РАЗРАБОТКА СИМУЛЯТОРА ВЫСОКОПРОИЗВОДИТЕЛЬНЫХ ВЫЧИСЛЕНИЙ ДЛЯ РЕШЕНИЯ ЗАДАЧИ ОПРЕДЕЛЕНИЯ ЭФФЕКТИВНОСТИ АЛГОРИТМОВ ПЛАНИРОВАНИЯ

И.Н. Лозгачев, А.В. Сенин

Нижегородский госуниверситет им. Н.И. Лобачевского
E-mail: ivan.lozgachev@itlab.unn.ru, andrey.senin@itlab.unn.ru

Введение

Симуляция широко используется для моделирования реальных процессов в самых различных областях применения, включая производство, строительство, информатику. Она обеспечивает решение широкого круга вопросов, таких, как проверка реализуемости идеи, оценки поведения и производительности без построения самой системы. Важный и широко используемый класс симуляторов – симуляторы высокопроизводительных распределенных систем. Например, некоторые инструменты созданы для моделирования Grid-систем, к реальным аналогам которых очень трудно получить доступ. Кроме того, почти невозможно оценить их производительность при различных сценариях использования, учитывая разное количество ресурсов и изменение их доступности с течением времени.

Требования к симулятору

В ННГУ в рамках учебно-исследовательского проекта разрабатывается собственная система управления кластерами – Метакластер [1]. Важнейшими отличительными особенностями данной системы являются возможность одновременного управления несколькими кластерами, работающими на базе основных операционных систем семейств Windows и Linux, а также интеграция с другими системами управления (в настоящий момент реализована интеграция с Microsoft Windows HPC Server 2008).

Система управления поддерживает различные стратегии распределения заданий по вычислительным ресурсам. Для сравнения эффективности этих стратегий необходимо использовать симулятор, так как реальный запуск заданий сделал бы недоступным ресурсы для пользователей на время эксперимента, что, конечно, недопустимо. Кроме задачи моделирования, от симулятора также требуется предоставление информации о планируемом времени выполнения пользовательских заданий для подсистемы планирования. Таким образом, задача моделирования ставит следующие требования к симулятору:

- задание модели произвольной гетерогенной вычислительной среды, включающей описание параметров узлов и топологию сети передачи данных;
- выполнение симуляции заданного набора вычислительных задач на выбранной вычислительной системе;
- предоставление пользователю результатов тестов в виде набора метрик и диаграммы распределения задач по узлам вычислительной системы.

Требование интеграции с подсистемой планирования выдвигает следующие требования к симулятору:

- высокая скорость работы при одновременном моделировании небольшого объема задач (до 100 активных задач в очереди);

- простота и краткость задания модели задача – параметры модели должны задаваться пользователем при постановке задачи.

Обзор существующих решений

В настоящее время разработано большое количество программных систем, позволяющих моделировать распределенные вычислительные ресурсы. Часть разработок поддерживает моделирование вычислительных ресурсов опционально (например, это одна из множества решаемых задач), часть систем изначально проектировалась как симуляторы грид или кластера. Нами был проведен анализ существующих систем с целью выбора лучшего кандидата. Часть рассмотренных симуляторов перечислена в табл. 1.

Таблица 1. Список наиболее распространенных симуляторов вычислительных сред

№	Средство	Описание	Разработчик	Направленность
1.	SimJava	SimJava [2] предоставляет процесс, основанный на дискретном моделировании событий с анимацией взаимодействий через коллекцию сущностей, взаимодействующих друг с другом.	University of Edinburgh, UK	Дискретная симуляция событий
2.	Bricks	Bricks [3] является инструментом оценки эффективности, который анализирует и сравнивает различные схемы планирования в высокопроизводительных системах типа Грид.	Tokyo Institute of Technology, Japan	Грид-симуляция
3.	MicroGrid	MicroGrid [4] предоставляет средство моделирования, которое поддерживает масштабируемую симуляцию Грид-приложений с использованием кластерных ресурсов в сетевых вычислительных исследованиях.	University of California at San Diego, USA	Грид-симуляция
4.	Simgrid	Simgrid [5] имитирует распределенные приложения в гетерогенных распределенных средах.	University of California at San Diego, USA	Грид-симуляция
5.	GridSim	GridSim [6] моделирует и симулирует процессы, происходящие в распределенных параллельных системах, на основе создания и мониторинга различных ресурсов.	Monash University, Australia	Грид-симуляция

К сожалению, нам не удалось найти симулятор, полностью удовлетворяющий всем выдвигаемым требованиям. К числу основных недостатков существующих систем можно отнести:

- Ограниченные возможности по моделированию вычислительных ресурсов. Так, GridSim предполагает связи внутри кластера однородными, что существенно ограничивает возможности по исследованию эффективности алгоритмов планирования на гетерогенных системах.
- Платформозависимость. Так, ядро SimGrid написано на C с использованием платформозависимых функций ОС Linux, что затрудняет использование симулятора на ОС Windows.

- Низкая скорость работы. Скорость моделирования в большинстве рассмотренных симуляторов даже небольших объемов задач делает невозможным использование этих симуляторов для планирования на реальных вычислительных системах.

В результате проведенного анализа нами было принято решение о написании собственного симулятора.

Реализация симулятора

Разрабатываемый симулятор включает 6 основных компонент:

- Планировщик. Модуль Метакластера, отвечающий за распределение вычислительных ресурсов по активным задачам.
- Стартер задач. Модуль Метакластера, запускающий задачи во время, назначенное планировщиком.
- Ядро симулятора. Основной модуль, несущий ответственность за инициализацию, заполнение очереди задач, обслуживание событий, запуск и остановку симуляции.
- Предсказатель времени работы задач. В случае использования пользовательской модели задачи данный модуль определяет время работы каждой задачи на текущей конфигурации системы.
- Подсистема контроля дискретной модели времени.
- Модуль отчетов. По результатам симуляции строится диаграмма Ганта, отображающая временную зависимость распределения ресурсов вычислительной системы по задачам.

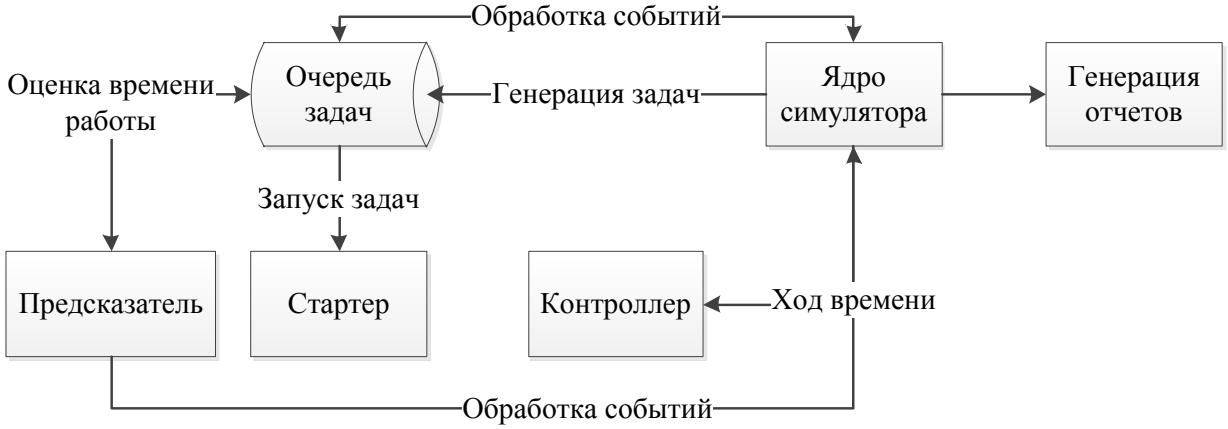

Рис. 1. Архитектура симулятора

Модель ресурсов и задачи

Основным требованием к модели вычислительных ресурсов и модели параллельной задачи была минимизация числа используемых параметров, чтобы сделать возможным задание описания задачи пользователем при постановке через веб-интерфейс системы управления.

Задача состоит из N параллельных процессов. Для каждого процесса задается:

- Количество инструкций I_{i_j}, которые выполняет процесс j программы i.
- Число передаваемых байт данных. Параметр $S_{i_{(a,b)}}$ определяет число байт, которое передаст процесс a процессу b параллельной программы i.
- Число пересылок данных $L_{i_{(a,b)}}$. Данный параметр необходим для оценки влияния латентности сети передачи данных.

Для задания вычислительного ресурса в симуляторе определены следующие сущности:

- узлы, задающиеся параметрами: объем памяти, скорость выполнения инструкций процессором, скорость и задержка доступа к памяти;
- сетевые коммутаторы, задающиеся числом портов, пропускной способностью и временем задержки;
- топология сети передачи данных.

Стратегия симуляции

Разрабатываемый симулятор рассматривает ход выполнения параллельной задачи как последовательность стабильных состояний, в каждом из которых задача выполняется с фиксированной скоростью, зависящей от внешних факторов (загруженность системы, наличие других задач на узлах, использование общей сети передачи данных) и внутренних (обмен данными между процессами). Количество инструкций, выполняемых процессом, скорость пересылки между каждой парой коммуницирующих процессов предполагается постоянной. Скорость выполнения пересчитывается лишь тогда, когда в расписании появляются новые задачи или завершается часть текущих. Этот подход позволяет достичь более высокой скорости симуляции и лучших показателей масштабируемости по сравнению с аналогами, многие из которых рассматривают каждую пересылку данных в отдельности.

Сравнение эффективности алгоритмов

Сравнение эффективности работы алгоритмов планирования проводится на основании метрик, подсчитанных по результатам планирования. Существуют различные метрики, усредненные значения которых используются для оценки эффективности алгоритма планирования. К наиболее распространенным критериям относятся:

- Время ожидания запуска.
- Время отклика – сумма времени ожидания запуска (T_w) и времени работы (T_r) задачи $T_w + T_r$.
- Время замедления – отношение времени отклика ко времени работы задачи $\frac{T_w + T_r}{T_r}$.
- Предельное время замедления – метрика, значение которой определяется следующей формулой $\max\left(\frac{T_w + T_r}{\max(T_r, \tau)}, 1\right)$, где τ – некоторая граничная константа времени работы задачи (как правило, устанавливается равной 10 с).

Проведение эксперимента

Разработанный симулятор был использован для оценки эффективности алгоритмов планирования, реализованных в Метакластере. В данном эксперименте возможности по предсказанию времени выполнения программ не использовались. Время выполнения задач предполагалось постоянным.

В приведенном эксперименте сравниваются алгоритм планирования, используемый в системе управления кластерами Метакластер и алгоритмы FIRSTFIT и BESTFIT, реализованные в открытой системе планирования Maui. Сравнение производится по критериям: среднее время ожидания и среднее время замедления для пяти наборов SWF-трасс [7]. Основное различие между наборами задач заключено в количестве небольших задач, длительность которых не превышает 1000 секунд.

Таблица 2. Список экспериментов

Название набора	Среднее время работы задач (с)	Общее количество задач	Количество небольших задач	Процент небольших задач
CTC-SP2-1995-1	9778.27	70918	40403	57
NASA-iPSC-1993-2.1-cln	764.9	18239	15936	87.4
SDSC-DS-2004-1	6696.23	96089	58470	60.8
DAS2-fs3-2003-1	716.07	66112	64395	97.4
DAS2-fs4-2003-1	2819.097	32953	24797	75.2

По второму критерию на двух последних наборах Метакластер значительно проигрывает Maui. Характерной особенностью двух последних наборов является большая доля недлительных задач. На третьем наборе, который содержит больше крупных заданий, планировщик Метакластера показал лучшие результаты.

Таким образом, можно сделать вывод, что планировщик Maui лучше использовать, если заранее известно, что доля недлительных задач значительна. Планировщик Метакластера нужно использовать в противном случае.

В настоящее время ведется тестирование и доработка компоненты предсказателя на реальных вычислительных задачах.

Работа выполнена в лаборатории ITLab факультета ВМК ННГУ.

Литература

1. Гергель В.П., Сенин А.В. Разработка системы управления интегрированной средой высокопроизводительных вычислений Метакластер // Вестник Нижегородского университета им Н.И. Лобачевского. Нижний Новгород, Изд-во Нижегородского госуниверситета, 2010. №6.
2. Официальный сайт SimJava. – [http://www.dcs.ed.ac.uk/home/hase/simjava].
3. Официальный сайт Bricks. – [http://matsu-www.is.titech.ac.jp/~takefusa/bricks].
4. Официальный сайт MicroGrid. – [http://www-csag.ucsd.edu/projects/grid/microgrid.html].
5. Официальный сайт Simgrid. – [http://grail.sdsc.edu/projects/simgrid].
6. Официальный сайт GridSim. – [http://www.buyya.com/gridsim].
7. Официальный сайт Parallel Workloads Archive. – [http://www.cs.huji.ac.il/labs/parallel/workload].