УДК 004.89, 004.85

ТЕХНОЛОГИЯ СОЗДАНИЯ ИНТЕЛЛЕКТУАЛЬНЫХ САМООБУЧАЮЩИХСЯ СИСТЕМ УПРАВЛЕНИЯ АВТОНОМНЫМИ РОБОТАМИ

Поспелов Сергей Михайлович[1], Бондаренко Иван Юрьевич[2]

Введение

Интеллектуальные системы управления используются во многих прикладных областях, таких как компьютерные игры, автоматическое управление автомобилями, самолетами или даже роботами. Примеры внедрения этих технологий – робот-пылесос, робот-трейдер, торгующий акциями, а также беспилотный автомобиль.

Повышение уровня интеллектуальности агентов в настоящее время является одной из важнейших проблем, как для создателей автономных роботов, так и для разработчиков искусственного интеллекта в целом. Задача приспособления к незнакомой окружающей среде для выполнения поставленных целей является наиболее актуальной. Такие стандартные методы, как конечные автоматы или экспертные системы с продукционной архитектурой не слишком хорошо подходят для управления агентом в незнакомой среде, особенно если среда является нестационарной [1].

Более подходящим методом является метод обучения с подкреплением. Он позволяет автономному агенту обучаться на основе собственного опыта без вмешательства разработчика. Для управления агентом в анимат-подобной среде небольшой и средней размерности хорошо подходит один из вариантов обучения с подкреплением – алгоритм Q-обучения. В данной работе приводятся экспериментальные сравнения двух видов алгоритма Q-обучения: алгоритма интерактивного обучения и алгоритма обратного переигрывания.

Формализация задачи управления автономным агентом в анимат-подобной среде

Среда, в которой производились эксперименты с аниматом в данной работе, представлена на рис. 1. Данная среда представляет собой замкнутое пространство, в которое помещается анимат. Пространство ограничено с четырёх сторон стеной, в центре которой с каждой стороны есть углубление, содержащее «пищу». Задача анимата – найти «пищу» за наименьшее количество шагов.

На рис. 1 используются следующие условные обозначения: стена (W), «пища» (F) и пустое пространство(-).

Робот имеет 8 сенсоров, определяющих содержимое смежных с ним ячеек. Таким образом, состояние робота определяется только состоянием соседних ячеек и кодируется по ходу часовой стрелки, начиная с севера.

Если робот(R) находится в положении, изображенном на рис. 2, то он получает следующую сенсорную информацию: -----WFW.

[1] магистрант кафедры прикладной математики и информатики Донецкого национального технического университета, тел. 063 190 4000, pospielov@gmail.com

[2] ассистент кафедры прикладной математики и информатики Донецкого национального технического университета, тел. 050-425-01-61 bond005@yandex.ru

Ф

W	W	W	W	W	W	W	W	W
W	W	W	W	F	W	W	W	W
W	W	-	-	-	-	-	W	W
W	W	-	-	-	-	-	W	W
W	F	-	-	-	-	-	F	W
W	W	-	-	-	-	-	W	W
W	W	-	-	-	-	-	W	W
W	W	W	W	F	W	W	W	W
W	W	W	W	W	W	W	W	W

Рисунок 1. Анимат-подобная среда

-	-	-	W	W
-	-	-	W	W
-	-	R	F	W
-	-	-	W	W
-	-	-	W	W

Рисунок 2. Пример положения робота

Всего возможно 17 различных состояний робота, учитывая то, что 9 центральных клеток кодируются одним состоянием.

В каждый момент времени анимат может двигаться в одном из 8-ми возможных направлений: 1 – север, 2 – северо-восток , 3 – восток , 4 – юго-восток , 5 – юг, 6 – юго-запад, 7 – запад, 8 – северо-запад.

Принципы взаимодействия робота и среды таковы:

1) если ячейка, в которую перемещается робот, пустая, то среда позволяет ему переместиться в эту ячейку;
2) если ячейка, в которую перемещается робот, содержит стену, то среда не позволяет ему переместиться в эту ячейку и робот получает наказание (отрицательное вознаграждение);
3) если ячейка, в которую перемещается робот, содержит «пищу», то среда позволяет роботу в неё переместиться и робот получает вознаграждение.

Анализ различных алгоритмов Q-обучения для оптимизации управления автономным агентом.

Алгоритм Q-обучения был предложен Воткинсом (Watkins) в 1989 году [2].

Данный алгоритм работает с Q-функцией, аргументами которой являются состояние и действие. Это позволяет итерационным способом построить Q-функцию и тем самым найти оптимальную политику управления. Целью обучения является максимизация награды r. Выражение для обновления Q-функции имеет следующий вид:

$$Q(s_t, a_t) \leftarrow r_t + \gamma \cdot \max_{a \in A} Q(s_{t+1}, a)$$

Оценки Q-значений хранятся в 2-х мерной таблице, входами которой являются состояние и действие.

При использовании одношагового Q-обучения коррекция Q-значений производится на основе оценок текущего и последующего состояний, т.е. на оценках смежных состояний. Существует два вида алгоритма Q-обучения – алгоритм интерактивного обучения и алгоритм обратного переигрывания [3].

В алгоритме интерактивного обучения, подстройка Q-значений таблицы производится после каждого действия анимата. Таким образом, при одной итерации на текущее подстраивание Q-значения влияет только значение Q-фактора следующего состояния. Это может оказаться минусом, т.к. для нахождения длинных стратегий может потребоваться много итераций. Однако интерактивное обучение позволяет менять свое поведение сразу после выполнения определенных действий. Алгоритм интерактивного обучения описан на рис. 3.

1) Обследовать текущее состояние *s*
2) Повторять пока не будет достигнуто поглощающее состояние:
   - выбрать действие *a* такое что:
     $$a = \max_{a \in A} Q(s_{t+1}, a);$$
   - получить награду r;
   - обследовать новое состояние *s';*
   - обновить ячейку таблицы для пары состояние-действие (*s*-*a*) следующим образом:
     $$Q(s_t, a_t) \leftarrow r_t + \gamma \cdot \max_{a \in A} Q(s_{t+1}, a);$$
   - *s*←*s';*

Рисунок 3. Алгоритм интерактивного Q-обучения

При использовании алгоритма обратного переигрывания, обновление Q-значений производится только после того, как агент достигнет поглощающего состояния (нахождение пищи). Поэтому агент после каждого шага должен запоминать текущее состояние, выбранное действие и непосредственную награду. После достижения агентом поглощающего состояния, у него должен быть сформирован список всех совершенных переходов. Этот список прокручивается в обратном порядке и осуществляется коррекция Q-функции. Алгоритм описан на рис. 4.

Для всех сохраненных $\{(s_0, a_0, s_1, r_0)\ldots(s_n, a_n, s_{n+1}, r_n)\}$ выполнить
1. $t \leftarrow n$;
2. $Q_{real,t} \leftarrow Q(s_t, a_t)$;
3. $u_{t+1} \leftarrow Q(s_{t+1}, a_{t+1})$;
4. $Q_{target} \leftarrow r_t + \gamma * [(1-\lambda) * u_{t+1} + \lambda * Q_{real,t+1}]$;
5. Корректировка *Q*-таблицы, $Q(s_t, a_t) = Q_{target} - Q_{real}$;
6. Если $t = 0$ выход; иначе $t \leftarrow t-1$ и переход на 2-ой шаг.

Рисунок 4. Алгоритм Q-обучения с обратным переигрыванием

Ф

Алгоритм обратного переигрывания отличается от алгоритма интерактивного обучения тем, что на каждую подстройку Q-значения влияют все последующие значения Q-фактора, что позволяет находить оптимальную стратегию за меньшее количество итераций. Минусами этого алгоритма можно назвать обязательное присутствие поглощающего состояния и необходимость сохранять список переходов.

Экспериментальные исследования

Среда, в которой производятся эксперименты, является немарковской, или принадлежит к классу 2 по классификации, используемой Вильсоном в [4], т. к. данных сенсорного входа не всегда достаточно для однозначного определения действия, которое должен совершить робот в направлении ближайшей “пищи”. Это происходит потому, что в центре данной среды имеется пустое пространство, и, находясь в одной из 9-ти центральных ячеек, робот получает одинаковую сенсорную информацию.

Исследования проводились в среде, описанной в разделе «Формализация задачи управления автономным агентом в анимат-подобной среде». Для обучения использовались коэффициенты обучения $\gamma = 0{,}02$ и $\lambda = 0{,}9$.

Критерием оценки качества обучения агента будет служить сумма шагов от начальной точки до пищи с каждой клетки пространства. Минимальная такая сумма составит 46 шагов. Схема переходов при минимальном количестве шагов изображена на рис. 5. В этом случае, максимальное расстояние от начального положения до «пищи» составит 5 шагов.

В случае, изображенном на рис. 6, максимальное необходимое количество шагов до «пищи» составит 4 шага, а сумма шагов из всех клеток 47 шагов.

Оба алгоритма в результате обучения анимата как в первом, так и во втором случаях анимат достигал цели независимо от применяемого алгоритма обучения.

График, показывающий зависимость суммы переходов к «пище» от количества обучающих итераций для обоих алгоритмов изображен на рис. 7.

Как видно из графика, оба алгоритма обучения позволяют достичь стратегии поведения, близкой к оптимальной, уже на 50 итерации. Оптимальной стратегии оба алгоритма достигают при 200 итераций.

W	W	W	W	W	W	W	W	W
W	W	W	W	F	W	W	W	W
W	W	↓	↗	↑	↖	↓	W	W
W	W	↙	↖	↖	↖	↘	W	W
W	F	←	↖	↖	↖	→	F	W
W	W	↖	↖	↖	↖	↗	W	W
W	W	↑	↘	↓	↙	←	W	W
W	W	W	W	F	W	W	W	W
W	W	W	W	W	W	W	W	W

Рисунок 5. Схема переходов при минимальном количестве шагов до «пищи»

W	W	W	W	W	W	W	W	W
W	W	W	W	F	W	W	W	W
W	W	→	↗	↑	↖	↓	W	W
W	W	↙	←	←	←	↘	W	W
W	F	←	←	←	←	→	F	W
W	W	↖	←	←	←	↗	W	W
W	W	→	↘	↓	↙	←	W	W
W	W	W	W	F	W	W	W	W
W	W	W	W	W	W	W	W	W

Рисунок 6. Схема переходов при минимальном максимальном необходимом количеством шагов до «пищи»

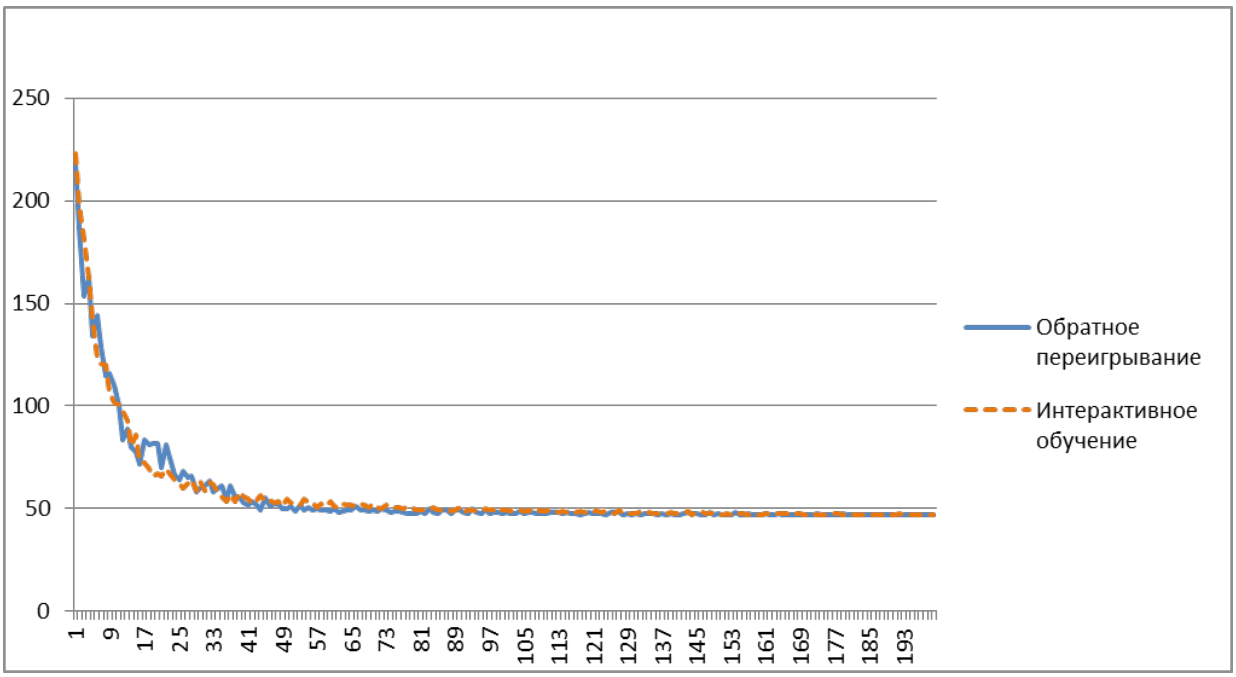

Рисунок 7. Сравнение эффективности обучения алгоритмов обратного переигрывания и интерактивного обучения

Заключение

В результате экспериментов показано, что данные алгоритмы хорошо справляются с самообучением в незнакомой среде. Это позволит внедрять эту технологию во многие прикладные отрасли.

При сравнении двух алгоритмов Q-обучения - обратного переигрывания и интерактивного обучения – оказалось, что оба они достаточно эффективны для обучения агента достижению цели в незнакомой среде. И алгоритм обратного переигрывания, и алгоритм интерактивного обучения сходятся к оптимальному поведению за достаточно небольшое количество итераций.

Дальнейшие исследования будут направлены в сторону усложнения аниматподобной среды для проверки эффективности работы алгоритмов на ситуациях, более приближенных к реальным ситуациям внешнего мира.

Список источников

1. Поспелов С.М., Бондаренко И.Ю. Анализ проблем моделирования интеллектуального поведения персонажей в компьютерных играх // Сб. тр. междунар. научно-техн. конференции «Информатика и компьютерные технологии 2010». – Донецк: ДонНТУ. – 2010.
2. Watkins, C., Dayan P., “Q-Learning”, // In: Machine Learning 8, Kluwer Academic Publishers, Boston, 1992 – pp. 279-292.
3. Поспелов С.М., Бондаренко И.Ю. Разработка модели интеллектуального поведения персонажа в компьютерной игре robocode на основе метода нейродинамического программирования. // Сб. тр. междунар. научно-техн. конференции «Информационные управляющие системы и компьютерный мониторинг 2011». – Донецк: ДонНТУ. – 2011
4. Wilson S. W., “The Animat Path to AI. // In: From Animals to Animats: Proceeding of the First International Conference on the Simulation of Adaptive Behavior, Cambridge, MA: The MIT Press/Bradford Books, 1991.