doi 10.17059/2015-2-24
УДК: 314.114:004.94

***В. В. Окрепилов*** [б], ***В. Л. Макаров*** [а], ***А. Р. Бахтизин*** [а], ***С. Н. Кузьмина*** [б]

[а] Центральный экономико-математический институт РАН

[б] Государственный региональный центр стандартизации, метрологии и испытаний в г. Санкт-Петербурге и Ленинградской области

# ПРИМЕНЕНИЕ СУПЕРКОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ ДЛЯ МОДЕЛИРОВАНИЯ СОЦИАЛЬНО-ЭКОНОМИЧЕСКИХ СИСТЕМ[1]

*К настоящему времени накоплен значительный опыт в исследовании проблематики в области качества, оценки систем управления, моделирования устойчивости экономических систем.*

*Проведенные исследования создали основу для формирования нового научного направления — экономики качества, инструменты которого позволяют использовать возможности имитационного моделирования при построении математических моделей, адекватно отражающих роль качества в естественных, технических, социальных закономерностях функционирования сложных социально-экономических систем.*

*Широкое использование и разработка моделей, а также системное моделирование с использованием суперкомпьютерных технологий, по нашему глубокому убеждению, выведут проводимые исследования социально-экономических систем на принципиально новый уровень. Кроме того, данное научное направление вносит заметный вклад в имитационное моделирование мультиагентных социальных систем, и, что не менее важно, оно относится к приоритетным направлениям в развитии науки и техники в нашей стране.*

*В статье уделено внимание вопросам применения суперкомпьютерных технологий в общественных науках, в первую очередь, — в части технической реализации крупномасштабных агент-ориентированных моделей (АОМ). Суть данного инструмента в том, что благодаря увеличению мощности компьютеров стало возможным описывать поведение многих отдельных фрагментов сложной системы, каким являются социально-экономические системы.*

*Также в статье будет рассмотрен опыт зарубежных ученых и практиков в запуске АОМ на суперкомпьютерах, а также пример АОМ, разработанной в ЦЭМИ РАН. Проанализированы этапы и методы эффективного отображения счетного ядра мультиагентной системы на архитектуру современного суперкомпьютера. В заключение представлены эксперименты на основе имитационного моделирования по трем сценариям прогнозирования численности населения Санкт-Петербурга как одного из важнейших факторов, влияющих на развитие социально-экономической системы и качество жизни населения.*

**Ключевые слова:** качество жизни, модель, прогноз, система, суперкомпьютерные технологии, экономика качества

## Основная часть

Для имитации поведения сложных социально-экономических систем, а также оценки многоуровневых систем управления качеством [1] наиболее адекватным инструментом являются активно развивающиеся в последнее время агент-ориентированные модели (АОМ). Основная идея, лежащая в основе моделей этого класса, заключается в построении вычислительного инструмента, представляющего собой совокупность агентов с определенным набором свойств и позволяющего проводить симуляции реальных явлений. От объектоориентированных моделей АОМ отличаются активностью своих элементов, каждый из которых обладает не только заданным набором личностных характеристик (ресурсов), но и целевой функцией (интересами), на основе чего имитируется его реакция на изменения внешней среды, затрагивающие сферу его интересов (поведение). Появление АОМ можно рассматривать как результат эволюции методологии моделирования: переход от мономоделей (одна модель — один алгоритм) к мультимоделям (одна модель — множество независимых алгоритмов). Таким образом, АОМ представляет собой искусственное общество, состоящее из взаимодействующих между собой самостоятельных агентов, что позволяет смоделировать систему, максимально приближенную к ре-

[1] © Окрепилов В. В., Макаров В. Л., Бахтизин А. Р., Кузьмина С. Н. Текст. 2015.

альности. Агентный подход к моделированию очень универсален и удобен для прикладников в силу свой наглядности, но его также отличает и требовательность к вычислительным ресурсам. Очевидно, что прямое моделирование достаточно длительных социальных процессов в масштабах страны (или планеты) в целом требует весьма значительной вычислительной мощности.

В свою очередь, суперкомпьютеры позволяют на несколько порядков увеличить число агентов и других количественных характеристик (узлов сети, величины территории) в моделях, первоначально разработанных для использования в обычных настольных компьютерах. Поэтому суперкомпьютерное моделирование является логичным и крайне желательным шагом для тех упрощенных моделей, которые уже прошли успешную практическую апробацию на обычных компьютерах. Однако специфика архитектуры современных компьютеров не гарантирует, что программное обеспечение компьютерной модели немедленно заработает и на суперкомпьютере. Требуется как минимум распараллеливание счетного ядра, а зачастую и его глубокая оптимизация, поскольку в ином случае применение дорогостоящего суперкомпьютерного счета будет не так уж оправдано.

В этой связи отметим, что практически на всех мировых конгрессах, посвященных разработкам АОМ, одной из главных проблем в этой области знаний, ставят проблему запуска АОМ на суперкомпьютерах.

Компьютерное моделирование — широчайшая, интереснейшая и интенсивно развивающаяся область исследований, востребованная сегодня практически во всех сферах человеческой деятельности. Агент-ориентированный подход к моделированию универсален и удобен для прикладных исследователей и практиков в силу своей наглядности, но при этом предъявляет высокие требования к вычислительным ресурсам. Очевидно, что для прямого моделирования достаточно длительных социальных процессов в масштабах страны и планеты в целом нужны весьма значительные вычислительные мощности.

## 1. Опыт зарубежных ученых и практиков

Рассмотрим наиболее известные примеры запусков агент-ориентированных моделей на суперкомпьютерах. В сентябре 2006 г. стартовал проект по разработке крупномасштабной АОМ европейской экономики — EURACE, то есть Europe ACE (Agent-based Computational Economics) с очень большим числом автономных агентов, взаимодействующих в рамках социально-экономической системы [2]. В проект вовлечены экономисты и программисты из восьми научно-исследовательских центров Италии, Франции, Германии, Великобритании и Турции, а также консультант из Колумбийского университета США — нобелевский лауреат Джозеф Стиглиц.

Чтобы преодолеть ограничения широко распространенных моделей, рассматривающих агрегированных агентов, а также предполагающих их рациональное поведение и состояние равновесия, в основу исследования была положена методология ACE (Agent Based Computational Economics — агент-ориентированная экономика).

Для модели используется географическая информационная система, охватывающая широкий перечень объектов: предприятия, магазины, школы, транспортные сети и т. д.

Как отмечают разработчики, практически все существующие АОМ рассматривают либо отдельную отрасль, либо относительно небольшой географический район и, соответственно, небольшую популяцию агентов, а в EURACE представлен весь Европейский союз, так что по масштабам и сложности эта модель является уникальной, а ее численное разрешение требует использования суперкомпьютеров, а также специального программного обеспечения.

Для наполнения модели статистической информацией использовались данные (в виде геоинформационных карт) статистической службы Европейского союза уровня NUTS-2[1], представляющие сведения о 268 регионах 27 стран.

В модели три типа агентов: домашние хозяйства (около 107), предприятия (около 105) и банки (около 102). Все они имеют географическую привязку, а также связаны друг с другом посредством социальных сетей, деловых отношений и т. д.

EURACE реализована с помощью гибкой масштабируемой среды для моделирования агентных моделей — FLAME (Flexible Large-scale Agent Modeling Environment), разработан-

[1] NUTS (фр. Nomenclature des unités territoriales statistiques) — Номенклатура территориальных единиц для целей статистики, представляющая собой стандарт территориального деления стран для статистических целей, разработанный Европейским союзом и охватывающий входящие в него страны. Существуют NUTS-единицы трех уровней, при этом второй уровень (NUTS-2) соответствует административным округам в Германии, графствам в Великобритании и т. д.

ной Симоном Коакли и Майклом Холкомбом[1] первоначально для моделирования роста клеток, выращиваемых в различных условиях. Используемый в FLAME подход основан на так называемых X-машинах, напоминающих конечные автоматы, но отличающихся от них тем, что у каждой X-машины есть набор данных (своего рода память), а также тем, что переходы между состояниями являются функциями не только состояния, но и набора данных памяти.

Итак, каждый агент в системе FLAME представлен X-машиной, причем предусмотрено общение между агентами посредством передачи сообщений. При работе в параллельном режиме на суперкомпьютере обмен сообщениями между агентами требует больших вычислительных затрат, в связи с чем агенты изначально были распределены по процессорам в соответствии с их географическим положением. Тем самым разработчики программы минимизировали вычислительную нагрузку, исходя из предположения, что в основном общение между агентами происходит в рамках небольшой социальной группы, локализованной в определенной местности. Таким образом, весь модельный ландшафт был поделен на небольшие территории и распределен между узлами суперкомпьютера.

С помощью разработанной модели был проведен ряд экспериментов с целью исследования рынка труда. Не рассматривая детально полученные числовые результаты, отметим, что, по мнению авторов, основной вывод исследования заключается в том, что макропоказатели двух регионов со схожими условиями (ресурсы, развитие экономики и т. д.) в течение продолжительного периода (10 лет и более) могут значительно разойтись за счет первоначальной неоднородности агентов.

В АОМ EpiSims, разработанной исследователями из Института биоинформатики Вирджинии (Virginia Bioinformatics Institute), рассматриваются как перемещения агентов, так и их контакты в рамках среды, максимально приближенной к реальности и содержащей дороги, здания и прочие инфраструктурные объекты. Для построения модели потребовался большой массив данных, включающий информацию о здоровье отдельных людей, их возрасте, доходе, этнической принадлежности и т. д.

Исходная цель исследования заключалась в построении для запуска на суперкомпьютере АОМ большой размерности, с помощью которой можно будет изучать распространение болезней в обществе. Однако впоследствии в процессе работы также решалась задача, связанная с созданием специализированного программного обеспечения ABM++, которое позволяет осуществлять разработку АОМ на языке C++, а также содержит функции, облегчающие распределение исполняемого программного кода на узлах кластеров суперкомпьютера. Помимо этого, ABM++ предоставляет возможность динамического перераспределения потоков вычислений, а также синхронизации происходящих событий.

ABM++, первая версия которого появилась в 2009 г., представляет собой результат модернизации инструмента, разработанного в 1990–2005 гг. в Лос-Аламосской национальной лаборатории в процессе построения крупномасштабных АОМ (EpiSims, TRANSIMS, MobiCom).

Межпроцессорные связи между вычислительными узлами в АОМ часто требуют синхронизации происходящих в модели событий. ABM++ позволяет разрабатывать модели, отвечающие этому требованию. К примеру, в социальных моделях агенты часто перемещаются между различными точками пространства (работа, дом и т. д.), а на программном уровне этому соответствует смена узла кластера, и здесь важно, чтобы модельное время принимающего узла было синхронизировано со временем узла, который агент только что покинул.

Также в ABM++ реализована библиотека MPIToolbox, которая соединяет интерфейс C++ API (Application Programming Interface) и Message Passing Interface (MPI) суперкомпьютера и ускоряет передачу данных между узлами кластеров.

ABM++ создавалось в Ubuntu Linux — операционной системе с компиляторами gcc/g++. В качестве интегрированной среды разработки рекомендуется пакет Eclipse с плагином для поддержки C и C++, а также с плагином PTP (Parallel Tools Platform), обеспечивающим разработку и интеграцию приложений для параллельных компьютерных архитектур. Eclipse поддерживает интеграцию с TAU (Tuning and Analysis Utilities) Performance System — инструментом для разностороннего анализа и отладки программ для параллельных вычислений, что также упрощает разработку агентных моделей.

Специалисты другой исследовательской группы из того же Института биоинформатики Вирджинии создали инструмент для изучения особенностей распространения инфекци-

[1] Более подробно см. www.flame.ac.uk.

онных заболеваний внутри различных групп населения — EpiFast, к достоинствам которого можно отнести масштабируемость и высокую скорость исполнения. К примеру, имитация социальной активности жителей Большого Лос-Анджелеса (агломерации с населением свыше 17 млн чел.) с 900 млн связей между людьми на кластере с 96 двухъядерными процессорами POWER5 заняла менее пяти минут. Такая достаточно высокая производительность обеспечивается предложенным авторами оригинальным механизмом распараллеливания [3].

Как правило, для реализации моделирования социума на группе процессоров используют несколько естественных способов. К примеру, один из них базируется на представлении социума в виде набора неструктурированных связей между индивидуумами. При распараллеливании эти связи равномерно распределяются на группы, количество которых соответствует числу процессоров. Минус такого подхода заключается в сложности отслеживания статуса отдельного человека, иными словами, если, к примеру, инфицирован индивидуум i, то информация об этом должна быть синхронизирована во всех группах, так как мы не знаем, в каких из них содержатся связи данного человека. Необходимость такой синхронизации влечет за собой большую вычислительную нагрузку. Другой подход основан на разделении людей — агентов модели — в соответствии с их географическим местоположением. Однако в этом случае, поскольку население обычно расселено неравномерно, распределение вычислительной нагрузки требует применения достаточно сложных алгоритмов для ее выравнивания, что также создает дополнительную вычислительную нагрузку.

Предлагаемый авторами метод заключается в равномерном распределении агентов с соответствующими им односторонними исходящими связями по группам, число которых равно числу процессоров. Вычислительный алгоритм основан на взаимодействии ведущих (master) и ведомых (slave) процессоров, организованном следующим образом: ведущий процессор знает, какой из процессоров обслуживает конкретного агента, а каждый из ведомых процессоров знает только обслуживаемых (локальных) агентов. В процессе вычислений ведомые процессоры посылают ведущим процессорам запросы относительно связей с нелокальными агентами. Ведущие процессоры не осуществляют никаких расчетов, кроме поиска нелокального агента для каждой внешней связи, и пересылают запросы соответствующим процессорам.

По мнению разработчиков, предложенный ими алгоритм позволяет проводить эффективные и высокоскоростные симуляции систем с очень большим числом агентов.

Классические модели распространения эпидемий преимущественно основывались на использовании дифференциальных уравнений, однако такой инструментарий затрудняет учет связей между отдельными агентами и их многочисленными индивидуальными особенностями. АОМ позволяют преодолеть указанные недостатки. В 1996 г. Джошуа Эпштейн и Роберт Акстелл опубликовали описание одной из первых АОМ, в которой рассматривается процесс распространения эпидемий [4]. Агенты модели, отличающиеся друг от друга восприимчивостью к заболеванию, которая зависит от состояния иммунной системы, распределены на определенной территории. При этом в данной модели агенты, число которых составляет всего несколько тысяч, реализуют достаточно примитивное поведение.

В дальнейшем под руководством Джошуа Эпштейна и Джона Паркера в Центре социальной и экономической динамики Брукингского института (Center on Social and Economic Dynamics at Brookings) была построена одна из самых больших АОМ, включающая данные о всем населении США, То есть порядка 300 млн агентов [5]. Данная модель имеет ряд преимуществ. Во-первых, она позволяет предсказать последствия распространения заболеваний различного типа. Во-вторых, она ориентирована на поддержку двух сред для вычислений: одна среда состоит из кластеров с установленной 64-битной версией Linux, а другая — из серверов с четырехъядерными процессорами и установленной системой Windows (в этой связи в качестве языка программирования был выбран Java, хотя разработчики и не уточняют, какую именно реализацию Java они использовали). В-третьих, модель способна поддерживать численность агентов от нескольких сотен миллионов до 6 миллиардов.

Способ распределения агентов между аппаратными ресурсами состоит из двух фаз: сначала агенты распределяются по компьютерам, задействованным в работе, а затем по потокам на каждом из компьютеров. В процессе работы модели каждый поток может остановиться (в заранее обусловленное время) для передачи сообщений другим потокам. Все подготовленные к отправке сообщения до определенного момента хранятся в пуле сообщений, а затем

одновременно отправляются. Также в реализации модели применяются две вспомогательные утилиты: первая управляет потоками на отдельном компьютере, вторая следит за тем, чтобы все сообщения между потоками были выполнены до момента возобновления вычислительного процесса.

При распределении агентов по аппаратным ресурсам следует учитывать два обстоятельства: 1) производительность узла напрямую зависит от числа инфицированных агентов; 2) контакты, предполагающие передачу сообщений между потоками, требуют гораздо больше вычислительных затрат, чем контакты, ограничивающиеся локальной информацией. Исходя из этого, возможны различные распределения агентов. С одной стороны, можно поделить все рассматриваемое географическое пространство на равные части, число которых должно соответствовать числу узлов, а затем определить для каждого узла какой-либо географический регион. Такое распределение позволяет сбалансировать вычислительную нагрузку между узлами. С другой стороны, можно закрепить определенную территорию, представляющую собой единую административную единицу, за конкретным узлом — в этом случае вычислительную нагрузку удастся сократить за счет снижения числа контактов, предполагающих передачу сообщений между потоками. Если первый способ влечет за собой увеличение вычислительной нагрузки за счет ресурсоемких контактов, то второй в ряде случаев чреват значительным дисбалансом между аппаратными ресурсами. К примеру, вспышка какого-либо заболевания в одном из регионов загрузит один из вычислительных узлов, в то время как некоторые другие будут простаивать.

Рассматриваемая модель (US National Model) включает 300 млн агентов, перемещающихся по карте страны в соответствии с матрицей корреспонденций размерностью 4000×4000, специфицированной с помощью гравитационной модели. На US National Model был проведен вычислительный эксперимент, имитирующий 300-дневный процесс распространения болезни, которая характеризуется 96-часовым периодом инкубации и 48-часовым периодом заражения. В исследовании, в частности, было установлено, что распространение заболевания идет на спад, после того как 65 % зараженных выздоровели и приобрели иммунитет. Эту модель неоднократно использовали специалисты Университета Джонса Хопкинса (Johns Hopkins University), а также Департамента национальной безопасности США для исследований, посвященных стратегии быстрого реагирования на различного рода эпидемии [6].

В 2009 г. была создана вторая версия US National Model, включающая 6,5 млрд агентов, спецификация действий которых проводилась с учетом имеющихся статистических данных. С ее помощью имитировали последствия распространения вируса гриппа A(H1N1/09) в масштабах всей планеты.

Ранее подобная модель была разработана в Лос-Аламосской национальной лаборатории (США), и результаты работы с ней были представлены в широкой печати 10 апреля 2006 г. [7]. Для технической реализации модели был использован один из мощнейших по тем временам суперкомпьютеров с именем Pink, состоящий из 1024 узлов, с двумя процессорами с частотой 2,4 ГГц и 2 ГБ памяти на каждом. С помощью этой крупномасштабной модели, включающей 281 млн агентов, были рассмотрены сценарии распространения различных вирусов, в том числе и H5N1, с учетом тех или иных оперативных вмешательств: вакцинации, закрытия школ и введения карантина на некоторых территориях.

Естественно, для подобных задач можно использовать не только суперкомпьютеры, но и более дешевые решения. Показателен пример ученых Университета Ньюкасла из Австралии (The University of Newcastle, Australia), построивших кластер, включающий 11 узлов — отдельных персональных компьютеров, объединенных в сеть с пропускной способностью до 100 Мбит/с [8]. Программное обеспечение узлов кластера было одинаковое: операционная система Debian GNU/Linux и JavaParty — кроссплатформенное программное расширение языка Java (ПО JavaParty располагает средствами для распределения потоков выполнения кода по узлам кластеров). Для оценки производительности кластера были использованы три версии одной АОМ, в рамках которой тестировались различные стратегии агентов, принимающих участие в аукционе. Версии различались сложностью стратегий агентов.

Основной вывод тестирования заключается в том, что использование кластеров оправданно лишь тогда, когда вычислительная нагрузка отдельных узлов достаточно интенсивная; в противном случае распределение потоков, наоборот, уменьшает скорость работы модели. Отметим от себя, что такое поведение кластерных приложений неспецифично.

Рассмотрим RepastHPC, которая является средой проектирования агент-ориентированных моделей для высокопроизводительных вычислений. Отдельного внимания заслуживает программное обеспечение, разработанное для проектирования АОМ с целью последующего запуска на суперкомпьютерах, — Repast for High Performance Computing (RepastHPC). Данный пакет реализован с использованием языка C++ и MPI — программного интерфейса для обмена сообщениями между процессами, выполняющими задачу в параллельном режиме, а также библиотеки Boost, расширяющей C++.

В рамках RepastHPC реализован динамический дискретно-событийный планировщик выполнения программных инструкций с консервативными алгоритмами синхронизации, предусматривающими задержку процессов для соблюдения определенной очередности их выполнения.

В RepastHPC агенты распределяются между процессами, и каждый процесс связан с агентом, являющимся локальным по отношению к данному процессу. В свою очередь, агент локален к процессу, выполняющему программный код, описывающий поведение данного агента. При этом копии остальных — нелокальных — агентов могут присутствовать в любом процессе, что позволяет агентам всей модели взаимодействовать с этими копиями. К примеру, пусть пользователь в своей модели, предполагающей параллельные вычисления, использует два процесса — P1 и P2, каждый из которых создает определенное количество агентов и имеет собственный планировщик выполнения программных инструкций. Агенты, поведение которых рассчитывается на процессе P1, являются локальными по отношению к данному процессу, и только в рамках данного процесса программный код может изменить их состояние (аналогично и для процесса P2). Предположим, процесс P1 запрашивает копию агента A2 из процесса P2. Агент A2 не является локальным по отношению к процессу P1, и, соответственно, программный код, выполняемый в рамках процесса P1, не может изменить состояние агента A2. При этом агенты, реализуемые в рамках процесса P1, при необходимости могут запросить состояние агента A2, но копия A2 останется неизменной. Изменение оригинального A2 возможно только в рамках процесса P2, но в этом случае RepastHPC синхронизирует изменения состояния агента между всеми процессами.

## 2. Адаптация агент-ориентированных моделей для суперкомпьютера: наш подход

На суперкомпьютере «Ломоносов» была запущена модель, имитирующая развитие социально-экономической системы России на протяжении последующих 50 лет. Эта АОМ основана на взаимодействии 100 млн агентов, условно представляющих социально-экономическую среду России. Поведение каждого агента задано набором алгоритмов, которые описывают его действия и взаимодействие с другими агентами в реальном мире.

В проекте участвовали специалисты ЦЭМИ РАН (В.Л. Макаров, А.Р. Бахтизин) и сотрудники МГУ (В.А. Васенин, В.А. Роганов, И.А. Трифонов). Данные для моделирования были предоставлены Федеральной службой государственной статистики и Российским мониторингом экономического положения и здоровья населения. Модель для обычного компьютера была конвертирована в суперкомпьютерную версию [9]. Ниже мы рассмотрим основные этапы и методы эффективного отображения счетного ядра мультиагентной системы на архитектуру современного суперкомпьютера, наработанные нами в процессе решения соответствующей задачи.

*Проблема масштабирования.* Важно понимать, что масштабирование программ для суперкомпьютеров является фундаментальной проблемой. Хотя обычная и суперкомпьютерная программы реализуют один и тот же функционал, целевые функции их разработки, как правило, разные.

При начальной разработке сложного прикладного программного обеспечения в первую очередь стараются сократить издержки на программирование, обучение персонала, повысить переносимость между платформами и т. д., а оптимизацию откладывают на потом. И это вполне разумно, так как на ранних стадиях приоритетом разработки является исключительно функционал.

Однако когда разработанное программное обеспечение уже начало внедряться, часто выясняется, что на больших реальных данных не хватает производительности. А поскольку современные суперкомпьютеры — это вовсе не разогнанные в тысячи раз персональные компьютеры, для запуска на суперкомпьютере программу приходится существенно видоизменять. Причем эффективно сделать это без специальных знаний и навыков удается далеко не всегда.

При грамотно проведенной работе значительное повышение эффективности обычно достигается на трех уровнях:

1) распараллеливание счета;

2) специализация вычислительных библиотек по задачам;

3) низкоуровневая оптимизация.

*Специализация и низкоуровневая оптимизация*. Прежде чем всерьез говорить об использовании суперкомпьютеров, программу следует максимально оптимизировать и адаптировать к целевой аппаратной платформе. Если этого не сделать, параллельная версия будет лишь хорошим тестом для суперкомпьютера, а сам счет будет весьма неэффективным.

Использовать суперкомпьютер без оптимизации и адаптации программы к целевой аппаратной платформе — все равно, что послать на боевое задание полк новобранцев: их нужно сначала хорошо обучить выполнять их задачу (специализация, оптимизация программного обеспечения), а также эффективно владеть оружием (низкоуровневая оптимизация программного обеспечения), и вот тогда это будет эффективным использованием ресурсов.

В универсальных системах моделирования типа AnyLogic предоставляемые процедуры универсальны. А универсальный код часто можно оптимизировать для конкретного семейства задач.

*Выбор системы поддержки моделирования*. Безусловно, АОМ можно программировать и без специальной среды, на любом объектоориентированном языке. При этом основным недостатком существующих пакетов для создания АОМ, кроме RepastHPC, является невозможность разработки проектов, выполняющихся на вычислительном кластере (то есть не предусмотрен механизм распараллеливания процесса выполнения программного кода).

Однако более разумным подходом будет использование одной из хорошо зарекомендовавших себя систем для АОМ по причине унифицированной реализации типичных способов взаимодействия агентов. В целях экономии места здесь мы рассмотрим только систему ADEVS[1].

ADEVS представляет собой набор низкоуровневых библиотек для дискретного моделирования, выполненных на языке C++. Из достоинств стоит отметить:

— простоту реализации;

— высокую производительность моделей;

— поддержку основных численных методов при построении моделей;

— встроенное распараллеливание симуляции при помощи OpenMP;

— возможность применения стандартных средств распараллеливания;

— достаточно быстрое развитие библиотек в текущий момент;

— кроссплатформенность;

— низкоуровневость (текущий функционал не накладывает никаких ограничений на модель);

— независимость скомпилированного кода от нестандартных библиотек;

— открытый исходный код.

Однако существенными недостатками этого продукта являются полное отсутствие средств презентации и достаточно сложная, по сравнению, например, с AnyLogic, разработка моделей. Поэтому этот продукт не может использоваться для построения моделей на уровне заказчика, однако представляет собой эффективную платформу для реализации параллельных симуляторов.

Основными элементами программы с использованием библиотеки ADEVS при построении АОМ обычно являются:

— симулятор adevs:: Simulator < X >;

— примитив агентов adevs:: Atomic< X >;

— модель (контейнер агентов) adevs:: Digraph < VALUE, PORT >.

В силу перечисленных выше достоинств системы ADEVS суперкомпьютерную версию описываемой ниже программы было решено реализовывать именно на ее базе. В рамках данной работы были разработаны MPI-версия симулятора ADEVS, а также система визуализации процесса счета на базе библиотеки Qt — кроссплатформенного инструментария разработки ПО на языке программирования C++.

Далее мы переходим к краткому описанию разработанной модели и процедуры ее последующего запуска на суперкомпьютере.

*Исходная агент-ориентированная модель*. Первый этап разработки описываемой ниже АОМ заключался в построении инструмента, эффективно решающего задачу исследования на обычных компьютерах, а также в настройке параметров модели. После ее успешной апробации с небольшим числом агентов (около 20 тыс. — такова численность агентов, над которой способен производить вычисления с удовлетворительной скоростью персональный компьютер с хорошей производительностью, учитывая сложность агентов) было решено кон-

[1] Система ADEVS и ее описание [Электронный ресурс]. URL: http://www.ornl.gov/~1qn/adevs, свободный. Загл. с экрана. Яз. англ. (дата обращения: май 2014 г.).

вертировать модель для суперкомпьютера — в этом состоял второй этап разработки. На первом этапе был использован пакет AnyLogic, технические возможности которого позволили достаточно быстро отладить модель и настроить ее параметры. Затем модель для обычного компьютера была конвертирована в суперкомпьютерную версию.

Изображение рабочего окна разработанной АОМ представляет собой картинку, на которой отображаются точками агенты, отдельной областью предоставляется в правом верхнем углу информация о количестве агентов, а также картографическая информация и параметры выбранного региона, а именно: название, численность населения, ВВП и другие данные, которые задает для отображения на экране пользователь. В процессе работы системы можно получать оперативную информацию о социально-экономическом положении всех регионов России, в том числе с использованием картографической информации, меняющейся в режиме реального времени в зависимости от значений эндогенных параметров.

Спецификация агентов модели осуществлялась с учетом следующих основных параметров: возраст; продолжительность жизни; специализация родителей; место работы; регион проживания; доход и др.

Спецификация регионов осуществлялась с учетом следующих параметров: географические границы; число жителей; число работников (по типам); ВРП; ВРП на душу населения; объем инвестиций; объем инвестиций на душу населения; средняя заработная плата; средняя продолжительность жизни; показатель прироста населения и др. Все эти параметры учитывались при апробации методологических подходов к управлению экономическими системами [10], определению социальных закономерностей [11], а также при разработке Стратегии — 2030 с целью определения значений показателей, влияющих на главную цель Стратегии — качество жизни населения.

Для наполнения модели данными использовались статистические сборники Росстата, а также социологические базы данных RLMS.

*Конвертация модели в суперкомпьютерную программу.* Выше мы уже обсуждали проблемы, связанные с использованием средств разработки АОМ для реализации проектов, выполняющихся на вычислительных кластерах суперкомпьютера. У AnyLogic ввиду трудностей отделения счетной части от презентативной и из-за реализации кода на языке высокого уровня Java производительность выполнения кода существенно ниже, чем у ADEVS, и, кроме того, очень сложно (либо очень трудозатратно) переработать генерируемый код в параллельно выполняемую программу.

Ниже приведен алгоритм конвертации модели AnyLogic в суперкомпьютерную программу.

*Трансляция модели.* Модели в проекте AnyLogic хранятся в виде XML-файла, содержащего дерево необходимых для генерации кода параметров: классы агентов, параметры, элементы презентации, описания UML-диаграмм поведения агентов.

В процессе работы конвертера это дерево транслируется в код C++ программы, вычисляющей эту модель. Проход дерева совершается «в глубину», при этом выделяются следующие ключевые стадии, и происходит их совмещение с выполнением задачи трансляции.

1. Генерация основных параметров. Поиск корня дерева и считывание параметров дочерних вершин, таких как имя модели, адрес сборки, тип модели, тип презентации.

2. Генерация классов. Генерация классов (более подробно):

1) построение списка классов;

2) считывание основных параметров класса;

3) считывание переменных;

4) считывание параметров;

5) считывание функций;

6) генерация списка функций;

7) считывание кода функций;

8) преобразование кода функций Java -> C++;

9) считывание используемых фигур и элементов управления;

10) генерация кода инициализации фигур и элементов управления;

11) генерация кода конструктора, деструктора, визуализатора;

12) генерация структуры класса;

13) генерация кода заголовочного и source-файлов.

3. Генерация симулятора. Поиск вершины, хранящей информацию о процессе симуляции (управляющие элементы, значения важных констант, элементы презентации и т. д.).

4. Генерация общих файлов проекта (main.cpp, mainwindow.h, mainwindow.cpp и т. д.)

*Импортирование входных данных.* В качестве выходных данных в модель загружаются данные из геоинформационной составляющей исходной модели (карты России), содержащей всю необходимую информацию.

*Генерация классов и преобразование кода функций.* При генерации функций из дерева

считываются: имя функции, возвращаемый тип, параметры и ее тело.

На основании построенного ранее списка классов в аргументах функций производится замена «тяжеловесных классов», то есть всех сгенерированных классов, классов фигур и прочих классов, не входящих в стандартное множество, на соответствующие указатели с целью экономии памяти и во избежание ошибок при работе с нею. Далее генерируются заголовки функций, которые впоследствии вставляются в заголовочные и source-файлы. При таком считывании тело функции посредством соответствующей функции (листинг 5) преобразуется из Java-кода в аналогичный ей C++ код (это возможно ввиду достаточно узкого класса используемых функций, а в случае более сложных функций требуется ручная доработка транслированного кода), после чего она добавляется в список тел для данного класса.

В ходе трансляции неоднократно возникает задача преобразования исходного кода функций из языка Java в язык C++. Его можно представить в виде последовательных замен конструкций, например следующих:

1. Преобразование циклов: Java-формат.

2. Преобразование указателей. В Java нет столь явного, как в C++, различия между объектом и указателем на объект, поэтому структуры работы с ними не различаются. Поэтому вводится список классов, в которых важно использовать операции с указателем на объект, а не с самим объектом, и отслеживаются все переменные этих классов с последующей заменой в пределах данной функции при обращении к ним на соответствующие обращения к указателям.

3. Раскрытие черных ящиков. В Java, и конкретно в библиотеке AnyLogic, есть некоторое число функций и классов, аналогов которым нет ни в самом C++, ни в библиотеке ADEVS. В связи с этим были созданы дополнительные библиотеки shapes.h, mdb-work.h, в которых и реализованы недостающие функции.

4. На этапе генерации основных параметров списка классов получаются название основного класса и названия моделируемых классов-агентов. В код основного класса добавляется процедура добавления агента в зону видимости симулятора.

*Генерация внешних объектов.* В процессе генерации внешних объектов создается отдельная функция Main::initShapes(), в которой содержится вся «графическая информация», то есть в рамках функции происходит инициализация всех фигур, классы которых реализованы также в shapes.h. Соответствующий пример приведен в следующем отрывке кода.

*Генерация классов, кода заголовочного и source-файлов.* На основе всех считанных и сгенерированных данных создаются заголовочный и source-файл соответствующего класса.

*Генерация симуляции.* Для генерации симуляции оказалось достаточно заранее написанных файлов main.cpp, mainwindow.cpp, mainwindow.h, в которых шаблоны задают тип основного класса и подключаемые заголовочные файлы. При компоновке исходного кода шаблоны заменяются на полученные ранее (на стадии генерации) названия классов. Этого достаточно для двухпоточной симуляции, которую позднее можно заменить на соответствующий модуль для многопроцессорной симуляции.

*Дополнительные параметры.* На этапе разбора дерева (см. выше) параллельно формируется сходное по строению дерево для генерации C++ кода, с помощью которого на этапе подготовки к трансляции можно задать необходимые параметры компиляции (визуализацию тех или иных частей, визуальную валидацию распознавания кода, дополнительные флаги сборки и т. д.).

После этого при команде трансформации с учетом данных параметров и происходит окончательная компиляция.

*Сборка готового проекта.* Для сборки транслированного проекта применяется QtCreator — кроссплатформенная свободно распространяемая интегрированная среда разработки для работы с фреймворком Qt.

*Код агента.* При помощи описанного выше транслятора из данных файлов проекта AnyLogic (model.alp и др.) был сгенерирован исходный код (за исключением характера поведения агента).

Характер поведения агента должен быть сгенерирован из диаграммы состояний, однако на данный момент автоматизация этого процесса еще не реализована. Таким образом, к сгенерированному коду необходимо было добавить еще некоторый объем кода.

После внесения необходимых изменений в результате компиляции получилось кроссплатформенное приложение, повторяющее основную функциональность данной модели.

*Статистика и визуализация временных срезов.* Ввиду неинтерактивности запуска программы на больших суперкомпьютерах сбор выходных данных и визуализация были разделены (это связано с неравномерностью нагрузки на кластеры в разное время суток, а

Таблица 1

**Доступные для исследовательской группы суперкомпьютеры**

| Позиция в топ-50 | Суперкомпьютеры | CPU | Ядра | TFlops |
|---|---|---|---|---|
| 1 | «Ломоносов» (МГУ) | 12 422 | 82 468 | 902 |
| 3 | МВС-100К (МСЦ РАН) | 416 | 28 704 | 376 |
| 4 | «Чебышев» (МГУ) | 256 | 6 400 | 320 |

монопольный доступ попросту невозможен). После пересчета модели получившаяся на выходе информация может быть снова визуализирована с учетом произведенных расчетов и изменений в системе. При этом на экране наглядно можно отобразить агентов точками различных цветов, также можно отобразить в цвете и отдельные регионы в зависимости от полученных результатов, что позволяет наглядно представить динамику произошедших в процессе моделирования изменений.

*Доступные для расчетов суперкомпьютеры.* На момент проведения расчетов для нас были доступны три суперкомпьютера (табл. 1), входящие в первую пятерку суперкомпьютерного рейтинга топ-50 в странах СНГ (в редакции от 23.09.2014 г.).

При проведении исследований использовали для расчетов два суперкомпьютера — «Ломоносов» и МВС-100К.

*Результаты.* За счет применения суперкомпьютерных технологий и оптимизации программного кода удалось достичь очень высокой производительности.

Сама по себе оптимизация кода и использование C++ вместо Java позволили увеличить скорость выполнения программы. Так, модель тестировалась при следующих начальных условиях: 1) число агентов — 20 тыс.; 2) прогнозируемый период — 16 лет (до 2030 г.). По итогам расчетов оказалось, что время счета модели с использованием ADEVS составило 22 сек. на одном процессоре, а время счета модели также на одном процессоре с использованием AnyLogic составило 2 мин. 18 сек., — а это значит, что среда разработки была выбрана удачно.

Как отмечалось выше, обычный персональный компьютер с хорошей производительностью способен осуществлять вычисления с удовлетворительной скоростью над совокупностью агентов числом около 20 тыс. (поведение каждого из них задается приблизительно 20 функциями), и при этом среднее время пересчета одной единицы модельного времени (один год) составляет около минуты. При большем числе агентов, к примеру 100 тыс., компьютер попросту зависает.

Задействование 800 процессоров суперкомпьютера и выполнение оптимизированного кода позволило увеличить число агентов до 5 млн. При этом такой массив вычислений был выполнен за период времени, приблизительно равный 1 мин. 12 сек. (этот показатель может несколько меняться в зависимости от типа используемых процессоров).

В рамках исследования проводились расчеты по трем сценариям. Первый из них предполагал развитие текущей ситуации (с неизменными значениями параметров модели)

Таблица 2

**Расчеты численности населения Санкт-Петербурга по трем сценариям, тыс. чел.**

| Год | Базовый сценарий | Снижение заработной платы | Изменение миграционной политики |
|---|---|---|---|
| 2015 | 5180,03 | 5150,02 | 5145,98 |
| 2016 | 5251,19 | 5148,02 | 5143,88 |
| 2017 | 5322,72 | 5144,17 | 5141,18 |
| 2018 | 5395,20 | 5133,57 | 5134,91 |
| 2019 | 5468,38 | 5133,23 | 5123,12 |
| 2020 | 5542,41 | 5125,41 | 5121,50 |
| 2021 | 5617,37 | 5122,19 | 5114,65 |
| 2022 | 5693,14 | 5120,09 | 5097,87 |
| 2023 | 5769,18 | 5101,64 | 5087,11 |
| 2024 | 5845,51 | 5094,12 | 5077,02 |
| 2025 | 5921,90 | 5092,93 | 5052,44 |
| 2026 | 5998,31 | 5083,76 | 5052,23 |
| 2027 | 6075,31 | 5056,77 | 5029,27 |
| 2028 | 6153,12 | 5041,34 | 5023,12 |
| 2029 | 6231,06 | 5034,31 | 5011,92 |
| 2030 | 6309,54 | 5023,77 | 4989,83 |

— базовый сценарий. Второй сценарий предполагал снижение заработной платы в Санкт-Петербурге на 15 % относительно среднероссийского уровня. Таким образом, у агентов модели, предполагающим мигрировать в Санкт-Петербург из других регионов с целью поиска более высокооплачиваемой работы снижается стимул к переезду. Третий сценарий предусматривал принятие мер, направленных на ужесточение миграционной политики, что также снижает вероятность переезда агентов из других регионов. В модели это выражается в снижении средних значений вероятностных параметров, определяющих возможность переезда на 30 % от их начальных значений. Полученные результаты приведены в табл. 2.

Результаты моделирования показали, что при сохранении текущих тенденций население Санкт-Петербурга к 2030 г. прирастет почти на 22 %. Между тем, многое зависит от экономических условий (в частности, уровня заработной платы) и миграционной политики. Так, оба других сценария демонстрируют снижение численности населения на 2,5 % и 3,1 % соответственно. Вообще говоря, последние сценарии представляются малореалистичными, но и они демонстрируют некритичное снижение количества проживающих в Санкт-Петербурге человек, в то время как базовый сценарий предполагает ощутимый прирост.

Таким образом, полученные результаты исследований возможно использовать для совершенствования теории и практики управления социально-экономическим развитием регионов, в том числе макрорегиона Северо-Запад, повышения эффективности управленческих решений, экономии всех видов ресурсов [12]. Это, в свою очередь, будет способствовать повышению темпов социально-экономического развития выбранного региона, повышению устойчивости развития, снижению рисков и улучшению параметров качества жизни. Также полученные результаты могут найти применение при определении стратегических направлений развития региона как социально-экономической системы, определения перспектив такого развития и его роли в общем экономическом развитии страны.

## Список источников

1. *Окрепилов В. В.* Экономика качества как методологическая основа управления регионами // Экономика и управление. — 2013. — № 1 (87) — С. 8-14.

2. *Deissenberg C.*, *Hoog S. van der, Herbert D.* (2008). EURACE: A Massively Parallel Agent-Based Model of the European Economy // Document de Travail No. 2008. Vol. 39. 24 June.

3. *Roberts D. J.*, *Simoni D. A.*, *Eubank S.* (2007). A National Scale Microsimulation of Disease Outbreaks. RTI International. Research Triangle Park. Blacksburg: Virginia Bioinformatics Institute.

4. *Bisset K.*, *Chen J.*, *Feng X.*, *Kumar V. S. A.*, *Marathe M.* (2009). EpiFast: A fast algorithm for large scale realistic epidemic simulations on distributed memory systems. Yorktown Heights, New York; 2009: 430–439. Proceedings of 23rd ACM International Conference on Supercomputing (ICS'09).

5. *Epstein J. M.* (2009). Modeling to Contain Pandemics // Nature, volume 460, p. 687, 6 August.

6. *Collier N.* (2012). Repast HPC Manual. [Электронный ресурс] February 23. Режим доступа: http://repast.sourceforge.net, свободный. Загл. с экрана. Яз. англ. (дата обращения: май 2013).

7. *Keith R. B.*, Jiangzhuo C., Xizhou F., *Kumar A. V.* S., *Madhav V. M.* (2009). EpiFast: A Fast Algorithm for Large Scale Realistic Epidemic Simulations on Distributed Memory Systems ICS'09. June 8–12. N.Y.: Yorktown Heights.

8. *Ambrosiano N.* (2006). Avian Flu Modeled on Supercomputer // Los Alamos National Laboratory NewsLetter. Vol. 7. No. 8, p. 32.

9. Средства суперкомпьютерных систем для работы с агент-ориентированными моделями / Макаров В. Л., Бахтизин А. Р., Васенин В. А., Роганов В. А., Трифонов И.А. // Программная инженерия. — 2011. — №3. — С. 2-14.

10. *Бабкин А. В.*, *Шамина Л. К.* Анализ применения методологических подходов к управлению экономическими системами // Научно-технические ведомости СПбГПУ. Экономические науки. — 2008. — № 1 (53). — С. 18-22.

11. *Зусев Г. Ю.*, *Плотников В. А.* Социальные закономерности и роль человека в современном экономическом развитии // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. — 2011. — Т. 2. — № 119. — С. 22-26. — (Экономические науки).

12. *Gnevko V.* Municipalities — roots of democracy and economics. USA: Society and Science press. 2012, p. 295.

13. *Epstein J. M.*, *Axtell R. L.* (1996). Growing Artificial Societies: Social Science from the Bottom Up. Ch. V. Cambridge, Massachusetts: MIT Press.

14. *Parker J.* (2007). A Flexible, Large-Scale, Distributed Agent Based Epidemic Model. Center on Social and Economic Dynamics. Working Paper No. 52, p.25.

15. *Lynar T. M.*, *Herbert R. D.*, *Chivers W. J.* (2009): Implementing an Agent Based Auction Model on a Cluster of Reused Workstations // International J. of Computer Applications in Technology. Vol. 34. Issue 4, p.13-24.

## Информация об авторах

**Окрепилов Владимир Валентинович** (Санкт-Петербург, Россия) — доктор экономических наук, профессор, академик Российской академии наук, генеральный директор ФБУ Государственного регионального центра стандартизации,

метрологии и испытаний в г. Санкт-Петербурге и Ленинградской области (190103, Санкт-Петербург, Курляндская, 1; e-mail: letter@rustest.spb.ru).

**Макаров Валерий Леонидович** (Москва, Россия) — доктор физико-математических наук, профессор, академик Российской академии наук, директор Центрального экономико-математического института РАН (117418 Москва, Нахимовский проспект, 47; e-mail: makarov@cemi.rssi.ru).

**Бахтизин Альбер Рауфович** (Москва, Россия) — доктор экономических наук, ведущий научный сотрудник лаборатории экспериментальной экономики Центрального экономико-математического института РАН (117418 Москва, Нахимовский проспект, 47; e-mail: albert.bakhtizin@gmail.com).

**Кузьмина Светлана Николаевна** (Санкт-Петербург, Россия) — доктор экономических наук, главный специалист ФБУ «ФБУ Государственного регионального центра стандартизации, метрологии и испытаний в г. Санкт-Петербурге и Ленинградской области (190103, Санкт-Петербург, Курляндская, 1; e-mail: kuzmina2003@bk.ru).

**V. V. Okrepilov, V. L. Makarov, A. R. Bakhtizin, S. N. Kuzmina**

## Application of Supercomputer Technologies for Simulation Of Socio-Economic Systems

*To date, an extensive experience has been accumulated in investigation of problems related to quality, assessment of management systems, modeling of economic system sustainability.*

*The performed studies have created a basis for development of a new research area — Economics of Quality. Its tools allow to use opportunities of model simulation for construction of the mathematical models adequately reflecting the role of quality in natural, technical, social regularities of functioning of the complex socio-economic systems.*

*Extensive application and development of models, and also system modeling with use of supercomputer technologies, on our deep belief, will bring the conducted research of socio-economic systems to essentially new level. Moreover, the current scientific research makes a significant contribution to model simulation of multi-agent social systems and that is not less important, it belongs to the priority areas in development of science and technology in our country.*

*This article is devoted to the questions of supercomputer technologies application in public sciences, first of all, — regarding technical realization of the large-scale agent-focused models (AFM). The essence of this tool is that owing to the power computer increase it has become possible to describe the behavior of many separate fragments of a difficult system, as socio-economic systems are.*

*The article also deals with the experience of foreign scientists and practicians in launching the AFM on supercomputers, and also the example of AFM developed in CEMI RAS, stages and methods of effective calculating kernel display of multi-agent system on architecture of a modern supercomputer will be analyzed.*

*The experiments on the basis of model simulation on forecasting the population of St. Petersburg according to three scenarios as one of the major factors influencing the development of socio-economic system and quality of life of the population are presented in the conclusion.*

**Keywords:** Agent-based model, demographic projection, model simulation, quality of life, quality management methods, socio-economic system, supercomputer technologies, scenario estimates, economics of quality.

## References

1. Okrepilov V. V. (2013). Ekonomika kachestva kak metodologicheskaya osnova upravleniya regionami [Economy of ualityas methodological basis of management of regions]. *Ekonomika i upravleniya [Economics and management],* 1 (87), 8-14.

2. Deissenberg, C., Hoog, S. van der & Herbert, D. (2008, June). EURACE: A Massively Parallel Agent-Based Model of the European Economy. *Document de Travail,* 39.

3. Roberts, D. J., Simoni, D. A. & Eubank, S. (2007). *A National Scale Microsimulation of Disease Outbreaks.* RTI International. Research Triangle Park. Blacksburg: Virginia Bioinformatics Institute.

4. Bisset, K., Chen, J., Feng, X., Kumar, V. S. A. & Marathe, M. (2009). EpiFast: A fast algorithm for large scale realistic epidemic simulations on distributed memory systems. Yorktown Heights, New York; 2009:430–439. *Proceedings of 23rd ACM International Conference on Supercomputing (ICS'09).*

5. Epstein, J. M. (2009, August). Modeling to Contain Pandemics. *Nature, volume 460,* 687.

6. Collier, N. (2012, February 23). *Repast HPC Manual.* Available at: http://repast.sourceforge.net (date of access: 2013, May).

7. Keith, R. B., Jiangzhuo, C., Xizhou, F., Kumar, A. V. S. & Madhav, V. M. (2009). *EpiFast: A Fast Algorithm for Large Scale Realistic Epidemic Simulations on Distributed Memory Systems ICS'09.* June 8–12. N.Y.: Yorktown Heights.

8. Ambrosiano, N. (2006). Avian Flu Modeled on Supercomputer. *Los Alamos National Laboratory NewsLetter,* 7, 8, 32.

9. Makarov, V. L., Bakhtizin, A. R., Vasenin, V. A., Roganov, V. A. & Trifonov I. A. (2011). *Sredstva superkompyuternykh sistem dlya raboty s agent-orientirovannymi modelyami [Services of supercomputer systems for work with agent-focused models],* 3.

10. Babkin, A. V. & Shamin, L. K. (2008). Analiz primeneniya metodologicheskikh podkhodov k upravleniyu ekonomicheskimi sistemami [The methodological approache application analysis to economic system management]. *Nauchno-tekhnicheskie vedomosti SPbGU. Ekonomicheskie nauki [Scientific and Technical Journal of Peter the Great St. Petersburg Polytechnical University. Economic Sciences],* 1(53), 18-22.

11. Zusev, G. Yu. & Plotnikov, V. A. (2011). Sotsialnyye zakonomernosti i rol cheloveka v sovremennom ekonomicheskom razvitii [Social regularities and role of himan being in modern economic development]. *Nauchno-tekhnicheskie vedomosti Sankt-Peterburgskogo gosudarstvennogo politekhnicheskogo universiteta [Scientific and Technical Journal of Peter the Great St. Petersburg Polytechnical University. Economic Sciences],* 2, 119, 22-26.

12. Gnevko, V. (2012). Municipalities — roots of democracy and economics. *USA: Society and Science press,* 295.

13. Epstein, J. M. & Axtell, R. L. (1996). *Growing Artificial Societies: Social Science from the Bottom Up.* Ch. V. Cambridge, Massachusetts: MIT Press.

14. Parker, J. (2007). A Flexible, Large-Scale, Distributed Agent Based Epidemic Model. Center on Social and Economic Dynamics. *Working Paper,* 52, 25.

15. Lynar, T. M., Herbert, R. D. & Chivers, W. J. (2009). Implementing an Agent Based Auction Model on a Cluster of Reused Workstations. *International J. of Computer Applications in Technology,* 34, 4, 13-24.

## Information about the authors

**Okrepilov Vladimir Valentinovich** (Saint Petersburg, Russia) — Doctor of Economics, Professor, Member of the Russian Academy of Sciences, General Director of the State Regional Centre for Standardization, Metrology and Testing in Saint Petersburg and the Leningrad Oblast (1, Kurlyandskaya St., 190103 Saint Petersburg, Russia; e-mail: letter@rustest.spb.ru).

**Makarov Valeriy Leonidovich** (Moscow, Russia) — Doctor of Physics and Mathematics, Professor, Member of the Russian Academy of Sciences, Head of the Central Economic Mathematical Institute of the Russian Academy of Sciences (47, Nakhimovskiy Av., 117418 Moscow, Russia; e-mail: makarov@cemi.rssi.ru).

**Bakhtizin Albert Raufovich** (Moscow, Russia) — Doctor of Economics, Senior Research Associate at the Laboratory of Experimental Economics, the Central Economic Mathematical Institute of the Russian Academy of Sciences (47, Nakhimovskiy Av., 117418 Moscow, Russia; e-mail: albert.bakhtizin@gmail.com).

**Kuzmina Svetlana Nikolayevna** (Saint Petersburg) — Doctor of Economics, Chief Specialist at the State Regional Centre for Standardization, Metrology and Testing in Saint Petersburg and the Leningrad Oblast (1, Kurlyandskaya St., 190103 Saint Petersburg, Russia; e-mail: kuzmina2003@bk.ru).