УДК 004.896

Модульный монолит как архитектурный шаблон для создания интегрированных сред разработки и визуализации мультиагентных систем обучения с подкреплением

Моргунов Егор Феликсович(*) morgunov-egor@mail.ru
SPIN-код: 7620-0191

МГТУ им. Н.Э. Баумана, Москва, Россия

Аннотация. *Рассмотрены различные методы разработки, обучения и настройки интеллектуальных систем с использованием глубокого обучения с подкреплением на основе нейросетевых алгоритмов. Предложен подход к проектированию мультиагентной среды в виде монолитной программной архитектуры, направленный на повышение качества обучения, адаптивности, эффективности управления ресурсами и возможностей мониторинга. Проведены эксперименты по обучению агентов в разработанной системе с применением классических алгоритмов глубокого* Q-*обучения. Сделаны выводы, что предложенный способ интеграции мультиагентного обучения с подкреплением и классических архитектур информационных систем может повысить качество и надежность обучения интеллектуальных систем, ориентированных на задачи дополненной реальности.*

Ключевые слова: *архитектура информационных систем, модульный монолит, мультиагентная среда, обучение с подкреплением, глубокое* Q-*обучение*

Modular monolith as an architectural template for creating integrated development and visualization environments of multi-agent reinforcement learning systems

Morgunov Egor Feliksovich(*) morgunov-egor@mail.ru
SPIN-code: 7620-0191

BMSTU, Moscow, Russia

Abstract. Various methods of developing, training, and configuring intelligent systems using deep reinforcement learning based on neural network algorithms are considered. An approach to designing a multi-agent environment as a monolithic software architecture is proposed, aimed at improving quality of learning, adaptability, resource management efficiency, and monitoring capabilities. Experiments on agent training were conducted in the developed system using classical Deep Q-learning algorithms. It is concluded that the proposed method of integrating multi-agent reinforcement learning with classical information system architectures can enhance the quality and reliability of training for intelligent systems oriented toward augmented reality applications.

Keywords: information systems architecture, modular monolith, multi-agent environment, reinforcement learning, deep Q-learning

Введение. Искусственный интеллект является неотъемлемым инструментом при разработке многих современных сред виртуальной и дополненной реальности [1]. Применение нейросетевых алгоритмов позволяет улучшить качество симуляций, повысить эффективность взаимодействия с пользователем и расширить способы взаимодействия между системой и человеком — от распознавания голоса и движений до полноценной стимуляции мозга [2]. Сочетание искусственного интеллекта и виртуальной реальности создает интерактивные и адаптивные среды, обеспечивающие высокую степень реализма, приближающуюся к реальным физическим условиям.

Одним из эффективных способов масштабирования и наполнения виртуальных симуляций является использование интеллектуальных агентов, улучшающих свои стратегии поведения с помощью алгоритмов обучения с подкреплением (Reinforcement Learning, RL) [3]. Обучение может проходить как в игровых средах, так и в сложных системах с динамически изменяющимися условиями (климат, физика движений, аэродинамические характеристики). Агенты учатся находить оптимальные стратегии поведения, взаимодействовать друг с другом и адаптироваться к изменениям системы.

Для разработки и тестирования классических алгоритмов RL, как правило, используются стандартизированные платформы и фреймворки, вроде OpenAI Gym и PettingZoo [4]. Несмотря на простоту реализации, подобные системы не предоставляют гибких средств контроля над процессом обучения, характеризуются жесткой связанностью компонентов и ограниченными возможностями мониторинга. Одним из способов повышения гибкости и адаптивности RL-сред является использование классических архитектур информационных систем [5]. Интеграция с монолитной и микросервисной архитектурой позволяет расширить возможности для обучения, обеспечить эффективный мониторинг вычислительных ресурсов и повысить модульность системы [6].

Целью исследования является изучение влияния архитектуры программного обеспечения на функционирование мультиагентной системы. В задачи исследования входит разработка монолитной программной архитектуры, интеграция мультиагентного обучения с подкреплением в разработанную модульную систему, проведение экспериментов по обучению агентов с помощью классического алгоритма централизованного Q-обучения, анализ полученных результатов.

Материалы и методы. Основой обучения с подкреплением является взаимодействие интеллектуального агента с виртуальной средой методом проб и ошибок [7]. За каждое совершенное действие агент получает числовое вознаграждение (подкрепление). Анализируя полученную информацию о состоянии среды, агент должен выработать стратегию, которая максимизирует его награду в процессе обучения. С активным развитием алгоритмов RL появилось мультиагентное обучение с подкреплением (MARL). В MARL несколько агентов помещаются в одну среду и взаимодействуют не только с ней, но и друг с другом. Кооперативное и конкурентное взаимодействие

агентов друг с другом открыло возможности для решения задач управления поведением и распределения ресурсов.

Одним из популярных методов обучения с подкреплением является алгоритм глубокого Q-обучения (DQN). DQN объединяет оценку Q-функций для нахождения максимальной долгосрочной награды, и глубокие нейронные сети для обработки многомерных пространств состояний [8]. Для повышения сходимости и стабильности обучения в DQN используется буфер воспроизведения, сохраняющий все кортежи (s, a, r, s', где s — состояние агента, a — совершаемое действие, r — получаемая награда, s' — следующее состояние) в качестве будущих обучающих данных, а также дополнительная целевая нейронная сеть, которая дублирует основную, но с определенной задержкой.

Для мультиагентного обучения с подкреплением существует несколько модификаций DQN, одной из которых является алгоритм централизованного Q-обучения (CDQN). В CDQN буфер воспроизведения, основная и целевая нейронные сети являются едиными для всех агентов. Такой подход позволяет агентам обучаться согласованно и преодолевать проблему нестационарности обучения.

Для обучения интеллектуальных агентов спроектирована двухмерная виртуальная среда на языке программирования Python с использованием библиотек NumPy и PyTorch. В ходе обучения агентам необходимо поймать динамическую цель, которая двигается по карте в горизонтальном направлении. Каждый шаг среды агенты исследуют окружающее пространство с помощью ограниченной зоны видимости, совершают действие (движение в одну из сторон) и получают целочисленную награду.

На рис. 1 представлен внешний вид пользовательского интерфейса разработанной системы по управлению MARL. Для повышения уровня контроля за обучением разработано программное приложение, которое является единой точкой входа в систему. При запуске проекта пользователь может настроить ключевые гиперпараметры, которые передаются алгоритму CDQN для обучения — число эпизодов обучения, скорость обучения (Learning Rate, LR), фактор дисконтирования (discount factor, γ). Также пользователь может выбрать режим работы MARL-системы — запуск обучающего алгоритма CDQN_train, или запуск алгоритма тестирования CDQN_test на основе обученной модели.

Для повышения наглядности обучения и контроля ресурсов системы добавлен функционал мониторинга загрузки CPU и оперативной памяти устройства, а также награды каждого агента за эпизод. Мониторинг проходит в режиме реального времени, а результаты выводятся на графики пользовательского интерфейса в процессе обучения и тестирования.

Для интеграции с классической архитектурой программного обеспечения все компоненты системы распределены по функциональности. Декомпозиция по модулям повышает надежность и отказоустойчивость системы путем минимизации взаимосвязей между компонентами. Подобная архитектура является модульным монолитом, который состоит из обособленных, взаимозаменяемых элементов, отвечающих за различные аспекты бизнес-функциональности [9].

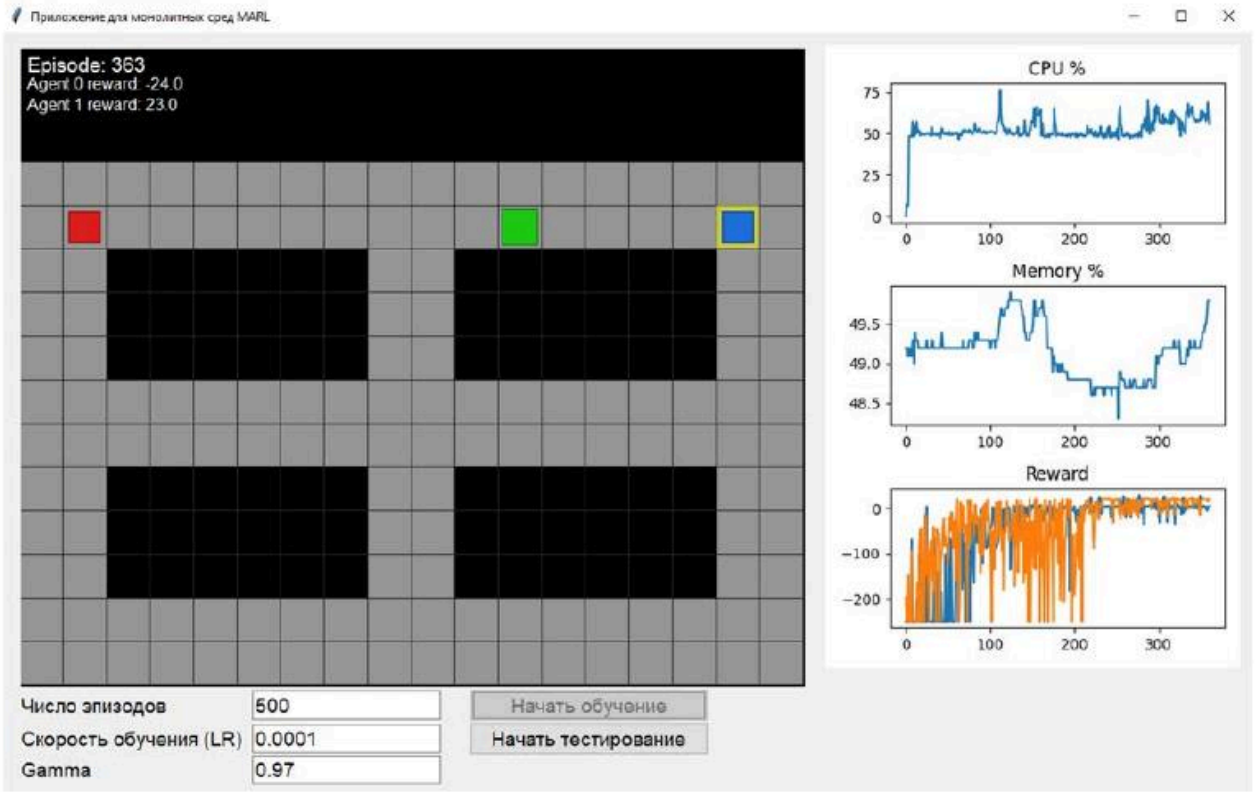

Рис. 1. Внешний вид пользовательского интерфейса монолитной MARL-системы

Результаты. На рис. 2 представлены результаты обучения монолитной MARL-системы с помощью алгоритма CDQN. Эксперимент был разбит на 500 эпизодов, для обучения были заданы следующие гиперпараметры: α = 0,0001, γ = 0,97, ε = 1, εd = 0,9995, εmin = 0,01, batch_size = 64, где α — скорость обучения, γ — фактор дисконтирования, ε — параметр эпсилон в начале обучения, εd — уменьшение эпсилон за каждый эпизод, εmin — минимальное значение эпсилон, batch_size — размер пакета.

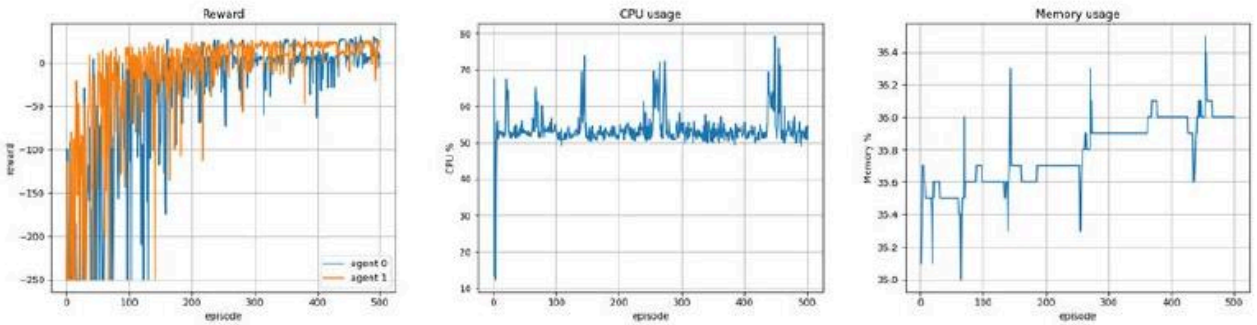

Рис. 2. Результаты обучения агентов в разработанной монолитной MARL-системе

Как видно из графиков, оба агента смогли прийти к оптимальной стратегии, позволяющей максимизировать награду и достигать цели за минимальное время. График потребления оперативной памяти показывает скачки в те

моменты, когда агенты обновляют свою стратегию и находят новые способы достижения цели. Таким образом, алгоритм CDQN сумел успешно обучить агентов достигать цели и ориентироваться в пространстве в разработанной мультиагентной среде, а применение модульной декомпозиции монолитной архитектуры повысило наглядность обучения и уровень контроля пользователя.

Обсуждение. Архитектура программного обеспечения является основой разработки современных комплексных информационных систем. Правильно выбранный метод проектирования веб-сайтов, приложений и сервисов позволяет снизить затраты на разработку и эксплуатацию, повысить масштабируемость и надежность системы, а также расширить технологические возможности приложений. С развитием технологий разработки и проектирования грамотный выбор программной архитектуры стал необходимостью не только для веб-приложений, но и для интеллектуальных сервисов и систем дополненной и виртуальной реальности.

Алгоритмы искусственного интеллекта становятся важным компонентом при создании и наполнении виртуальных симуляций. Для повышения производительности и качества таких систем необходима оптимизация нейросетевых алгоритмов и сред обучения интеллектуальных агентов. Предложенный подход по проектированию мультиагентной системы доказывает, что архитектура программного обеспечения может играть ключевую роль для обучения агентов в системах, где необходим подробный мониторинг производительности и качества обучения.

Заключение. В данной работе проведено исследование интеграции мультиагентного обучения в монолитную архитектуру. В ходе экспериментов был использован алгоритм централизованного глубокого обучения CDQN, который достиг сходимости в процессе поиска оптимальной стратегии. Результаты экспериментов обосновывают способность мультиагентного обучения успешно интегрироваться в монолитные системы, а совмещение MARL и классических архитектур программного обеспечения позволяет повысить надежность, функциональность и адаптивность распределенных систем.

В будущих работах планируется рассмотреть другие методы проектирования мультиагентных систем. Использование микросервисной архитектуры может повысить отказоустойчивость и технологическую неоднородность системы, что позволит эффективнее обучать агентов для комплексных сред виртуальной реальности.

Список источников

[1] Lampropoulos G. Combining artificial intelligence with augmented reality and virtual reality in education: current trends and future perspectives. Multimodal Technologies and Interaction, 2025, vol. 9(2), art. no. 11.

[2] Oumaima D., Mohamed L., Hamid H., Mohamed H. Application of artificial intelligence in virtual reality. International Conference on Trends in Sustainable Computing and Machine Intelligence. Singapore, Springer Nature Singapore, 2023, pp. 67–85.

[3] Morgunov E.F., Alfimtsev A.N. The “Stag Hunt” Social Dilemma in Multi-Agent Reinforcement Learning. 6th International Youth Conference on Radio Electronics, Electrical and Power Engineering (REEPE), 2024, pp. 1–5. https://doi.org/10.1109/REEPE60449.2024.10479770

[4] Terry J., Black B., Grammel N., Jayakumar M., Hari A., Sullivan R., Santos L.S., Dieffendahl C., Horsch C., Perez-Vicente R., Pettingzoo W.N. Gym for multi-agent reinforcement learning. Advances in Neural Information Processing Systems, 2021, vol. 34, pp. 15032–15043.

[5] Wang H., Hu X., Yu Q., Gu M., Zhao W., Yan J. and Hong T. Integrating reinforcement learning and skyline computing for adaptive service composition. Information Sciences, 2020, vol. 519, pp. 141–160.

[6] Моргунов Е.Ф., Алфимцев А.Н. Мультиагентная микросервисная архитектура. *Вестник МГТУ им. Н.Э. Баумана. Сер. Приборостроение*, 2025, № 2 (151), с. 78–101.

[7] Sutton R.S., Barto A.G. Reinforcement learning: An introduction. MIT press, 2018, 552 p.

[8] Алфимцев А.Н. *Мультиагентное обучение с подкреплением*. Москва, Изд-во МГТУ им. Н.Э. Баумана, 2021, 224 с.

[9] Newman S. Building microservices: designing fine-grained systems. O'Reilly Media Inc., 2021.