зеленым цветом. Таким образом, тестовый лабиринт совмещает в себе 5 лабиринтов различной сложности.

Экспериментальным путем выбрана нейронная сеть с оптимальным фиксированным количеством нейронов в скрытом слое. Большее количество нейронов в скрытом слое увеличивает временные и ресурсные затраты, не добавляя существенной эффективности. Конфигурация нейронной сети, которую необходимо обучить, представлена на Рисунке 2. Она имеет 8 входов (расстояние до ближайшей стены), 4 выхода (направление движения) и один скрытый слой, в котором 6 нейронов.

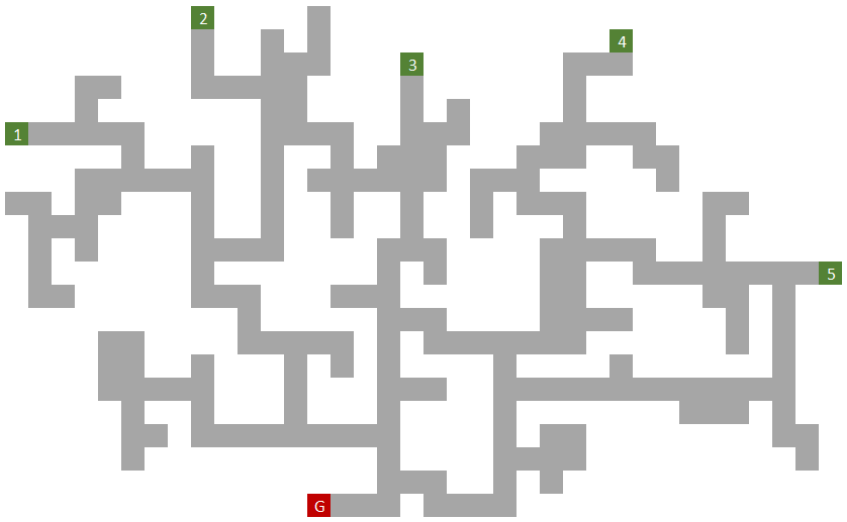


Рисунок 1 – Тестовый лабиринт
Figure 1 – Test maze

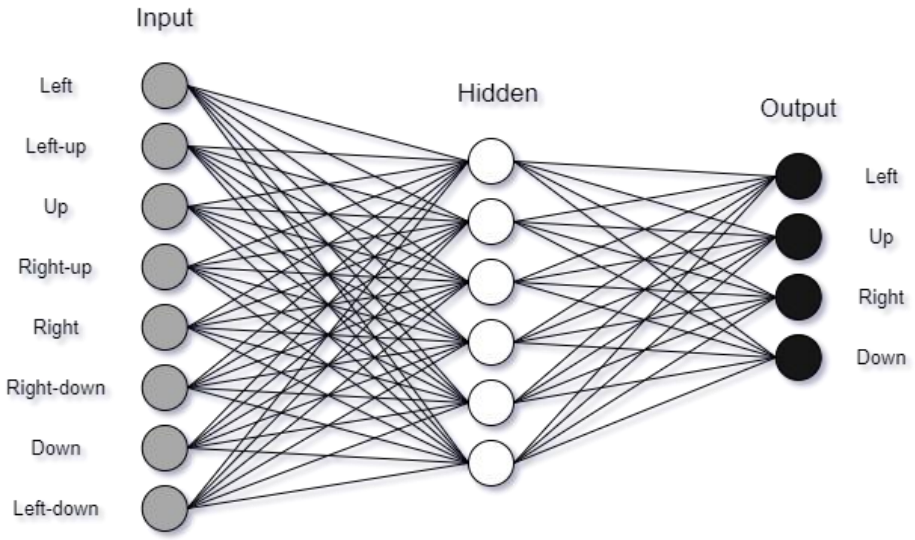


Рисунок 2 – Конфигурация нейронной сети
Figure 2 – Neural network configuration

Схема примененного в данном исследовании алгоритма отображена на Рисунке 3 и представляет собой алгоритм дифференциальной эволюции (начальная популяция, расчет целевой функции, воспроизводство потомков и т. д.) с применением поиска новинок на различных этапах и добавления нескольких дополнительных шагов в алгоритме.

Каждая особь популяции представляют собой весовую матрицу. Пригодность решения определяется как евклидово расстояние между конечной точкой решения и целевой точкой. Решение считается оптимальным, если евклидово расстояние меньше заранее определенного значения. Максимальное количество поколений – 200000. Параметры DE: CR = 0.9, F = 0.5 и количество агентов = 28.

Нейронная сеть написана на языке Python с использованием библиотек Keras и Tensorflow.

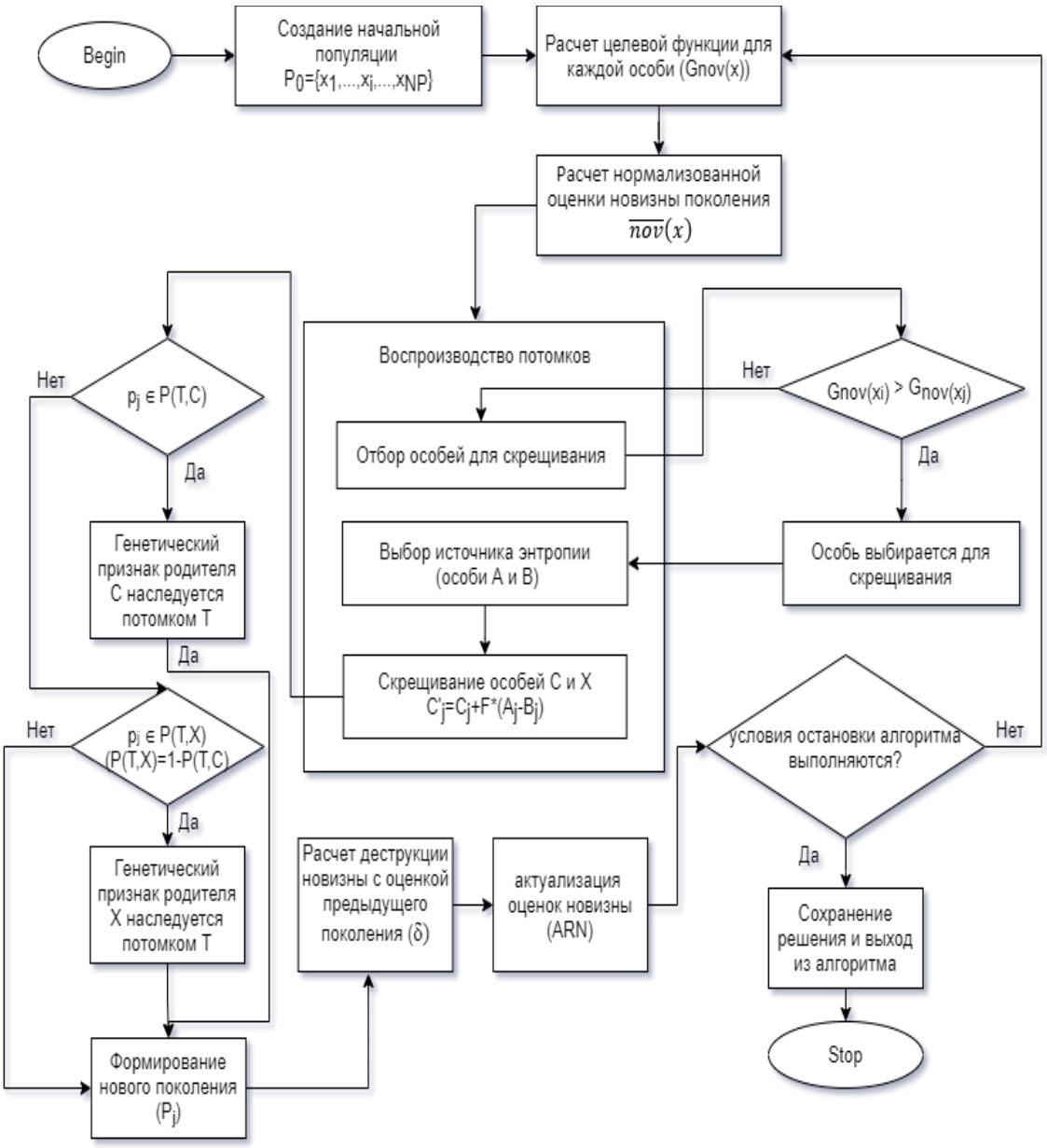
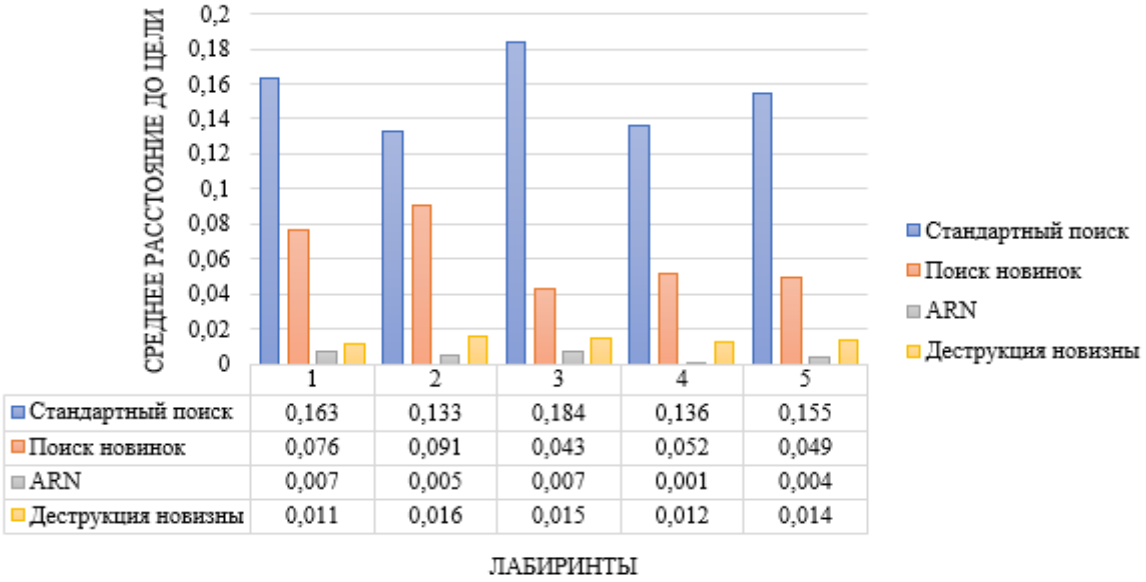


Рисунок 3 – Схема алгоритма гибридной нейроэволюции
Figure 3 – Hybrid neuroevolution algorithm scheme

Как видно из Рисунка 3, поиск новинок в дифференциальной эволюции применяется на этапе расчета целевой функции. Помимо этого, добавляется два дополнительных шага: деструкция новизны и актуализация оценок новизны для поколений.

Результаты и обсуждение (Results and Discussion)

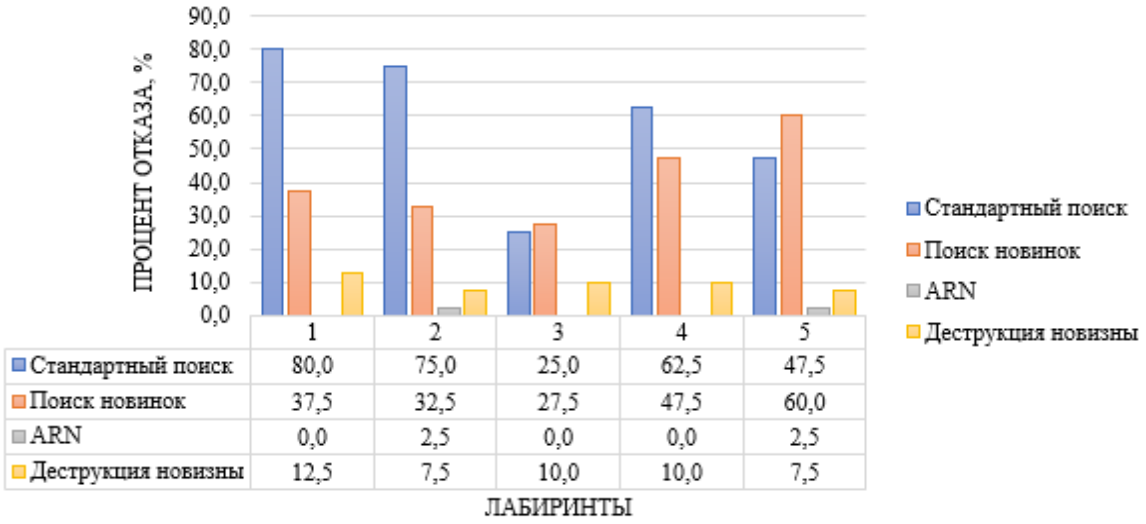
Поиск новинок с деструкцией новизны работает значительно лучше, чем без деструкции. На Рисунке 4 отображено среднее расстояние до цели для каждого алгоритма в каждом из пяти лабиринтов. Лучшие показатели у предложенных подходов: актуализация оценок новизны (ARN) и деструкция новизны. Хуже всего работает стандартный поиск.



	1	2	3	4	5
Стандартный поиск	0,163	0,133	0,184	0,136	0,155
Поиск новинок	0,076	0,091	0,043	0,052	0,049
ARN	0,007	0,005	0,007	0,001	0,004
Деструкция новизны	0,011	0,016	0,015	0,012	0,014

Рисунок 4 – Среднее расстояние до цели для каждого алгоритма в каждом лабиринте
Figure 4 – Average distance to the target for each algorithm in each maze

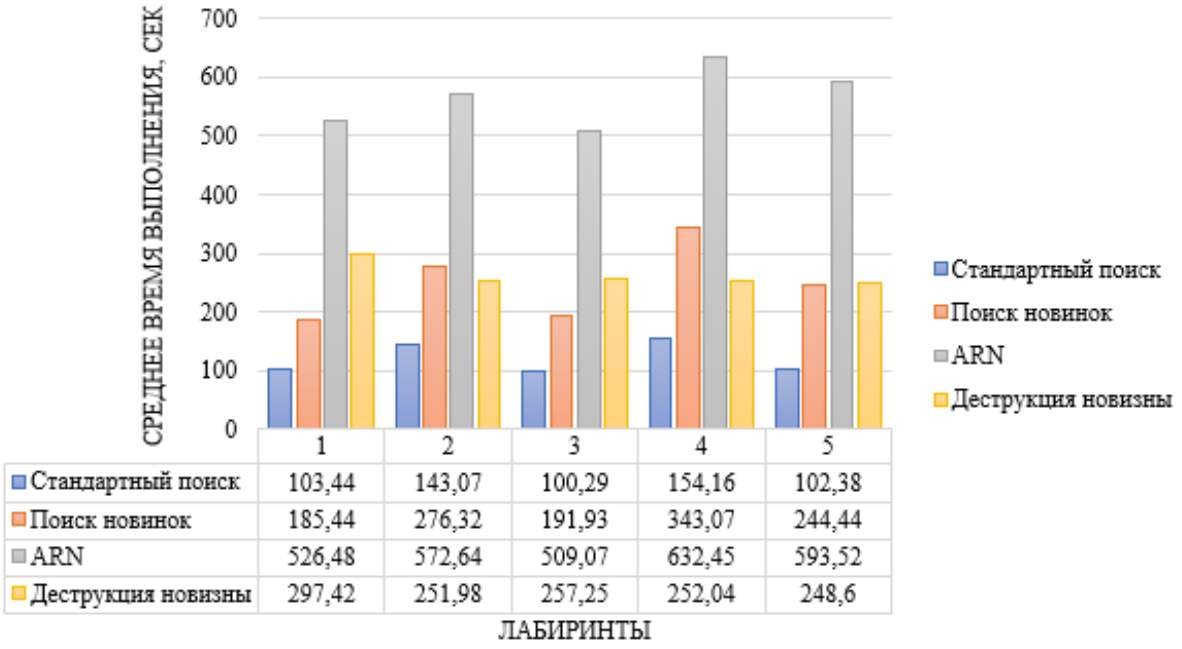
На Рисунке 5 отражен процент отказов каждого алгоритма в каждом лабиринте. Для каждого алгоритма было произведено по 40 запусков. Отказом следует считать превышение максимального количества поколений и недостижение цели. Большее количество отказов у стандартного поиска. Меньше всего отказов у алгоритмов ARN и деструкции новизны. Тем не менее, нельзя выделить ни одного алгоритма, который бы не отказал хотя бы один раз.



	1	2	3	4	5
Стандартный поиск	80,0	75,0	25,0	62,5	47,5
Поиск новинок	37,5	32,5	27,5	47,5	60,0
ARN	0,0	2,5	0,0	0,0	2,5
Деструкция новизны	12,5	7,5	10,0	10,0	7,5

Рисунок 5 – Процент отказов каждого алгоритма в каждом лабиринте из 40 запусков
Figure 5 – The failure rate of each algorithm in each maze of 40 runs

Среднее время выполнения каждого алгоритма в каждом лабиринте представлено на Рисунке 6. Диапазон временного разброса от 100 секунд до 650 секунд. Больше времени было потрачено на алгоритм ARN для каждого лабиринта. Быстрее всех выполняет стандартный поиск в то время, как поиск новинок и деструкция новизны держатся в одном диапазоне.



	1	2	3	4	5
Стандартный поиск	103,44	143,07	100,29	154,16	102,38
Поиск новинок	185,44	276,32	191,93	343,07	244,44
ARN	526,48	572,64	509,07	632,45	593,52
Деструкция новизны	297,42	251,98	257,25	252,04	248,6

Рисунок 6 – Среднее время выполнения (сек.) каждого алгоритма в каждом лабиринте
Figure 6 – Average execution time (sec.) of each algorithm in each maze

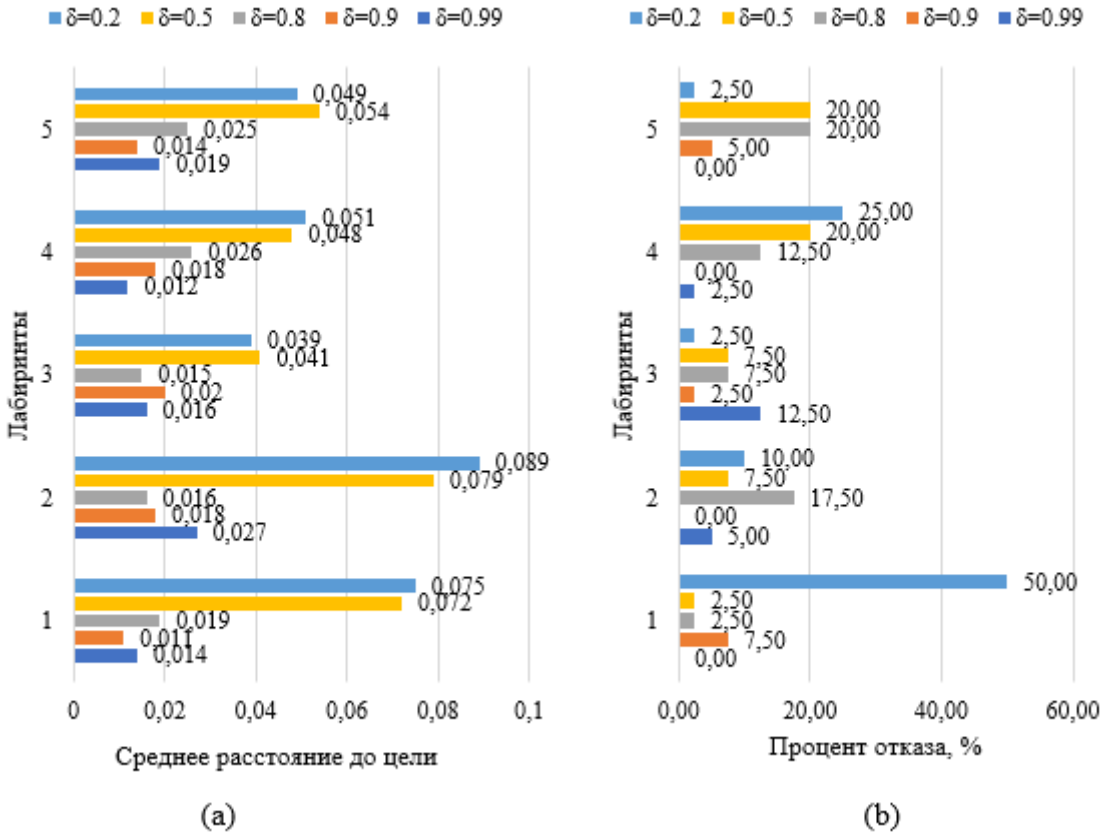


Рисунок 7 – Характеристики алгоритма деструкции новизны при разных значениях δ: (a) среднее расстояние до цели в каждом лабиринте; (b) процент отказа в каждом лабиринте из 40 запусков
Figure 7 – Characteristics of the novelty destruction algorithm for different values of δ: (a) the average distance to the target in each maze; (b) the failure rate in each maze of 40 runs

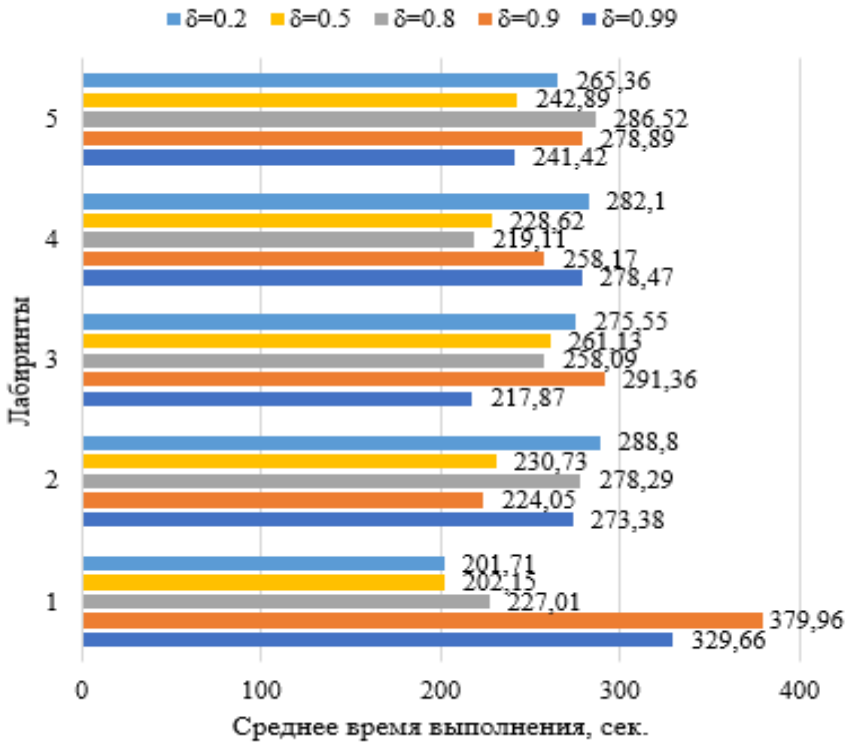


Рисунок 8 – Среднее время выполнения алгоритма деструкции новизны при разных значениях δ в каждом лабиринте
Figure 8 – Average execution time of the novelty destruction algorithm for different values of δ in each labyrinth

На Рисунке 7 и Рисунке 8 отображены характеристики алгоритма деструкции новизны при разных значениях δ в каждом лабиринте. Как видно, невозможно выделить однозначно лучшее значение параметра δ тестовых лабиринтов. Деструкция новизны показывает лучшие решения при значении параметра δ в диапазоне от 0.8 до 0.99. Чем меньше значение параметра δ, тем меньше данный параметр влияет на алгоритм и, соответственно, тем больше схожесть его работы с классическим поиском новинок. В целом наблюдается, что высокое значение δ, близкое к 1, дает наилучший результат.

Заключение (Conclusion)

С точки зрения вычислительной мощности, как и предполагалось, метод актуализации оценок новизны (ARN) более ресурсозатратен, чем деструкция новизны. Таким образом, ARN выполняется значительно медленнее всех других алгоритмов из-за дополнительных вычислений. Стандартный поиск значительно быстрее, но, значительно хуже, с точки зрения точности.

Полученные результаты показывают, что поиск новинок однозначно совместим с дифференциальной эволюцией. Стандартная реализация поиска новинок работает лучше, чем стандартный поиск по целевой функции. Недостатком поиска новинок является отсутствие обновления оценок новизны. Оба подхода, предложенных в данном исследовании, решают эту проблему. У каждого из этих подходов есть преимущества и недостатки. Деструкция новизны нересурсозатратна и проста в реализации, в то время как, актуализация оценок новизны сходится быстрее. Данные подходы могут быть

применимы к другим алгоритмам, таким как генетические алгоритмы, грамматическая эволюция, генетическое программирование, оптимизация роя частиц и т. д.

В дальнейшем планируется применение описанных подходов к управлению робототехническими системами и изучение динамической корректировки значения δ. Для решения различных задач требуются разные значения δ, чтобы деструкция новизны работала оптимально. Возможно, динамическая корректировка значения δ по мере обучения сети для адаптации к уровню сложности задачи устранит проблему выбора наиболее эффективного значения δ.

ЛИТЕРАТУРА

1. Lehman J., Stanley K.O. Abandoning objectives: Evolution through the search for novelty alone. Evolutionary computation. 2011;19(2):189-223.
2. Ahn C.W., Ramakrishna R.S. Elitism-based compact genetic algorithms. IEEE Transactions on Evolutionary Computation. 2003;7(4):367-385.
3. Bratton D., Kennedy J. Defining a standard for particle swarm optimization. IEEE Swarm Intelligence Symposium. 2007. SIS 2007. IEEE.
4. Storn R.M., Price K. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization. 1997;11(4):341-359.
5. Corucci F., Calisti M., Hauser H., Laschi C. Evolutionary discovery of self-stabilized dynamic gaits for a soft underwater legged robot. Advanced Robotics (ICAR), 2015 IEEE International Conference.
6. Reehuis E., Olhofer M., Sendhoff B., Back T. Novelty-guided restarts for diverse solutions in optimization of airfoils. A Bridge between Probability, Set-Oriented Numerics, and Evolutionary Computation, EVOLVE 2013.
7. Naredo E., Trujillo L., Martínez Y. Searching for novel classifiers. European Conference on Genetic Programming. EUROGP 2013.
8. Liapis A., Yannakakis G., Togelius J. Enhancements to constrained novelty search: Two-population novelty search for generating game content. Proceedings of the Genetic and Evolutionary Computation Conference.2013.
9. Cuccu G., Gomez F. When novelty is not enough. Applications of Evolutionary Computation. EvoApplications 2011. Lecture Notes in Computer Science, vol 6624. Springer, Berlin, Heidelberg.
10. Yao X. Evolving artificial neural networks. Proceedings of the IEEE. 1999;87(9):1423-1447.
11. Stanley K.O., Miikkulainen R. Evolving neural networks through augmenting topologies. Evolutionary computation. 2002;10(2):99-127.

REFERENCES

1. Lehman J., Stanley K.O. Abandoning objectives: Evolution through the search for novelty alone. Evolutionary computation. 2011;19(2):189-223.
2. Ahn C.W., Ramakrishna R.S. Elitism-based compact genetic algorithms. IEEE Transactions on Evolutionary Computation. 2003;7(4):367-385.
3. Bratton D., Kennedy J. Defining a standard for particle swarm optimization. IEEE Swarm Intelligence Symposium. 2007. SIS 2007. IEEE.
4. Storn R.M., Price K. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization. 1997;11(4):341-359.

5. Corucci F., Calisti M., Hauser H., Laschi C. Evolutionary discovery of self-stabilized dynamic gaits for a soft underwater legged robot. Advanced Robotics (ICAR), 2015 IEEE International Conference.
6. Reehuis E., Olhofer M., Sendhoff B., Back T. Novelty-guided restarts for diverse solutions in optimization of airfoils. A Bridge between Probability, Set-Oriented Numerics, and Evolutionary Computation, EVOLVE 2013.
7. Naredo E., Trujillo L., Martínez Y. Searching for novel classifiers. European Conference on Genetic Programming. EUROGP 2013.
8. Liapis A., Yannakakis G., Togelius J. Enhancements to constrained novelty search: Two-population novelty search for generating game content. Proceedings of the Genetic and Evolutionary Computation Conference.2013.
9. Cuccu G., Gomez F. When novelty is not enough. Applications of Evolutionary Computation. EvoApplications 2011. Lecture Notes in Computer Science, vol 6624. Springer, Berlin, Heidelberg.
10. Yao X. Evolving artificial neural networks. Proceedings of the IEEE. 1999;87(9):1423-1447.
11. Stanley K.O., Miikkulainen R. Evolving neural networks through augmenting topologies. Evolutionary computation. 2002;10(2):99-127.

ИНФОРМАЦИЯ ОБ АВТОРАХ / INFORMATION ABOUT THE AUTHORS

Березина Виктория Андреевна, аспирантка, Северо-Кавказский федеральный университет, Ставрополь, Российская Федерация
e-mail: *BerezinaVA@yandex.ru*

Victoria A. Berezina, post-graduate Student, North-Caucasus Federal University, Stavropol, Russian Federation

Мезенцева Оксана Станиславовна, кандидат физико-математических наук, Профессор кафедры информационных систем и технологий, Северо-Кавказский федеральный университет, Ставрополь, Российская Федерация
e-mail: omezentceva@ncfu.ru

Oksana S. Mezentseva, Candidate of physical-mathematical sciences, Professor of the Department of Information Systems and Technologies, North-Caucasus Federal University, Stavropol, Russian Federation

Ганьшин Константин Юрьевич, аспирант, Северо-Кавказский федеральный университет, Ставрополь, Российская Федерация
e-mail: magnuskos@gmail.com

Konstantin Y. Ganshin, post-graduate Student, North-Caucasus Federal University, Stavropol, Russian Federation