УДК 004.942

Е.Б.Замятина, Д.Ф.Каримов, А.А.Митраков

АРХИТЕКТТУРА АГЕНТНО-ОРИЕНТИРОВАННОЙ СИСТЕМЫ ИМИТАЦИИ С АГЕНТАМИ, ОСНОВАННЫМИ НА НЕЙРОННЫХ СЕТЯХ

В работе рассматриваются принципы построения агентно-ориентированной системы имитационного моделирования. Известно, что агентные модели расширяют возможности применения метода имитационного моделирования при решении ряда задач, которые методами системной динамики или событийно-ориентированного моделирования решены быть не могли. Особое внимание при проектировании агентно-ориентированной системы имитации авторы уделяют проблемам реализации распределенного моделирования, реализации интеллектуальных агентов и использованию онтологий на всех этапах имитационного моделирования.

This paper discusses the problems of the agent-based simulation system design. It is wellknown that agent models extend the capabilities of simulation for solving some problems that can't be solved by the methods of system dynamics and discrete event simulation. Particular attention in the design and the implementation of agent-based simulation authors pay to the problems of distributed simulation, the problems of intelligent agent's implementation and the use of ontologies at all stages of the simulation experiments.

Ключевые слова: имитационное моделирование, агентная модель, распределенное моделирование, интеллектуальные агенты, онтологии

Keywords: simulation, agent-based model, distributed simulation, intelligent agent, ontology

Введение

Проблемы, возникающие в повседневной жизни (в промышленном производстве, торговле, управлении и т.д.), постоянно усложняются, что вызывает необходимость разрабатывать новые и совершенствовать имеющиеся методы решения этих проблем. Одним из методов решения сложных проблем в различных областях деятельности является имитационное моделирование.

Известно, что метод имитационного моделирования применяется для исследования сложных *динамических* систем. Применение этого метода целесообразно в том случае, когда решаемую задачу трудно формализовать и аналитические и численные методы не могут достаточно детально исследовать изучаемую систему, объект, процесс, явление.

Различают непрерывное моделирование и дискретно-событийное (DES - discrete-event simulation). Однако существуют задачи, которые трудно описать с помощью непрерывных и дискретных имитационных моделей. Это верно по отношению к экономико-математическим моделям (эти модели описывают процессы, протекающие в экономических и эколого-экономических системах в виде системы математических выражений (уравнений и неравенств)). Однако современные экономико-математические модели имеют динамический характер и большую размерность. При моделировании возникают ситуации, когда построенные модели, несмотря на соответствие моделируемому объекту, оказываются противоречивыми.

В этом случае актуальным становится применение новых методов, которые позволяют решать задачи управления сложными объектами. Одним из таких методов является метод агентного моделирования. Суть агентного моделирования состоит в том, что локальное поведение агентов, работающих по своим собственным правилам, формирует глобальное поведение системы в целом (концепция проектирования «снизу-вверх»). Это отличается от традиционных подходов проектирования имитационной модели «сверху-вниз», когда заданы глобальные законы функционирования системы, на базе которых работают её элементы. Глобальное функционирование системы заведомо неизвестно исследователю. Два-три простейших правила уже могут привести к весьма разнообразным формам поведения в группе

агентов. Примером может послужить boids-моделирование и теория клеточных автоматов[1,2].

Еще одним аргументом в пользу применения агентно-ориентированного моделирования является необходимость проектирования, анализа и реинжиниринга бизнес-процессов [3]. Часть операций в бизнес-процессе выполняются объектами, которые самостоятельно определяют свое поведение (лица, принимающие решения, автоматизированные и роботизированные комплексы, объекты, управляемые человеком). При этом поведение объекта определяется на основании набора правил, обычно неизменного в ходе выполнения поставленной задачи. Кроме того, следует учитывать воздействие других объектов на принятие решения конкретного объекта о своем поведении. Еще одной сложностью является то, что зачастую нельзя свести процесс принятия решения к единственной оценке решения или к функции полезности, которую можно вычислить, используя численные методы. Однако подобная оценка может быть вычислима в результате работы с набором правил, последовательно применяемых к текущей ситуации.

Итак, под агентом будем понимать самостоятельную (автономную) систему, имеющую возможность принимать воздействие от внешнего мира, определять свою реакцию на это воздействие и осуществлять эту реакцию, а под интеллектуальным агентом - агент, который обладает рядом знаний о себе и окружающем мире, при этом целесообразное его поведение определяется этими знаниями.

Существует большое количество работ, которые показывают актуальность применения агентного подхода в маркетинге[4], в моделировании ситуаций, которые происходят в торгах[5], управлении запасами, цепочками поставок и т.д.

В настоящее время существует большое количество программных систем (чаще всего – это библиотеки программ), которые позволяют реализовать агентное моделирование. Ниже рассматриваются различные программные системы как отечественные, так и зарубежные, как платные, так и свободно распространяемые. Рассматриваются архитектурные особенности построения программных систем, реализующих агентный подход, предлагаются решения, позволяющие повысить адаптируемость программного продукта, эффективность и надежность агентного имитационного эксперимента

Обзор существующих программных систем агентного моделирования

ANYLOGIC

Одна из наиболее известных агентно-ориентированных систем моделирования - AnyLogic[6,8]. Система поддерживает непрерывное моделирование, процессо-ориентированное и агентное, располагает современным графическим интерфейсом и предоставляет пользователю набор стандартных библиотек, что упрощает процесс разработки и позволяет создавать модели для широкого спектра задач. Помимо встроенного графического языка возможно расширение созданных моделей с помощью языка Java (а это позволяет добиться кроссплатформенности). Интеграция компилятора Java в AnyLogic предоставляет более широкие возможности при создании моделей, а также создание Java-апплетов, которые могут быть открыты любым браузером. Среда моделирования AnyLogic поддерживает проектирование, разработку, документирование модели, выполнение компьютерных экспериментов с моделью, включая различные виды анализа — от анализа чувствительности до оптимизации параметров модели относительно некоторого критерия. Однако в AnyLogic нет средств создания интеллектуальных агентов, кроме того, настоящая версия не поддерживает распределенное моделирование (но, судя по публикациям, работы такие ведутся[7]).

BPSim

Еще одна отечественная разработка агентной системы имитации – BPSim [9,10,11]. Система имитационного моделирования предназначена для моделирования бизнес-процессов и организационно-технических систем. Программная система BPSim представляет

собой программную реализацию модели мультиагентного преобразования потребления ресурсов (МППР). Агенты управляют объектами процесса преобразования ресурсов. Агенты анализируют текущую ситуацию, обращаются к базе знаний, вырабатывают решение, контролируют достижение целей, обмениваются сообщениями. Программные средства BPSIM позволяют разработать концептуальную модель, динамическую модель, провести имитационный эксперимент и экспортировать результаты в EXCEL. В системе реализованы (а) *реактивные агенты* (их описывают с помощью диаграмм деятельности (конечный автомат), используют для динамического моделирования МППР), (б) *реактивно-интеллектуальные* агенты (описывают с помощью продукционной базы знаний, используют для динамического моделирования МППР, для описания моделей ЛПР (лицо принимающее решение), управляющих процессами), (в) *интеллектуальные агенты* (поведение их описывают планирующей системой, знания хранятся в фреймовой базе знаний, используют для построения сложных советующих ЭС и т.д.), (г) *гибридные* (построение сложных систем планирования).

Однако вряд ли можно говорить о кроссплатформенности этих программных средств BPSIM (генерируется код на Delphi[11]) и о поддержке параллельного (распределенного) моделирования.

REPAST

REcursive Porous Agent Simulation Toolkit (Repast)[12] – это открытый и свободно распрстраняемый источник библиотек для крупномасштабного агентного моделирования. Repast поддерживает разработку гибких моделей из агентов и используется в моделировании социальных процессов, в маркетинге, логистике. Пользователь строит свою модель, включая в свои программы компоненты из библиотеки Repast или используя визуальный Repast для среды Scripting (существует три версии Repast, названных Repast for Python (Repast Py) (Python), Repast for Java (Repast J) и Repast for the Microsoft.NET framework (Repast .NET)). Так Repast J – это кроссплатформенная среда моделирования, написанная на Java и предназначенная для разработки крупномасштабных агентных моделей, обладающими свойствами интероперабельности, интегрируемости и мобильности. Repast J включает (а) параллельный дискретный планировщик по времени (календарь событий); (б) среду для визуализации модели; (в) средства интеграции с географическими информационными системами с целью моде0лирования агентов на реальных картах); (г) средства описания поведения агентов (с применением компонентов из библиотек нейронных сетей, генетических алгоритмов, например). Модели Repast могут разрабатываться различными способами, включая использование ReLogo (диалект языка мультиагентного моделирования Logo), блок-схем, языка Groovy (динамического языка виртуальной машины Java) или самого языка Java. Также все указанные способы редактирования модели можно чередовать без потери качества в ходе разработки модели. Кроме того, среда моделирования располагает средствами для обработки результатов моделирования (например, MatLab, SQL, Excel). Однако для того, чтобы создать модель, описать поведение агентов, требуется подготовка, надо быть специалистом-программистом.

NetLogo

NetLogo[13] – среда моделирования, предназначенная для создания моделей, описывающих естественные и социальные феномены. Данная система позволяет пользователям оперировать моделями “на лету”, динамически меняя поведение системы под действием различных условий. Система достаточно проста, что позволяет пользователям без квалификации программиста открывать и запускать уже готовые модели, или строить их самим. Но также система достаточно “продвинута”, чтобы удовлетворить запросы исследователей из многих областей. Среда моделирования NetLogo является кроссплатформенной и поставляется с библиотекой программных компонентов, которая представляет собой большой набор заранее написанных моделей. Эти модели могут быть использованы вновь или модифицированы.

Mason

Агентно-ориентированная система моделирования MASON[14] – это мультиагентная среда моделирования, представленная в виде набора библиотек на Java (кроссплатформенность) и поддерживающая дискретно-ориентированную парадигму моделирования (выпущена под лицензией Academic Free License, version 3.0). MASON включает мощные программные средства визуализации (2D,3D). В то же время, пользователь имеет возможность не подключать эти средства. Система MASON устойчива к взломам.

Однако MASON не поддерживает распределенное моделирование (но в настоящее время ведутся работы по созданию распределенной версии) и требует серьезных знаний языка Java, т.е. не является удобным для малоопытного пользователя.

Ascape

Система Ascape [15]– еще одна кроссплатформенная агентная система моделирования, написана на Java и является свободно распространяемой. Инструментальные средства, предоставленные конечному пользователю, делает возможным пользователям без программистских навыков освоить многие аспекты моделирования.

SWARM

Swarm [16](стая, рой) был первой средой разработки АМ приложений, впервые запущен в 1994г. Swarm стремится создать распределенную платформу для моделирования АМ и содействовать разработке широкого круга моделей. Пользователь создает модели путем включения компонентов из библиотек Swarm в свои программы. Больший объем информации про Swarm, также как и дистрибутивы, могут быть найдены на домашней странице SDG.

Требования к построению агентно-ориентированных систем имитационного моделирования

Итак, обзор позволяет сделать вывод о том, что при разработке систем моделирования важно, чтобы модели могли разрабатывать не только пользователи с квалификацией программиста, но и обычные конечные пользователи[17,18]. Таким образом, агентно-ориентированные системы должны обладать визуальными программными средствами конструирования и редактирования моделей. Кроме того, для агентных систем моделирования характерны: (а) кроссплатформенность (чаще всего используют язык Java или C); (б) модульность, использование объектно-ориентированного подхода (агентов чаще всего представляют в виде объектов, которые обмениваются информацией друг с другом); (в) возможность динамического изменения моделей (изменение настроек, например) во время выполнения имитационного эксперимента. При разработке агентно-ориентированной системы моделирования авторы пытались учесть опыт перечисленных выше разработок и добиться выполнения требований, изложенных ниже:

(а) *Операции над моделью*

При разработке агентной модели целесообразно предусмотреть возможность изменения набора агентов, совместное функционирование которых отображает протекающие в реальном мире процессы, возможность изменения связей между агентами[19,20,21].

(б) *Иерархическое представление, детализация*

Кроме того, имитационная модель должна быть иерархической и предоставлять возможность детализировать модель или, наоборот, заменять группу агентов одним, при этом новый агент должен реализовывать групповое поведение объединенных в группу агентов[19,20,21].

(в) *Выделение структуры взаимодействия агентов*

Имитационная модель отображает взаимосвязи агентов (структура моделируемой системы), поведение агентов. Кроме того, агенты обмениваются сообщениями.

Целесообразно разработать программное обеспечение, которое в готовой имитационной модели выделяет ее структуру. Выделив структуру агентов, их взаимосвязи, можно про-

вести дополнительный анализ имитационной модели, исследуя ее структурные характеристики (возможно, методами теории графов)[19,20,21].

(г) Сбор и обработка статистических данных

Сбор и обработка статистических данных должны выполняться специальными агентами, которые играют роль датчиков, мониторов и составляют «алгоритм исследования». Алгоритм исследования должен быть отделен от самой имитационной модели[21].

(д) Верификация и валидация агентных имитационных моделей

В большинстве программных систем имитационного моделирования либо не существует, либо существуют ограниченные по своей функциональности средства верификации и валидации моделей. Целесообразно разработать специальные программные средства для отладки и тестирования имитационных моделей с целью получения достоверной, валидной модели[22]

(е)Адаптируемость

Системе имитации приходится решать задачи, связанные с различными предметными областями. Кроме того, с имитационной моделью можно работать в разных «терминах», так например, при моделировании компьютерной сети можно работать в «терминах» систем массового обслуживания, можно «в терминах» сетей Петри или теории графов. В этом случае целесообразно использовать онтологический подход, опыт применения онтологий при работе с имитационными моделями приводится в [23,24,25,26].

(ж) Оптимизация имитационного эксперимента по времени

Сложность решаемых задач требует использования вычислительных ресурсов целого набора вычислительных узлов (это может быть кластер, суперкомпьютер, вычислительная сеть). Использование вычислительных ресурсов нескольких вычислительных узлов позволит сократить время выполнения имитационного эксперимента[27]. Для сохранения каузальности событий в распределенной модели используют оптимистический и консервативный алгоритмы[28,29,30]. Еще одним способом сокращения времени распределенного имитационного эксперимента является балансировка нагрузки вычислительных узлов[31,32,33,34].

(з) Оптимизация имитационного эксперимента по надежности

В условиях распределенного имитационного эксперимента целесообразно разработать дополнительное программное обеспечение для обеспечения надежности проведения имитационного эксперимента. Программное обеспечение должно обнаруживать отказавшее устройство и перемещать агенты на другие «живые» вычислительные узлы.

(и) Удаленный доступ

При разработке системы агентного имитационного моделирования (как и систем с другой парадигмой моделирования) целесообразно реализовать удаленный доступ, разработав соответствующие WEB-сервисы. Удаленный доступ позволит не только оперативно получать пользователям результаты моделирования, создавать модели с помощью графического или текстового редактора, но и организовывать совместную работу пользователей, которые географически находятся на удаленном расстоянии друг от друга[35].

(к) Интеллектуальная обработка результатов моделирования

В результате имитационного эксперимента пользователь получает большое количество зачастую неструктурированной информации. Целесообразно в этом случае выполнять дополнительную обработку результатов моделирования, используя методы Data Mining и Knowledge Discovery. Дополнительная обработка результатов моделирования позволит оптимизировать имитационный эксперимент, выявляя зависимости и взаимосвязи между входными параметрами[36].

(*л) Интеллектуальные агенты*

Для того, чтобы наиболее адекватно отображать моделируемые процессы в экономике, маркетинге, логистике очень важно реализовать интеллектуальных агентов. Большинство из рассмотренных выше систем предоставляет пользователю инструментальные средства для

описания реактивного поведения, позволяют имитировать достаточно простое поведение. А необходимо сделать так, чтобы интеллектуальные агенты могли изменять свое поведение в зависимости от изменения обстановки (обучаться), преследовать цели, выбирать ту или иную стратегию в зависимости от роли, которую они выполняют.

В настоящей версии разрабатываемой авторами агентной системы имитации особое внимание уделялось разработке инструментальных средств, поддерживающих распределенное моделирование и разработке интеллектуальных агентов.

Реализация агентной системы имитации

Настоящая версия мультиагентной системы моделирования выполнена на языке *Scala*[37]. Язык *Scala* был выбрана в качестве языка программирования для реализации агентной платформы моделирования. Язык *Scala* представляет собой кросспарадигменный объектно-функциональный язык программирования, совмещающий в себе ООП и функциональное программирование и специализирующийся на создании легкомасштабируемого компонентного программного обеспечения (*Scala* была создана в 2004г. под руководством Мартина Одерски в Университете *EPFL* (Lausanne, Switzerland)). В настоящее время доступна для платформ *Java* и *.NET Framework*. Среди ключевых особенностей языка можно отметить: единую объектную модель, наличие примесей (*traits*), технику сопоставления с образцом, лямбда-исчисление, виды (*type views bounds*), линеаризацию типов (*type linearization*), параметрический и функциональный полиморфизм, вариантность типов, кейс-классы (*case classes*), вывод типов, поддержка хвостовой рекурсии, наличие многочисленных инструментов по созданию новых языковых конструкций и обработку списков. Обобщённая структурная схема симулятора изображена на рис. 1.

Таким образом, архитектура симулятора является не многокомпонентной, а многослойной. Достигается это за счёт того, что к модулю, реализующему некоторую базовую функциональность, «примешиваются» другие компоненты, расширяющие структуру, поведение и семантику.

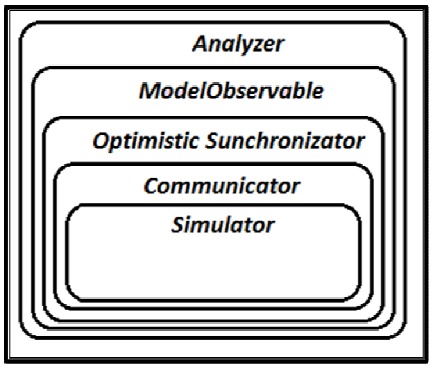

Рис. 1. Многослойная архитектура симулятора

Трейт Simulator является базовым модулем, к которому подмешиваются другие элементы, уточняя его поведение. Фактически, Simulator, – это каркас для логического процесса. Трейт Communicator представляет собой модуль для работы с акторской системой. Он содержит в себе экземпляр актора, который занимается посылкой/приёмом сообщений. Трейт OptimisticSynchronizator является ключевым звеном для организации PDES – он реализует оптимистический алгоритм синхронизации Time Warp. Трейт ModelObservable является реализацией концепции информационных процедур, принятых в области имитационного моделирования для организации сбора статистики (информационные процедуры – программные компоненты, накладываемые на модель с целью сбора статистики в ходе имитационного

прогона). Трейт Analyzer добавляет «интеллектуальность» алгоритмам синхронизации, используемых для реализации распределенного моделирования.

Особое внимание в разработанной системе имитации уделяется алгоритму синхронизации распределенной модели. Известно, что распределенная модель представляет собой совокупность логических процессов, которые выполняются на различных вычислительных узлах высокопроизводительной вычислительной системы. Логические процессы обмениваются сообщениями друг с другом. При выполнении распределенной модели очень важно поддерживать каузальность событий. С этой целью разработаны алгоритмы, которые стали классическими. Алгоритмы разделяют на две группы: консервативные и оптимистические. Консервативные алгоритмы предполагают продвижение времени только после того, как становится очевидным, что очередное событие является безопасным. Событие является безопасным, если есть уверенность в том, что не появится другое событие с меньшим “штампом времени”[]. Оптимистический алгоритм состоит в следующем: процесс выполняется, продвигаясь от одного события к другому о тех пор, пока он не получит от другого процесса сообщение с меньшим штампом времени, нежели штамп времени очередного события[].

В системе реализован оптимистический алгоритм синхронизации, основанный на знаниях об имитационной модели. Существует ряд работ [38, 39], в которых используются знания об имитационной модели для сокращения времени выполнения распределенного имитационного эксперимента. Сокращение времени выполнения распределенного алгоритма авторы достигают за счет реализации эффективного алгоритма синхронизации логических процессов, управляемого продукционной экспертной системой, и за счет балансировки нагрузки на вычислительных узлах. Классические алгоритмы также используют «знания» о модели: *lookahead* в консервативных алгоритмах, *lookback* в оптимистических, циклы в алгоритме Work Flow и т.д. Использование знаний об агентной имитационной модели, извлекаемые из онтологий, дали хорошие результаты при реализации алгоритмов синхронизации в агентной системе имитации. Были проведены эксперименты с одной и той же моделью многократно при одних и тех же параметрах. Общая длительность моделирования составляла 300 (±15) ед. модельного времени.

Эксперименты показали, что метрики, характеризующие скорость проведения распределенного эксперимента, управляемого модифицированным алгоритмом синхронизации (алгоритм KBASA, основанный на знаниях о модели) существенно сократились по сравнениями с метриками классического оптимистического алгоритма Time Warp. Действительно, (а) Количество откатов сократилось с 71.2 до 3.8(рис.2.). (б) Общее число неглубоких откатов сократилось практически до нуля (в среднем 0.1-0.2 против 7-11). (в) Число глубоких откатов также сократилось с 19.4 до 3.2. (г) Количество отправленных антисообщений сократилось с 108.4 до 12.8 (рис.).

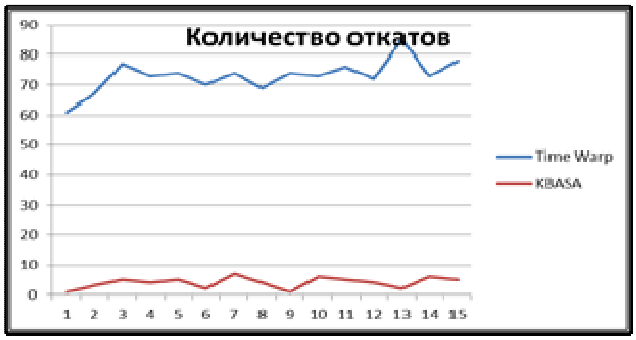

Рис.2.Количество откатов за 300 ед. модельного времени

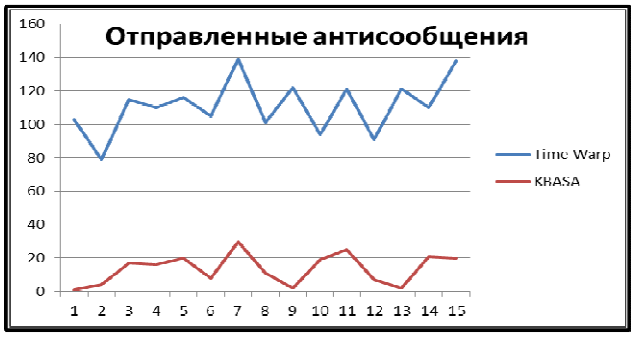

Рис.3.Отправлено антисообщений за 300 ед. модельного времени

Реализация интеллектуальных агентов

Известно, что создание интеллектуальных агентов[44] является весьма сложной задачей, требующей теоретического фундамента для концептуального представления агентов. Таким фундаментом служат модели интеллектуальных агентов, по-разному описывающие знания, способы рассуждений, планирование поведения и непосредственные действия агентов.

Модели принято рассматривать в двух точек зрения: с точки зрения анализа свойств и поведения агентов в процессе функционирования системы в целом; с точки зрения изучения и конструирования свойств агента, определяющих его внутренние процессы (получение знаний, выработка целей, принятие решений и т.д.).

Выделено три вила архитектур: (а) делиберативные архитектуры и модели; (б) реактивные архитектуры и модели; (в) гибридные архитектуры и модели.

Реактивный подход позволяет использовать множество достаточно простых сценариев поведения агентов. Эти сценарии являются реакцией на появление того или иного события внешней среды. Недостатком является то, что достаточной полный ситуативный анализ всех возможных активностей агентов.

Делиберативные модели и архитектуры позволяют применять строгие формальные методы и хорошо отработанные технологии традиционного искусственного интеллекта, позволяющие относительно легко представлять знания в символьной форме и переносить их в агентно-ориентированные системы.

Гибридный тип архитектуры совмещает преимущества упомянутых выше архитектур. Таким образом, интеллектуальный агент обладает высокоуровневым выводом и низкоуровневыми реактивными способностями.

Итак, разработка агентной системы моделирования предполагала разработку инструментальных средств набора базовых классов для представления интеллектуальных агентов. Было принято решение представить архитектуру агентов в виде гибридной и делиберативной схем. В качестве средств логического вывода авторы используют продукционные системы и нейронные сети. Известно, что нейронные сети обладают свойством самообучаться, что актуально для реализации интеллектуальных агентов, которые должны приспосабливаться к изменению внешней среды и менять свое поведение, принимая то или иное решение. На первом этапе исследование было использовать три типа нейронных сетей: многослойный персептрон, сеть Хопфилда и сеть Хемминга[42,43]. Для обучения нейронных сетей предложен генетический алгоритм.

Разрабатываемые инструментальные средства были протестированы. В качестве тестовых задач была выбрана задача поиска достопримечательностей в парке [40,41] и «Искусственная жизнь».

Задача о поиске достопримечательностей имеет следующую формулировку: некая персона ищет достопримечательность в парке и пытается добыть информацию об ее месторасположении с помощью карты местности и информаторов. Получив информацию от информатора-человека, персона направится к остановке (трамвая), вместо того, чтобы исключительно пешком добираться до интересующей достопримечательности. Если в качестве источника информации используется карта, то персона добирается до цели пешком и т.д. Задача «Искусственная жизнь» достаточно хорошо известна.

Результаты тестовой задачи «Поиск достопримечательностей» приведены ниже в таблице.

Было проведено моделирование в среде размерами 400 на 600 точек, при условии наличия одной цели-достопримечательности, и всего было совершено 100 тестовых прогонов модели. В результате моделирования были получены результаты, показывающие возможность использования такого типа агентов для решения данной задачи (таблица 1).

Таблица 1. Результаты моделирования “поиск в парке” продукционными агентами

Кол-во агентов-информаторов	Среднее время поиска цели (в секундах)
1	29.6
3	27.9
5	19.5

Модель агента, основанная на нейронных сетях, была реализована в виде 2-слойного персептрона, имеющего 6 нейронов входного уровня и 2 нейрона выходного слоя. В качестве входных данных на входной слой сети от рецепторов передавалась следующая информация: (а) факт наличия информатора в поле видимости – info; (б) расстояние до информатора (если в поле видимости их несколько, то учитывается расстояние до ближайшего) - $d_{inf.}$;(в) расстояние до цели - d_f.; и т.д.

Был проведен имитационный эксперимент в среде размерами 400 на 600 точек, при условии наличия одной цели-достопримечательности, и всего было совершено 100 тестовых прогонов модели. В результате моделирования были получены результаты, показывающие возможность использования такого типа агентов для решения данной задачи (таблица 2).

Таблица 2. Результаты моделирования “поиск в парке” нейронными агентами

Количество агентов-информаторов	Среднее время поиска цели (в секундах)
1	37.6
3	33.5
5	25.1

Заключение

Итак, авторами была спроектирована агентно-ориентированная система имитации и разработан ее прототип. При разработке прототипа системы имитации авторы старались придерживаться требований к агентным системам имитации (кроссплатформенность, иерархическое представление модели, адаптируемость, операции над моделью (добавление, удаление объектов, добавление, удаление связей между агентами), поддержка распределенного (параллельного) моделирования и т.д).

Кроссплатформенность в разрабатываемой агентной системе достигается за счет использования языка Scala.

Особое внимание уделялось разработке алгоритма синхронизации, который поддерживает распределенное моделирование. Известно, что распределенная модель представляет собой совокупность логических процессов, расположенных на различных вычислительных узлах. Логические процессы необходимо синхронизировать, чтобы обеспечить каузальность событий распределенной модели. В результате проведенных исследований в агентной системе имитации был разработан алгоритм синхронизации, основанный на классическом оптимистическом алгоритме. Алгоритм использовал знания о модели. Исследования показали, что использование знаний о модели значительно увеличивает эффективность алгоритма. В качестве формальной модели была использована модель акторов, наиболее адекватным образом отображающее поведение агентов в распределенной (параллельной) среде.

Еще одним важным звеном в построении агентной модели является разработка интеллектуальных агентов. Большинство систем агентного моделирования не располагают средствами, поддерживающими функционирование интеллектуальных агентов, а наличие их

в системе дает возможность строить более адекватные модели. Были разработаны агенты, построенные на продукционных правилах и искусственных нейронных сетях. Были построены две тестовые модели и проведено по 100 прогонов каждой из разработанных тестовых моделей в течение определенного времени: для моделей задачи “искусственная жизнь” – 300 временных тактов среды, для задачи о поиске в парке – пока агент не найдет цель (но не более 1000 тактов). Результаты экспериментов показали состоятельность разработанных инструментальных средств и принятых решений при их проектировании.

Выполненная работа по проектированию и реализации агентной системы имитации является актуальной, поскольку в настоящее время стоит вопрос о широком практическом применении имитационной экспертизы[17]. Считается, что любые проектные работы по созданию и модернизации систем должны быть предварены имитационным исследованием «с получением практических выводов и методических рекомендаций по вопросам целесообразности существования, построения, функционирования или модернизации системы».

Список литературы

1. Macal, C. M., and M. J. North. 2005. Tutorial on agent-based modeling and simulation. //Proceedings of the 2005 Winter Simulation Conference. eds. M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines. 2-15. Piscataway, New Jersy: Institute of Electrical and Electronics Engineers
2. Macal, C. M., and M. J. North. 2009. Agent-based modeling and simulation. //Proceedings of the 2009 Winter Simulation Conference. eds. M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R.G. Ingalls. 86-98. Piscataway, New Jersy: Institute of Electrical and Electronics Engineers.
3. Клышинский Э.С. Проектирование элементов принятия решений в имитационных моделях бизнес-систем. Автореферат на соискание канд. диссертации. [Электронный ресурс] [Режим доступа: http://www.dissercat.com] [Проверено:25.04.2014]
4. Ивашкин Мультиагентное имитационное моделирование маркетинговых ситуаций. Автореферат на соискание канд. диссертации. [Электронный ресурс] [Режим доступа: http://www.dissercat.com] [Проверено:25.04.2014]
5. Грибанова Е.Б. Алгоритмы и комплекс программ для решения задач имитационного моделирования объектов прикладной экономики. Автореферат на соискание канд. диссертации. [Электронный ресурс] [Режим доступа: http://www.dissercat.com] [Проверено:25.04.2014]
6. Борщёв А. От системной динамики и традиционного имитационного моделирования к практическим агентным моделям: причины, технология, инструменты. [Электронный ресурс] [Режим доступа: http://www.gpss.ru/paper/borshevarc.pdf] [Проверено:21.06.2013]
7. Borshchev A., Karpov Y., Kharitonov V. Distributed Simulation of Hybrid Systems with AnyLogic and HLA. // Parallel computing technologies. – 2002. № 18(6). – P.829-839.
8. Система моделирования “AnyLogic” [Электронный ресурс] [Режим доступа: http://www.xjtek.ru] [Проверено:25.04.2014]
9. Аксенов К.А., Антонова А.С., Киселева М.В. Моделирование процесса выпуска металлургической продукции в системах AnyLogic и BPSim.MAS // Сборник докладов шестой всероссийской научно-практической конференции «Имитационное моделирование. Теория и практика» (ИММОД-2013). Том 2. // ISBN 978-5-9690-0221-0 // Издательство «ФЭН» Академии наук РТ, Казань, 2013, с. 13-18.
10. Аксенов К.А., Ван Кай, Аксенова О.П. Разработка и применение метода анализа узких мест на основе мультиагентного имитационного моделирования // Сборник докладов шестой всероссийской научно-практической конференции «Имитационное моделирование. Теория и практика» (ИММОД-2013). Том 2. // ISBN 978-5-9690-0221-0 // Издательство «ФЭН» Академии наук РТ, Казань, 2013, с. 19-23.

11. Аксенов К.А., Смолий Е.Ф., Аксенова О.П. BPSim – система мультиагентного имитационного моделирования бизнес-процессов и организационно-технических систем // Сборник докладов шестой всероссийской научно-практической конференции «Имитационное моделирование. Теория и практика» (ИММОД-2013). Том 1. // ISBN 978-5-9690-0221-0 // Издательство «ФЭН» Академии наук РТ, Казань, 2013, с. 13-20.
12. Система моделирования “Repast” [Электронный ресурс] [Режим доступа: http://repast.sourceforge.net] [Проверено:25.04.2014]
13. Система моделирования “NetLogo” [Электронный ресурс] [Проверено:25.04.2014]
14. Система моделирования “MASON” [Электронный ресурс] [Режим доступа: http://cs.gmu.edu/~eclab/projects/mason] [Проверено:25.04.2014]
15. Система моделирования “Ascape” [Электронный ресурс] [Режим доступа: http://ascape.sourceforge.net/index.html#Contact] [Проверено:25.04.2014]
16. Система моделирования SWARM. [Электронный ресурс][режим доступа:www.swarm.org] [Проверено:25.04.2014]
17. Власов С.А., Девятков В.В., Назмеев М.М. Имитационная экспертиза: опыт применения и перспективы // Сборник докладов шестой всероссийской научно-практической конференции «Имитационное моделирование. Теория и практика» (ИММОД-2013). Том 1. // ISBN 978-5-9690-0221-0 // Издательство «ФЭН» Академии наук РТ, Казань, 2013, с. 54-63.
18. Замятина Е.Б., Миков А.И. Программные средства системы имитации Triad.Net для обеспечения ее адаптируемости и открытости. Информатизация и связь. №5, 2012, АНО «Редакция журнала «Информатизация и связь», ISSN 2078-8320, С.130-133.
19. Mikov A.I. Simulation and Design of Hardware and Software with Triad// Proc.2nd Intl.Conf. on Electronic Hardware Description Languages. Las Vegas, USA, 1995. P. 15 20.
20. Mikov A.I. Formal Method for Design of Dynamic Objects and Its Implementation in CAD Systems //Gero J.S. and F.Sudweeks F.(eds), Advances in Formal Design Methods for CAD, Preprints of the IFIP WG 5.2 Workshop on Formal Design Methods for Computer-Aided Design. Mexico, 1995. P.105-127.
21. Замятина Е.Б., Миков А.И., Михеев Р.А. Лингвистические и интеллектуальные инструментальные средства симулятора компьютерных сетей TRIADNS. International Journal “Information theories & Applications (IJ ITA). Vol 19, Number 4, 2012, pp.355-368. ITHEA, Sofia, 1000, P.O.B. 775, Bulgaria. ISSN 1310-0513 (printed)
22. Mikov A.,Zamyatina E.,Mikheev. Linguistic and Program Tools For Debugging and Testing Of Simulation Models Of Computer Networks. International Journal “Information Models and Analyses, V.2., N 1,2013, ITHEA, ISSN 1314-6416, Sofia., 1000., P.O.B. 775, Bulgaria, pp. 70-80
23. Mikov A., Zamyatina E., Kubrak E. An Ontology-based Approach to the Incomplete Simulation Model Analisis and its Automatic Completion. International Journal “Information Technologies & Knowledge”, 2009, Volume 3, Number 2, pp. 169-186.
24. Сухов А.О. Трансформация визуальных моделей в системе MetaLanguage / Современные проблемы математики и ее прикладные аспекты – 2013. Сборник тезисов конференции. – Пермь: Пермский государственный национальный исследовательский университет, 2013. – С. 44.
25. Сухов А.О. Разработка предметно-ориентированных языков на основе онтологий / Современные проблемы математики и ее прикладные аспекты – 2013. Сборник тезисов конференции. – Пермь: Пермский государственный национальный исследовательский университет, 2013. – С. 45.
26. Замятина Е.Б, Лядова Л.Н., Сухов А.О.. Мультиязыковое моделирование с использованием DSM платформы MetaLanguage. Информатизация и связь. №5, 2013, АНО «Редакция журнала «Информатизация и связь», ISSN 2078-8320, С.11-15.

27. Fujimoto R.M. Distributed Simulation Systems. In Proceedings of the 2003 Winter Simulation Conference S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, eds. The 2003 Winter Simulation Conference 7-10 December 2003. The Fairmont New Orleans, New Orleans, LA, pp. 124-134
28. Bryant R.E. Simulation of packet communication architecture computer systems. Technical Report MIT-LCS-TR-188, Massachusetts Institute of Technology, 1977.
29. Chandy K.M, Misra J. Distributed simulation: A case study in design and verification of distributed programs. IEEE Transactions on Software Engineering, SE-5(5): p.440-452, September, 1979.
30. Jefferson D.R. Virtual Time. ACM Transactions on Programming Languages and Systems, 1985. 7(3): p.404-425.
31. Wilson L. F., Shen W. Experiments in load migration and dynamic load balancing in Speedes // Proc. of the Winter simulation conf. /Ed. by D. J. Medeiros, E. F.Watson, J. S. Carson, M. S. Manivannan. Piscataway (New Jersey): Inst. of Electric. and Electron. Engrs, 1998. P. 487–490.
32. Zheng G. Achieving high performance on extremely large parallel machines: Performance prediction and load balancing: Ph.D. Thesis. Department Comput. Sci., Univ. of Illinois at Urbana-Champaign, 2005. 165 p. [Electron. resource]. http://charm.cs.uiuc.edu/.
33. Миков А.И., Замятина Е.Б., Козлов А.А. Мультиагентный подход к решению проблемы равномерного распределения вычислительной нагрузки. Natural and Artificial Intelligence, ITHEA, Sofia, Bulgaria, 2010, pp.173-180.
34. Миков А.И., Замятина Е.Б. Проблемы повышения эффективности и гибкости систем имитационного моделирования. Проблемы информатики,№4(8), Новосибирск, Институт вычислительной математики и математической геофизики СО РАН, 2010, стр.49-64
35. Замятина Е.Б., Миков А.И. Применение онтологий и принципов организации сервис-ориентированно архитектуры при проектировании и реализации системы имитационного моделирования. Материалы 3-ьей Международной научно-технической конференции «Технологии разработки информационных систем ТРИС-2012», Т.1., Таганрог, издательство Технологического института ЮФУ, Ростов –на-Дону, 2012, 9 сентября, стр. 61-65.
36. Kolevatov G.A., Zamyatina E.B. Simulation Analysis Framework Based on Triad.Net. Proceedings of the 6-th Spring/Summer Young Reseachers' Colloquium on Software Engineering. SYRCoSE 2012, Perm, May 30-31, 2012-Perm, Russia,pp.160-163.
37. Odersky M. The Scala Programming Language [Электронный ресурс]. URL: http://www.scala-lang.org/node/25 (дата обращения: 06.06.2013)
38. Замятина Е., Ермаков С. Алгоритм синхронизации объктов распределенной имитационной модели в TRIAD.Net. Applicable Information Models. ITHEA, Sofia, Bulgaria, 2011, ISBN: 978-954-16-0050-4, стр.211-220
39. Миков А.И., Замятина Е.Б. Использование GPU для повышения производительности симулятора компьютерных сетей. Высокопроизводительные вычисления на графических прцессорах: тезисы докл. Науч.-практ.конф. с междунар. Участием с элементами науч. шк. для молодежи, 21-25 мая 2012 г. Перм. гос. нац. иссл. ун-т..-Пермь, 2012, стр.53-57.
40. Замятина Е.Б., Чудинов Г.В. Разработка и использование программных средств для построения и исследования агентных имитационных моделей. / Е.Б. Замятина, Г.В. Чудинов // Вестник пермского университета. Математика, механика, информатика, 2010, №2(2), С. 80 – 84.
41. Dubiel B., Tsimhoni O. Integrating agent based modelling into a discrete event simulation . In the Proceedings of the 2005 Winter Simulation Conference, ed. Kuhl M. E., Steiger N. M., Armstrong F. B., and Joines J. A., 2005, pp. 1029 1037. URL: http://www.informs-cs.org/wsc05papers/123.pdf. [Проверено:25.04.2014]
42. Хайкин С. Нейронные сети: Полный курс. – Москва: Вильямс, 2006. – 31-89 с.
43. Круглов В.В. Искусственные нейронные сети. – Москва: Телеком, 2002. – 63-79 с.

Благодарности: Работа была выполнена при финансовой поддержке гранта РФФИ №12-07-00302- а, гранта РФФИ №13-07-96506_юг_а

Информация об авторах:

Замятина Елена Борисовна, Пермский государственный национальный исследовательский университет (ПГНИУ), Пермский филиал НИУ ВШЭ, e_zamyatina@mail.ru, 614017, Пермь, Тургенева, 33, 40, +7(342)2826304, доцент, доцент, к.ф.-м.н.

Каримов Данил Фаризович, выпускник Пермского национального исследовательского исследовательского университета (ПГНИУ), магистр прикладной математики и информатики электронный адрес: googlicus@gmail.com, 614010, Пермь, ул. Куйбышева, 101, кв. 93.

Митраков Артем Андреевич, выпускник Пермского национального исследовательского исследовательского университета (ПГНИУ), магистр прикладной математики и информатики электронный адрес: mitrakov-artem@yandex.ru, 614070, Пермь, Ким 109,19.