*ТЕХНИЧЕСКИЕ НАУКИ / TECHNICAL SCIENCES*

УДК 004.021 https://doi.org/10.33619/2414-2948/61/25

# СРАВНИТЕЛЬНЫЙ АНАЛИЗ АЛГОРИТМОВ ПОИСКА ОПТИМАЛЬНОГО ПУТИ

**©Султанова А. Б.,** *ORCID: 0000-0003-3230-6349, кан. тех. наук,*
*Азербайджанский государственный университет нефти и промышленности;*
*Институт систем управления НАН Азербайджана, г. Баку, Азербайджан, saxira@mail.ru*

# COMPARATIVE ANALYSIS OF OPTIMAL PATH SEARCH ALGORITHMS

**©Sultanova A.,** *ORCID: 0000-0003-3230-6349, Ph.D., Azerbaijan State University of Oil and Industry; Institute of Control Systems of Azerbaijan NAS, Baku, Azerbaijan, saxira@mail.ru*

*Аннотация.* В данной статье рассмотрены широко используемые алгоритмы поиска оптимальных путей. В настоящее время существует довольно широкий список алгоритмов поиска кратчайшего пути, которые активно применяются в мобильной робототехнике для поиска оптимального маршрута. Предлагается двухуровневая система, осуществляющая планирование движения. Произведен сравнительный анализ различных методов поиска пути: их длины, сложности, числа точек поворота. Целью статьи является исследование и сравнительный анализ алгоритмов из области искусственного интеллекта для поиска кратчайшего пути в лабиринте и шестиугольная сетке. Изучаемые алгоритмы: A* (звезда), алгоритм Дейкстры, алгоритм BFS (Breadth first search), DFS (Depth First Search) и Greedy. Алгоритмы сравниваются по двум критериям: длина найденного пути и время нахождения пути. Результаты, представленные аналитически и графически, показывают применение пять алгоритмов для лабиринтов с различным размером и количеством препятствий.

*Abstract.* This article discusses widely used algorithms for finding optimal paths. Currently, there is a fairly wide list of algorithms for the problem of finding the shortest path, and is actively used in mobile robotics to find the optimal route. The article offers a two-level system that performs traffic planning. Comparative analysis of various search methods was carried out: their length, complexity, and a number of turning points. The purpose of the article is to study and compare algorithms from the field of artificial intelligence for finding the shortest path in a maze and a hexagonal grid. Algorithms under study: A* (star), Dijkstra algorithm, BFS, DFS, and Greedy algorithm. Algorithms are compared based on two criteria: the length of the found path and the time it takes to find the path. The results presented analytically and graphically, show the application of five algorithms for mazes with different size and number of obstacles.

*Ключевые слова*: интеллектуальный робот, интеллектуальный интерфейс, система управления поведением, неопределенные среды, поиск пути, алгоритм Дейкстры, алгоритм A*, алгоритм BFS; алгоритм DFS; алгоритм Greedy.

*Keywords:* intelligent robot, intelligent interface, behavior management system, undefined environments, path search, A* algorithm, Dijkstra algorithm, BFS algorithm, DFS algorithm, algorithm Greedy.

Т. 6. №12. 2020
https://doi.org/10.33619/2414-2948/61

*Введение*

Хотя за последние два десятилетия ряд разработок позволил повысить точность и эффективность методов поиска путей, эта проблема по-прежнему привлекает большое внимание исследователей. В общем случае существуют различные вариации задачи поиска пути [6, 7], такие как одноагентный поиск пути, многоагентный поиск пути, состязательный поиск пути, динамические изменения в окружающей среде, неоднородный рельеф местности, мобильные подразделения и неполная информация.

У каждой из этих проблем есть разные приложения в разных областях. Как правило, поиск пути состоит из двух основных этапов: генерация графов и алгоритм поиска путей.

Задача генерации графов для «топологии местности” рассматривается как основа приложений робототехники. Поиск пути является фундаментальным компонентом многих важных приложений в области GPS [1], робототехники [2, 3], логистики [4, 5] и может быть реализован в статическом, динамическом и реально -временные среды.

Поиск пути определяется как процесс перемещения объекта из его предыдущего положения в конечное положение. В различных областях применения используются алгоритмы поиска путей (PFA). К ним относятся игры и виртуальные туры, беспилотные транспортные средства, движение роботов и навигация.

Алгоритмы поиска пути полезны в области роботизированных манипуляций, так как они могут быть использованы для управления роботом по сложной местности без необходимости вмешательства человека [6].

Мобильные роботы, перемещающиеся в наружной или внутренней среде, должны иметь свое расписание движения и навигационные схемы, чтобы иметь возможность определять свое направление. Обычно блоки навигации и поиска пути размещаются на роботах так, чтобы они могли двигаться самостоятельно.

Для решения поставленной задачи используются методы теории нечетких множеств, технология нечеткой логики, теория автоматического управления, методы обработки цифровых изображений, теория нейронных сетей.

*Обзор алгоритмы поиска оптимального пути*

Алгоритмы поиска пути полезны в области роботизированных манипуляций, так как они могут быть использованы для управления роботом по сложной местности без необходимости вмешательства человека. Мобильные роботы, перемещающиеся в наружной или внутренней среде, должны иметь свое расписание движения и навигационные схемы, чтобы иметь возможность определять свое направление.

Существует большое количество алгоритмов, позволяющих определить маршрут, по которому можно попасть из одной точки в другую. Эти алгоритмы можно разбить на две группы:

-алгоритмы, позволяющие определить оптимальный путь [2-4];

-алгоритмы, позволяющие найти субоптимальный путь [2–5].

В первой группе для нахождения решения требуется полностью исследовать некоторую область. Самым простым способом поиска оптимального пути является полный перебор всех возможных маршрутов. В этом случае найденный путь будет кратчайшим. Однако такой способ неприменим в большинстве случаев из-за чрезмерных накладных расходов, так как требуется полное исследование всей карты и хранение ее в памяти.

В связи с этим на первый план выходит разработка алгоритмов поиска субоптимальных путей. Примером являются эвристические алгоритмы, которые на каждом шаге приближаются к конечной точке. Однако при поиске одного из близких к оптимальному пути следует учитывать, что изначально трудно точно предсказать, какой именно вариант будет выбран. К тому же, одним из требований к маршруту является его реалистичный внешний вид. В этом случае можно использовать различные алгоритмы при выборе направления на каждом шаге либо различные алгоритмы постобработки маршрутов.

Кениг и Лихачев предложили алгоритм поиска оптимального пути в неопределенной среде. Этот метод сильно отличается от существующих алгоритмов, но очень полезен при реализации системой в неизвестных условиях [7].

Если бы можно было получить точные и полные карты, было бы целесообразно использовать обычный метод поиска, такой как A* [9].

D. D. Harabor, A. Grastien [4] предложили одноагентный алгоритм поиска точек перехода (GPS) для решения общей задачи в робототехнике.

Uras et al. [5] опубликовали метод ускорения поиска пути путем генерации графов подцелей.

Björnsson Y., Enzenberger M., Holte R., Schaeffer J., Yap P. показали, что гексагональные сетки обладают многими желательными свойствами квадратных сеток [6].

Отман и др. [7] обсудили проблемы, с которыми сталкиваются исследователи, использующие алгоритмы поиска путей в робототехнике. Одной из главных проблем мобильной робототехники является избегание реальных объектов.

D. D. Harabor, A. Grastien и Ю. И. Кудинов представили иерархическое аннотированное расширение алгоритма A* (HAA*), которое может быть использовано в среде реального времени [4, 8]. Этот алгоритм может работать с одним абстрактным графом и может быть использован для планирования пути для агентов с неоднородными размерами. Точки маршрута широко используются в видеоиграх и робототехнике.

C. Niederberger, D. Radovic, M. Gross предложили алгоритм поиска пути на основе A* в статических ландшафтах с полигональными препятствиями [9].

H. Burchardt, R. Salomon предложили скорректированный генетический алгоритм поиска путей, основанный на малогабаритных роботах RoboCup [11].

Рассмотрим наиболее известные алгоритмы поиска оптимального пути [1].

1. A* (A-стар);
2. Алгоритм Дейкстры;
3. Поиск в глубину (Depth-first search, DFS);
4. Поиск в ширину (Breadh-first search, BFS);
5. Лучший Поиск (Greedy).

*Алгоритм A** является одним из наиболее известных алгоритмов поиска субоптимального пути [2].

Алгоритм A*, разработан П. Хартом, Н. Нильсоном и Б. Рафаэлем в 1968 году. A* — это модификация алгоритма Дейкстры, оптимизированная для единственной конечной точки. Алгоритм Дейкстры может находить пути ко всем точкам, A* находит маршрут от начальной вершины к конечной с наименьшей стоимостью. Он отдает приоритет путям, которые ведут ближе к цели, использует модель поиска «лучший первый» в сочетании с эвристической функцией:

$$F(x) = G(x) + H(x)$$

где G(x) — путь от стартовой точки до текущей, H(x) — эвристическая функция, оценивающая расстояние до финишной на каждом шаге, в качестве которой выбрано Чебышевское расстояние, равное наибольшему из удалений по вертикали и горизонтали (ортогональная метрика). A* пошагово просматривает все пути, ведущие от начальной вершины в конечную, пока не найдет минимальный. Сначала рассматриваются те маршруты, которые «кажутся» ведущими к цели. В начале работы просматриваются узлы, смежные с начальным; выбирается тот из них, который имеет минимальное значение эвристической функции, после чего этот узел раскрывается

Практически, алгоритм A* отличается от алгоритма Дейкстры направленностью обхода узлов графа за счет использования эвристической функции, определяющей ориентировочное расстояние между данным узлом и концом пути. Иными словами, приоритет отдается тем узлам, которые согласно эвристической функции находятся ближе к концу пути.

Алгори́тм Де́йкстры (Dijkstra's algorithm) — алгоритм поиска пути на графах, изобретенный нидерландским ученым Э. Дейкстрой в 1959 году. Находит кратчайшее расстояние от одной из вершин графа до всех остальных или до заданной конечной. Алгоритм работает пошагово — на каждом шаге он «посещает» одну вершину и пытается уменьшать метки. Работа алгоритма завершается, когда все вершины посещены.

Поиском в глубину (DFS – depth first search) называется один из методов обхода графа G = (V, E), суть которого состоит в том, чтобы идти «вглубь» пока это возможно. В процессе поиска в глубину вершинам графа присваиваются номера, а ребра помечаются. Обход вершин графа происходит согласно принципу: если из текущей вершины есть ребра, ведущие в непройденные вершины, то идем туда, иначе возвращаемся назад.

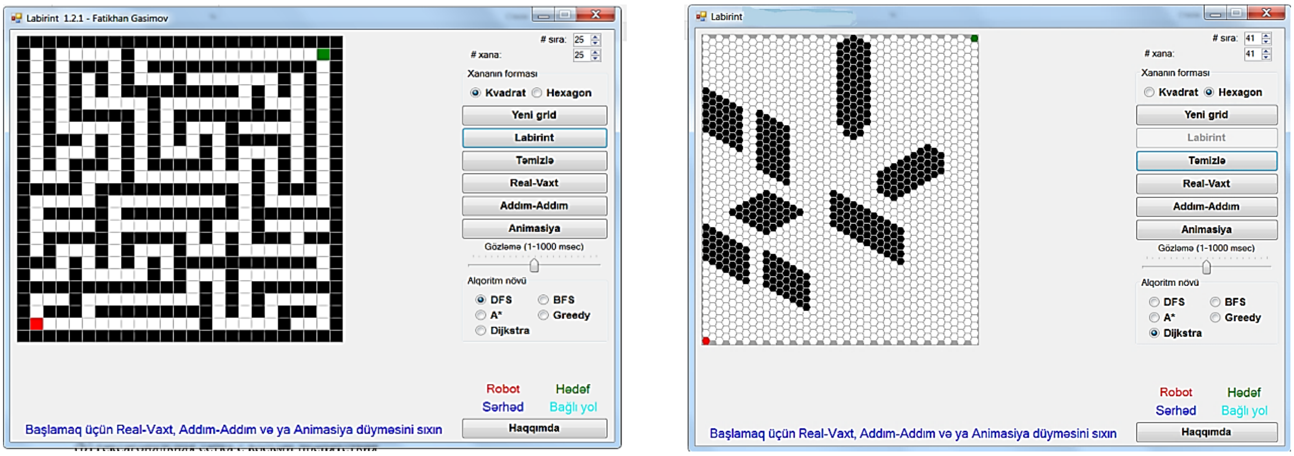
Поиск в ширину (Breadh-first search, BFS) известен как неинформированный поиск, потому что он не использует никакой информации . Основной идеей алгоритма является исследование на каждом этапе всех путей, ведущих из текущей вершины. Поиск в ширину выполняет исследование равномерно во всех направлениях. Это невероятно полезный алгоритм, не только для обычного поиска пути, но и для процедурной генерации карт, поиска путей течения, карт расстояний и других типов анализа карт.

Жадный алгоритм (Greedy Algorithm): Жадный алгоритм-это метод нахождения оптимального решения некоторой задачи, включающей большую однородную структуру данных (например, массив, дерево или граф), путем перехода от оптимального решения к некоторому компоненту или небольшой части структуры данных и расширения его, рассматривая дополнительные компоненты структуры данных один за другим, до оптимального глобального решения. Жадный алгоритм предполагает, что локальный оптимум является частью глобального оптимума. Жадный алгоритм-это любой алгоритм, который следует метаэвристике решения задачи принятия локально оптимального выбора на каждом этапе поиска глобального оптимума. Жадные алгоритмы дают хорошие решения для некоторых математических задач

*Постановка и решение вопроса*

*Структура используемых лабиринтов*

Сетка состоит из вершин или точек, соединенных ребрами для представления графа [7]. В большинстве алгоритмов поиска пути производительность навигации основана на атрибутах этого графического представления. Мы объясняем фундаментальную концепцию двух популярных сеточных подходов: квадратная сетка и гексагональных сетка. Квадратная сетка со статическим препятствиями -прямоугольный «лабиринт» (также «карта»), рассматриваемый в статье, является двумерным динамическим массивом, состоящим из элементов («ячеек» или «блоков»). Он имеет ширину, обозначаемую как $F_N$ и высоту — $D_M$. Начало координат отображается красным в левом нижнем углу карты, а целевая точка-зеленым цветом, расположенным в правом верхнем углу карты. Составленная программа помогает получить визуальный вид лабиринта (Рисунок 1).




a) b)

Рисунок 1. (a) квадратная сетка со статическим препятствиями, (b) гексагональная сетка с восьми препятствия

Лабиринт многофункциональный. Для нахождения кратчайшего пути используется квадратный лабиринт и гексагональная сетка. Можно создать различных размеров лабиринт и гексагональную сетку. Длина маршрута варьируется в зависимости от структуры лабиринта или гексагональной сетки. В составленном лабиринте реализованы пять алгоритмов: А * (звезда), алгоритм Дейкстры, алгоритм BFS (Breadth first search) , DFS (Depth First Search) и Greedy.

Для реализации задачи был выбран лабиринт размером 25 на 25. Полученные результаты приведены в Таблице.

Таблица.

РЕЗУЛЬТАТЫ, ПОЛУЧЕННЫЕ В ЛАБИРИНТЕ

| | *Алгоритм А** | *алгоритм Дейкстры* | *алгоритм BFS* | *DFS* | *Greedy.* |
|---|---|---|---|---|---|
| Найденный путь | 2995 | 2999 | 2985 | 2998 | 2993 |
| Шаг | 148 | 148 | 148 | 148 | 148 |

*Выводы*

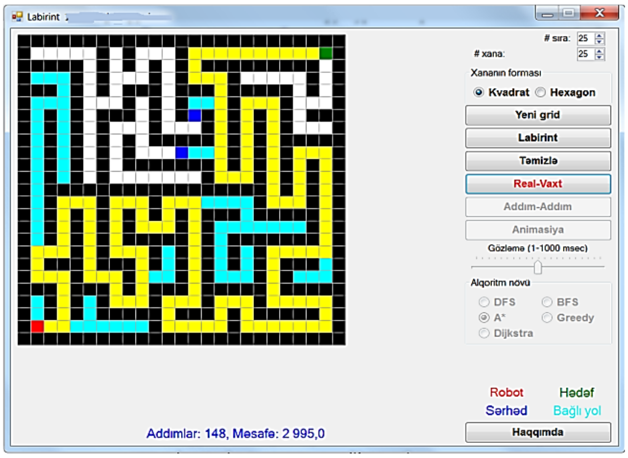
В статье реализован сравнительный анализ алгоритмов поиска оптимального пути. Все алгоритмы, рассмотренные выше, были сравнены по времени работы и длине находимого ими кратчайшего пути в лабиринте.

В статье был найден короткий путь с использованием различных алгоритмов.

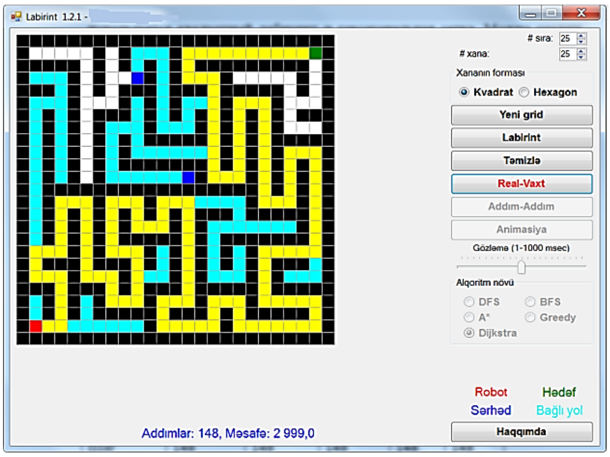
Сравнительный анализ показал, что алгоритм BFS достигает цели быстрее и кратчайшим путем.

Следует отметить, что это исследование также можно провести для гексагональной сетки.

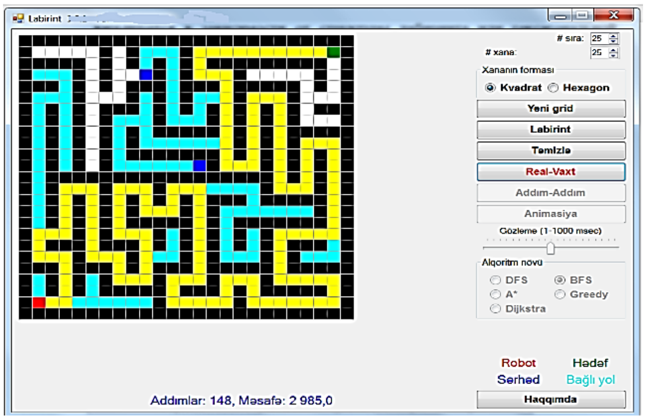
Т. 6. №12. 2020
https://doi.org/10.33619/2414-2948/61



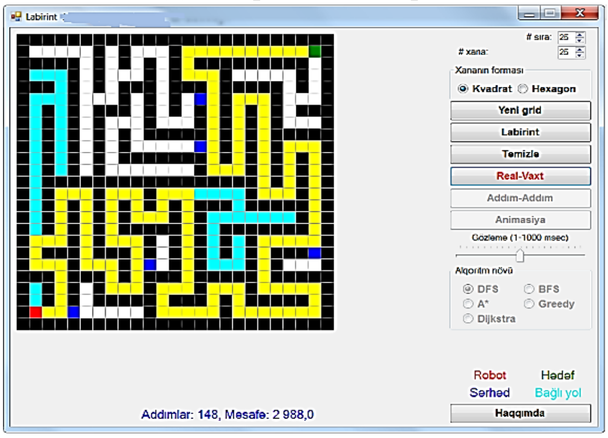

a) A * (звезда)



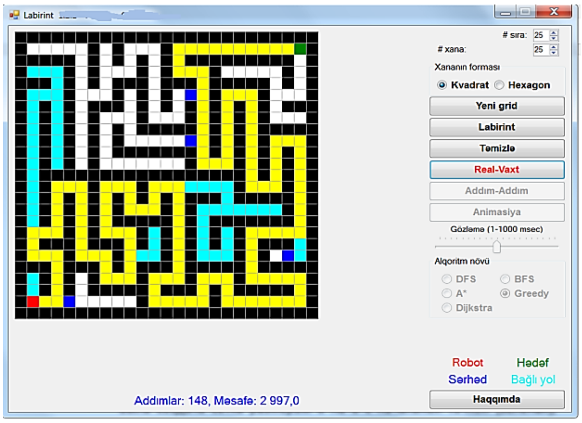

(b) алгоритм Дейкстры




(c) алгоритм BFS




(d) алгоритм DFS




(s) Greedy

Рисунок 2. a) A * (звезда), (b) алгоритм Дейкстры, (c) алгоритм BFS, (d) алгоритм DFS, (s) Greedy

*Список литературы:*

1. Peri V. M., Simon D. Fuzzy logic control for an autonomous robot //NAFIPS 2005-2005 Annual Meeting of the North American Fuzzy Information Processing Society. IEEE, 2005. P. 337-342. https://doi.org/10.1109/NAFIPS.2005.1548558

2. Van Den Berg J. et al. ANA*: anytime nonparametric A* // Proceedings of twenty-fifth AAAI conference on artificial intelligence (AAAI-11). 2011. V. 2. P. 1.

3. Bnaya Z., Stern R., Felner A., Zivan R., Okamoto S. Multi-agent path finding for self interested agents // Sixth Annual Symposium on Combinatorial Search. 2013.

4. Harabor D. D., Grastien A. Improving Jump Point Search // ICAPS. 2014.

5. Uras T., Koenig S., Hernández C. Subgoal graphs for optimal pathfinding in eight-neighbor grids // Twenty-Third International Conference on Automated Planning and Scheduling. 2013.

6. Björnsson Y., Enzenberger M., Holte R., Schaeffer J., Yap P. Comparison of different grid abstractions for pathfinding on maps // IJCAI. 2003. P. 1511-1512.

7. Othman M. F., Samadi M., Asl M. H. Simulation of dynamic path planning for real-time vision-base robots // FIRA RoboWorld Congress. Springer, Berlin, Heidelberg, 2013. P. 1-10. https://doi.org/10.1007/978-3-642-40409-2_1

8. Кудинов Ю. И. Нечеткие системы управления // Известия Академии наук. Техническая кибернетика. 1990. №5. С. 196-206.

9. Niederberger C., Radovic D., Gross M. Generic path planning for real-time applications // Proceedings Computer Graphics International, 2004. IEEE, 2004. P. 299-306. https://doi.org/10.1109/CGI.2004.1309225

10. Ma T., Yan Q., Liu W., Guan D., Lee S. Grid task scheduling: algorithm review // IETE Technical Review. 2011. V. 28. №2. P. 158-167.

11. Burchardt H., Salomon R. Implementation of path planning using genetic algorithms on mobile robots // 2006 IEEE International Conference on Evolutionary Computation. IEEE, 2006. P. 1831-1836. https://doi.org/10.1109/CEC.2006.1688529

*References:*

1. Peri, V. M., & Simon, D. (2005, June). Fuzzy logic control for an autonomous robot. In *NAFIPS 2005-2005 Annual Meeting of the North American Fuzzy Information Processing Society* (pp. 337-342). IEEE. https://doi.org/10.1109/NAFIPS.2005.1548558

2. Van Den Berg, J., Shah, R., Huang, A., & Goldberg, K. (2011, August). ANA*: anytime nonparametric A*. In *Proceedings of twenty-fifth AAAI conference on artificial intelligence (AAAI-11)* (2, 1).

3. Bnaya, Z., Stern, R., Felner, A., Zivan, R., & Okamoto, S. (2013, June). Multi-agent path finding for self interested agents. In *Sixth Annual Symposium on Combinatorial Search*.

4. Harabor, D. D., & Grastien, A. (2014, June). Improving Jump Point Search. In *ICAPS*.

5. Uras, T., Koenig, S., & Hernández, C. (2013, June). Subgoal graphs for optimal pathfinding in eight-neighbor grids. In *Twenty-Third International Conference on Automated Planning and Scheduling*.

6. Björnsson, Y., Enzenberger, M., Holte, R., Schaeffer, J., & Yap, P. (2003, August). Comparison of different grid abstractions for pathfinding on maps. In *IJCAI* (pp. 1511-1512).

7. Othman, M. F., Samadi, M., & Asl, M. H. (2013, August). Simulation of dynamic path planning for real-time vision-base robots. In *FIRA RoboWorld Congress* (1-10). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-40409-2_1

8. Kudinov, Yu. I. (1990). Nechetkie sistemy upravleniya. *Izvestiya Akademii nauk. Tekhnicheskaya kibernetika,* (5). 196-206.

9. Niederberger, C., Radovic, D., & Gross, M. (2004, June). Generic path planning for real-time applications. In *Proceedings Computer Graphics International, 2004.* (pp. 299-306). IEEE. https://doi.org/10.1109/CGI.2004.1309225

10. Ma, T., Yan, Q., Liu, W., Guan, D., & Lee, S. (2011). Grid task scheduling: algorithm review. *IETE Technical Review*, *28*(2), 158-167.

11. Burchardt, H., & Salomon, R. (2006, July). Implementation of path planning using genetic algorithms on mobile robots. In *2006 IEEE International Conference on Evolutionary Computation* (pp. 1831-1836). IEEE. https://doi.org/10.1109/CEC.2006.1688529

*Работа поступила в редакцию 08.11.2020 г.*

*Принята к публикации 12.11.2020 г.*

---

*Ссылка для цитирования:*

Султанова А. Б. Сравнительный анализ алгоритмов поиска оптимального пути // Бюллетень науки и практики. 2020. Т. 6. №12. С. 248-255. https://doi.org/10.33619/2414-2948/61/25

*Cite as (APA):*

Sultanova, A. (2020). Comparative Analysis of Optimal Path Search Algorithms. *Bulletin of Science and Practice, 6*(12), 248-255. (in Russian). https://doi.org/10.33619/2414-2948/61/25