УДК 004.89:004.94
DOI: 10.15827/0236-235X.141.046-059
Дата подачи статьи: 15.09.22, после доработки: 19.12.22
2023. Т. 36. № 1. С. 046–059

Моделирование поведения интеллектуальных агентов на основе методов машинного обучения в моделях конкуренции

А.О. Анохин [1], *аспирант, alex.anokhin.st@gmail.com*
Д.С. Парыгин [1], *к.т.н., доцент, dparygin@gmail.com*
Н.П. Садовникова [1], *д.т.н., профессор, npsn1@ya.ru*
А.А. Финогеев [2], *к.т.н., доцент, fanton3@ya.ru*
А.С. Гуртяков [1], *к.т.н., доцент, agurtyakov@gmail.com*

[1] *Волгоградский государственный технический университет, г. Волгоград, 400005, Россия*
[2] *Пензенский государственный университет, г. Пенза, 440026, Россия*

В настоящей статье рассматриваются аспекты применения методов машинного обучения к существующим способам моделирования поведения интеллектуальных агентов для обеспечения возможности агентам повысить показатели своей эффективности в моделях конкуренции.

Практическая значимость исследования представлена разработкой подхода к моделированию поведения интеллектуальных агентов, за счет которого можно повысить эффективность их функционирования в таких сферах деятельности, как компьютерные игры, разработка беспилотных летательных аппаратов и поисковых роботов, изучение городской и транспортной мобильности, а также в прочих сложных системах.

Проведен обзор существующих методов машинного обучения (обучение с подкреплением, глубокое обучение, Q-обучение) и способов моделирования поведения агентов (модель на правилах, конечно-автоматная модель поведения, деревья поведения). Выбрана наиболее подходящая к задаче комбинация метода обучения и модели поведения: деревья поведения и обучение с подкреплением.

Средствами Unity реализована тестовая платформа, разработаны модели поведения четырех основных архетипов агентов, которые должны соревноваться в задаче сбора ресурсов в условиях ограниченного времени. Реализован обученный агент с помощью средств Unity ML и TensorFlow.

На базе тестовой платформы проведена серия экспериментов в различных условиях: ограниченность, изобилие, среднее количество ресурсов. В рамках эксперимента тестировалась способность разработанной модели поведения интеллектуального агента выигрывать в условиях конкуренции с агентами, снабженными различными вариантами традиционных моделей поведения на базе деревьев поведения. Оценены работоспособность и преимущества использования разработанной модели поведения. Проанализированы результаты эксперимента, сделаны выводы относительно потенциала выбранной комбинации методов.

Ключевые слова: *моделирование поведения, интеллектуальный агент, искусственный интеллект, машинное обучение, дерево поведения.*

Моделирование поведения интеллектуальных агентов – задача, находящая применение в мультиагентных системах, моделирующих такие процессы, как потоки городского движения [1, 2], эпидемии, поведение толпы [3, 4]. Одно из направлений применения мультиагентного подхода – моделирование поведения неигровых персонажей в компьютерных играх. Рост потребности в качественной организации игрового искусственного интеллекта обусловлен большим количеством компьютерных игр, выходящих каждый год.

В данной работе предложен подход к моделированию поведения агентов, основанный на комбинации деревьев поведения и обучения с подкреплением, который позволяет повысить эффективность функционирования интеллектуальных агентов.

Целью исследования являлась разработка теоретических основ и инструментальных программных средств для моделирования поведения интеллектуальных агентов в игровой среде и повышения эффективности их функционирования.

Подходы к моделированию поведенческих моделей конкуренции

Существуют несколько основных подходов к моделированию поведения интеллектуальных агентов.

Известно, что первой появилась модель поведения на основе правил. Ключевая идея подхода состоит в задании поведения с помощью правил для каждой конкретной ситуации. Модели поведения на основе правил использовались в большинстве классических игр 80-х гг., включая Pac-Man. В этой игре поведение каждого из привидений задается своими правилами: например, одно привидение на всех развилках поворачивает только направо, а другое – только в сторону игрока.

В число моделей на основе правил входят и более сложные реализации, правила в которых задаются на основе текущего состояния агента, его параметров и окружающей среды. В таком случае поведение будет более разнообразным и вариативным, однако останется предсказуемым [5].

Подход к моделированию на основе конечных автоматов предоставляет больше возможностей для построения вариативной модели поведения. Суть его заключается в определении основных возможных состояний агента (например, голод) и построении связей, по которым агент сможет переходить из одного состояния в другое.

Одна из наиболее известных реализаций игрового искусственного интеллекта на основе конечного автомата – Goal Oriented Action Planning (GOAP) [6]. Система появилась в 2005 году и была использована студией Monolith Productions в игре F.E.A.R. Позже вариации на тему этого подхода были применены в других играх, в том числе в S.T.A.L.K.E.R. и Fallout 3.

Последовательность действий агента при такой реализации определяется целью и текущим состоянием агента и окружающей среды, то есть в разных обстоятельствах одна и та же цель может быть достигнута разными способами. Такой подход требует реализации модуля-планировщика, который и будет определять, каким образом агент оптимально достигнет цели.

Также распространен подход к моделированию на основе деревьев поведения. Деревья представляют собой математические модели выполнения плана и часто используются в информатике в задачах, где необходимо описывать переключения в конечном наборе действий по модульному принципу. Они являются мощным инструментом для моделирования поведения интеллектуальных агентов и использованы в таких играх, как Halo, Bioshock и Spore. Однако сфера применения деревьев не ограничивается играми: они используются для решения таких классов задач, как управление беспилотными техническими средствами, роботизированные манипуляции и другие.

Дерево поведения, являясь разновидностью конечного автомата, имеет перед ним весомое преимущество: добавление новых действий в дерево поведения не сопряжено с такими сложностями, как в случае конечного автомата, где для этого необходимо определять переходы между новым состоянием и каждым из существующих [7].

Графически дерево поведения представляется как набор узлов, связанных линиями. У корневого узла нет родителей, и он имеет ровно один дочерний узел, которому отсылает сигнал на выполнение с определенной частотой. С этого начинается выполнение дерева: сигнал, поданный корневым узлом, по определенным связям передается через узлы потока (например, последовательности или селекторы) к узлам выполнения, не имеющим дочерних узлов и описывающим выполнение конкретных действий. На выполнение цепочек действий влияет состояние агента и окружающей среды: каждый из узлов может вернуть состояния «успех», «провал» или «выполнение». В случае провала агент должен будет выбрать другой способ достижения цели [8].

Реализация деревьев поведения предполагает разработку набора классов узлов, из которых может быть построено само дерево. Пример реализации дерева поведения приведен в [9].

Рассмотрение различных подходов к моделированию поведения агентов позволило сделать выбор в пользу применения деревьев поведения в качестве модели поведения в силу их преимуществ. С помощью деревьев может быть реализована вариативная и реалистичная модель поведения, работать с которой проще, чем с конечными автоматами.

Существующие методы машинного обучения. Машинное обучение как направление в компьютерной науке возникло в середине прошлого века. Это обширный подраздел искусственного интеллекта, изучающий методы построения алгоритмов, способных обучаться. Машинное обучение может быть применено к моделированию поведения агентов с целью создания более результативной модели поведения. Выделяют несколько разновидностей методов машинного обучения [10].

В число основных входит обучение с подкреплением. В данном методе испытуемая система (агент) обучается при взаимодействии со

средой. Такое обучение является частным случаем обучения с учителем, но в роли учителя выступает сама среда. Она реагирует на воздействия агента посредством откликов, называемых сигналами подкрепления [11]. Алгоритм этого вида обучения включает следующие шаги:

– агент получает текущее состояние системы S0 (например, информацию о положении объектов вокруг него);

– агент выполняет действие A0 (например, движение в сторону одного из объектов);

– среда переходит в некоторое новое состояние S1 (положение агента изменилось относительно объектов в среде);

– среда посылает агенту отклик R1 (некоторое вознаграждение) [12].

Цель заключается в максимизации ожидаемого вознаграждения, что приводит агента к необходимости совершения наилучшего хода или действия. Обучение с подкреплением часто используется для создания игрового искусственного интеллекта [13].

Отдельно стоит рассмотреть разновидность обучения с подкреплением – Q-обучение. Этот вид отличается дополнительной функцией полезности Q, которая помогает агенту учитывать предыдущий опыт взаимодействия со средой и в дальнейшем выбирать шаги, оказавшиеся более удачными в прошлом. Таким образом, агент может исходить из ожидаемой полезности доступных действий. Этот вид обучения часто применяется при разработке агентов для компьютерных игр [14].

Помимо обучения с подкреплением, имеет смысл рассмотреть и другие подходы, в частности, глубинное обучение. Оно часто используется в области обработки лиц и эмоций, в задачах цветокоррекции и колоризации изображений и других, а также может применяться в обработке звуковых сигналов: приложение Magenta умеет создавать музыку, а сервис Google Voice – транскрибировать голосовую почту и управлять СМС [15]. Применение глубинного обучения в задаче моделирования поведения интеллектуальных агентов в играх может быть оправданным, однако зачастую используется комбинация нескольких подходов: так, в [13] предлагается подход к реализации поведенческой модели агента на основе глубокого обучения с подкреплением.

Среди наиболее развитых направлений глубинного обучения стоит отметить глубокие нейронные сети (Deep Neural Networks) и глубинное обучение с подкреплением (Deep Q-network, или DQN) [16]. Направление DQN представляет собой вариацию обучения с подкреплением без модели с применением Q-обучения. Так, DQN использовано компанией DeepMind, которая сумела создать агента, способного обыграть действующего чемпиона мира по игре AlphaGo в марте 2016 г. В планах компании – развить своего агента, чтобы он смог играть в более сложные игры, такие как Doom и гоночные симуляторы [17].

В работах [18, 19] рассмотрена возможность применения генетических алгоритмов к задаче разработки моделей поведения интеллектуальных агентов. Для реализации алгоритма необходимо задать целевую функцию приспособленности для агентов и сгенерировать первую популяцию. Затем поколение размножается (потомки агента предыдущей популяции наследуют его черты) и мутирует (черты потомка изменяются в заранее заданных границах). Для полученной популяции агентов вновь вычисляется значение функции приспособленности. Процесс повторяется до тех пор, пока значение функции не будет соответствовать целевому. Генетические алгоритмы могут успешно использоваться для построения моделей обучения, применяемых к поведенческим моделям агентов [20, 21].

Анализ разных методов машинного обучения и основных сфер их применения позволил сделать выбор в пользу обучения с подкреплением в силу его широкой применимости в схожей сфере задач и способности обеспечить достижимость требуемого результата.

Применимость методов машинного обучения к моделированию поведения интеллектуальных агентов. В рамках данного исследования проанализировано более 400 статей из Scopus, Springer и других баз, опубликованных в 2016–2021 гг. Из них определенным образом отобрана 51 статья.

В некоторых работах авторы задаются целью натренировать агента таким образом, чтобы он смог демонстрировать человеческий уровень игры или превосходить его [22, 23]. Для этих целей могут быть использованы глубинное обучение [24], комбинация глубокого обучения с подкреплением [25, 26] и различные вариации генетических алгоритмов [27] и нейронных сетей с Q-обучением [28]. Кроме того, одним из наиболее важных и перспективных направлений, где применяются мультиагентные модели, является разработка беспи-

лотных аппаратов, для которых также применяются обучение с подкреплением [29, 30], глубокое обучение с подкреплением [31], Q-обучение [32] и другие алгоритмы.

Получено распределение статей по языкам: русскоязычные – 5 (10 %), англоязычные – 46 (90 %), а также по источникам: Springer Link – 46 (90 %), прочие источники – 5 (10 %).

Из всех рассмотренных источников более половины (53 %) описывают применение обучения с подкреплением. Внимание исследователей привлекли нейросетевой подход (21 %), глубокое обучение и генетические алгоритмы (по 14 %), Q-обучение (12 %).

Таким образом, сделан вывод, что требованиям исследования наиболее отвечает подход, сочетающий применение обучения с подкреплением в качестве алгоритма обучения и деревьев поведения в качестве модели поведения агентов.

Выбранный подход актуален в силу того, что сочетает в себе как простоту и удобство, так и широкие возможности применения к моделям поведения интеллектуальных агентов. Совместное применение связки деревьев поведения и обучения с подкреплением позволяет добиться качественных результатов в силу возможностей как деревьев поведения, так и выбранного алгоритма обучения. В отличие от методов, сосредоточенных на более сложных и громоздких алгоритмах обучения, выбранный подход позволяет добиться схожих по качеству результатов, затрачивая меньше времени и усилий на разработку. За счет применения комбинации деревьев поведения и алгоритмов обучения с подкреплением становится доступным расширение возможностей существующих деревьев поведения и значительно повышается эффективность функционирования агентов, способных обучаться.

Модельная среда для исследования поведения интеллектуальных агентов

Задача настоящего исследования подразумевает разработку собственной программной платформы, необходимой для моделирования.

Один из подходов к реализации программной платформы предполагает программирование окружения с нуля, используя только базовые программные средства: IDE и выбранный язык программирования. Такой подход редко применяется в игровой индустрии, и его наиболее существенными недостатками являются чрезмерная трудоемкость и затраты по времени.

Существует также подход, заключающийся в использовании игрового движка. Он применяется в индустрии гораздо чаще и обладает существенными преимуществами: нет необходимости в разработке базового функционала и программных зависимостей. В большинстве современных игровых движков достаточно подключить один плагин, чтобы добавить в свой проект гравитацию или другие особенности, в то время как в собственноручно разработанной системе пришлось бы тратить часы на создание такого функционала. Среди недостатков можно выделить необходимость в тщательном выборе движка для работы и изучении его документации перед ее началом.

В открытом доступе можно найти множество игровых движков, поэтому задача выбора сводилась к формированию критериев, по которым можно было бы его осуществить. Среди требований – проприетарность, невысокий порог вхождения и возможность работы с большим количеством агентов.

В результате отбора по выделенным критериям остаются два наиболее популярных игровых движка: Unity3D и Unreal Engine.

Следует отметить, что Unity больше подходит для любителей, в то время как Unreal Engine предназначен для профессионалов и требует более углубленного изучения [33, 34]. Было принято решение использовать Unity, поскольку он эффективнее справляется с задачей одновременной работы с большим количеством агентов, а также обладает встроенным фреймворком для машинного обучения Unity ML.

Unity ML – это библиотека, разрабатываемая командой Unity. В комплекте с ней поставляется набор реализаций различных алгоритмов обучения на базе TensorFlow. В работе [35] описывается использование библиотеки в практических целях и содержатся рекомендации по внедрению SDK ML-Agents в проект.

Для функционирования агентов и возможности демонстрировать их способности необходимо разработать тестовую платформу, в пределах которой агенты могут свободно перемещаться и контактировать с окружением. Основные задачи – выделение структурных особенностей окружения и его наполнение.

В качестве основных характеристик проектируемого соревновательного окружения (далее – арена) приняты следующие решения: арена должна иметь квадратную форму и ограничиваться непроницаемой стеной; внутри арены не должно быть объектов, затрудняющих передвижение агентов.

Для наполнения арены были предложены категории объектов: пища, вода, аптечки и объекты сбора. Для каждого вида объектов обозначены индивидуальные правила применимости.

В зависимости от вида пищи (яблоко или хот-дог) при поглощении агент сможет восстановить 30 или 50 очков сытости. Вода восполняет 30 очков гидратации. Аптечки восполняют 50 очков здоровья. Объекты сбора разделены на четыре вида в зависимости от ценности (1, 3, 5 и 10 очков) и являются основной целью сбора агентов.

Расположение объектов на платформе должно подчиняться определенным законам. Для каждого вида объектов задается своя политика размещения.

Вся пища размещается случайно в пределах пространства арены, однако размещение зависит от ценности конкретного вида пищи (более ценная пища ближе к центру). Вода и аптечки располагаются хаотично по всей площади арены, а объекты сбора – тем ближе к центру, чем выше их ценность. За счет таких законов достигается высокая конкуренция ближе к центру арены, так как в этой области присутствует максимальное количество наиболее ценных объектов сбора.

После того как объект собран агентом, запускается последовательность действий по восполнению запаса этого объекта. В зависимости от ценности объекта пауза между его исчезновением и повторным появлением должна меняться: чем выше ценность размещаемого объекта, тем реже он должен появляться в пространстве арены. По истечении времени паузы объект должен быть размещен повторно в соответствии со своим законом размещения.

В центре арены должна располагаться опасная зона определенного радиуса. Находясь в ее пределах, агент теряет некоторое число очков здоровья в секунду. Размещение опасной зоны должно влиять на поведение агентов.

Реализация программной платформы средствами Unity включала в себя постройку арены по заданным правилам. Для всех необходимых объектов найдены 3D-модели на сайте Turbo-Squid в свободном доступе. Вид арены, наполненной всеми видами объектов, представлен на рисунке 1.

Разработка модели поведения интеллектуальных агентов

Разработка базового функционала агента. Наиболее важными для моделирования являются принадлежность агента к одному из архетипов (умный, осторожный, сбалансированный, рискованный), показатели сытости и гидратации, показатель здоровья и количество очков агента. Для показателей здоровья, сытости и жажды вводятся пороговые значения: если показатель опустился ниже порога, агент переключает приоритет на восполнение этого показателя. Пороговые значения различны для разных архетипов: осторожный тип агентов обладает самыми высокими пороговыми значениями (50 % от максимума: такой порог позволяет агенту реже оказываться в критической ситуации, когда какой-либо из показателей слишком мал и рядом нет ресурса для его пополнения), рискованная группа обладает сниженными пороговыми значениями (10–15 %: такой порог позволяет агенту больше концентрироваться на сборе объектов и реже переключаться на задачу поиска ресурсов), сбалансированный и умный типы агентов имеют средние пороговые значения (30 %: такой порог позволяет агенту расходовать время сбалансированно на задачу поиска объектов сбора и задачу поддержания собственной жизнеспособности).

Для функционирования агента необходим набор основных действий: детектирование объектов и определение их класса, перемещение к объектам, сбор объектов и контроль основных показателей агента.

Зона обнаружения объектов агента (поле зрения) реализована с помощью коллайдера, расположенного перед агентом. Каждый объект, попавший в поле зрения, оценивается агентом на предмет полезности в текущий момент времени. Если в настоящий момент объект счи-

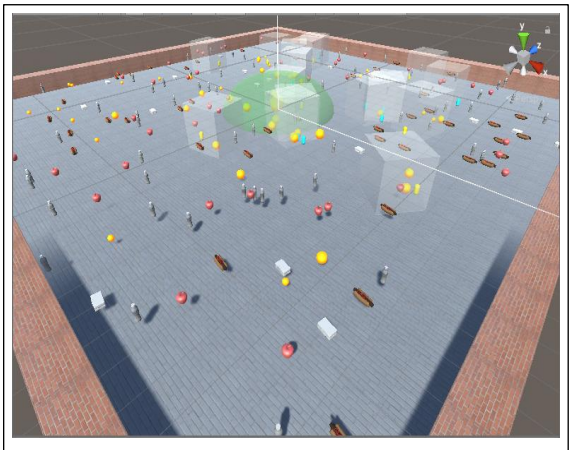

Рис. 1. Вид наполненной арены

Fig. 1. A view of the filled arena

тается полезным (например, еда в случае, если агент испытывает голод), агент приближается и собирает его.

Разработка модели поведения агента. Авторы приняли решение о самостоятельной разработке интерфейса деревьев поведения. Были определены основные необходимые типы узлов, из которых может быть построено дерево: корневой узел, базовый узел, селектор и секвенция. Корневой узел начинает выполнение дерева, с помощью селекторов и секвенций можно реализовать ветвления и последовательности действий, а на основе базового узла – конкретные действия агента.

Дерево поведения агента должно зависеть от его архетипа. Последовательность и приоритет действий должны различаться в зависимости от принадлежности агента к тому или иному архетипу.

Основное влияние на выполняемые агентом действия на начальном этапе оказывает нахождение агента в одном из критических состояний: голод, жажда или слабое здоровье. На основе факта нахождения в одном из этих состояний происходит выбор приоритета действий агента: поиск пищи, воды или аптечки. Если агент не находится ни в одном из этих состояний, он может переходить к поиску объектов сбора. Для проверки этих состояний были построены дополнительные узлы. Отдельные действия, необходимые для корректного функционирования агента, было решено представить также в виде узлов дерева. В число таких действий вошли перемещение, генерация случайной точки для движения к ней, поиск, подбор и поглощение объектов.

Общий вид модели поведения представлен на рисунке 2. Действия расположены сверху вниз в порядке убывания приоритета: специфика функционирования узла-селектора предполагает последовательную проверку необходимости выполнения каждого из дочерних действий сверху вниз. Таким образом, максимальный приоритет отдается заботе о собственном здоровье, а если такая необходимость в данный момент отсутствует, агент переходит к выполнению следующего по убыванию приоритета действия.

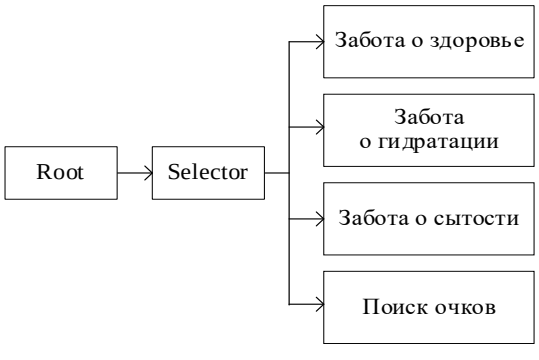

Рис. 2. Общий вид модели поведения (здесь и далее Root – корневой узел дерева, Selector – узел выбора одного из дочерних узлов)

Fig. 2. A general view of the behavior model

Было решено каждой из рассматриваемых популяций присвоить определенный цвет, что позволит их различать. Осторожные агенты окрашиваются в зеленый цвет, сбалансированные в желтый, рискованные в красный. Соответствующими цветами на рисунке 3 представлены детали реализации каждой модели поведения: цвет блока характерен только для популяции, которой присвоен тот же цвет. Бесцветные блоки являются общими для всех популяций.

Осторожный агент делает дополнительную проверку попадания в опасную зону: максимальный приоритет отдается покиданию ее пределов. Рискованный агент отличается доработанным алгоритмом поиска объектов сбора: приоритет отдается поиску наиболее ценных объектов в центре опасной зоны.

Задачу поиска объекта ввиду ее масштабности было решено выделить в отдельное дерево поведения. Это дерево связано с общим деревом через входной параметр: на вход дерева поиска подается объект, который необходимо найти.

Задача поиска объекта разбита на две ветви. Если объект найден поблизости от агента, подобрать его. Если вблизи найти искомый объект не удалось, агент будет генерировать случайные точки и двигаться к ним, осматриваясь, пока не заметит в поле зрения цель поиска, после чего подойдет и подберет требуемый объект. Графически дерево поиска представлено на рисунке 4.

Интеграция алгоритма машинного обучения

Разработка умной популяции агентов представляет собой отдельную задачу, связанную с необходимостью применения рассмотренных ранее методов машинного обучения. В качестве выбранного алгоритма машинного обучения рассматривается обучение с подкреплением. Его схема приведена на рисунке 5.

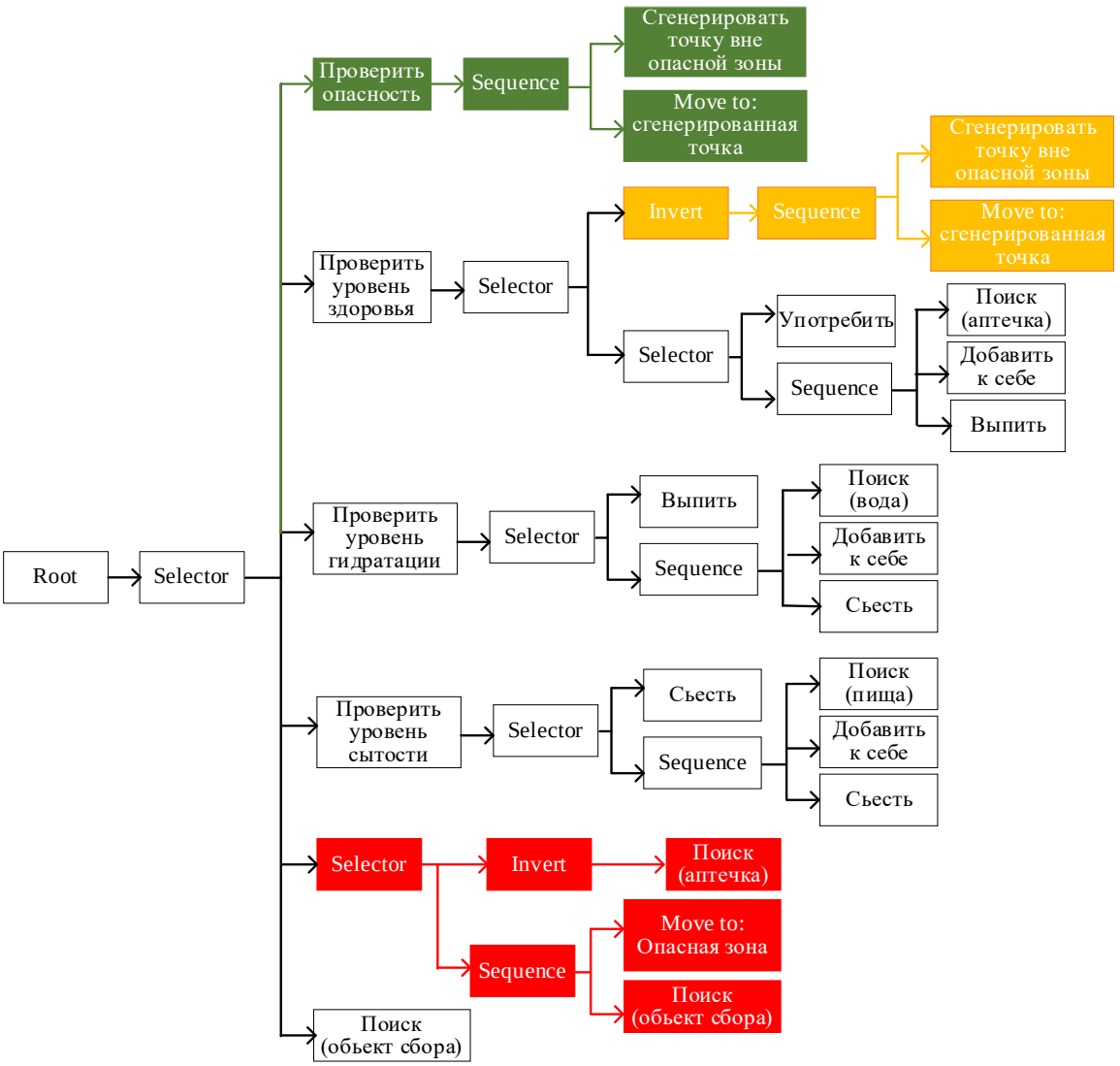

Рис. 3. Обобщенное дерево поведения (здесь и далее Sequence – узел последовательного выполнения всех дочерних узлов, Move to – действие передвижения к заданной точке)

Fig. 3. A generalized behavior tree

В роли окружающей среды для агента выступает арена, наполненная объектами сбора, пищей, водой и аптечками. Каждое действие, совершаемое агентом для достижения определенной цели, должно вознаграждаться средой.

Разрабатываемая модель обучения состоит из двух подмоделей, направленных на разные цели: жизнеобеспечение и сбор очков. Цель подмодели жизнеобеспечения – максимизация времени жизни агента, подмодели сбора очков – максимизация полученного вознаграждения. Взаимодействие подмоделей должно обеспечить достижение агентом обеих целей: дольше прожить и набрать больше очков. Для обеспечения взаимодействия необходимо подобрать верные коэффициенты обучения.

Подмодель жизнеобеспечения представляет следующие награды за действия агента:

– подбор пищи, воды или аптечки: 0.5 (необходимо для поддержания жизнеспособности в условиях потребности в ресурсе для восполнения показателя);

– подбор пищи, воды или аптечки, когда агент еще в них не нуждается: 0.7 (должно побуждать агента иметь ресурс про запас, чтобы в случае необходимости применить его сразу и не тратить время на поиск);

– каждая секунда выживания: 0.01 (побуждает агента максимизировать время собственной жизни);

– падение здоровья ниже 20 %: –0.1 в секунду (побуждает агента поддерживать высокий уровень здоровья);

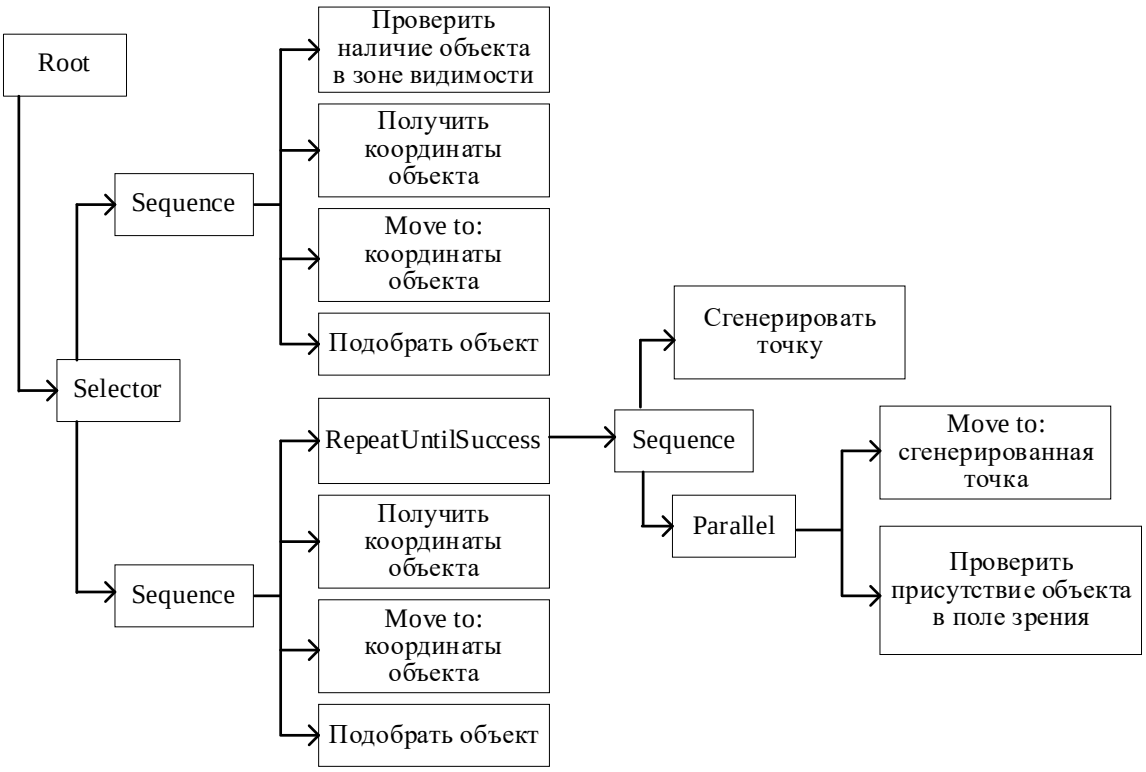

Рис. 4. Дерево поиска (RepeatUntilSuccess – узел, выполняющий действия дочерних узлов, пока не будет возвращено значение «успех», Parallel – узел, осуществляющий параллельное выполнение дочерних узлов)

Fig. 4. A search tree

– смерть: –2 (побуждение агента любым возможным способом избегать смерти).

Такой подбор коэффициентов подталкивает агента разумно распоряжаться ресурсами и прожить как можно дольше. Агент будет стараться запастись критическими для выживания вещами до наступления необходимости в них.

Подмодель сбора очков представляет следующие награды:

– подбор объекта стоимостью 1: 0.5;
– подбор объекта стоимостью 3: 0.7;
– подбор объекта стоимостью 5: 0.9;
– подбор объекта стоимостью 10: 1.3 (чем ценнее объект – тем выше награда: побуждает агента подбирать наиболее ценные награды из тех, что удается найти);
– каждая секунда в режиме поиска: –0.05 (если не удается найти более ценные объекты, агент должен подбирать менее ценные, а не тратить время впустую);
– каждая секунда в опасной зоне: –0.1 (побуждает агента проводить меньше времени в опасной зоне: если не удается найти наиболее ценные объекты, стоит покинуть зону).

Такие значения коэффициентов подталкивают агента искать более дорогие объекты сбора, но при этом не тратить лишнее время в опасной зоне и просто в режиме поиска: если не удается найти дорогой объект, можно поднять более дешевый, который находится рядом.

Модель обучения была запрограммирована на языке C# и присоединена как компонент

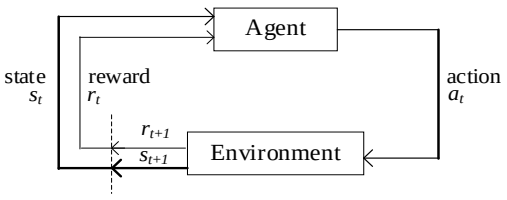

Рис. 5. Схема обучения с подкреплением (Agent – агент, Environment – среда, action – действие, осуществляемое агентом по отношению к среде, reward – награда от среды агенту, state – обновленное состояние среды и агента)

Fig. 5. A reinforcement learning scheme

агента в Unity. После настройки модели запущен процесс обучения с помощью связки Unity ML и Python.

Оценка качества построенной модели по результатам вычислительного эксперимента

В рамках эксперимента умная популяция сталкивалась с каждой из оставшихся. Победитель определялся по количеству набранных популяцией очков за отведенное время: этот же показатель выбирался в качестве критерия оценки эффективности популяции.

Для обеспечения объективности эксперимента был реализован механизм настройки модельной среды, с помощью которого можно установить количество размещаемых объектов сбора, еды, воды и аптечек, а также время эксперимента. Для каждой пары соперников проводилась серия матчей в разных условиях ради повышения объективности результата.

Сценарий эксперимента включал в себя три серии по три матча. В каждой серии умная популяция соревновалась с одной из оставшихся. Матчи одной серии отличались друг от друга начальными условиями: количеством размещаемых объектов, темпом их восполнения и ограничением на время матча. Первый матч проходил в условиях изобилия ресурсов и имел большую длительность, для второго матча были настроены средние показатели, третий проходил в условиях дефицита ресурсов и крайне ограниченного времени.

В первой серии матчей умная популяция соревновалась с осторожной. В первом матче ввиду изобилия ресурсов обе команды набрали много очков, умная популяция одержала верх за счет способности добывать наиболее дорогие ресурсы из опасной зоны, куда осторожные агенты не могут заходить. Во втором матче умная популяция также одержала победу за счет оптимизации процесса поиска ресурсов: умные агенты тратят основную часть времени на поиск объектов сбора, а не аптечек, как осторожные. В третьем матче фокус осторожной популяции был смещен в сторону выживания, однако получить преимущество по очкам ей не удалось.

Во второй серии матчей умная популяция соревновалась со сбалансированной. Результаты команд во всех матчах оказались схожими ввиду похожего алгоритма поиска ресурсов и определения приоритета действий. В условиях избытка ресурсов (матч № 1) победу одержала сбалансированная популяция с небольшим отрывом, однако в остальных матчах уверенная победа была за умной популяцией в силу более оптимального алгоритма посещения опасной зоны и переключения приоритетов с жизнеспособности на сбор ресурсов и обратно.

В третьей серии матчей умная популяция соревновалась с рискованной. Матч в условиях избытка ресурсов показал уверенное преимущество рискованных агентов, чье постоянное пребывание в опасной зоне привело к быстрому набору очков за счет сбора наиболее ценных ресурсов. В других же матчах стало очевидным, что умная популяция обладает преимуществом перед рискованной, посещая опасную зону лишь изредка в отличие от рискованных агентов, которые находились в ней практически постоянно даже тогда, когда все ресурсы в ней собраны. В силу этого рискованные агенты имели существенно меньшую продолжительность жизни и в долгосрочной перспективе набрали меньше очков.

Результаты всех серий экспериментов представлены в таблице (победа, если умная популяция победила, или поражение, если проиграла).

Соревнования проводились в идентичных для обоих соперников условиях, за счет чего достигнута объективность эксперимента. Умная популяция одержала победу в 78 % случаев (7 из 9 матчей), что доказывает эффективность применения алгоритмов машинного обучения в поставленной задаче.

Заключение

Цель настоящего исследования – разработка теоретических основ и инструментальных программных средств для моделирования поведения интеллектуальных агентов в игровой среде и повышения эффективности их функционирования.

Был проведен анализ существующих подходов к организации моделей поведения интеллектуальных агентов и методов машинного обучения, которые могут быть к ним применены, а также спроектированы и разработаны необходимые модели поведения и обучающий алгоритм.

Разработка программной платформы позволила добиться следующих результатов.

- Полностью функциональная программная платформа на игровом движке Unity позволяет устраивать матчи между популяциями интеллектуальных агентов в идентичных условиях, что может быть полезно для проверки эф-

Результаты серий экспериментов

Results of a series of experiments

Номер матча	Соперник	Количество ресурсов	Темп восполнения ресурсов, 1/мин.	Время симуляции, мин.	Результат
Первая серия					
1	Осторожный	100	2	5	Победа
2	Осторожный	50	1	3	Победа
3	Осторожный	10	0,5	1	Победа
Вторая серия					
1	Сбалансированный	100	2	5	Поражение
2	Сбалансированный	50	1	3	Победа
3	Сбалансированный	10	0,5	1	Победа
Третья серия					
1	Рискованный	100	2	5	Поражение
2	Рискованный	50	1	3	Победа
3	Рискованный	10	0,5	1	Победа

фективности функционирования определенных моделей поведения или новых подходов к моделированию.

- Платформа способна обеспечивать возможность автономного функционирования двух популяций интеллектуальных агентов в течение длительного времени путем обеспечения постоянного восполнения объектов сбора, пищи, воды и аптечек.
- Разработаны четыре модели поведения для тестовых прогонов на базе созданной платформы. Каждая из моделей обладает своими уникальными параметрами и способна демонстрировать поведение, отличное от других моделей.
- Одна из разработанных моделей поведения снабжена обучающими алгоритмами, что позволило получить новую модель, существенно отличающуюся от разработанных хардкод-моделей.

Вычислительный эксперимент на базе разработанного ПО включал в себя проведение серии матчей, в которых агенты, использующие алгоритмы машинного обучения, соревновались с агентами других популяций в задаче набора очков в условиях ограниченного времени.

Эксперимент показал, что применение методов машинного обучения к моделированию поведения интеллектуальных агентов может существенно повысить эффективность их функционирования и позволит им иметь преимущество над другими способами организации игрового искусственного интеллекта, демонстрируя достаточно разнообразное и реалистичное поведение.

В перспективе планируются исследования с использованием других подходов к организации игрового искусственного интеллекта на базе разработанной платформы.

Исследование выполнено за счет гранта Российского научного фонда и Администрации Волгоградской области № 22-11-20024, https://rscf.ru/project/22-11-20024/. Результаты части «Модельная среда для исследования поведения интеллектуальных агентов» получены в рамках гранта Российского научного фонда (РНФ, проект № 20-71-10087).

Литература

1. Parygin D., Usov A., Burov S., Sadovnikova N., Ostroukhov P., Pyannikova A. Multi-agent approach to modeling the dynamics of urban processes (on the example of urban movements). Communications in Computer and Information Science. Proc. Int. Conf. EGOSE, 2019, vol. 1135, pp. 243–257. DOI: 10.1007/978-3-030-39296-3_18.

2. Бурова А.А., Буров С.С., Парыгин Д.С., Финогеев А.Г., Смирнова Т.В. Панель администрирования платформы многоагентного моделирования с возможностью построения графических отчетов // Int. J. of Open Information Technologies. 2021. Т. 9. № 12. С. 4–14.

3. Парыгин Д.С., Голубев А.В., Буров С.С., Анохин А.О., Финогеев А.Г. Платформа для моделирования массовых перемещений объектов и субъектов в условиях городской среды // Программные продукты и системы. 2021. Т. 34. № 2. С. 354–364. DOI: 10.15827/0236-235X.134.354-364.

4. Anokhin A., Burov S., Parygin D., Rent V., Sadovnikova N., Finogeev A. Development of scenarios for modeling the behavior of people in an urban environment. studies in systems, decision and control. In: Society 5.0: Cyberspace for Advanced Human-Centered Society, 2021, vol. 333, pp. 103–114. DOI: 10.1007/978-3-030-63563-3_9.

5. Анохин А.О., Садовникова Н.П., Катаев А.В., Парыгин Д.С. Моделирование поведения агентов для реализации игрового искусственного интеллекта // Прикаспийский журнал: управление и высокие технологии. 2020. № 2. С. 85–99. DOI: 10.21672/2074-1707.2020.50.2.096-110.

6. Orkin J. Applying Goal-Oriented Action Planning to Games. URL: http://alumni.media.mit.edu/~jorkin/GOAP_draft_AIWisdom2_2003.pdf (дата обращения: 05.09.2022).

7. OPSIVE. Behavior Trees or Finite State Machines. URL: https://opsive.com/support/documentation/behavior-designer/behavior-trees-or-finite-state-machines/ (дата обращения: 25.08.2022).

8. Behavior Trees for AI: How They Work. URL: https://www.gamasutra.com/blogs/ChrisSimpson/20140717/221339/Behavior_trees_for_AI_How_they_work.php (дата обращения: 15.08.2022).

9. Building Your Own Basic Behavior Tree in Unity. URL: https://hub.packtpub.com/building-your-own-basic-behavior-tree-tutorial/ (дата обращения: 11.08.2022).

10. Ray S. A quick review of machine learning algorithms. Proc. COMITCon, 2019, pp. 35–39. DOI: 10.1109/COMITCon.2019.8862451.

11. Botvinick M., Ritter S., Wang J.X., Kurth-Nelson Z., Blundell Ch., Hassabis D. Reinforcement learning, fast and slow. Trends in Cognitive Sciences, 2019, vol. 23, no. 5, pp. 408–422. DOI: 10.1016/j.tics.2019.02.006.

12. Yang Y., Wang J. An overview of multi-agent reinforcement learning from game theoretical perspective. ArXiv, 2020, art. 2011.00583. URL: https://arxiv.org/abs/2011.00583v1 (дата обращения: 16.08.2022).

13. How to Teach AI to Play Games: Deep Reinforcement Learning. URL: https://towardsdatascience.com/how-to-teach-an-ai-to-play-games-deep-reinforcement-learning-28f9b920440a (дата обращения: 17.08.2022).

14. Jang B., Kim M., Harerimana G., Kim J.W. Q-learning algorithms: A comprehensive classification and applications. IEEE Access, 2019, vol. 7, pp. 133653–133667. DOI: 10.1109/ACCESS.2019.2941229.

15. Глубинное обучение: возможности, перспективы и немного истории. URL: https://habr.com/company/it-grad/blog/309024/ (дата обращения: 18.08.2022).

16. An Introduction to Deep Q-Learning: Let's Play Doom. URL: https://medium.freecodecamp.org/an-introduction-to-deep-q-learning-lets-play-doom-54d02d8017d8 (дата обращения: 31.08.2022).

17. Human-Level Control Through Deep Reinforcement Learning. URL: https://deepmind.com/research/dqn/ (дата обращения: 18.08.2022).

18. Гладков Л.А., Курейчик В.В., Курейчик В.М. Генетические алгоритмы. Ростов-на-Дону: Рост-Издат, 2006. 367 с.

19. Mirjalili S. Genetic algorithm. In: SCI, 2019, pp. 43–55. DOI: 10.1007/978-3-319-93025-1_4.

20. Жаравин Д.Е., Козин Д.Ю., Фомичев Д.Ю., Федотовский С.Б. Использование генетических алгоритмов для обучения искусственной нейронной сети // Вестн. ВоГУ. Сер.: Технические науки. 2019. № 2. С. 41–43.

21. Трокоз Д.А. Алгоритм машинного обучения широких нейронных сетей с использованием алгебры гиперразмерных двоичных векторов и генетических алгоритмов // Южно-Сибирский научн. вестн. 2020. № 6. С. 148–154.

22. Jaderberg M., Czarnecki W.M., Dunning I., Marris L. et al. Human-level performance in 3D multiplayer games with population-based reinforcement learning. Science, 2019, vol. 364, no. 6443, pp. 859–865.

23. Ye D., Liu Z., Sun M., Shi B. et al. Mastering complex control in moba games with deep reinforcement learning. Proc. AAAI Conf. on Artificial Intelligence, 2020, vol. 34, no. 04, pp. 6672–6679. DOI: 10.1609/AAAI.V34I04.6144.

24. Gudmundsson S.F., Eisen Ph., Poromaa E., Nodet A. et al. Human-like playtesting with deep learning. Proc. IEEE Conference on CIG, 2018, pp. 1–8. DOI: 10.1109/CIG.2018.8490442.

25. Brown N., Bakhtin A., Lerer A., Gong Q. Combining deep reinforcement learning and search for imperfect-information games. Advances in Neural Information Processing Systems, 2020, vol. 33, pp. 17057–17069.

26. Torrado R.R., Bontrager Ph., Togelius J., Liu J., Pérez-Liébana D. Deep reinforcement learning for general video game AI. Proc. IEEE Conf. on CIG, 2018, pp. 1–8. DOI: 10.1109/CIG.2018.8490422.

27. Jin S. Research and application of multi-agent genetic algorithm in tower defense game. AIP Conf. Proc., 2018, vol. 1955, no. 1, art. 040138. DOI: 10.1063/1.5033802.

28. Zhou M., Chen Y., Wen Y., Yang Y. et al. Factorized q-learning for large-scale multi-agent systems. Proc. I Int. Conf. on Distributed Artificial Intelligence, 2019, pp. 1–7. DOI: 10.1145/3356464.3357707.

29. Qie H., Shi D., Shen T., Xu X., Li Y., Wang L. Joint optimization of multi-UAV target assignment and path planning based on multi-agent reinforcement learning. IEEE Access, 2019, vol. 7, pp. 146264–146272. DOI: 10.1109/access.2019.2943253.

30. Wenhong Z., Li J., Liu Z., Shen L. Improving multi-target cooperative tracking guidance for UAV swarms using multi-agent reinforcement learning. CJA, 2022, vol. 35, no. 7, pp. 100–112. DOI: 10.1016/j.cja. 2021.09.008.

31. Zhang Y., Zhuang Z., Gao F., Wang J., Han Z. Multi-agent deep reinforcement learning for secure UAV communications. Proc. IEEE WCNC, 2020, pp. 1–5. DOI: 10.1109/WCNC45663.2020.9120592.

32. Zhao W., Fang Z., Yang Z. Four-dimensional trajectory generation for UAVs based on multi-agent Q learning. J. of Navigation, 2020, vol. 73, no. 4, pp. 874–891. DOI: 10.1017/S0373463320000016.

33. Unreal Engine. URL: https://www.unrealengine.com/en-US (дата обращения: 25.08.2022).

34. Движок Unreal Engine – особенности, преимущества и недостатки. URL: https://cubiq.ru/dvizhok-unreal-engine/ (дата обращения: 25.08.2022).

35. Juliani A., Berges V.-P., Teng E. Unity: a general platform for intelligent agents. ArXiv, 2018, art. 1809.02627. URL: https://arxiv.org/abs/1809.02627 (дата обращения: 25.08.2022).

Software & Systems
DOI: 10.15827/0236-235X.141.046-059

Received 15.09.22, Revised 19.12.22
2023, vol. 36, no. 1, pp. 046–059

Modeling the intelligent agent behavior based on machine learning methods in competition models

A.O. Anokhin [1], *Postgraduate Student, alex.anokhin.st@gmail.com*
D.S. Parygin [1], *Ph.D. (Engineering), Associate Professor, dparygin@gmail.com*
N.P. Sadovnikova [1], *Dr.Sc. (Engineering), Professor, npsn1@ya.ru*
A.A. Finogeev [2], *Ph.D. (Engineering), Associate Professor, fanton3@ya.ru*
A.S. Gurtyakov [1], *Ph.D. (Engineering), Associate Professor, agurtyakov@gmail.com*

[1] *Volgograd State Technical University, Volgograd, 400005, Russian Federation*
[2] *Penza State University, Penza, 440026, Russian Federation*

Abstract. The article discusses the aspects of applying machine learning methods to existing methods for modeling the behavior of intelligent agents to enable agents to improve their performance in competition models.

The practical significance of the study is represented by developing an approach to modeling the behavior of intelligent agents in order to increase the efficiency of their functioning in such areas as computer games, developing unmanned aerial vehicles and search robots, studying urban and transport mobility, as well as other complex systems.

There is a review of the existing machine learning methods (reinforcement learning, deep learning, Q-learning) and methods for modeling the agents' behavior (a rule-based model, a finite automaton model of behavior, behavior trees). The authors have chosen the most appropriate combination of a learning method and a behavior model for the task: behavior trees and reinforcement learning.

A test platform was implemented using Unity tools, behavior models were developed for the four main archetypes of agents that must compete in collecting resources in a limited time. A trained agent was implemented using Unity ML and TensorFlow tools.

The test platform has become a basis for a series of experiments under various conditions: limited resources, resource abundance, average amount of resources. As part of the experiment, the authors tested the ability of the developed intelligent agent's behavior model to win in a competitive environment with agents equipped with various variants of traditional behavior models based on behavior trees. The efficiency and advantages of using the developed behavior model were evaluated. The paper analyzes the experimental results and draws conclusions regarding the potential of the selected combination of methods.

Keywords: behavior modeling, intelligent agent, artificial intelligence, machine learning, behavior tree.

Acknowledgements. *The study has been supported by the grant from the Russian Science Foundation (RSF) and the Administration of the Volgograd Oblast (Russia) no. 22-11-20024, https://rscf.ru/en/project/22-11-20024/. The results of part "Model environment for studying the behavior of intelligent agents" were obtained within the Russian Science Foundation (RSF) grant (project no. 20-71-10087).*

References

1. Parygin D., Usov A., Burov S., Sadovnikova N., Ostroukhov P., Pyannikova A. Multi-agent approach to modeling the dynamics of urban processes (on the example of urban movements). *Communications in Computer and Information Science. Proc. Int. Conf. EGOSE*, 2019, vol. 1135, pp. 243–257. DOI: 10.1007/978-3-030-39296-3_18.
2. Burova A.A., Burov S.S., Parygin D.S., Finogeev A.G., Smirnova T.V. Administration panel of the multi-agent modeling platform with the ability to build graphical reports. *Int. J. of Open Information Technologies*, 2021, vol. 9, no. 12, pp. 4–14 (in Russ.).
3. Parygin D.S., Golubev A.V., Burov S.S., Anokhin A.O., Finogeev A.G. The platform for modeling large-scale movement of objects and subjects in an urban environment. *Software & Systems*, 2021, vol. 34, no. 2, pp. 354–364. DOI: 10.15827/0236-235X.134.354-364 (in Russ.).
4. Anokhin A., Burov S., Parygin D., Rent V., Sadovnikova N., Finogeev A. Development of scenarios for modeling the behavior of people in an urban environment. studies in systems, decision and control. In: *Society 5.0: Cyberspace for Advanced Human-Centered Society*, 2021, vol. 333, pp. 103–114. DOI: 10.1007/978-3-030-63563-3_9.
5. Anokhin A.O., Sadovnikova N.P., Kataev A.V., Parygin D.S. Modeling of agents behavior to implement gaming artificial intelligence. *Caspian J.: Control and High Technologies*, 2020, no. 2, pp. 85–99. DOI: 10.21672/2074-1707.2020.50.2.096-110 (in Russ.).
6. Orkin J. *Applying Goal-Oriented Action Planning to Games*. Available at: http://alumni.media.mit.edu/~jorkin/GOAP_draft_AIWisdom2_2003.pdf (accessed September 05, 2022).
7. *OPSIVE. Behavior Trees or Finite State Machines*. Available at: https://opsive.com/support/documentation/behavior-designer/behavior-trees-or-finite-state-machines/ (accessed August 25, 2022).
8. *Behavior Trees for AI: How They Work*. Available at: https://www.gamasutra.com/blogs/ChrisSimpson/20140717/221339/Behavior_trees_for_AI_How_they_work.php (accessed August 15, 2022).
9. *Building Your Own Basic Behavior Tree in Unity.* Available at: https://hub.packtpub.com/building-your-own-basic-behavior-tree-tutorial/ (accessed August 11, 2022).
10. Ray S. A quick review of machine learning algorithms. *Proc. COMITCon*, 2019, pp. 35–39. DOI: 10.1109/COMITCon.2019.8862451.
11. Botvinick M., Ritter S., Wang J.X., Kurth-Nelson Z., Blundell Ch., Hassabis D. Reinforcement learning, fast and slow. *Trends in Cognitive Sciences*, 2019, vol. 23, no. 5, pp. 408–422. DOI: 10.1016/j.tics.2019.02.006.
12. Yang Y., Wang J. An overview of multi-agent reinforcement learning from game theoretical perspective. *ArXiv*, 2020, art. 2011.00583. Available at: https://arxiv.org/abs/2011.00583v1 (accessed August 16, 2022).
13. *How to Teach AI to Play Games: Deep Reinforcement Learning*. Available at: https://towardsdatascience.com/how-to-teach-an-ai-to-play-games-deep-reinforcement-learning-28f9b920440a (accessed August 17, 2022).
14. Jang B., Kim M., Harerimana G., Kim J.W. Q-learning algorithms: A comprehensive classification and applications. *IEEE Access*, 2019, vol. 7, pp. 133653–133667. DOI: 10.1109/ACCESS.2019.2941229.
15. *Deep Learning: Opportunities, Perspectives, and a Bit of History*. Available at: https://habr.com/company/it-grad/blog/309024/ (accessed August 18, 2022) (in Russ.).
16. *An Introduction to Deep Q-Learning: Let's Play Doom*. Available at: https://medium.freecodecamp.org/an-introduction-to-deep-q-learning-lets-play-doom-54d02d8017d8 (accessed August 31, 2022).
17. *Human-Level Control Through Deep Reinforcement Learning*. Available at: https://deepmind.com/research/dqn/ (accessed August 18, 2022).
18. Gladkov L.A., Kureychik V.V., Kureychik V.M. *Genetic Algorithms*. Rostov-on-Don, 2006, 367 p. (in Russ.).
19. Mirjalili S. Genetic algorithm. In: *SCI*, 2019, pp. 43–55. DOI: 10.1007/978-3-319-93025-1_4.
20. Zharavin D.E., Kozin D.Yu., Fomichev D.Yu., Fedotovsky S.B. Using genetic algorithms for training an artificial neural network. *Bull. of the VoSU. Ser.: Tech. Sci.*, 2019, no. 2, pp. 41–43 (in Russ.).
21. Trokoz D.A. Algorithm of machine learning wide neural networks using the algebra of hyperdimensional binary vectors and genetic algorithms. *South-Siberian Sci. Bull.*, 2020, no. 6, pp. 148–154 (in Russ.).
22. Jaderberg M., Czarnecki W.M., Dunning I., Marris L. et al. Human-level performance in 3D multiplayer games with population-based reinforcement learning. *Science*, 2019, vol. 364, no. 6443, pp. 859–865.
23. Ye D., Liu Z., Sun M., Shi B. et al. Mastering complex control in moba games with deep reinforcement learning. *Proc. AAAI Conf. on Artificial Intelligence*, 2020, vol. 34, no. 04, pp. 6672–6679. DOI: 10.1609/AAAI.V34I04.6144.

24. Gudmundsson S.F., Eisen Ph., Poromaa E., Nodet A. et al. Human-like playtesting with deep learning. *Proc. IEEE Conference on CIG*, 2018, pp. 1–8. DOI: 10.1109/CIG.2018.8490442.

25. Brown N., Bakhtin A., Lerer A., Gong Q. Combining deep reinforcement learning and search for imperfect-information games. *Advances in Neural Information Processing Systems*, 2020, vol. 33, pp. 17057–17069.

26. Torrado R.R., Bontrager Ph., Togelius J., Liu J., Pérez-Liébana D. Deep reinforcement learning for general video game AI. *Proc. IEEE Conf. on CIG*, 2018, pp. 1–8. DOI: 10.1109/CIG.2018.8490422.

27. Jin S. Research and application of multi-agent genetic algorithm in tower defense game. *AIP Conf. Proc.*, 2018, vol. 1955, no. 1, art. 040138. DOI: 10.1063/1.5033802.

28. Zhou M., Chen Y., Wen Y., Yang Y. et al. Factorized q-learning for large-scale multi-agent systems. *Proc. I Int. Conf. on Distributed Artificial Intelligence*, 2019, pp. 1–7. DOI: 10.1145/3356464.3357707.

29. Qie H., Shi D., Shen T., Xu X., Li Y., Wang L. Joint optimization of multi-UAV target assignment and path planning based on multi-agent reinforcement learning. *IEEE Access*, 2019, vol. 7, pp. 146264–146272. DOI: 10.1109/access.2019.2943253.

30. Wenhong Z., Li J., Liu Z., Shen L. Improving multi-target cooperative tracking guidance for UAV swarms using multi-agent reinforcement learning. *CJA*, 2022, vol. 35, no. 7, pp. 100–112. DOI: 10.1016/j.cja.2021.09.008.

31. Zhang Y., Zhuang Z., Gao F., Wang J., Han Z. Multi-agent deep reinforcement learning for secure UAV communications. *Proc. IEEE WCNC*, 2020, pp. 1–5. DOI: 10.1109/WCNC45663.2020.9120592.

32. Zhao W., Fang Z., Yang Z. Four-dimensional trajectory generation for UAVs based on multi-agent Q learning. *J. of Navigation*, 2020, vol. 73, no. 4, pp. 874–891. DOI: 10.1017/S0373463320000016.

33. *Unreal Engine*. Available at: https://www.unrealengine.com/en-US (accessed August 25, 2022)..

34. *Unreal Engine – Features, Advantages and Disadvantages*. Available at: https://cubiq.ru/dvizhok-unreal-engine/ (accessed August 25, 2022) (in Russ.).

35. Juliani A., Berges V.-P., Teng E. Unity: A general platform for intelligent agents. *ArXiv*, 2018, art. 1809.02627. Available at: https://arxiv.org/abs/1809.02627 (accessed August 25, 2022).

Для цитирования

Анохин А.О., Парыгин Д.С., Садовникова Н.П., Финогеев А.А., Гуртяков А.С. Моделирование поведения интеллектуальных агентов на основе методов машинного обучения в моделях конкуренции // Программные продукты и системы. 2023. Т. 36. № 1. С. 046–059. DOI: 10.15827/0236-235X.141.046-059.

For citation

Anokhin A.O., Parygin D.S., Sadovnikova N.P., Finogeev A.A., Gurtyakov A.S. Modeling the intelligent agent behavior based on machine learning methods in competition models. *Software & Systems*, 2023, vol. 36, no. 1, pp. 046–059 (in Russ.). DOI: 10.15827/0236-235X.141.046-059.