Расширенная постановка задачи исследования местности с использованием многоагентной системы

М. Р. Гарифуллин
Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)
m.m.gari@mail.ru

Н. О. Турсуков
Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)
stepingnik@gmail.com

Т. А. Турушев
Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)
ta.turushev@gmail.com

С. С. Чупров
Рочестерский технологический институт (RIT)
drmyscull@gmail.com

Аннотация. **В статье представлен экспериментальный анализ сравнения алгоритмов планирования маршрутов для многоагентных систем в рамках задачи исследования местности. В качестве основного критерия качества работы алгоритмов выбран критерий пройденного расстояния до полного покрытия местности. Для сравнения алгоритмов используется разработанная имитационная среда исследуемой местности. Проведены испытания с использованием группы, имеющей в составе от двух до четырёх агентов. В дополнение к сравнению уже существующих алгоритмов, разработан новый алгоритм планирования маршрутов. В результате проведения сравнительного анализа предложены различные сценарии использования каждого из алгоритмов на практике.**

Ключевые слова: агент, многоагентная система, исследование местности, имитационная среда, сравнение алгоритмов, экспериментальный анализ

I. Введение

Использование многоагентных систем становится все более актуальным в современном мире, благодаря их способности автономно выполнять широкий спектр задач без необходимости присутствия человека. В частности, наблюдается значительное ускорение развития в области робототехники, которая включает в себя различные формы автоматизации и интеллектуальных систем. Эти системы способны эффективно решать задачи, связанные с мониторингом окружающей среды, анализом данных, управлением процессами и выполнением сложных операций.

В последние годы многоагентные системы активно применяются в различных сферах, включая медицину [1], образование [2] и сельское хозяйство. В медицине они используются для создания инновационных решений, таких как экзоскелеты и миниатюрные роботы для вживления в организм человека, что позволяет улучшить качество жизни пациентов. В образовании многоагентные комплексы служат инструментами для обучения и исследований, позволяя исследователям работать над реальными проектами и применять комплексные знания. В сельском хозяйстве многоагентные системы помогают автоматизировать уход за культурами и осуществлять управление сельхозтранспортом, что способствует повышению эффективности и снижению затрат.

Многоагентные системы в целом играют ключевую роль в современном обществе, обеспечивая автоматизацию и оптимизацию различных процессов, что позволяет снизить затраты, повысить качество работы и безопасность, а также расширить возможности для инноваций и развития.

При работе с группой агентов на местности требуется формировать планируемый маршрут движения, задавая его вручную, или с использованием специально разработанных программных средств, чтобы каждый участник группы исследовал определённый участок местности. Вместе с этим при вычислении маршрута каждого отдельного агента следует учитывать возможные перемещения других участников группы и имеющиеся препятствия, чтобы избежать опасных ситуаций, которые могли бы привести к столкновению агентов друг с другом или с препятствием. Для задач имитационного моделирования оптимальным является ситуация, когда исследуемая территория была разделена на приблизительно равные части для эффективного распределения ресурсов каждого из агентов.

Во время планирования маршрутов удобно представлять изначальную территорию в виде множества секторов, которые необходимо посетить агентами, чтобы полностью покрыть местность. Такое представление территории позволяет работать с ней, как с графом, где сектора представляют из себя вершины графа, рёбра соединяют два прилегающих друг к другу

сектора, а имеющиеся препятствия можно убрать, чтобы обозначить вершину, которую не следует рассматривать при планировании маршрута.

В качестве алгоритмов, формирующих маршруты движения агентов, рассматривается алгоритм, основанный на обходе графа в глубину, алгоритм, генерирующий случайный маршрут, и специально разработанный для приблизительно равномерного распределения областей между агентами алгоритм, основанный на использовании библиотеки KaHIP (Karlsruhe High Quality Partitioning) [3], представляющей из себя многоуровневый алгоритм разбиения графа на части [4]. Основным критерием для сравнения работы алгоритмов был выбран критерий пройденного расстояния до полного покрытия местности.

II. Имитационная среда

Для проведения тестирования алгоритмов была разработана имитационная среда. Данная среда позволяет в наглядной форме визуализировать работу алгоритмов, отображая перемещение агентов на карте. Имитационная среда представляет собой модуль, встраиваемый в программные решения и оперирующий со сформированными данными перемещений агентов.

Программа предполагает получение данных от оператора, который, в ходе взаимодействия определяет конфигурацию агентов (на каких участках происходит движение и какие изначальные координаты занимают агенты). Полученные данные передаются алгоритму нахождения пути и итоговый путь отображается в графическом интерфейсе пользователя.

Для создания модуля был выбран Python 3.9 в качестве основного языка программирования. В составе модуля был задействован фреймворк Django, основанный на языке программирования Python. Django предназначен для создания веб-приложений, обеспечивающих как скорость работы, так и безопасность. Использование данной платформы позволило организовать взаимодействие между клиентом и сервером через веб-приложение, доступное для загрузки в веб-браузере.

Для создания визуализации карты территории применялась интерактивная библиотека Bokeh, предназначенная для отображения данных веб-браузерами. Эта библиотека обеспечивает возможность работы с различными типами графиков, включая сложные и простые, а также позволяет реализовывать интерактивные модули для визуализации данных. В дальнейшем использовался язык программирования JavaScript для создания взаимодействия с отображаемыми данными на графике. Использование этого языка программирования позволяет клиенту получить удовлетворительный ответ от сервера всего лишь один раз, после чего проигрывать визуализацию непосредственно в своем браузере без необходимости дополнительных запросов к серверу (рис. 1).

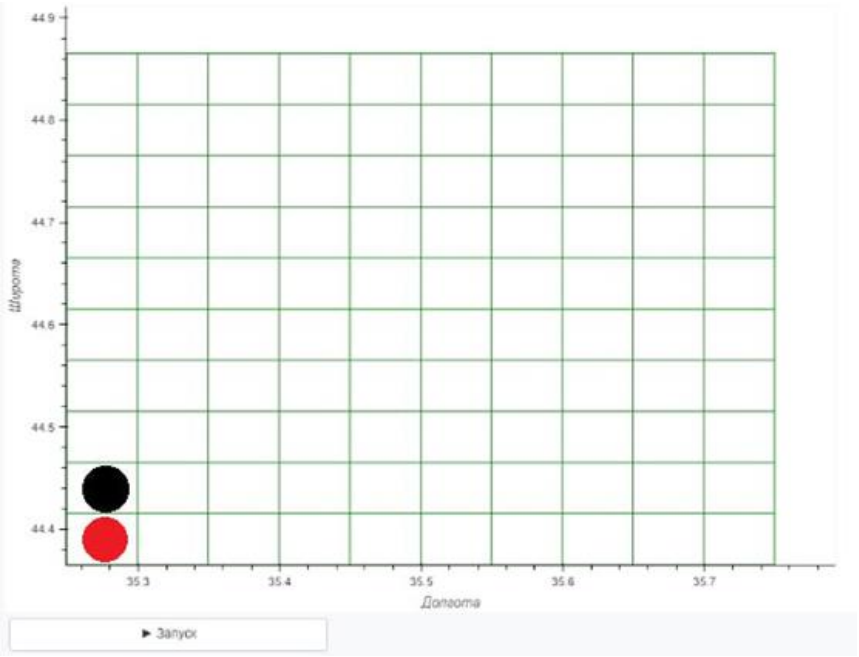

Рис. 1. Внешний вид имитационной среды

III. Сравнение алгоритмов

A. Описание работы алгоритмов

Формирование маршрутов агентов начинается с того, что оператором задаются начальные местоположения агентов и имеющиеся на местности препятствия, после чего формируется порядок, в котором каждый из агентов будет делать выбор следующей вершины, которую он собирается посетить. Дальнейшее построение маршрута затем определяется выбранным алгоритмом.

Следует отметить, что в работе каждого алгоритма могут возникать «тупиковые» ситуации, когда смежные для текущего местоположения агента вершины оказались посещёнными ранее, но на местности ещё остались вершины, которые необходимо посетить, в таком случае агент выбирает ближайшую, никем не

посещённую ранее вершину, и формирует до неё кратчайший маршрут, при этом, когда такая вершина выбирается, она сразу же помечается как посещённая, чтобы в процессе работы другие агенты не включали её в свой путь, тем самым уменьшая количество случаев, когда одна вершина посещается несколько раз.

При работе алгоритма, генерирующего случайный маршрут, каждая следующая вершина, которую посетит агент, выбирается случайным образом, в результате чего при одних и тех же исходных данных алгоритм может формировать разные маршруты при каждом запуске. Это может быть полезно в случаях, когда может потребоваться рассмотреть несколько различных маршрутов и определить среди них наиболее подходящий по какому-либо критерию, например, если необходимо уменьшить количество вершин, проходимых одним агентом, или снизить количество пересечений маршрутов разных агентов друг с другом.

Алгоритм формирования маршрута, основанный на обходе графа в глубину, лишён элемента случайности, так как во время его работы выбирается первая в списке смежности вершина, которая не была посещена ранее. Такой алгоритм имеет больше шансов избежать возникновения «тупиковой ситуации», так как в большинстве случаев агент будет двигаться по прямой.

Последний из рассматриваемых алгоритмов планирования маршрутов основан на использовании библиотеки KaHIP, которая по своей сути является семейством программ, специально предназначенных для разбиения графа на части. В случае с использованием этой библиотеки для планирования маршрутов в начале работы происходит первоначальное разделение местности на части (области), где каждой области присваивается свой агент для её исследования. После того, как агент был присвоен конкретной области, строится кратчайший маршрут движения в эту область. В процессе движения агентов до областей, они могут прокладывать свой маршрут через области им не принадлежащие, тогда в таком случае каждая пройденная вершина из чужой области помечается как посещённая, чтобы исключить её повторное посещением агентом, назначенным на область, в которую эта вершина входит. Когда агент доходит до своей области исследования, он продолжает своё движение, не выходя за её границы, это движение может быть основано как на алгоритме обхода в глубину, так и на случайном алгоритме, в рассматриваемом конкретном случае используется алгоритм обхода в глубину.

B. Сравнение работы алгоритмов

Для сравнения алгоритмов имитационная среда была специально подготовлена для автоматического формирования случайных первоначальных местоположений агентов и различных препятствий. Сами тесты проводились на местности в конфигурации с размером 10x10, количество агентов для формирования маршрутов варьировалось в диапазоне от двух до четырёх, число препятствий фиксировано и равно шести, количество проводимых тестов для каждого из вариантов конфигурации – 10000. В результате тестирования высчитывалось среднее пройденное расстояние до полного покрытия местности агентами.

На рис. 2 представлен пример сформированных маршрутов для трёх агентов алгоритмом, генерирующим случайный маршрут. Изначальные координаты агентов [1, 1], [1, 2] и [5, 5], каждому агенту назначен свой цвет, первому агенту – красный, второму – черный, третьему – зелёный, в процессе работы маршруты агентов пересекались между собой, соответствующие пересечения обозначены кругами с несколькими цветами. Препятствия на рисунке обозначены серыми квадратами.

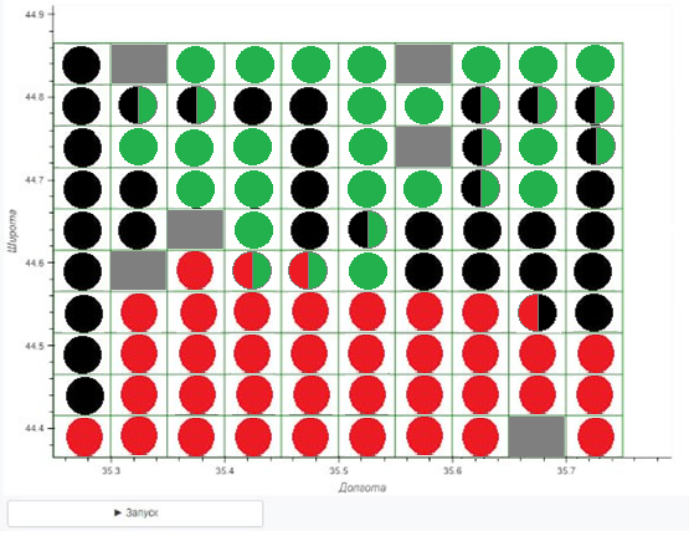

Рис. 2. Пример работы алгоритма, генерирующего случайный маршрут

Аналогично на рис. 3 представлен пример работы алгоритма, основанного на обходе в глубину, и на рис. 4 пример работы алгоритма, использующего библиотеку KaHIP. На рис. 4 можно заметить, что пересечений маршрутов агентов друг с другом не было, это связано с тем, что агенты в процессе своего движения достигли своих областей и продолжали движение в них же, при этом уже посещённые вершины, лежащие в чужих областях, повторно не посещались.

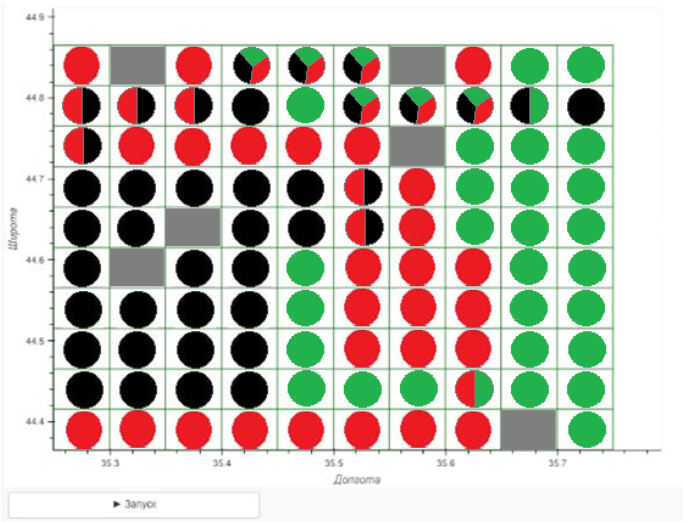

Рис. 3. Пример работы алгоритма, основанного на обходе в глубину

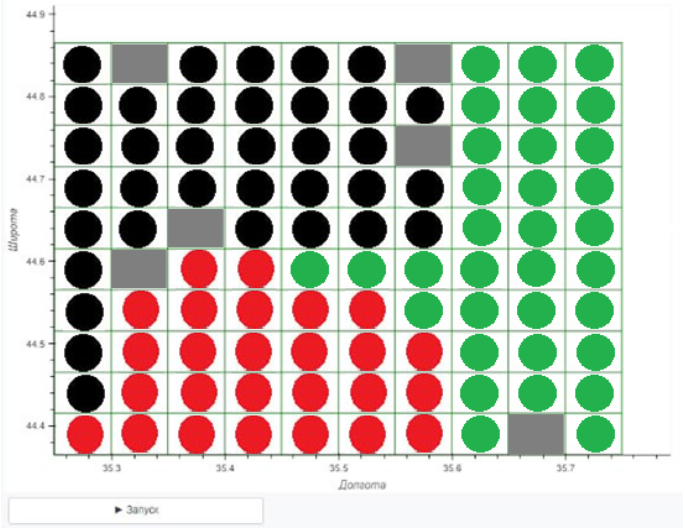

Рис. 4. Пример работы алгоритма, основанного на использовании библиотеки KaHIP, для трёх агентов

На рис. 5 представлен вариант работы алгоритма, использующего библиотеку KaHIP, для четырёх агентов. Изначальные координаты агентов [1, 1], [1, 10], [10, 1] и [10, 10], аналогично каждому агенту назначен свой цвет, первому агенту – красный, второму – черный, третьему – зелёный, четвёртому – синий. Как и в предыдущем случае использования этого алгоритма, пересечений маршрутов агентов друг с другом не происходило.

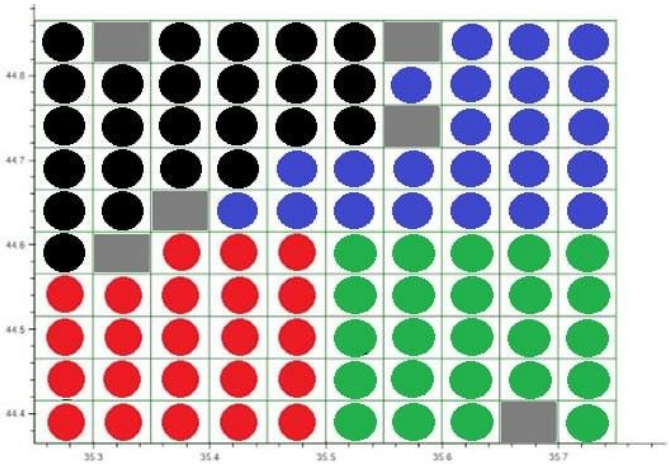

Рис. 5. Пример работы алгоритма, основанного на использовании библиотеки KaHIP, для четырёх агентов

В табл. 1 представлены результаты тестирования алгоритмов. Из полученных средних расстояний для каждого алгоритма видно, что хуже всего себя показал алгоритм, генерирующий случайный маршрут, это связано с тем, что в процессе своей работы полученные маршруты каждого из агентов часто пересекались из-за частого возникновения «тупиковых» ситуаций, в результате чего некоторые вершины могли быть посещены дважды. Алгоритм, основанный на обходе в глубину, и алгоритм, использующий библиотеку KaHIP, показали примерно одинаковые результаты, что вполне ожидаемо, так как в основе обоих лежит обход в глубину, однако во втором случае присутствует небольшое усложнение в виде равномерного распределения исследуемых территорий между агентами и необходимости движения агентов в сторону присвоенных им территорий.

ТАБЛИЦА I. РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

Алгоритм	**Случайный маршрут**			**Обход в глубину**			**Обход с использованием библиотеки KaHIP**		
Количество агентов	2	3	4	2	3	4	2	3	4
Количество тестов	10000	10000	10000	10000	10000	10000	10000	10000	10000
Среднее пройденное расстояние (в клетках)	64	44	34	56	39	30	56	40	33

Несмотря на то, что алгоритм, генерирующий случайный маршрут, показал себя хуже всего, его использование всё равно может быть полезно в сценариях, где необходимо проверить различные маршруты и выбрать среди них наилучший. Алгоритм, основанный на обходе в глубину можно брать в качестве вспомогательного или основного инструмента при создании более сложных алгоритмов. Алгоритм, основанный на использовании библиотеки KaHIP, может быть полезен в ситуациях, когда необходимо эффективно распределить ресурсы каждого из агентов для успешного выполнения задачи.

IV. ЗАКЛЮЧЕНИЕ

В результате проведения экспериментального анализа сравнения алгоритмов планирования маршрутов для многоагентных систем в рамках задачи исследования местности, были протестированы три алгоритма и предложены возможные варианты использования этих алгоритмов в различных сценариях.

СПИСОК ЛИТЕРАТУРЫ

[1] Диане С.А.К. Интеллектуальные роботы и многоагентные робототехнические системы: перспективы социальной интеграции // Философские проблемы информационных технологий и киберпространства. 2016. №2 (12).

[2] Родзин Сергей Иванович, Родзина Лада Сергеевна Многоагентные приложения для среды e-leaming // Открытое образование. 2014. №3.

[3] The graph partitioning framework KaHIP -- Karlsruhe High Quality Partitioning. // URL: https://github.com/KaHIP/KaHIP

[4] Пирова А.Ю. Параллельные алгоритмы разделения графов: учебно-методическое пособие. Нижний Новгород: Нижегородский госуниверситет, 2019. 20 с.