Материал опубликован в тезисах научно-технической конференции «Научно-программное обеспечение в образовании и научных исследованиях». СПбГУ ПУ. 2008, с. 46-52.

К.В. Егоров, П.М. Райков, А.А. Шалыто

ПРИМЕНЕНИЕ АВТОМАТНОГО ПОДХОДА ДЛЯ СОЗДАНИЯ ОДНОГО КЛАССА МУЛЬТИАГЕНТНОЙ СИСТЕМЫ

Санкт-Петербургский государственный университет информационных технологий, механики и оптики, г. Санкт-Петербург, Россия,
egorovk@rain.ifmo.ru, raikov@rain.ifmo.ru, shalyto@mail.ifmo.ru

С появлением компьютерных игр резко возрос интерес к обеспечению сложного поведения персонажей игр. Однако, в большинстве случаев интеллект персонажей реализуется без использования формальных методов, что может приводить к несоответствию поведения персонажей с желаемым поведением. Трудности обеспечения интеллекта на примере одной из игр («Freedom Fighters») рассмотрены в работе [1]. Открытие кода этой игры, видимо, связано с тем, что текущее состояние дел в игровой индустрии в плане создания поведенческой модели *NPC* (Non-player-character [2]) неудовлетворительно. Поэтому любое новое эффективное решение в этой области может представлять интерес для компаний, которые разрабатывают игры.

Современные подходы к созданию *NPC* представлены в книгах [3–5], где автоматный подход рассматривается как одно из самых надежных и удобных решений проблемы *NPC*. Там же отмечается несколько важнейших свойств автоматов при программировании игр:

- удобны при написании кода;
- просты в отладке;
- обладают высоким быстродействием;
- легко модифицируются и поддаются анализу.

Однако, в известных работах практически не рассматривались вопросы о формальном переходе от автоматов к их программной реализации и их использовании при создании мультиагентных систем. Эти вопросы рассматриваются в настоящей работе на примере игры «Побег». Это игра для одного игрока (преступника), задача которого состоит в том, чтобы уехать на своей машине от машин полицейских. Игрок управляет машиной мышью, а полицейские машины – искусственным интеллектом. Полицейские машины образуют мультиагентную систему, взаимодействуя между собой для поимки преступника. При этом каждая из полицейских машин может посылать сообщение всем осталь-

ным полицейским для координации действий. В работе используется четыре тактики координаций действий полицейских.

Основные правила игры. Игровое пространство представляет собой двумерную бесконечную плоскость (пустыню), «заселенную» редкой растительностью. Присутствие кактусов и камней на экране обусловлено исключительно эстетическим вкусом разработчиков, которые хотели внести разнообразие в игровой процесс. При этом если любая машина переедет их, то это никак не отразится на ее дальнейшем движении.

В начале раунда игрок появляется где-то в пустыне. После этого сразу же начинается погоня, полицейские машины постоянно появляются из-за пределов экрана и стремятся поймать игрока. Тактики, используемые полицейскими могут быть выбраны и изменены в настроечном XML-файле. При этом полицейские машины периодически сталкиваются и они взрываются. Игра заканчивается, когда какая-нибудь из полицейских машин врезается в машину игрока. После этого на экран выводится результат игры, который зависит от времени, которое игрок «продержался» и числа полицейских машин, столкнувшихся за это время.

Игровой мир устроен так, что полиция не замечает своих потерь, и уже через несколько секунд на смену погибшим машинам приходят новые, которые появляются на пути игрока. Кроме того, игрок может уехать от какой-то машины, и она, скрывшись за пределами экрана, прекратит погоню.

Проектирование и реализация игры на основе автоматного подхода. Данная программа была написана на языке *Java* в среде *Eclipse* на базе инструментального средства *UniMod* [6]. Это средство обеспечивает визуальное проектирование автоматных программ на основе SWITCH-технологии [7, 8]. Эта технология позволяет вынести практически всю логику программы в автоматы и разбить все остальные классы на два основных типа: поставщики событий и объекты управления. В программе также используется ряд вспомогательных классов, например, для реализации отрисовки. При этом автоматы могут содержать как вложенные автоматы, так и вложенные состояния. Это, фактически, является разбиением логики работы *NPC* на уровни [9].

При разработке игры создано пять *UniMod*-моделей: модель игрового мира и четыре модели управления машинами полицейских. На рис. 1, 2 представлена одна из таких моделей управления машиной полиции. Она представляет собой двухуровневую логическую модель, в которой в состояние «Move to player» первого автомата вложен второй автомат.

Эффективность. Отметим, что в связи с ростом числа ядер на одном процессоре появляется возможность распараллеливания приложений. При автоматном подходе это легко реализуется, так как независимые автоматы, обменивающиеся сообщениями, могут выполняться параллельно. Таким образом, игра, реализованной на автоматах, на многоядерном процессоре будет исполняться быстрее, чем реализованная без использования автоматов. Как увеличивается производительность реальных программ при использовании автоматов описано в статье [10].

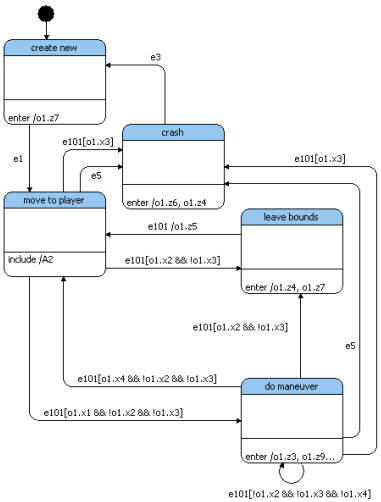

Рис. 1. Автомат управления машиной полиции (A1)

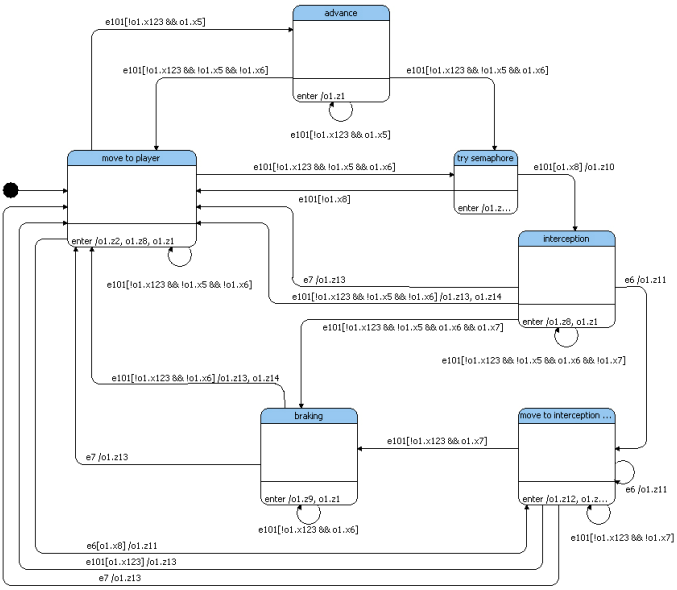

Рис. 2. Вложенный автомат управления машиной полиции (A2)

Реализация мультиагентности. В данной работе, как отмечалось выше, полицейские машины представляют собой мультиагентную систему: они взаимодействуют двумя способами (избегают столкновений и участвуют в «перехвате»). При «перехвате» полицейская машина пытается вычислить вероятную точку пересечения своей траектории с траекторией машины игрока («точку перехвата»). После этого она посылает сообщение всем полицейским машинам с координатами «точки перехвата» и предложением участвовать в этом маневре. Далее одна из других машин принимает это сообщение и начинает выполнение маневра. Таким образом, две машины пытаются создать «барьер» на пути игрока, остановившись перед «точкой перехвата». Заметим, что машины, которые образовали такой «барьер», могут продолжить движение, поняв, что игрок больше не движется в «точку перехвата».

Для ухода от столкновений агенты используют максимально простую тактику – они узнают координаты других агентов, и если их возможные траектории движения будут иметь общую точку пересечения через заданный промежуток времени, то они повернут в необходимом направлении, для того чтобы не встретится в этой в точке. Однако, им может не хватить радиуса поворота и им не удастся избежать столкновения.

Выше описана одна из возможных тактик. Другие тактики поведения полицейских машин отличаются методом поимки преступника. Эти методы подробно описаны в соответствующем проекте (http://is.ifmo.ru/unimod-projects/la_redada/).

Тестирование и анализ работы. Как отмечалось выше – одно из достоинств автоматов является простота отладки, что их существенно отличает от таких методов искусственного интеллекта как, например, «генетические алгоритмы» [11] или «муравьиные алгоритмы» [12]. В инструментальном средстве *UniMod* каждый автомат может выводить лог своих действий на экран, что резко упрощает анализ программы. Кроме того, автоматное программирование позволяет проводить автоматическую верификацию программ [13].

В данной работе помимо лога работы автоматов предлагается его графическое представление с помощью программы, написанной на языке *Java*. После выполнения погони за машиной игрока по этому представлению можно посмотреть траектории движения машин и их взрывы (рис. 3). Здесь используются следующие обозначения: линии – траектории движения машин, черные круги – места взрывов машин.

Диаграмма на рис. 3 не представляет полной картины погони, зато дает возможность исследовать поведение *NPC* визуально. Например, с ее помощью можно установить, что все черные кружки располагаются

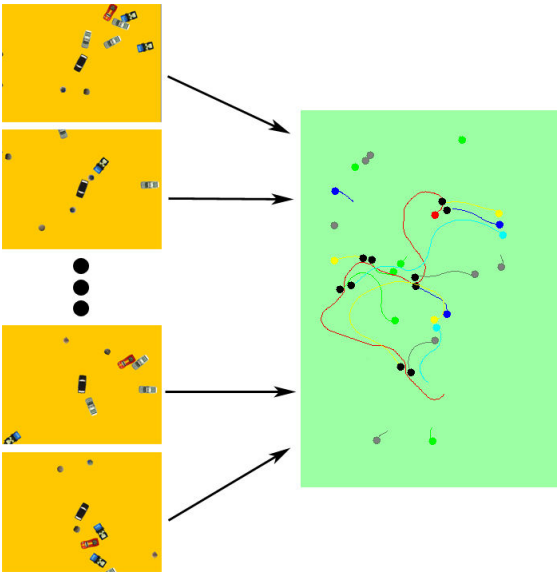

Рис. 3. Построение диаграммы хода игры

парами – полицейские машины взрываются при столкновении друг с другом. Также видны очень короткие траектории и кружки вообще без выходящих линий – это полицейские машины, которые просто не успели развернуться и начать погоню, и поэтому быстро выбывшие из игры.

Выводы. Автоматный подход может использоваться в качестве одного из основных средств для разработки искусственного интеллекта игр. С помощью автоматов создаются роботы [14], имитируется работа мультиагентных систем в работах [15, 16]. Дальнейшее развитие данного подхода с использованием инструментального средства *UniMod* и добавление новых возможностей по графической отладке может внести существенный вклад в решение проблемы создания искусственного интеллекта.

Источники

1. *Andreasen P.* Top 5 reasons not to use AI in computer games. http://itu.dk/courses/MAIS/E2005/Lectures/ai_top_5_reasons.pdf
2. *Wiki.* Non-player-charachter. http://en.wikipedia.org/wiki/Non-player_character
3. *Champandard A.* AI Game Development: Synthetic Creatures with Learning and Reactive Behaviors. New Riders Publishing. 2003.
4. *Buckland M.* Programming Game AI by Example. Wordware Publishing. 2004. http://ai-junkie.com/architecture/state_driven/tut_state1.html
5. *AI Game Programming Wisdom*. Charles River Media. 2002.
6. *UniMod project*. http://unimod.sourceforge.net
7. *Шалыто А. А.* SWITCH-технологии. Алгоритмизация и программирование задач логического управления. СПб.: Наука, 1998. http://is.ifmo.ru/books/switch/1
8. *Гуров В., Мазин М., Нарвский А., Шалыто А.* UML. SWITCH-технология. Eclipse //Информационно-управляющие системы. 2004, № 6. http://is.ifmo.ru/works/uml-switch-eclipse
9. *van Waveren J.M.P.* The Quake III Arena Bot. 2001. http://www.kbs.twi.tudelft.nl/docs/MSc/2001/Waveren_Jean-Paul_van/thesis.pdf
10. *Seinstra F., Koelma D., Bagdanov A.* Finite State Machine Based Optimization of Data Parallel Regular Domain Problems Applied in Low Level Image Processing // IEEE Transactions on Parallel and Distributed Systems. 2004. Vol. 15. № 10, pp. 865–877. http://staff.science.uva.nl/~fjseins/Papers/Journals/i3e-tpds2.pdf
11. *James G.* Using Genetic Algorithms for Game AI. 2005. http://gignews.com/gregjames1.htm
12. *Gordon D.* Ant-based Pathfinding. 2002. http://david.gordon.name/

projects/Ant-based%20Pathfinding.pdf
13. *Вельдер С. Э., Шалыто А. А.* Введение в верификацию автоматных программ на основе метода Model checking. СПбГУ ИТМО. 2006. http://is.ifmo.ru/verification/modelchecking/
14. *Argall B., Browning B., Veloso M.* Learning to Select State Machines using Expert Advice on an Autonomous Robot / Proceedings of IEEE International Conference on Robotics and Automation. Rome. 2007. http://www.cs.cmu.edu/~bargall/docs/07icra-argall.pdf
15. *Паращенко Д. А., Царев Ф. Н., Шалыто А. А.* Технология моделирования одного класса мультиагентных систем на основе автоматного программирования на примере игры «Соревнование летающих тарелок». http://is.ifmo.ru/unimod-projects/plates/
16. *Hamel A., Attonaty J., Suzanne P.* An Instrumetalized Participatory Approach for Cooperative Knowledge Acquisition to Build a Social MABS / Proceedings of European Simulation Multiconference. Magdeburg. 2004. http://scs-europe.net/services/esm2004/pdf/esm-40.pdf