СОВРЕМЕННЫЕ ПОДХОДЫ К РАЗРАБОТКЕ ИГРОВОГО ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

В.А. Коровкин, аспирант гр. А7-39
Д.Е. Ткаченко, студент гр. 8В7Б
Томский политехнический университет
E-mail: det4@tpu.ru

Введение

Методы машинного обучения и искусственный интеллект (ИИ), как явление, сегодня является одной из самых передовых областей исследований и разработки. Целью исследований в области традиционного ИИ является создание полноценного цифрового «разума», т. е. способного к обучению, к социальному взаимодействию и т. д.

Специфика искусственного интеллекта в играх достаточно сильно отличается от традиционной ИИ. В играх основная цель ИИ – это имитация разумного и реалистичного поведения, которое предложит пройти игроку определенной сложности испытание. Стоит также отметить, что игровой ИИ должен быть легко регулироваться и модифицироваться в эксплуатации, так как процесс разработки игры является итерационным. Традиционные средства разработки ИИ, такие как различные методы машинного обучения для решения такого класса задач и соблюдения выдвинутых условий не всегда подходят. Сегодня можно выделить несколько наиболее распространенных архитектур, которые используются для создания игрового ИИ: конечный автомат, дерево поведения (behavior tree) и utility-based AI.

Конечный автомат

Конечный автомат (FSM – finite state machine) является наиболее простой архитектурой игрового ИИ. Для FSM базовым элементом является состояние. Это конкретное действие (или набор действий), которое агент выполняет в данный момент. Непосредственно поведение агента реализуется при помощи логики определения действий (или набора действий) и смены (или сохранения) состояния, при этом логика может, как и оставаться простой, так и усложняться.

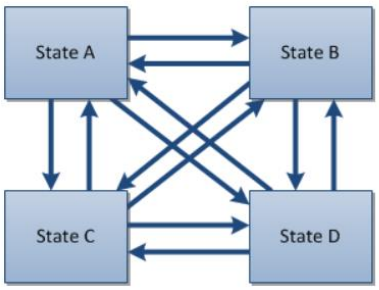

Рис. 1. Схематическое представление конечного автомата

К сожалению, такой подход имеет явные недостатки. С ростом числа состояний количество переходов и связей возрастает с огромной скоростью. Добавление каждого нового состояния будет требовать разработать еще больше переходов. К тому же, данное действие может требовать пересмотра уже существующей логики. А если учесть то, что современные игры, как правило, состоят из десятков состояний, то становится понятно, что такой подход слабо подходит для сложных систем имитирующих виртуальную «жизнь».

Дерево поведения

Далее рассмотрим архитектуру дерево поведения или behavior tree. В конечном автомате, агенты находятся в конкретном состоянии в каждый момент времени и внутри каждого состояния находится логика принятия решений. Как говорилось выше, такой подход приводит к увеличению сложности и объема кода. В архитектуре деревьев поведения всю логику принятия решения отделили от непосредственно самих состояний. Таким образом, вся логика поведения находится в одном месте, что позволяет как угодно изменять и усложнять логику поведения с гораздо меньшими трудозатратами, чем это было бы в конечном автомате.

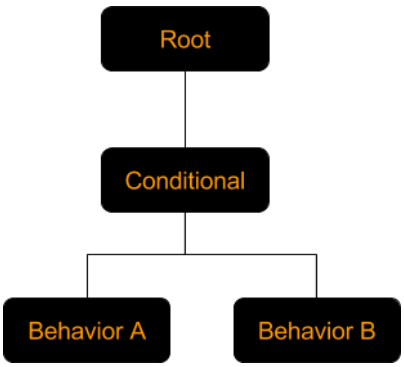

Рис. 2. Типичное дерево поведения

К тому же дерево поведения — это отличный формальный способ построения модели поведения, так как делает модель более прозрачной и интуитивно понятной. На данный момент дерево поведения является наиболее широко применимой архитектурой ИИ в играх.
Данный подход также не лишен недостатков. Одна из главных проблем деревьев поведений состоит в том, что они не располагают потенциалом для «продвинутого» принятия решений, потому что процесс принятия решений привязан к узлам условий, и нельзя указать, как именно принимаются решения, чтобы задействовать различные ответвления. Вместе с этим сложность игровых симуляций постоянно возрастает.

Utility AI

Utility AI – архитектура игрового ИИ, получившая немалую популярность в последние годы. Многие считают данную архитектуру заменой деревьев поведения, потому что данная система позволяет совершать выгодные тактические действия без досконального знания всего игрового пространства. Система выявляет доступные для ИИ действия и начисляет им очки в зависимости от сложившихся обстоятельств. В конце выбирается самое выгодное действие. Данный подход имеет ряд значительных преимуществ. Во-первых, простота проектирования: искусственный интеллект на основе Utility AI зачастую может быть описан на естественном языке, что сильно упрощает коммуникацию между программистом и дизайнером. Нет необходимости затрагивать такие специфические понятия, как условия, состояния, последовательности и декораторы. Вместо этого можно объяснить планируемое поведение в виде «если персонаж под огнём, главная его задача – поиск укрытия». Во-вторых, упрощенная расширяемость: правила, которые в данном подходе называются маркерами, можно свободно добавлять поверх уже существующих, не опасаясь при этом, что произойдут наложения или ошибки. Вышеуказанные пункты значительно сокращают количество возможных ошибок и увеличивают продуктивность. Данный факт, в свою очередь, оставляет простор для совершенствования ИИ в пределах прежних сроков и бюджета, что повышает его итоговое качество в целом.

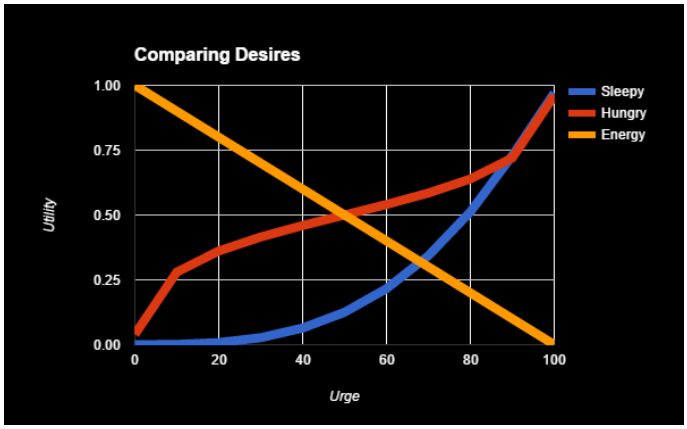

Рис. 3. Кривые начисления очков в Utility AI

В Utility AI для каждого действия есть входная функция, которая нормализуется в пределах от 0 до 100, при этом выгодность нормализуется в пределах от 0 до 1. Кривые позволяют Utility AI принимать решения на основе широкого спектра входных данных, и качество системы выходит на

уровень нечёткой логики. На практике это означает, что Utility AI может принимать неплохие решения даже в ситуациях, не предусмотренных программистом ИИ.

Простота использования Utility AI распространяется и на те области, с которыми обычно возникают проблемы у деревьев поведения. Сравнение и сортировка по приоритету сотен тактических позиций было бы сложной задачей для них, так как новые ветви нельзя добавлять динамически. Однако, правила Utility AI позволяют начислять очки любой отдельно взятой группе объектов, будь то тактическая позиция, планета, поставленная задача или выбор оружия. Таким образом методология принятия решений становится очень гибкой.

Ограничения данного подхода связаны главным образом с тем, что разрабатываемый ИИ базируется на дизайне. Как и в большинстве технологий игровых ИИ, идея в том, что искусственный интеллект должен действовать согласно предусмотренным дизайнером сценариям. Поэтому, ИИ на основе Utility AI может работать в условиях недостатка информации и действовать непредсказуемо благодаря способности к интерполяции и принятию решений на основе нечеткой логики, но он редко будет «умнее» своего создателя. Кроме того, разумность решений и правдоподобность поведения зависят от умелой настройки разных начислителей, а, следовательно, на разработку и особенно тестирование потребуется большое количество времени. В поиске альтернативных технологий, обходящих эти ограничения, следует взглянуть в сторону машинного обучения.

Заключение

В данной статье были описаны основные архитектурные подходы, применяемые для создания игрового искусственного интеллекта. На данный момент наиболее популярное решение в данной сфере это деревья поведений, хотя данная технология постепенно уступает роль конкурентам. В то же время Utility AI уже не раз успешно зарекомендовала себя в популярных продуктах с различными видами симуляций. Она избавлена от многих проблем, присущих деревьям поведения и выглядит очень перспективной для игр с комплексным, правдоподобным и более разумным поведением.

Список литературы

1. Utility AI: Поведенческие деревья уходят в прошлое? [Электронный ресурс] – URL: https://www.progamer.ru/dev/utility-ai.htm (дата обращения: 12.03.21)
2. Кулинарный путеводитель по архитектурам AI [Электронный ресурс] – URL: https://habr.com/ru/post/173187/ (дата обращения: 12.03.21)
3. Создание искусственного интеллекта для игр – от проекта до оптимизации [Электронный ресурс] – URL: https://habr.com/ru/company/intel/blog/265679/ (дата обращения 12.03.21)
4. B. Scwab AI Game Engine Programming – M.: Course Technologies, 2009. – 710 с.
5. Ian Millington Artificial Intelligence for games / Ian Millington, John Funge – M.: Elsevier Inc.? 2009. – 870 с.