УДК 004.4:57

В. М. Пентковский

Московский физико-технический институт (государственный университет)

Разработка проблемно-ориентированных архитектур и приложений для экзофлопных вычислений в биологии и фармацевтике

Рассмотрены проблемы, возникающие при оптимизации кластерных архитектур вычислительных систем для решения прикладных задач биоинформатики. Модель многопроцессорной ЭВМ строится на основе параллельной распределённой системы симуляции Graphite. Описаны текущее состояние и ближайшие планы развития «Лаборатории суперкомпьютерных технологий в области биоинформатики».

Ключевые слова: архитектуры вычислительных систем, высокопроизводительные расчёты, модели вычислительных комплексов, биоинформатика.

1. Введение

Проблема разработки новейших лекарств, а также выяснение механизмов и факторов, влияющих на развитие различного рода заболеваний, являются наиболее актуальными для прогресса всего человечества и, в частности, для биофармацевтической промышленности.

Классические подходы, основанные в основном на «скрининге», себя практически исчерпали, и стремительный прогресс таких областей, как фармацевтика, биотехнологии, медицина, химия, происходящий в настоящее время, уже во многом стал возможен благодаря компьютерному моделированию.

Развитие теоретических методов, разработка высокоэффективных компьютерных алгоритмов, а также непрерывно происходивший на протяжении последних десятилетий экспоненциальный рост вычислительных мощностей делают возможным разработку и моделирование свойств сложных органических молекул, имеющих важное значение для жизнедеятельности биологических организмов и разработки новейших лекарственных средств, например, протеинов, активных ферментов, антител.

Современные методы квантовой химии в сочетании с возникающими вычислительными ресурсами позволяют предсказать свойства создаваемых молекул, структур, материалов или разрабатываемых процессов, основываясь на детальном понимании их структур на атомистическом уровне. В рамках такого подхода моделирование предшествует созданию прототипа, позволяя значительно уменьшить количество вариантов, подлежащих экспериментальной реализации, и, как следствие, сократить стоимость разработки, время, необходимое для её осуществления, и существенно снизить риски.

Однако в результате специализации в последние годы возник разрыв между биологами и фармацевтами, с одной стороны, и разработчиками вычислительных систем, программ и приложении — с другой. Таким образом, для создания новых прорывных технологий и продуктов требуется объединить усилия специалистов этих отраслей для создания оптимального «железа» и ПО для решения задач фармацевтики, биотехнологии, медицины и оказания услуг компаниям, специализирующимся в этих областях. Именно на это нацелена работа лаборатории суперкомпьютерных технологий в области биоинформатики», созданной в 2010 году в МФТИ.

Таким образом, целью нашей лаборатории является разработка такой архитектуры вычислительного комплекса, которая была бы оптимальна для решения наиболее важных прикладных задач биоинформатики и фармакологии, создание такого вычислителя и решение на нём задач, которые не могли бы быть решены за разумное время на других суперкомпьютерах. Однако прежде чем строить подобный супервычислитель, необходимо

построить его модель и оптимизировать под конкретную задачу. Причём сделать это надо на доступном сегодня «железе», которое тоже требовалось создать. Таким образом, работа велась в трёх направлениях: построение первого прототипа оптимальной вычислительной системы, подбор актуальных задач, которые не могли быть решены на доступных вычислительных ресурсах, и построение модели вычислительного комплекса. Причём такая модель должна имитировать поведение существенно большей (вплоть до петафлопса) машины.

В качестве базового подхода к построению вычислителя мы выбрали кластерный подход.

2. Архитектура суперкомпьютера

Кластерная архитектура на сегодняшний день является наиболее массово применяемой для строительства высокопроизводительных вычислительных комплексов: в списке самых мощных суперкомпьютеров мира Top–500 более 80% систем являются кластерами. В отличие от массивно-параллельных систем — суперкомпьютеров с традиционной архитектурой — кластер строится на базе массово выпускаемых компонентов в большей степени и состоит из стандартных серверов — вычислительных узлов, объединённых высокопроизводительной коммуникационной сетью — интерконнектом.

Кластерная архитектура решений предоставляет пользователям вычислительных систем с суперкомпьютерным уровнем производительности ряд существенных преимуществ:

- наиболее выгодное соотношение «цена/производительность»;
- прекрасные возможности расширения: производительность кластера можно увеличить путём простого добавления стандартных вычислительных узлов;
- практически не ограниченная масштабируемость системы в рамках выделенной площади и энергетики ЦОДа;
- возможность построения широчайшего спектра конфигураций системы.

К недостаткам кластерных систем относятся в первую очередь:

- сложность достижения высокой производительности приложений при использовании большого числа узлов (масштабирование до десятков и сотен тысяч узлов);
- сравнительно высокая сложность обслуживания;
- сравнительно невысокая надёжность, в особенности при наличии большого числа узлов. Впрочем, при выходе из строя вычислительного узла его легко заменить без остановки системы в целом.

Поскольку суперкомпьютеры, построенные по принципу вычислительных кластеров, и завоевали сегодня львиную долю рынка благодаря своей универсальности и экономической эффективности, мы выбрали именно этот подход. Суть принципа — построение вычислительной системы на базе большого числа относительно простых стандартных блоков и обеспечение за счёт некоторых программных и архитектурных особенностей их согласованной и слаженной работы над единой задачей.

Структура вычислительного кластера

- Вычислительный кластер — сложная система, состоящая из большого числа разнообразных компонентов. Эффективность и работоспособность её зависит прежде всего от того, из каких «кубиков» она состоит и насколько связанно и согласованно они объединены между собой.

- Вычислительные узлы — компьютеры, составные элементы вычислительного поля, на которых производятся все вычисления. Мощность вычислительной системы определяется количеством и качеством вычислительных узлов, а также тем, как они между собой связаны в единое целое. Программная среда — системные программные средства, работающие на вычислительных узлах и обеспечивающие их взаимодействие: операционная система, необходимые драйверы коммуникационной сети, соответствующий сетевой программный стек, библиотека передачи сообщений (MessagePassingInterface, MPI).

- Коммуникационная сеть — набор сетевых элементов (коммутаторы, кабели, серверные адаптеры), с помощью которых строится сеть передачи сообщений, используемая для общения между собой процессов, работающих на вычислительных узлах. Коммуникационная сеть, как правило, характеризуется низкой задержкой и высокой пропускной способностью — именно эти показатели зачастую определяют скорость работы программы в параллельной среде.

- Транспортная и управляющая сети — служебные сети, служащие для объединения вычислительных узлов и их контроллеров удалённого управления, а также других компонентов системы; позволяют осуществлять удалённый консольный доступ к узлам, включение, выключение, перезагрузку узлов, диагностические действия.

- Управляющее ПО — набор программных средств, осуществляющих управление и мониторинг вычислительных узлов, управление задачами и учётными записями, организацию доступа пользователей и т. д.

- Управляющие серверы — набор серверов, на которых работает управляющее ПО.

- Система хранения данных — устройство или совокупность устройств, обеспечивающих хранение данных и доступ к этим данным с вычислительных узлов. Быстродействие системы хранения зачастую является одним из основных факторов, определяющих производительность вычислительной системы в целом.

- Система визуализации — набор систем, призванных отображать информацию в визуальном формате, осуществлять рендеринг больших сложных изображений. Система не является обязательной и определяющей, но её наличие помогает интерпретировать результаты расчётов.

- Средства разработки — компонент, необходимый для исследовательских систем. Включает в себя компиляторы, профилировщики, отладчики, математические библиотеки, анализаторы производительности и т. д.

Современная компьютерная индустрия ориентируется на стандарты. Стандартные процессорные архитектуры, стандартные средства межузловых взаимодействий, стандартные протоколы и библиотеки — всё это позволяет строить суперкомпьютеры из распространённых и недорогих строительных блоков. Стандартные технологии, применяемые и реализуемые в кластерных решениях, обладают отличными характеристиками с точки зрения производительности и масштабируемости и вполне успешно могут конкурировать с закрытыми специализированными решениями.

В качестве первого шага с помощью компании «РСК» был инсталлирован энергоэффективный суперкомпьютер производительностью 3 Тф с жидкостным охлаждением, оптимизированный для решения задач биоинформатики. Платформа «РСК Торнадо» изначально разрабатывалась как основа для высокопроизводительных кластерных систем большого масштаба (уровня петафлопса в 2011 г.). Установка, разработанная «РСК» для МФТИ, является пилотным проектом по развёртыванию системы такого класса для проблемно-ориентированных вычислений.

2.1. Вычислительная часть

Кластерное решение «РСК Торнадо» с жидкостным охлаждением призвано сочетать в себе все преимущества кластерного решения с рядом свойств, характерных для платформ следующего уровня — массивно-параллельных систем:

- Жидкостная система охлаждения обеспечивает высокую энергоэффективность решения (коэффициент PowerUsageEfficiency, PUE<1,2 — до 1,06), позволяет сохранить до 60% электроэнергии, которая бы тратилась на охлаждения кластерной установки той же мощности с воздушным охлаждением.

- Жидкостная система охлаждения обеспечивает компактность размещения установки, сокращает расходы на переоборудование помещений под центр обработки данных.

- Повышение надёжности вычислителя за счёт

  – тестирования узлов на всех стадиях реализации проекта;

  – пластины охлаждения оснащены быстроразъёмными муфтами, что позволяет демонтировать отдельный вычислительный узел без демонтажа системы охлаждения корзины (шасси) в целом;

  – каждый узел оснащён твердотельным накопителем объёмом 80 Гбайт. Использование твердотельных накопителей также направлено на повышение надёжности вычислителя – отказы шпиндельных дисковых накопителей составляют львиную долю причин отказов узлов в кластерных установках и вычислительных фермах.

- Использование жидкостного охлаждения позволило оснастить узлы прототипа суперкомпьютера (16 узлов) мощными процессорами IntelXeon5680 (3,33 GHz/130 Wt), которые благодаря такому охлаждению работают на повышенной частоте (технология TurboBoost) в 3,47 GHz *неограниченное* время. В результате МФТИ получает более мощное вычислительное поле и уникальный для кластерных установок уровень эффективности в **92**% на тесте LINPACK.

2.2. Сети

Сети суперкомпьютера «РСК Торнадо» основаны на открытых стандартах: Infiniband — для коммуникационной и GigabitEthernet — вспомогательной сетей.

Коммуникационная сеть реализована при помощи решений, поставляемых компанией Qlogic, скорость работы сети составляет 40 Гбит/с. Если на уровне прототипа (16-узловой кластер) использовался только один коммутатор Infiniband, при модернизации установки предполагается реализовать топологию FatTree с половинной бисекционной пропускной способностью.

Вспомогательная сеть — сеть GigabitEthernet — используется в основном для доступа к функциям IPMI-контроллеров узлов, а также как при загрузке системы. Возможно также предоставление доступа к сетевым дискам системы, использование сети как запасного канала для доступа пользователей на узлы и т. п.

2.3. Подсистема управления, набор ПО

Подсистема мониторинга и управления реализована на основе стандарта IPMI. Производителем на материнских платах узлов интегрированы специализированные контроллеры (BMC), осуществляющие мониторинг основных параметров физического состояния узлов (температура процессора, платы, напряжения и т. п.), а также предоставляющие базовые функции управления узлом: включения, выключения, горячей перезагрузки, KVMoverIP.

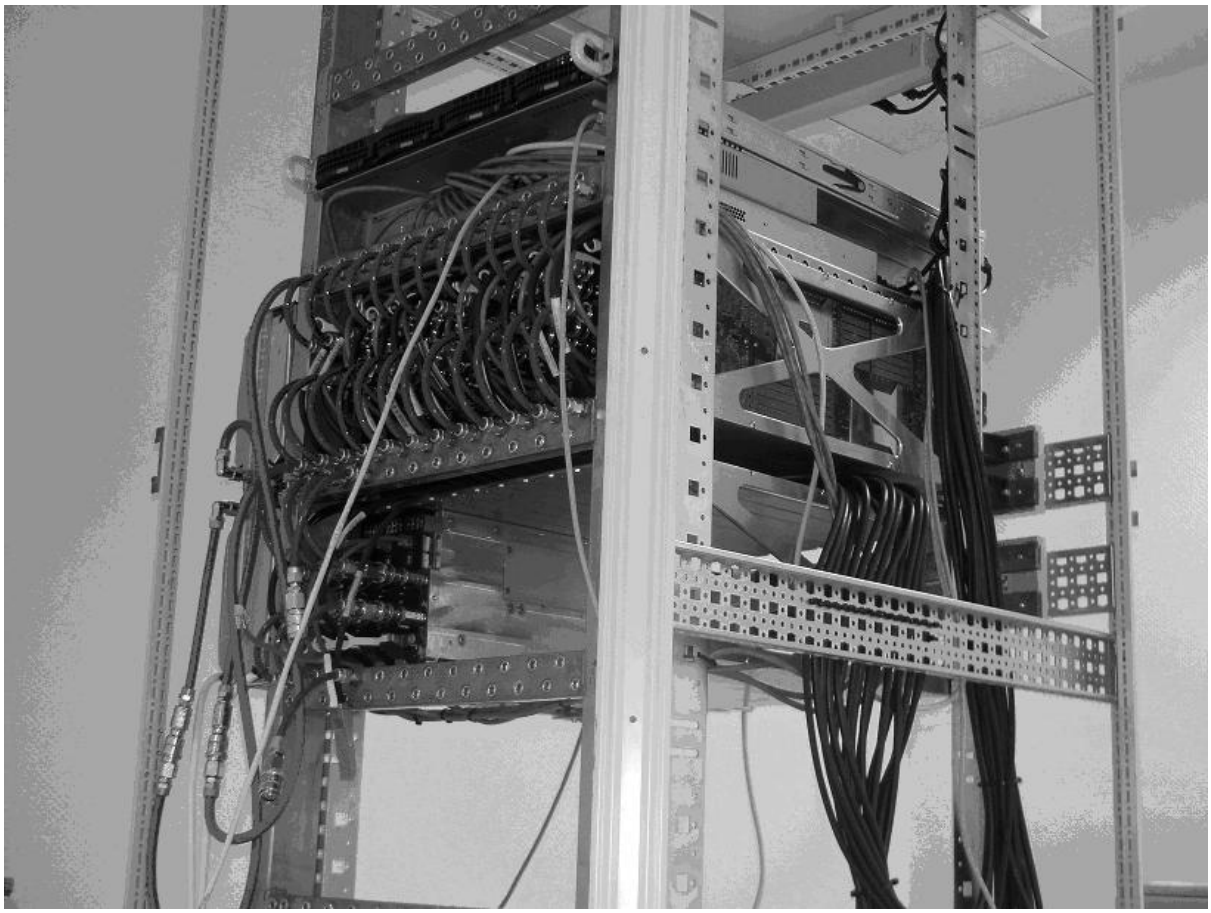

Рис. 1. Внешний вид прототипа высокопроизводительной системы решения задач в области биоинформатики

Система управления программной конфигурацией реализована на основе собственных решений «РСК» и ПО с открытым исходным кодом xCat. Загрузка операционной системы (ОС Linux семейства Debian) на узлах осуществляется всегда по сети с головного узла (вспомогательного сервера), за счёт чего обеспечивается единство программной конфигурации для всех узлов вычислительного кластера.

Таким образом, созданный кластер обладает целым рядом привлекательных характеристик:

- сокращение затрат на электроэнергию для охлаждения системы на 60% в год («зелёный» ЦОД);
- очень низкий показатель эффективности использования электроэнергии (PUE) на уровне $<$1,2 (до 1,06);
- повышенная надёжность, отсутствие шума и вибрации;
- сокращение площади ЦОД в 2,5–3 раза;
- упрощение настройки и конфигурации за счёт использования автоматических средств.

Планируется расширение системы в 2011 г. на основе узлов с процессорами E5–2600 (архитектура SandyBridge), общая производительность системы должна составить более 30 TFLOPS, общее число вычислительных узлов — более 100. Примерный вид системы «РСК Торнадо» после расширения показан на рисунке.

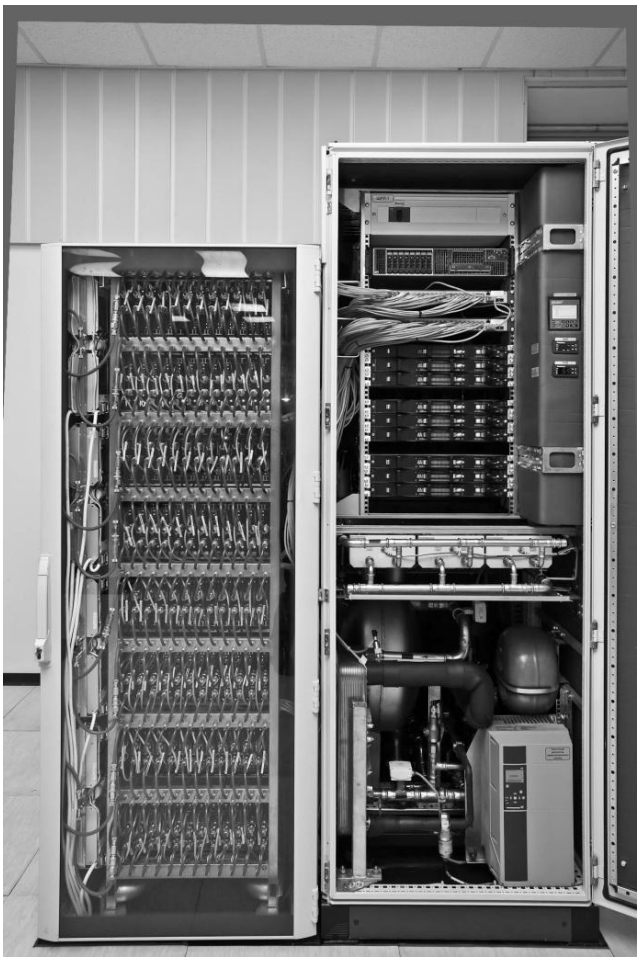

Рис. 2. Примерный вид системы «РСК Торнадо» после расширения

3. Модель вычислительной системы

Очевидно, что не существует универсального рецепта построения многопроцессорной системы. Для каждого класса вычислительных задач решение должно приниматься с учётом используемых алгоритмов, объёма обрабатываемых данных и т. п. Поэтому является актуальной задача предсказания эффективности предложенной архитектуры многопроцессорного вычислительного комплекса на выбранном классе вычислительных задач. Перспективным представляется подход с компьютерной симуляцией новой ЭВМ ещё до её постройки. Подобная практика хорошо зарекомендовала себя в области так называемой pre-Silicondevelopment [1, 2].

Дополнительное преимущество симуляционного подхода — возможность оценить производительность таких ЭВМ, реализация которых «в железе» является затруднительной по ряду причин (например, такая машина будет слишком дорога или ещё не существует компонент, из которых её можно было бы построить). Примером подобных исследований могут служить работы по построению моделей систем с миграцией исполняемого кода на узел с данными, подлежащими обработке (архитектура EM2 [3]), а также исследования компьютеров с транзакционной памятью [4, 5].

Модель многопроцессорной ЭВМ строится на основе параллельной распределённой системы симуляции Graphite [6]. Данная система предназначена для моделирования многоядерных процессоров, имеющих вплоть до нескольких тысяч ядер. Graphite позволяет запускать многопоточные приложения, использующие для распараллеливания библиотеку POSIXthreads под управлением ОС Linux, при этом требуется минимальное изменение их исходного кода (добавление двух вызовов функций в начале и в конце функции main()

для обозначения интересующего нас региона программы) и процесса сборки (необходимо добавить библиотеки поддержки Graphite). Работа распределяется среди многих машин, соединённых по сети, таким образом увеличивается скорость симуляции. Вместе с тем для исследуемого приложения сохраняется иллюзия того, что оно работает на одном компьютере как один процесс с единым адресным пространством.

Разработанный для моделирования многоядерных процессоров, симулятор Graphite предоставляет возможность моделировать разные части исследуемой архитектуры: вычислительный конвейер, подсистему памяти (кэши и динамическую память), сеть, соединяющую вычислительные ядра. По завершении работы приложения симулятор создаёт файл статистики, в котором содержится информация от этих моделей: число выполненных инструкций, эффективность работы кэшей, количество пакетов, прошедших по сетям, соединяющей вычислительные ядра, и т. п. Ещё раз отметим, что Graphite не предоставляет виртуализации на уровне операционной системы, лишь на уровне одного приложения. Для научных задач этого, как правило, достаточно, так как в большинстве случаев они не требуют знаний о системном уровне ЭВМ, на которой они выполняются. При проектировании модели ЭВМ, которая будет моделироваться под управлением Graphite, имеется возможность варьировать различные характеристики модели, касающиеся как системы межпроцессорных соединений, так и функций отдельного вычислительного ядра.

Для системы в целом мыслимы следующие параметризации:

1) варьировать полное число ядер в системе;

2) перераспределять ядра между уровнями блоков с общей памятью и группами ядер с независимой памятью, соединённых в сеть;

3) изменять топологию системы, т.е. определять, коммуникация между какими узлами возможна и какова её стоимость в терминах производительности: пропускная способность (throughput), задержка (latency).

Следующие возможности усматриваются в конфигурации одиночного ядра:

1) изменять количество и параметры частных кэшей ядра;

2) Изменять архитектуру ядра. Здесь возможности Graphite существенно ограничены, так как он не предоставляет средств к моделированию инструкций, не присутствующих в наборе команд используемого процессора. Фактически они сводятся к изменению ширины набора векторных инструкций SIMD, использованных при сборке приложения, и таким образом зависят от компилятора. Следует отметить, что этот вопрос требует отдельного тщательного рассмотрения.

Предлагается следующая методика для учёта всех видов взаимодействия вычислительного ядра с иерархией памяти и сетевой подсистемой. Каждый доступ по адресу в память регистрируется в модели системы кэшей общей памяти. С функциональной точки зрения отличие оперативной памяти от кэш-памяти состоит лишь в том, что чтение/запись в неё видны в соседних ядрах; с точки зрения потактовой модели разница состоит в увеличении времени доступа для более удалённых от ядра уровней памяти. Доступ достигает одного или нескольких уровней этой системы, в зависимости от характеристик модели. При этом определяется суммарное время доступа. Похожим образом, при вызове API модели сети соответствующая модель ответственна за маршрутизацию доступа, чтение/запись данных и регистрацию полной задержки, вызванной этим доступом. Иерархичность сетевой системы не показана, но она также может присутствовать.

В настоящее время в симуляторе уже присутствуют модели кэшей и сетей. Потребуется некоторая их модификация для того, чтобы получить более сложные модели, соответствующие реальному оборудованию; в первую очередь это касается топологий межпроцессорных соединений.

Суммируя всё вышеописанное, можно сказать, что наша задача состоит в измерении характеристик производительности для некоторого приложения на различных конфигурациях модели и нахождении оптимальной конфигурации, при которой есть основания считать, что время исполнения на реальной аппаратуре будет наиболее быстрым. Для её решения необходимо ввести метрики, позволяющие сравнивать конфигурации между собой. Фактически единственная ключевая характеристика построенной системы — это симулированная длительность исполнения программы. Не менее важной является профильная информация, собранная для отдельных подсистем, например, максимальная и средняя загруженность вычислительных ядер, пропускная способность сетевых соединений. Знания этих чисел позволяют определять «узкие места» и направленно оптимизировать систему.

В процессе работы по адаптации LINPACK мы обнаружили, что в настоящее время у Graphite существует несколько ограничений и нереализованных функций. Нами предложено несколько подходов к решению описанных выше проблем.

В результате симулятор Graphite является достаточно перспективным инструментом для задач исследований архитектуры многопроцессорных систем. В настоящее время, как активно развивающийся проект, он не лишён ряда недостатков, которые представляются нам исправимыми. Рассмотренный в данной работе подход позволяет достигнуть нескольких важных результатов при проектировании таких машин:

1) оценить производительность ЭВМ для различных её конфигураций с целью поиска наилучшей;

2) изучить поведение многопроцессорной системы на интересующем классе реальных приложений через симуляцию их выполнения;

3) ускорить процесс моделирования с помощью распределения симулятора на несколько компьютеров, прозрачного для изучаемого приложения.

В качестве класса задач, для которых будет производиться исследование архитектуры, были выбраны: приложения моделирования молекулярной динамики, используемые в биологии для изучения *расчётов белок-мембранных систем*, моделирование электропорации клеточной мембраны, а также построение полной модели вируса (на примера вируса энцефалита). Эта часть работы будет выполняться совместно с факультетом молекулярной и биологической физики, ИБХ РАН и химическим факультетом МГУ.

Литература

1. *Uhlig R., Fishtein R., Gershon O. et al.* SoftSDV: A presilicon software development environment for the IA–64 architecture // Intel Technology Journal. — 1999. — P. 112–126.

2. *Magnusson P.S., Christensson M., Eskilson J.et al.* Simics: A full system simulation platform // Computer. — 2002, Feb. — V. 35. — P. 50–58.

http://portal.acm.org/citation.cfm?id=619072.621909.

3. *Khan O.* EM2: A scalable shared-memory multicore architecture: Tech. rep. / Khan O., Liz M., Devadas S.: Massachusetts Institute of Technology. — 2010, Jun.

4. *Rajwar R.* Speculation-based techniques for transactional lock-free execution of lock-based programs: Ph.D. thesis / University of Wisconsin. — Madison. — 2002.

5. *Yen L., Bobba J., Marty M.R. et al.* LogTM-SE: Decoupling hardware transactional memory from caches // The 13th Annual International Symposium on High Performance Computer Architecture (HPCA-13). — 2007, Feb.

6. *Miller J.E., Kasture H., Kurian G. et al.* Graphite: a distributed parallel simulator for multicores // The 16th IEEE International Symposium on High-Performance Computer Architecture (HPCA). — 2010, Jan.

Поступила в редакцию 21.09.2011