Моделирование суперкомпьютерной вычислительной системы объектно-атрибутной архитектуры с управлением потоком данных

С.М. Салибекян, П.Б. Панфилов

Аннотация. Описаны принципы моделирования предлагаемых атрибутной (А) и объектно-атрибутной (ОА) архитектур вычислительных систем (ВС), реализующих dataflow-модель организации вычислений, и результаты имитационного моделирования суперкомпьютерной системы ОА-архитектуры. Для математического моделирования dataflow-ВС предложен новый формализм «атрибутных сетей» (А-сетей), существенно отличающийся от аппарата сетей Петри. Этот формализм положен в основу среды ОА-программирования и имитационного моделирования dataflow-ВС, которая использовалась при разработке, программировании и тестировании суперкомпьютера ОА-архитектуры.

Ключевые слова: dataflow-парадигма, модель атрибутной сети (А-сети), среда ОА-программирования и моделирования, суперкомпьютер.

Введение

В области вычислительных архитектур и программирования компьютеров вычислительные архитектуры, которые работают по принципу управления вычислительным процессом с помощью потока данных (dataflow-парадигма организации вычислений), рассматриваются как архитектуры, которые более приспособлены к реализации параллельных вычислений, чем классические системы, действующие по принципу управления вычислениями с помощью потока команд (controlflow-парадигма). Работа над милликомандной или атрибутной (А) и объектно-атрибутной (ОА) архитектурами вычислительной системы (ВС) [1, 2], реализующими dataflow модель вычислений, активно ведется последние 10 лет в Московском институте электроники и математики (МИЭМ), ставшим недавно составной частью Национального исследовательского университета «Высшая школа экономики» (НИУ ВШЭ). Разработанные коллективом исследователей МИЭМ НИУ ВШЭ архитектуры ВС обладают рядом преимуществ, которые, предположительно, позволят им составить коммерческую конкуренцию классической (фон-неймановской) архитектуре ВС, реализующей control flow модель вычислений.

Вычислительные системы А- и ОА-архитектур представляют собой набор функциональных устройств (ФУ), выполняющих вычислительную работу и обменивающихся между собой данными, оформленными в виде информационной пары (ИП) – простейшего токена, состоящего из двух полей: атрибута (идентификатора, представляющего собой целое число) и данных или указателя на ячейку оперативной памяти (ОП), где хранятся данные. Единовременно ФУ может принимать только одну информационную пару. Вычислительный процесс задается с помощью описания обмена данными между ФУ (описание обмена данными между ФУ будем называть **А-** или **ОА-образом**).

Основным отличием разработанных архитектур от существующих универсальных dataflow-систем является то, что исполняемый пакет (полный комплект данных для операции) собирается не в отдельном блоке поиска совпадений (англ. matching), а непосредственно во внутренних регистрах самих функциональных устройств (ФУ считывает данные с линии передачи информации, идентифицируя их по атрибутам, формирует исполняемый пакет и само же выполняет операцию). Данное техническое решение позволяет:

- реализовать распределенное управление вычислительным процессом (т.е. в составе ВС нет централизованного блока, который контролирует весь вычислительной процесс) [3, 4];
- реализовывать многооперандные команды (операнды поступают в ФУ последовательно, и поэтому нет, как в фон-неймановской архитектуре, ограничений по их количеству);
- существенно упростить ВС (нет необходимости в применении коммутационных сред, множества пересылок данных между блоками, входящими в состав ВС; в частности, можно обойтись без применения громоздкой ассоциативной памяти и т.д.) и тем самым снизить объем оборудования, входящего в состав ВС, что приведет к уменьшению стоимости системы;
- обеспечить масштабируемость ВС;
- обеспечить эффективность работы как в режиме крупнозернистого, так и мелкозернистого параллелизма;
- обеспечить эффективность работы как на параллельных задачах, так и на последовательных;
- создавать реконфигурируемые ВС (причем, реконфигурацию можно осуществлять непосредственно во время вычислительного процесса);
- обеспечить возможность динамической балансировки вычислительной нагрузки.

В то время как разработанная А-архитектура более подходит к аппаратной реализации dataflow-ВС, ОА-архитектура ВС оказалась применимой как к аппаратной, так и к программной частям ВС. Для описания алгоритма работы ОА-ВС был разработан ОА-язык параллельного программирования [5], на основе которого затем была создана среда программирования и компилятор языка. Благодаря тому, что язык программирования и «железо» функционируют по одним и тем же принципам, в ОА-системе практически отсутствует проблема семантического разрыва между языком программирования высокого уровня и аппаратной частью ВС.

В ходе выполнения НИР по разработке и моделированию суперкомпьютерной dataflow-ВС ОА-архитектуры в условиях отсутствия макетного образца ОА-системы в качестве основного инструмента для получения и оценки вычислительных характеристик и параметров ОА-ВС по производительности, масштабируемости и др. активно использовалось имитационное моделирование [6]. Для этих целей были отдельно решены задачи по разработке подходов к математическому и имитационному моделированию А- и ОА-архитектур ВС и собственно dataflow-ВС.

1. Принципы математического моделирования dataflow-ВС

В настоящее время для моделирования параллельных ВС широко используются сетевые модели и, в частности, сети Петри [7]. Однако математический аппарат сетей Петри в чистом виде оказался недостаточно подходящим для моделирования параллельных ВС и, в частности, dataflow-систем. Как отмечает профессор Массачусетского технологического института, США, Карл Хьюит (англ. Carl Hewitt), сетям Петри присущи следующие недостатки:

- сети Петри имеют следующее ограничение: они моделируют управление потоком, но не сам поток данных;
- сложность описания одновременных действий, происходящих во время вычислительного процесса;
- физическая интерпретация перехода в сетях Петри весьма сомнительна.

Для преодоления этих недостатков потребовалось разработать новый формализованный аппарат описания параллельных и dataflow-ВС, получивший название модели «атрибутной сети» или «А-сети» (Рис. 1 и Рис. 2), которая представляет собой восьмерку:

$$A=\{I, C, O, EC, CO, OI, IM, OM\},$$

где I – множество входных узлов;

C – множество вычислительных узлов;

O – множество выходных узлов;

IC: I → C – множество дуг из входных вершин в вычислительные узлы;

CO: C → O – множество дуг из вычислительных узлов в выходные узлы;

OI: O → I – множество дуг, соединяющих выходные вершины со входными;

IM – вектор маркировок входных узлов;

OM – вектор маркировок выходных узлов.

В теоретико-графовой интерпретации [8] A-сеть – это трёхдольный граф, где одна доля (множество C) – это вычислительные вершины (на них производится исполнение команды: в реальной ВС это так называемые исполнительные устройства или ИУ), вторая доля (множество I) – это входные данные (в реальной системе это внутренние регистры ИУ для хранения промежуточных данных), а третья доля (множество O) – это выходные данные для вычислительных вершин (внутренние регистры, в которые исполнительные устройства помещают результат вычислений) (Рис. 1).

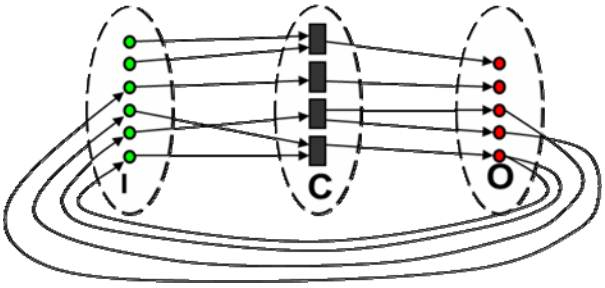

Рис. 1. 3-дольный граф атрибутной сети

На Рис. 2 атрибутная сеть представлена в другом (ярусном) виде. Здесь вычислительная вершина обозначена прямоугольником, входные и выходные узлы обозначены кружками зеленого и красного цвета соответственно. Для входных и выходных вершин вводится маркировка, имеющая только два значения – «пусто» и «заполнено». «Пусто» означает, что во входную вершину еще не поступили данные, «заполнено» - поступили. Маркировка задается с помощью векторов маркировки входных (IM) и выходных (OM) вершин. Первый ярус входных вершин на Рис. 2 обозначает входные данные для вычислительного процесса, последний ярус выходных вершин – выходные данные, полученные на выходе из ВС.

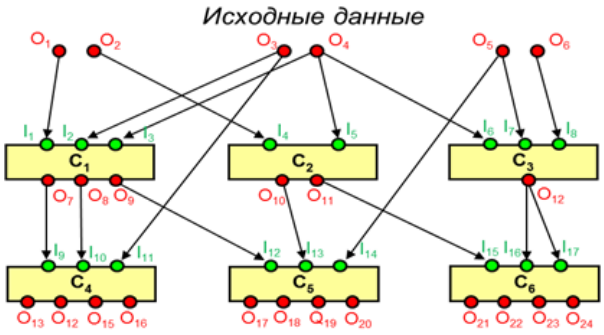

Рис. 2. Атрибутная сеть

Выполнение A-сети представляет собой последовательность событий двух видов:

- активация (выполнение) k-й вычислительной вершины ($OM' = OM+CO(c_k)$, где $c_k \in C$, CO – функция активации вершины, OM′ – новая маркировка выходных вершин A-сети, которая получается после активации вычислительного узла – все смежные с c_k выходные вершины помечаются как «заполненные»). Вершина может быть активирована только в том случае, когда смежные с ней входные вершины являются помеченными как «заполненные», т.е. для вычислительного узла поступили все необходимые для выполнения операции данные. В реальной ВС это событие есть окончание вычислений, выполняемых ИУ;

- передача данных из i-й выходной вершины во входные вершины ($IM'=IM+TO(o_i)$, где $o_i \in O$, TO – функция активации перехода, IM′ – массив маркировок входных вершин после активации передачи данных). Это событие может быть активизировано лишь в том случае, когда вершина o_i помечена как «заполненная». В реальной ВС это событие описывает передачу данных по линии связи (внутрипроцессорная шина, шина на печатной плате, вычислительная сеть и т.п.).

Разработанная модель A-сети оказалась свободной от недостатков сетей Петри, указанных Карлом Хьюитом, а именно:

- A-сеть моделирует поток данных, т.к. маркировка отображает перемещение данных в ВС (так событие «передача данных» есть не что иное, как передача токена, т.е. данных, снабженных служебной информацией, от одного узла ВС к другому).

- Простота описания одновременных действий (высокий параллелизм ВС не приводит к сильному усложнению математической модели

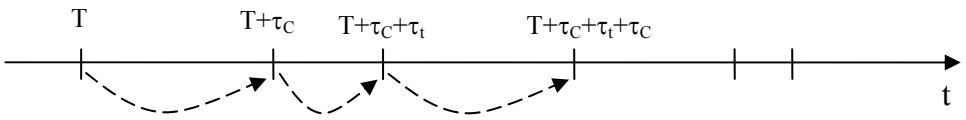

Рис. 3. Временное моделирование A-сети

вычислительного процесса, как это происходило в случае сети Петри).

- Физическая интерпретация событий в A-сети весьма конкретна: активация вычислительной вершины – это окончание вычислений, производимых в вычислительном узле (в реальности вычислительный узел представляет собой ИУ), и запись полученного результата в выходные регистры ИУ; а передача данных – это пересылка токена с результатом вычислений из выходного регистра ИУ во входные узлы (в реальности входными узлами A-сети являются входные регистры ИУ).

Разработанная модель может быть применена в том числе и для временно́го моделирования вычислительного процесса, разворачивающегося в dataflow-системе. Для этих целей используется временная ось, на которой отмечаются предстоящие события (Рис. 3), и времена пересылки данных и выполнения операции вычислительной вершиной. При активации вычислительной вершины, которая происходит в момент модельного времени T, на временную ось наносится метка в точке $T+\tau_C$, где τ_C – время выполнения операции (в реальности $T+\tau_C$ есть время появления результатов вычисления ИУ в его выходных регистрах). Аналогичным образом при активации пересылки данных в момент T на временную ось наносится метка в момент $T+\tau_t$, где τ_t – время выполнения пересылки токена с данными на входные узлы (т.е. во входные регистры ИУ, которые принимают данные).

Само моделирование строится по событийному принципу. Перед началом моделирования на временную ось наносятся метки всех предстоящих событий. Далее начинается цикл моделирования. На очередном шаге моделирования выбирается событие, метка которого имеет наименьшее время. В момент активации события на временно́й оси появляются новые метки (например, при активации вычислительной вершины появятся будущие события доставки данных во входные вершины, которые должны произойти с задержкой, равной времени пересылки данных между выходными и входными вершинами).

Например (Рис. 3), после активации вычислительной вершины в момент T на временную ось в момент времени $T+\tau_C$ ставится метка момента пересылки данных. В момент же времени $T+\tau_C$ происходит запись полученных результатов вычислений в соответствующие выходные вершины A-сети. Заполнение данными выходных вершин приводит к запуску процесса пересылки данных во входные вершины, поэтому на временной оси через время τ_t (время пересылки данных между выходными и входными вершинами) помещается метка. Затем в момент времени $T+\tau_C+\tau_t$ происходит запись результата вычислений во входную вершину. Наличие всех необходимых выходных данных для активизации вычислительной вершины может породить активизацию новых вычислительных вершин и т.д. Процесс моделирования заканчивается, когда на временно́й оси не остается меток.

Разработанный подход (на базе формализма A-сетей) к математическому моделированию dataflow-ВС лег в основу изложенных ниже принципов имитационного моделирования dataflow-ВС A- и OA-архитектур.

2. Принципы имитационного моделирования ВС A- и OA-архитектур

Имитационное моделирование вычислительных систем A- и OA-арихитектур произвести намного легче, чем классических ВС ввиду нескольких причин. Во-первых, функциональные устройства (ФУ) в OA-ВС могут быть реализованы как аппаратно, так и программно, поэтому имитационное моделирование ВС сводится к созданию виртуальных копий аппаратных ФУ. Во-вторых, в OA-ВС во время имитационного моделирования не требуется при-

менения специальных средств, производящих распараллеливание вычислений и отслеживание работы параллельно работающих устройств: вычислительный процесс описывается как обмен данными между ФУ, то есть, одни ФУ, получив результат вычислений, пересылают его на другие ФУ, и при наличии полного комплекта данных ФУ активизируются и пересылают свои данные другим ФУ. Таким образом, вычислительный процесс несколько напоминает собой «цепную реакцию», в результате чего происходит самораспараллеливание вычислений, и, поэтому, не возникает потребности специально разрабатывать сложный алгоритм распараллеливания вычислений. В-третьих, усилия, затраченные на создание имитационной модели, не пропадают зря, т.к. созданный ОА-образ (описание информационного обмена между ФУ) после моделирования практически без изменения переносится на реальную ВС (аппаратно реализованные ФУ имеют точно такую же логику работы, как и виртуальные) [3, 4]. Это существенно убыстряет и удешевляет процесс создания и отладки dataflow-ВС и их ОА-приложений.

Для осуществления имитационного моделирования была разработана среда программирования и имитационного моделирования объектно-атрибутной ВС, которая включает в себя:

- ОА-платформу (совокупность программ, реализующих логику работы виртуальных ФУ);
- компилятор ОА-языка параллельного программирования;
- инструментальные средства программирования (текстовый редактор ОА-программы, панели вывода результатов и служебных сообщений, панель инструментов, операционные подсказки и т.д.);
- средства, позволяющие осуществлять имитационное моделирование ОА-ВС.

Для осуществления имитационного моделирования параллельных dataflow-ВС потребовалось:

- Ввести в состав каждого виртуального ФУ очередь ожидания токенов. Токен попадает в очередь в том случае, когда ФУ, которому он адресован, уже занято вычислительной работой или ему недоступны аппаратные ресурсы (ИУ) для осуществления вычислений.
- Ввести в состав виртуальных ФУ настройки длительности выполнения каждой команды, выполняемой ФУ.
- Создать ФУ, эмулирующее работу распределителя вычислительных ресурсов (например, процессорных ядер) между ФУ.
- Создать ФУ, эмулирующее передачу данных между вычислительными устройствами (вычислительное устройство может, например, представлять собой многопроцессорную/многоядерную ВС с общей памятью). Будем называть такое ФУ «Шлюзом».
- Создать ФУ, контролирующее время наступления событий в А- или ОА-системе (англ. Eventser). В реальной (аппаратной или программной) dataflow-ВС данное ФУ присутствовать не будет, оно необходимо только для проведения имитационного моделирования. ФУ Eventser создается в среде программирования и моделирования в единственном экземпляре и контролирует время наступления событий во времени во всей ОА-ВС.

На Рис. 4 приведена схема организации имитационного моделирования ОА-ВС, которая базируется на модели А-сети: виртуальные ФУ заменяют собой вычислительные вершины А-сети, контекст ФУ (совокупность внутренних регистров ФУ) заменяет собой входные и выходные узлы А-сети. Контроллер же событий (Eventser) осуществляет функции контроля времени наступления событий в моделируемой ОА-системе.

Алгоритм имитационного моделирования в этом случае выглядит сделующим образом:

1. По приходу токена к ФУ оно информирует об этом событии планировщик и переходит в режим ожидания. Если в вычислительном узле имеется хотя бы одно свободное ИУ (например, процессорное ядро), то оно выделяется этому ФУ; в противном случае запрос на выполнение информационной пары ставится в очередь ожидания освобождения аппаратных ресурсов.
2. В том случае, когда распределителем ресурсов ФУ выделяется ИУ, на Eventser отправляется сообщение о длительности выполнения выполняемой ФУ операции, а Eventser прибавляет к текущему модельному времени время задержки выполнения операции и ставит для этого события метку на временно́й оси.

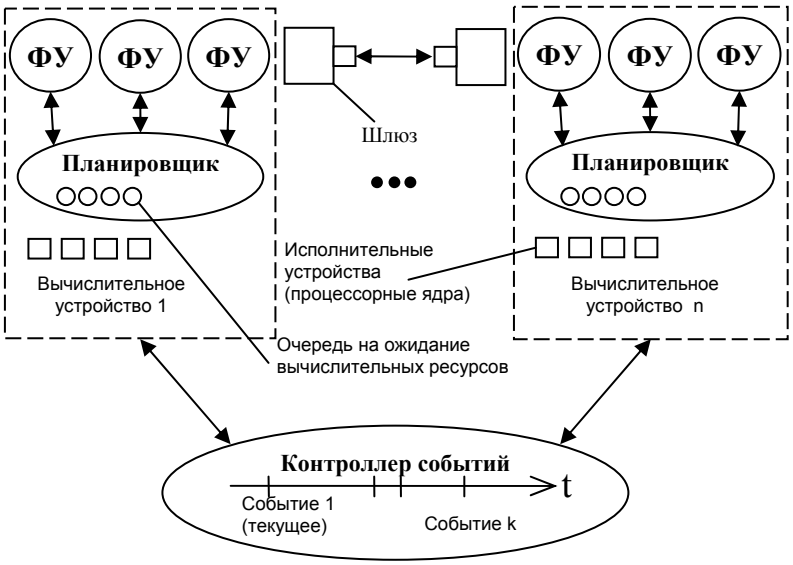

Рис. 4. Организация имитационного моделирования ОА-ВС

3. Если соответствующая метка активизируется (это происходит, когда уже активизировались все предыдущие метки, расположенные на временно́й оси), то Eventser выдает для ФУ, которому эта метка соответствует, сигнал о выходе из режима ожидания. ФУ производит вычисления и пересылку полученных результатов вычислений на другие ФУ.

Процесс моделирования заканчивается в тот момент, когда на временной оси не останется ни одной метки.

3. Результаты моделирования dataflow-ВС ОА-архитектуры

В рамках НИР по разработке и моделированию суперкомпьютерной dataflow-ВС были созданы семь тестовых пакетов (бенчмарков), прогоны которых были осуществлены в разработанной среде программирования и моделирования, включая тесты из широкого класса физических задач (4 тестовых примера), тест Graph500 на скорость обработки теоретико-графовых задач, тест на обработку строк GREP, тест из набора SPEC (решение системы уравнений симплекс-методом). На Рис. 5 приведен пример результатов моделирования dataflow-ВС ОА-архитектуры на тесте по обработке строк GREP: зависимость времени выполнения теста от числа ФУ и ИУ (число функциональных и исполнительных устройств варьируется от 1 до 50).

По результатам проведенных прогонов тестовых примеров можно сделать следующие выводы:

1. ОА-архитектура показывает хорошие результаты при реализации на распределенных ВС, в частности, при прогоне тестов из широкого класса физических задач (сеточные методы вычислений) при грамотном выборе конфигурации ВС удалось добиться линейного роста производительности вычислительной системы в зависимости от объема оборудования.

2. При моделировании расчетов тестов из широкого класса физических задач наибольшая загрузка ОП наступала на начальном этапе вычислительного процесса (промежуточные данные, для обработки которых не хватает исполнительных устройств, накапливаются в очередях ожидания вычислительных ресурсов).

3. В ходе выполнения НИР был разработан алгоритм планировки вычислений, который позволяет при реализации сеточных методов вычислений для решения широкого класса физических задач несколько снизить пик потребления ОП в начале вычислительного процесса и тем самым снизить требуемый для реализации вычислений объем ОП на вычислительных узлах.

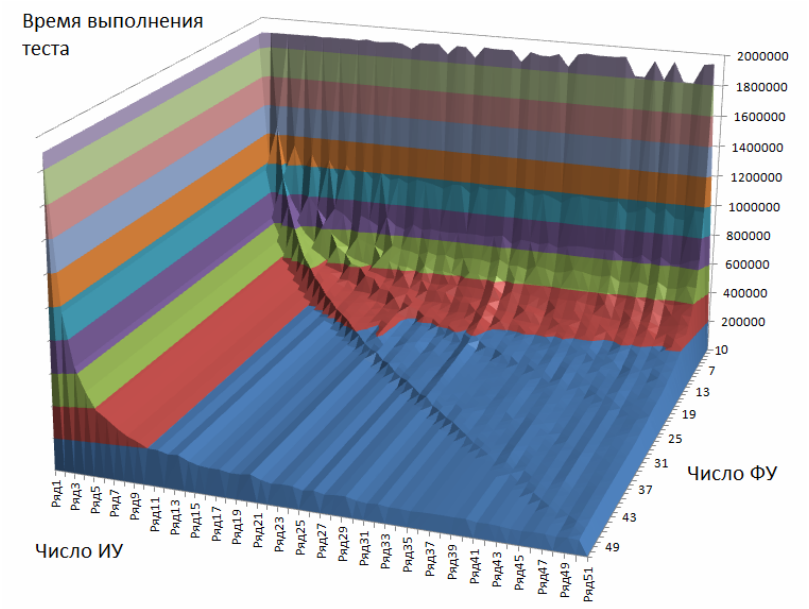

Рис. 5. Результаты моделирования ОА-ВС на тестовом наборе GREP

Создание среды программирования и имитационного моделирования и само моделирование ОА-ВС осуществлялось в рамках НИР по ФЦП «Исследования и разработки по приоритетным направлениям развития научно-технического комплекса России на 2007-2013 годы», финансируемой Министерством образования и науки РФ.

Заключение

В результате проведенной НИР по разработке и моделированию архитектуры суперкомпьютерной dataflow-ВС была предложена эффективная архитектура ВС с управлением вычислениями потоком данных, подходящая для решения конкретных вычислительных задач. В настоящее время ведутся подготовительные работы по инициированию и проведению ОКР по созданию макетного образца высокопроизводительной параллельной вычислительной системы ОА-архитектуры, реализующей dataflow-парадигму организации вычислений. Данная работа позволит найти удачные технические решения по созданию распределенных ОА-систем, и сделать задел для создания опытного мелкосерийного производства ВС ОА-архитектуры.

Литература

1. Салибекян С.М. Принципы милликомандной архитектуры как основа построения высокопроизводительных адаптивных вычислительных систем // Автоматизация и современные технологии. —2002. — Май. —No. 5.—С.25-32.
2. Салибекян С.М., Панфилов П.Б. Перспективная суперкомпьютерная система на основе объектно-атрибутной модели вычислений с управлением потоком данных // Международная конференция «Развитие суперкомпьютерных и грид-технологий в России» в рамках «Второго Московского Суперкомпьютерного форума» Россия, Москва, ВВЦ 26–27 октября 2011 года. URL: http://www.hpc-platform.ru/tiki-download_file.php?fileId=82
3. Салибекян С.М., Панфилов П.Б. ОА-архитектура построения и моделирования распределенных систем автоматизации // Автоматизация в промышленности. —2010.—Ноябрь.—No. 11.—С.51-56.
4. Салибекян С.М., Панфилов П.Б. Объектно-атрибутная архитектура - новый подход к созданию объектных систем // Информационные технологии.-2012.-No. 2 (186).-С.8-13
5. Салибекян С.М., Панфилов П.Б. Построение распределенных гетерогенных вычислительных систем на основе объектно-атрибутной архитектуры // Объектные системы - 2011 (зимняя сессия): материалы V Международной научно-практической конференции (Ростов-на-Дону, 10-12 декабря 2011 г.) / Под общ. ред. П.П. Олейника. - Ростов-на-Дону: ШИ ЮРГТУ (НПИ), 2011.-С.83-88.

6. Кельтон В., Лоу. А. Имитационное моделирование. Классика CS. 3-е изд. – СПб.: Питер; Киев: Издательская группа BHV, 2004.
7. Питерсон Дж. Теория сетей Петри и моделирование систем: Пер. с англ. - М.— 254 с.
8. Харари Фрэнк. Теория графов / пер. с англ. и предисл. В.П. Козырева. Под ред. Г.П. Гаврилова. Изд. 2-е. - М.: Едиториал УРСС, 2003. - 296 с.

Салибекян Сергей Михайлович. Старший преподаватель Московского института электроники и математики национальный исследовательский университет «Высшая школа экономики». Окончил МИЭМ в 2000 году. Кандидат технических наук. Автор более 20 печатных работ. Область научных интересов: распределенная и параллельная обработка данных, вычислительные системы и системы имитационного моделирования, компьютерная лингвистика, системы искусственного интеллекта. E-mail: salibek@yandex.ru

Панфилов Петр Борисович. Ведущий специалист-эксперт в Минэкономразвития РФ. Окончил МИЭМ в 1982 году. Кандидат технических наук, доцент. Автор более 70 печатных работ и 1монографии. Область научных интересов: распределенная и параллельная обработка данных, вычислительные системы и системы имитационного моделирования реального времени, визуальный компьютинг, интерактивные системы визуализации и виртуальные среды.
E-mail: panfilov@miem.edu.ru