УДК 519.685

ИССЛЕДОВАНИЕ АЛГОРИТМОВ БАЛАНСИРОВКИ МЕТОДОМ ДИСКРЕТНО-СОБЫТИЙНОГО МОДЕЛИРОВАНИЯ

© 2011 А. Р. Хайрутдинов, С. В. Востокин

Самарский государственный аэрокосмический университет
имени академика С. П. Королёва (национальный исследовательский университет)

Рассматривается использование метода дискретно-событийного моделирования алгоритмов балансировки в системе GraphPlus. Описан способ моделирования передачи трафика в компьютерных сетях на основе сетей Петри специального вида. Указанная система усовершенствована с целью расширения функциональных возможностей. Представлены результаты экспериментов балансировки нагрузки методом дискретно-событийного моделирования.

Распределённая система, алгоритм балансировки, имитационное моделирование, сети Петри.

Введение

В настоящее время для решения все большего числа как научных, так и инженерных задач требуется наличие мощных вычислительных ресурсов. Для удовлетворения потребности в них на рабочих станциях необходимо использовать более производительную аппаратную часть, либо соединять станции между собой высокоскоростной сетью для совместной работы. Второй подход зачастую более приемлем, так как позволяет использовать уже существующий парк оборудования.

В соответствии с этим разрабатываются системы, среды и модели программирования для организации нескольких рабочих станций в единую распределённую вычислительную систему [1]. Однако гетерогенная природа подобных систем может вызывать ряд проблем: изменение количества вычислительных узлов, в том числе и в результате отказа оборудования; загруженность каналов передачи данных, вызывающая задержку исполнения распределённых приложений; необходимость более сложной синхронизации между узлами и т.д. В связи с этим предпочтительно использовать балансировку нагрузки, то есть обеспечить равномерную загруженность вычислительных узлов [2, 3].

Система исследования алгоритмов балансировки имитационным методом

При исследовании алгоритмов балансировки возникают следующие трудности:

- натурные эксперименты имеют высокую стоимость;
- подсистема измерения оказывает влияние на ход вычислительного процесса;
- системы общего назначения не подходят по ряду критериев.

Поэтому возникает потребность в создании специализированной системы для имитации процесса балансировки, задача которой состоит в предоставлении инструментальных средств моделирования и анализа процесса балансировки для различных конфигураций сети. Подобная система описана в работе [2].

Она основывается на модели программного комплекса GraphPlus, представляющей собой модель взаимодействующих посредством сообщений (*M*-объектов) процессов (*P*-объектов) [3]. В основе комплекса лежит библиотека времени исполнения, на базе которой формируются распределённые приложения. При этом управляющий процессом исполнения алгоритм отделён от прикладного кода и данных, что позволяет встроить код имитационной подсистемы в готовую среду исполнения GraphPlus [3].

По нашему мнению, существенным недостатком системы исследования алгоритмов балансировки [2] является имитация лишь ограничения скорости и невозможность адекватно представить ситуацию переполнения очередей передатчика данных, что актуально при использовании множества

соединений типа "точка-точка". Кроме того, распределённые системы характеризуются гетерогенной структурой и нестабильной передачей данных. Поэтому требуется проводить моделирование внешних помех и вызванных ими повторных передач данных.

Моделирование вычислительных сетей на основе сетей Петри

Подход к моделированию вычислительных сетей на основе иерархических раскрашенных сетей Петри с временными задержками представлен в работе [4]. Цвет узлов соответствует типу трафика: "сообщение" или "маркер занятости" ресурса обработкой сообщения. Под иерархичностью понимается то, что любой объект в сети моделируется как подсеть Петри. Время в модели принимается дискретным и измеряется тактами; причём в качестве такта принимается временной интервал, в течение которого срабатывают все разрешённые переходы.

Формально можно дать следующее описание имитационной модели:

$$W = \{P, T, D, TR\}, \quad (1)$$

где P – множество позиций, T – множество переходов, D – множество дуг, TR – множество типов трафика.

Для каждого перехода и типа трафика существует (возможно, нулевая) временн*а*я задержка q, характеризующая скорость передачи данных по каналам связи. Для каждого перехода введена очередь поступивших меток.

В работе [4] для каждого сетевого устройства авторами выделяются специализированные "роли" передатчика, приёмника, генератора трафика и строятся элементарные сети, названные ими ролевыми функционалами. Авторы определяют алгебру операций над сетями и на основе операций, применяемых над ролевыми функционалами, показывают, как осуществляется представление различных устройств сети с помощью данного метода.

Описанный подход удобно применить для доработки имитационного исполнения эксперимента в программном комплексе [2]. При этом можно сделать допущение об отсутствии разного типа трафика в сети, в результате чего модель (1) упрощается: $W = \{P, T, D\}$.

Экспериментальная проверка

После доработки программного комплекса была осуществлена проверка адекватности его работы. Она состояла из трёх этапов. На первом этапе были воспроизведены все эксперименты, описанные в [2], расчёт эффективности работы статического и динамического алгоритмов балансировки и их сравнение между собой. В итоге были получены те же результаты.

На втором этапе проверялась правильность работы системы в условиях внешних помех на каналах передачи данных, что вызывало бы необходимость повторной передачи сообщений. На рис. 1 представлена модель сети, состоящая из трёх узлов. Канал связи между вторым и третьим узлом характеризуется сильным уровнем помех, из-за которых с вероятностью $P_{пов}$ следует осуществлять повторную передачу данных.

Следует отметить, что в реальных расчётах при построении статистических моделей случайных флуктуационных помех обычно используют гауссовский вероятностный закон [5], однако для простоты проверки было принято $P_{пов} = 0{,}1$. Обоснование возможности использования более простых видов флуктуационных помех в задачах балансировки давалось нами в работе [3].

С указанной моделью проводились серии экспериментов для случая разделения вычислительной задачи на четыре независимых подзадачи. При этом использовались статический, динамический и комбинированный алгоритмы балансировок и рассчитывалась эффективность их работы. Результаты

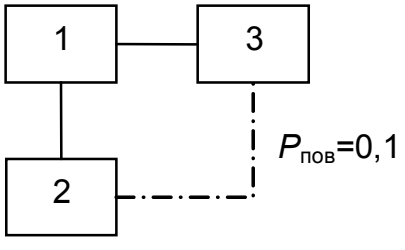

Рис. 1. Модель эксперимента

Таблица 1. Результаты исследования алгоритмов балансировок для разбиения задачи на четыре подзадачи при трёх вычислительных узлах

Тип балансировки	Эффективность, $T_1/(T_3\times 3)$	Эффективность, $T_1/(T_3\times 3)$ по работе [2]
Статическая	0,7065	0,7829
Динамическая	0,7140	0,6434
Комбинированная	0,8245	0,8121

приведены в табл. 1. По ним видно, что эффективность работы модифицированных алгоритмов не уступает результатам, приведенным в [2], при более простой реализации модели.

На третьем этапе были проведены эксперименты для проверки новой возможности системы моделирования получения точных решений статической задачи балансировки нагрузки путем перебора компоновок процессов (*P*-объектов в терминах [3]) по логическим процессорам.

Пусть $M: P \to C$ функция, ставящая в соответствие *P*-объекту $p \in P$ логический процессор $c \in C$. Решением задачи балансировки является $agr\ \min_{m \in M} T(m, t_{выч}, t_{пер})$, где T – функция, вычисляющая время исполнения алгоритма путем дискретно-событийной имитации. Данная функция минимизируется путем подбора оптимальной компоновки процессов по процессорам m при заданных постоянных параметрах: $t_{выч}$ - время вычисления элементарной операции алгоритма; $t_{пер}$ - время передачи сообщения между логическими процессорами. Компоновка влияет на временные затраты при доставке сообщений. Если $m(p_1) = m(p_2)$, то время доставки сообщений между процессами p_1 и p_2 принимается равным нулю. В данной серии экспериментов полагаем, что логические процессоры объединены полносвязным коммуникационным графом и время передачи сообщений постоянно.

В качестве примера алгоритма, для которого выполняется балансировка, мы рассмотрели алгоритм работы конвейера, каждая ступень которого выполняет элементарную операцию при получении данных от своего левого соседа и подтверждении доставки от правого соседа. После завершения элементарной операции, выполняется передача результатов правому соседу, а подтверждение доставки – левому. Практическим применением такого алгоритма может быть параллельное решение явно-неявных разностных схем, где ступень конвейера отвечает за локальные операции над фрагментом сеточной области.

В рамках третьего этапа нами выполнено решение задачи балансировки в разработанной имитационной модели методом полного перебора. В экспериментах определялись: реальное время решения задачи; длительность исполнения в модельном времени; вид компоновки, приводящий к наименьшему времени выполнения. Результаты экспериментов представлены в таблице 2. Рассматривалось статическое (постоянное в течение времени вычислений) распределение *P*-объектов по логическим процессорам. Задача решалась для трёх виртуальных процессоров и 12 *P*-объектов. Таким образом, мощность множества функций компоновок с учетом симметричных компоновок M составила $3^{12} = 531441$.

В таблице 2 используются следующие обозначения. Функция $m \in M$ кодируется перечислением своих значений. Например, тривиальная компоновка 000011112222 означает, что процессы 1-4 исполняются на процессоре 0, процессы 5-8 исполняются на процессоре 1, а процессы 9-12 – на процессоре 2. Число итераций $n_{итер}$ - сколько раз сигнал по конвейеру прошел от крайнего левого процесса до крайнего правого. Характеристика гранулярности $t_{выч} / t_{пер}$ - отношение времени выполнения элементарной операции ко

Таблица 2. Статическая балансировка методом полного перебора компоновок

M	$n_{итер}$	$t_{выч}/t_{пер}$	$t_{моделир}$	s	σ, %	$t_{,реальное}, c$
000000000000	6	∞	72	1	33,3	<0,01
000011112222	6	∞	35	2,06	68,7	<0,01
210012210012	**6**	∞	**28**	**2,57**	**85,7**	**42,9**
000011112222	6	10	35,4	2,03	67,7	<0,01
001122200112	**6**	**10**	**31,9**	**2,26**	**75,3**	**44,3**
000011112222	6	5	36,8	1,96	65,3	<0,01
001122200112	**6**	**5**	**34,6**	**2,08**	**69,3**	**44,4**
000000000000	12	∞	144	1	33,3	<0,01
000011112222	12	∞	59	2,44	81,3	<0,01
210012210012	**12**	∞	**52**	**2,77**	**92,3**	**83,6**
000011112222	12	10	60,6	2,37	79,0	<0,01
001122200112	**12**	**10**	**58,1**	**2,48**	**82,7**	**85,8**
000011112222	12	5	63,2	2,28	76,0	<0,01
200001111222	**12**	**5**	**61,6**	**2,34**	**78,0**	**85,8**

времени передачи сообщения. Более низкое значение означает большие коммуникационные издержки. Время $t_{моделир}$ - длительность счёта в модельном (безразмерном) времени; s – ускорение вычислений; σ – эффективность вычислений характеризует долю времени, в течение которого логические процессоры загружены. Время $t_{,реальное}$ - время в секундах, за которое подобрана оптимальная компоновка на компьютере с процессором Intel Core2 CPU T5300 @ 1.73GHz, RAM 1 GB, MS Vista SP2.

В таблицах выделены строки с оптимальными распределениями процессов (P-объектов) по логическим процессорам.

В силу высокой трудоёмкости полного числа прогонов имитационной модели для всех возможных функций $m \in M$, во второй группе тестов полный перебор был заменен вычислением времени T для случайно сгенерированных компоновок. Результаты балансировки методом случайного перебора представлены в таблице 3. В данной серии экспериментов количество попыток равня-

Таблица 3. Статическая балансировка методом частичного перебора компоновок

M	$n_{итер}$	$t_{выч}/t_{пер}$	$t_{моделир}$	s	σ, %	$t_{,реальное}, c$
000000000000	6	∞	72	1	33,3	<0,01
000011112222	6	∞	35	2,06	68,7	<0,01
021120120021	**6**	∞	**28**	**2,57**	**85,7**	**4,6**
000011112222	6	10	35,4	2,03	67,7	<0,01
001122200112	**6**	**10**	**32,3**	**2,23**	**74,3**	**4,7**
000011112222	6	5	36,8	1,96	65,3	<0,01
001122200112	**6**	**5**	**34,6**	**2,08**	**69,3**	**4,5**
000000000000	12	∞	144	1	33,3	<0,01
000011112222	12	∞	59	2,44	81,3	<0,01
021120120021	**12**	∞	**52**	**2,77**	**92,3**	**8,5**
000011112222	12	10	60,6	2,37	79,0	<0,01
001122200112	**12**	**10**	**58,1**	**2,48**	**82,7**	**8,7**
000011112222	12	5	63,2	2,28	76,0	<0,01
122220000111	**12**	**5**	**61,6**	**2,34**	**78,0**	**8,7**

лось 10% от мощности множества функций компоновок.

Сравнение значений эффективности σ и ускорений *s* по таблицам 2 и 3 позволяет сделать вывод о том, что на практике, когда эффективность определяет множество трудно учитываемых факторов, можно применять частичный перебор компоновок. Более того, при использовании случайного поиска на 10% значений, мы только в одном случае $n_{итер} = 6,\ t_{выч} / t_{пер} = 10$ получили незначительное отличие результата от оптимального (32,3 против 31,9).

Интересным результатом, полученным на имитационной модели, является не оптимальность тривиальных компоновок, в которых ступени конвейера сгруппированы по порядку (рис. 2). Наблюдаемую асимметрию оптимальных компоновок можно объяснить наличием фаз остановки и разгона конвейера, когда часть ступеней простаивает. Проведённые эксперименты позволяют оценить размерности задач, решаемых методом «грубой силы», так как трудоёмкость решения не связана с типом алгоритма, а лишь с числом составляющих его элементарных процессов.

Заключение

В работе проведено усовершенствование системы имитационного моделирования алгоритмов балансировки с целью расширения её функциональных возможностей и упрощения архитектуры. Выполнена серия экспериментов для сравнения с аналогичными исследованиями. Получены новые точные решения статической задачи балансировки с использованием полного и частичного перебора пространства компоновок. Это достигнуто благодаря более эффективному исполнению модели. Следовательно, можно сделать заключение о работоспособности и пригодности предложенной системы имитационного моделирования для исследования алгоритмов балансировки.

Данная статья написана по результатам проведения поисковой научно-исследовательской работы в рамках реализации ФЦП "Научные и научно-педагогические кадры инновационной России" на 2009-2013 годы.

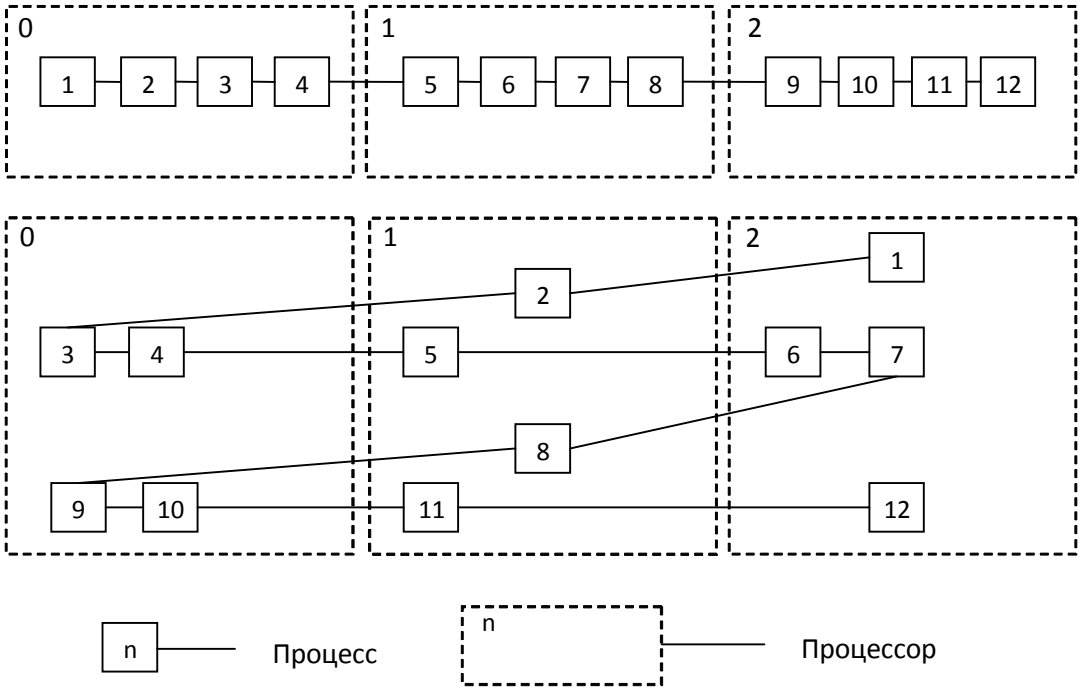

Рис. 2. Графическое представление неоптимальной (сверху) и оптимальной (снизу) компоновки

Библиографический список

1. Востокин, С. В. Применение метода парного взаимодействия объектов для построения сред разработки распределенных приложений [Текст] / С. В. Востокин // Вестник Самарского государственного технического университета. – 2005. – № 38. – С. 26-28.

2. Зекцер, И. Д. Разработка системы исследования алгоритмов балансировки имитационным методом [Текст] / И. Д. Зекцер // Инфокоммуникационные технологии. – 2010. – № 4. – С. 36-40.

3. Востокин, С. В. Графическая объектная модель параллельных процессов и ее применение в программных комплексах численного моделирования [Текст] // Дисс. д.т.н. – Самара: Издательство СНЦ РАН, 2007. – 286 с.

4. Гудов, А. М. Имитационное моделирование процессов передачи трафика в вычислительных сетях [Текст] / А. М. Гудов, М. В. Семехина // Управление большими системами. – 2010. – №31. – С. 130-161

5. Васильев, К. К. Математическое моделирование систем связи: учебное пособие [Текст] / К. К. Васильев, М. Н. Служивый. – Ульяновск: УлГТУ, 2008. – 170 с.

ANALYSIS OF LOAD BALANCING ALGORITHMS USING DISCRETE EVENT SIMULATION

© 2011 A. R. Khayrutdinov, S. V. Vostokin

Samara State Aerospace University
named after academician S. P. Korolyov (National Research University)

Discrete event simulation method is used to research load balancing algorithms in a program. Traffic transmission processes in computing networks are simulated with Petri nets. The program is changed to apply this method.

Distributed system, load balancing algorithm, simulation, Petri nets.

Информация об авторах

Хайрутдинов Андрей Ринатович, аспирант кафедры информационных систем и технологий, Самарский государственный аэрокосмический университет имени академика С. П. Королёва (национальный исследовательский университет). E-mail: khairutdinov@yandex.ru. Область научных интересов: распределённые системы, параллельное программирование.

Востокин Сергей Владимирович, доктор технических наук, профессор кафедры информационных систем и технологий, Самарский государственный аэрокосмический университет имени академика С. П. Королёва (национальный исследовательский университет). E-mail: easts@mail.ru. Область научных интересов: операционные системы, параллельное и распределенное программирование, инструменты и технологии.

Khayrutdinov Andrey Rinatovitch, post-graduate student of the department of information systems and technologies, Samara State Aerospace University named after academician S. P. Korolyov (National Research University). E-mail: khairutdinov@yandex.ru. Area of research: distributed systems, parallel programming.

Vostokin Sergey Vladimirovitch, doctor of technical sciences, professor of the department of information systems and technologies, Samara State Aerospace University named after academician S. P. Korolyov (National Research University). E-mail: easts@mail.ru. Area of research: operating systems, parallel and distributed computing, tools and techniques for automation in the field of concurrency.