Компьютерные инструменты в образовании, 2014
№ 4: 55–59
УДК: 621.320
http://ipo.spb.ru/journal

АЛГОРИТМ «ШПИОНАЖА» ДЛЯ КОМПЬЮТЕРНЫХ ИГР С МНОГОЧИСЛЕННЫМИ СОПЕРНИКАМИ

Жигулин Андрей Юрьевич, Сафонов Владимир Олегович

Аннотация

В статье рассматривается алгоритм, изменяющий поведение компьютерных соперников путем анализа действий пользователя и других компьютерных соперников.

Ключевые слова: *алгоритм, «шпионаж», ранжируемые сущноти, игры, игровая механика.*

1. ВВЕДЕНИЕ

При создании компьютерных игр одной из главных проблем является быстрое привыкание пользователя к игровому процессу. После привыкания пользователь чаще всего не видит смысла в дальнейшей игре, и решением данной проблемы обычно являются два метода оптимизации игры:

- Добавление нового контента в игру, с новой игровой механикой.
- Изменение механики уже существующего контента.

Оба этих метода требуют денежных и человеческих затрат, так как необходимо придумывать изменения «вручную». Наиболее эффективное решение, требующее наименьших затрат —- это создание динамической системы, которая меняется со временем, создавая непредсказуемую для пользователя механику, не добавляя при этом нововведений. В таком случае разработчики должны будут изначально заложить алгоритм изменения игры, после чего она будет подстраиваться под пользователя самостоятельно. Необходимость приближения мышления компьютерного соперника к мышлению пользователя описана в [1].

2. ОПИСАНИЕ ПРЕДЛАГАЕМОГО АЛГОРИТМА

В данной статье рассказывается об алгоритме, предназначенном для решения следующих задач:

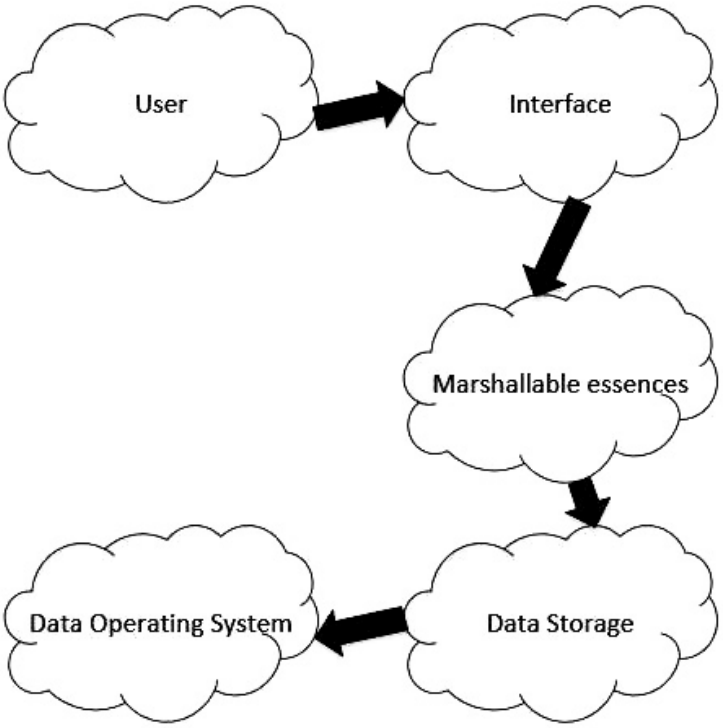
- Создание непредсказуемого для пользователя поведения компьютерных соперников.
- Усложнение игрового процесса.
- Поддержание заинтересованности пользователя в пользовании приложением.

Также данный алгоритм может использоваться как расширение пакета приложений, поставляемых одной компанией или группой компаний с похожими по функциональности приложениями. Расширение предоставляет действия пользователя, панели инструментов, горячие клавиши и другие настраиваемые элементы приложений в виде сохраняемых в отдельном файле настроек. При запуске аналогичных по функциональности приложений и анализе подобных файлов приложение может импортировать по желанию пользователя некоторые настройки из уже использовавшихся ранее приложений. Импорт поможет пользователю быстрее оптимизировать приложения для своих потребностей.

Рассматриваемая система состоит из следующих частей:

1. Интерфейс, определяющий основные возможные действия пользователя.
2. Ранжируемые сущности, определяющие все возможные действия пользователя (организация данного интерфейса и сущностей, ему удовлетворяющих, требует 90% всего времени на интегрирование алгоритма шпионажа в приложение и будет уникальна в каждой игре).
3. Функциональность, отвечающая за запись действий пользователя.
4. Функциональность, анализирующая действия пользователя и изменяющая поведение компьютерных соперников.

Взаимодействие описанных частей продемонстрировано на рис. 1



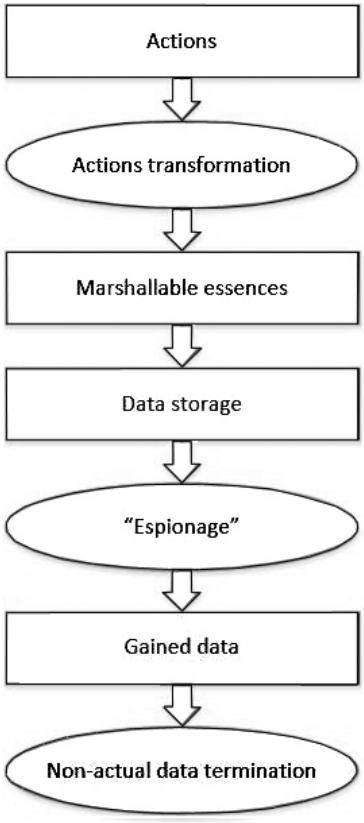
.

Рис. 1. Организация компонент системы

Алгоритм описывается следующими шагами:

1. Представление всех действий пользователя в виде ранжируемых сущностей.
2. Запись ранжируемых сущностей для последующего использования компьютерными соперниками.
3. Стадия шпионажа между компьютерными соперниками, в которой они с определенной долей вероятности, зависящей от успешности в игре против пользователя и количества полученных знаний, забирают определенные знания либо случайные знания.
4. Дальнейшее применение своих и украденных знаний против пользователя и других компьютерных соперников.
5. (Опциональное) уничтожение неактуальных стратегий. Необходимость планирования действий и создания собственной стратегии поведения для компьютерных соперников при симулировании действий пользователей описана в [1]. Более наглядно алгоритм описан на рис. 2.

Формулировка алгоритма выглядит следующим образом:



.

Рис. 2. Последовательность шагов алгоритма

1. Получение всех знаний пользователя, где n — число знаний пользователя (действия, нажатия на клавиши и т.д.).
2. Преобразование распознаваемых знаний пользователя в набор ранжируемых сущностей, где k — количество распознаваемых программой знаний.
3. Добавление полученных ранжируемых сущностей в базу знаний D (исключая добавление дубликатов, но учитывая актуальность данных при их совпадении). В базе знаний D хранятся элементы типа `<key, value>`, где `key` — это ранжируемая сущность, а `value` — объект, показывающий актуальность ее использования (количество использований, дата последнего использования, полезность). Изначально каждая ранжируемая сущность заносится в базу со значениями:
   1 — количество использований,
   0 — полезность.
   Текущая дата занесения в базу знаний — дата последнего использования.
   В дальнейшем подобное число помогает проверять актуальность ранжируемых сущностей и последующее удаление неактуальных сущностей. Такое детальное, но упрощенное в терминах хранения представление элементов позволяет экономить ресурсы и оставлять их для более важных задач (оптимизация данных и ресурсов рассмотрена в [2]).
4. Стадия шпионажа — процесс проверки каждого элемента базы знаний D на актуальность для компьютерного соперника. Сначала идет проверка на наличие данного элемента в базе знаний соперника, затем, в случае отсутствия данного элемента в базе, идет проверка его количества использований и полезности. В данном случае от разработчика зависит формула, проверяющая элемент на актуальность. Затем, в случае положительного результа-

та проверки элемента, он заносится в базу знаний компьютерного соперника D'. Способы проверки с оптимальным использованием ресурсов, а также различные алгоритмы выбора получаемых данных описаны в [3].
5. Использование всех знаний базы D' и выбор подходящих действий.
6. Уничтожение неактуальных объектов баз знаний D и D'.

В ходе проведенных исследований были получены следующие результаты:

1. Сформулирован и описан алгоритм шпионажа.
2. Создан прототип приложения, анализирующего определенные данные из файлов с уникальным расширением, отвечающих за хранение общих для всех приложений настроек. Приложение ищет все файлы определенного расширения, читает настройки, фильтрует их (отбрасывая не подходящие под него) и предлагает пользователю импортировать данные настройки в приложение.

3. ЗАКЛЮЧЕНИЕ

Так как актуальность компьютерных игр со временем растет, реализация алгоритмов, поддерживающих интерес пользователя к игре, является одной из самых актуальных и приоритетных задач. Верное использование описанного алгоритма способно создать конкурентоспособное приложение на рынке и сэкономить денежные и человеческие ресурсы компании.

Тем не менее, данная система очень зависима от навыков команды разработчиков и аналитиков, на которых ложится основная задача — предсказывание действий пользователя и создание правильного алгоритма в игре.

Если предсказывание действий пользователя будет ошибочным, то трудоемкая задача по интеграции алгоритма в игру будет тратой времени разработчиков. В таком случае целесообразно модифицировать ранжируемые сущности и строить игру уже на основании этих изменений.

Список литературы

1. *Madhav S.* Game Programming Algorithms and Techniques: A Platform-Agnostic Approach (Game Design). UK, 2013.
2. *Ericson C.* Real-Time Collision Detection. USA, 2004.
3. *Knuth D.E.* The Art of Computer Programming, Vols. 1–3. UK, 1998.

«Espionage» algorithm for computer games with multiple opponents

Zhigulin A. Y., Safonov V. O.

Abstract

The article discusses an algorithm that modifies the behavior of computer opponents by analyzing user and other computer opponents' behavior.

Keywords: *algorithm, «espionage», marshallable essences, games, game mechanics.*

Жигулин Андрей Юрьевич,
аспирант Санкт-Петербургского государственного университета,
dron.subzero@gmail.com

Сафонов Владимир Олегович,
доктор технических наук, профессор кафедры информатики СПбГУ,
vosafonov@gmail.com