УДК 004.272

Моделирование алгоритмов децентрализованной диспетчеризации параллельных задач в пространственно-распределенных мультикластерных вычислительных системах

М.Г. Курносов, А.А. Пазников

Сибирский государственный университет телекоммуникаций и информатики, Новосибирск

mkurnosov@gmail.com, apaznikov@gmail.com

Рассматриваются алгоритмы децентрализованной диспетчеризации параллельных программ в пространственно-распределенных вычислительных системах. Алгоритмы позволяют учитывать динамически изменяющуюся загрузку ресурсов системы. Описывается функциональная структура пакета GBroker и сеанс работы пользователя. Представлены результаты моделирования на действующей пространственно-распределенной мультикластерной вычислительной системе.

Ключевые слова: *диспетчеризация параллельных программ, пространственно-распределенные вычислительные системы, GRID-системы.*

Введение. На сегодняшний день для решения сложных задач науки и техники широкое распространение получили пространственно-распределенные вычислительные системы (ВС). Они представляют собой макроколлективы пространственно рассредоточенных ВС (вычислительных подсистем), взаимодействующих между собой через локальные и глобальные сети связи (включая всемирную сеть Internet) [1]. К таким системам относятся GRID-системы и мультикластерные ВС.

При организации функционирования пространственно-распределенных ВС возникает задача диспетчеризации параллельных программ, поступающих в очереди вычислительных подсистем. Для каждой программы необходимо определить, на каких вычислительных ресурсах каких подсистем она будет выполняться. В средствах диспетчеризации должны учитываться динамичность состава систем и переменная загрузка их вычислительных ресурсов.

Централизованные системы диспетчеризации имеют существенный недостаток: отказ управляющего узла может привести к неработоспособности всей ВС. Актуальной является задача разработки децентрализованных моделей, алгоритмов и системного программного обеспечения диспетчеризации параллельных задач в пространственно-распределенных вычислительных и GRID системах.

При децентрализованной схеме диспетчеризации задач в системе функционирует коллектив диспетчеров, принимающий решение о выделении ресурсов для программ. Это позволяет достичь живучести ВС, то есть ее способности продолжать работу при отказах отдельных подсистем.

На сегодняшний день существует несколько пакетов централизованной диспетчеризации параллельных программ в пространственно-распределенных системах: GridWay, Common Scheduler Framework (CSF), Nimrod/G, Condor-G, GrADS, AppLeS, DIRAC, WMS и др. В GridWay [2, 3] преследуется цель минимизации времени обслуживания задачи; реализован механизм миграции задач между подсистемами. Среда CSF [4] предоставляет средства резервирования ресурсов, на основе алгоритма Round-Robin и созданных пользователем алгоритмов. В основу Nimrod/G [5] положена

экономическая модель, целью которой является поиск равновесного состояния между поставщиками и потребителями ресурсов посредством механизма аукциона. В Condor-G [6] предполагается, что пользователь самостоятельно выбирает вычислительную подсистему из списка доступных; при этом поддерживаются пользовательские диспетчеры. Функционирование WMS [7] основывается на применении двух политик: в одном случае задача отправляется на подсистему для решения как можно быстрее, в другом – задача ожидает в очереди до тех пор, пока ресурс не станет доступным. В DIRAC [8] на вычислительных подсистемах установлены программные агенты, которые при освобождении ресурсов запрашивают задачи из глобальной очереди.

В работе рассматривается разработанный инструментарий децентрализованной диспетчеризации параллельных MPI-программ в пространственно-распределенных мультикластерных и GRID системах.

Постановка задачи. Пусть пространственно-распределенная ВС состоит из H вычислительных подсистем и включает N элементарных машин (ЭМ). Через n_i обозначим количество ЭМ, входящих в состав подсистемы $i \in S = \{1, 2, \ldots, H\}$. Известны значения показателей производительности каналов связи между подсистемами. Пусть $b_{ij} = b(i, j, m)$ – пропускная способность канала связи между подсистемами $i, j \in S$ при передаче сообщений размером m байт ($[b_{ij} = b(i, j, m)]$ = байт/с).

На каждой подсистеме присутствует диспетчер, который поддерживает очередь параллельных программ и осуществляет поиск вычислительных ресурсов для их выполнения. Коллектив диспетчеров представлен в виде ориентированного графа $G = (S, E)$, в котором вершинам соответствуют диспетчеры, а ребрам – логические связи между ними. Наличие дуги $(i, j) \in E$ означает, что диспетчер i может отправлять ресурсные запросы диспетчеру j. Множество всех вершин j, входящих в дуги (i, j), образует локальную окрестность $L(i) = \{j \in S : (i, j) \in E\}$ диспетчера i.

Пользователь направляет программу и запрос на выделение ресурсов одному из диспетчеров. Диспетчер в соответствие с реализованным в нем алгоритмом осуществляет поиск (суб)оптимальной подсистемы $i^* \in S$ для выполнения программы пользователя. Считаем, что программа пользователя характеризуется рангом r – количеством параллельных ветвей, ожидаемым временем t выполнения программы (walltime) и суммарным размером z исполняемых файлов и данных ([z] = байт).

Рассмотрим децентрализованный алгоритм диспетчеризации параллельных программ, ориентированный на минимизацию времени их обслуживания.

Алгоритм диспетчеризации. При поступлении программы в очередь диспетчера i, он опрашивает диспетчеры $j \in L(i)$ из своей локальной окрестности и получает от них информацию о текущем количестве задач в их очередях и оценку времени t_j, через которое программа может быть запущена на ресурсах подсистемы j в случае передачи задачи в её очередь. Через систему мониторинга определяются текущие значения пропускных способностей $b_{ij} = b(i, j, z)$ каналов связи между подсистемами. После этого выбирается подсистема i^*, обеспечивающая минимальное значение времени T обслуживания параллельной программы. Время обслуживания включает в себя время доставки исполняемых файлов и данных до выделенной подсистемы, время ожидания в очереди подсистемы и время выполнения программы.

$$i^* = \arg\min_{j \in L(i)} \{T(i)\}, \tag{1}$$

$$T(i) = z / b(i, j, z) + t_j + t. \tag{2}$$

После чего программа пересылается в очередь подсистемы i^*.

При поступлении задачи в очередь подсистемы i^*, для нее также периодически осуществляется поиск ресурсов. Это обеспечивает адаптацию под динамически изменяющуюся загрузку подсистем пространственно-распределенных ВС.

Программный пакет децентрализованной диспетчеризации GBroker. В Центре параллельных вычислительных технологий ГОУ ВПО “Сибирского государственного университета телекоммуникаций и информатики” (ЦПВТ ГОУ ВПО “СибГУТИ”) создан и развивается программный пакет GBroker [9] децентрализованной диспетчеризации программ в пространственно-распределенных ВС. Пакет создан на языке ANSI C для операционной системы GNU/Linux.

В пакет (рис. 1) входят диспетчер gbroker, который устанавливается на каждой из подсистем и взаимодействует с локальным менеджером задач, клиентская часть gclient, реализующую интерфейс между пользователем и распределенной ВС, и модуль netmon мониторинга производительности каналов связи на уровне стека протоколов TCP/IP.

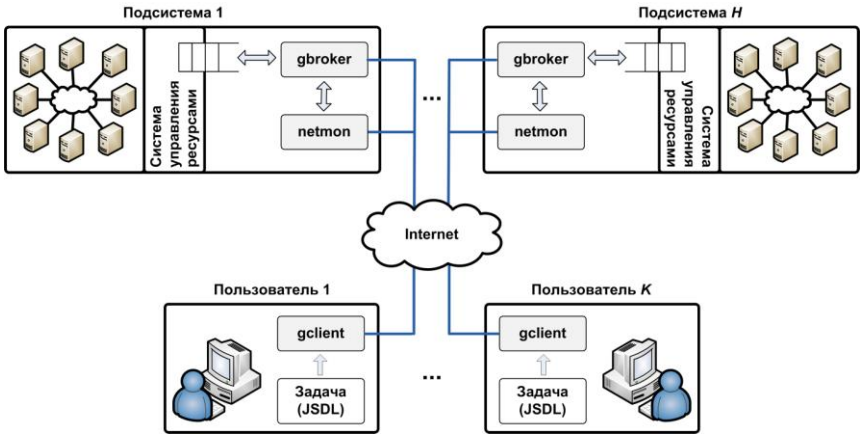

Рис. 1. Функциональная структура пакета GBroker

Модуль gbroker устанавливается на каждой из подсистем и обеспечивает интерфейс с локальной системой пакетной обработки заданий с помощью сервиса GRAM (Grid Resource Allocation Management), входящего в состав инструментария для организации функционирования GRID-систем Globus Toolkit 5.0. Gbroker взаимодействует с ограниченным числом других диспетчеров, образующих его локальную окрестность.

Модуль netmon устанавливается вместе с gbroker на подсистемах. Сервис netmon выполняет оценку производительности каналов связи между подсистемами. Модуль gclient реализует интерфейс между пользователем и системой.

Администратор настраивает локальные окрестности диспетчеров gbroker, указывая, какие диспетчеры с какими могут обмениваться программами из своих очередей, и настраивает службу netmon. Пользователь формирует задание, состоящее из параллельной MPI-программы и паспорта на языке ресурсных запросов JSDL (Job Submission Description Language), и отправляет его средствами gclient любому из диспетчеров gbroker. Диспетчер в соответствии с описанным алгоритмом выбирает подсистему, на которой затем выполняется программа.

Результаты экспериментов. Созданный инструментарий децентрализованной диспетчеризации параллельных задач исследован на действующей мультикластерной ВС, созданной ЦПВТ ГОУ ВПО “СибГУТИ” совместно с Лабораторией ВС ИФП СО РАН. В экспериментах было использовано 3 сегмента мультикластерной ВС (рис. 3):

– кластер Xeon16: 4 узла (2 x Intel Xeon 5150, 16 процессорных ядер);
– кластер Xeon32: 4 узла (2 x Intel Xeon 5345, 32 процессорных ядра);
– кластер Xeon80: 10 узлов (2 x Intel Xeon 5420, 80 процессорных ядер).

Сеть связи между сегментами ВС Fast Ethernet.

На сегментах системы был установлен пакет Gbroker и настроен компонент netmon (рис. 3). В локальную окрестность каждого диспетчеры были включены диспетчеры всех кластеров.

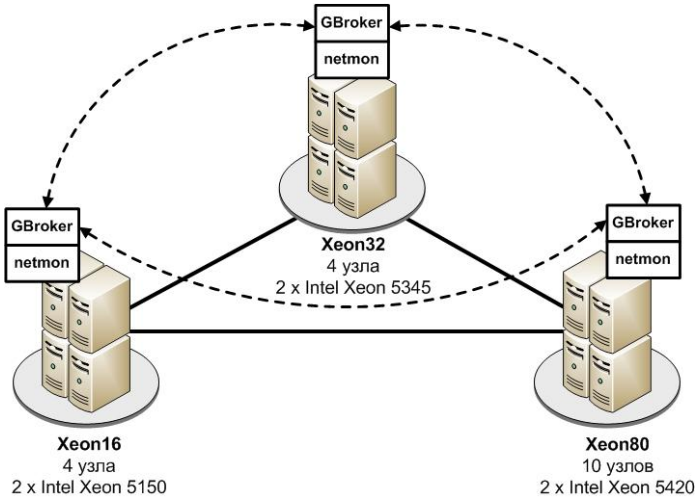

Рис. 2. Тестовая конфигурация пространственно-распределенной мультикластерной ВС: $H = 3$

В качестве тестовых задач использовались MPI-программы из пакета NAS Parallel Benchmarks (NPB). Моделирование проводилось следующим образом. На выбранную подсистему поступал стационарный поток из $M = 500$ параллельных задач. Тестовые задачи выбирались псевдослучайным образом. Ранг r_i параллельной программы генерировался в соответствии с распределением Пуассона со значениями математического ожидания $r \in \{4, 8, 12, 16\}$. Задачи поступали в очередь диспетчера кластера Xeon80 через интервалы времени t, экспоненциально распределенные со значениями интенсивности $\lambda \in \{0.1, 0.2, 0.3, 0{,}4\}$, $[\lambda] = c^{-1}$.

Обозначим через t_i – момент времени поступления задачи $i \in \{1, 2, \ldots, M\}$ на вход диспетчера, t_i' – момент времени начала решения задачи i, t_i'' – момент завершения решения задачи i. Тогда суммарное время τ обслуживания (ожидания в очереди и решения) M задач составит

$$\tau = \max_{i=1,2,\ldots,M} t_i'' - \min_{i=1,2,\ldots,M} t_i .$$

Для оценки эффективности алгоритма диспетчеризации использовались следующие показатели:

– среднее время T обслуживания задачи

$$T = 1/M \sum_{i=1}^{M} (t_i'' - t_i) ,$$

– среднее время W пребывания задачи в очереди

$$W = 1/M \sum_{i=1}^{M} (t_i' - t_i) ,$$

– пропускная способность B системы

$$B = M / \tau .$$

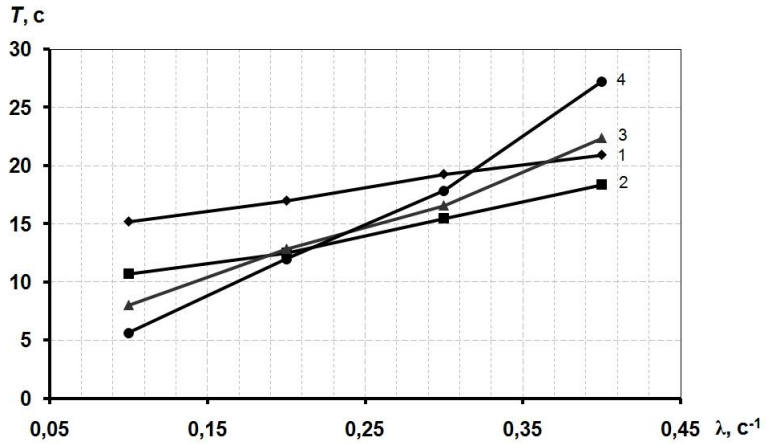

Рис. 3. Зависимости среднего времени обслуживания задач
от интенсивности λ потока задач

Из графиков (рис. 4) видно, что среднее время обслуживания задач увеличивается с ростом интенсивности потока поступления задач. Это связано с образованием очередей на обслуживание в системе пакетной обработки заданий. Такое влияние интенсивности потока особенно значительно при рангах параллельных программ, близких к размерам подсистем.

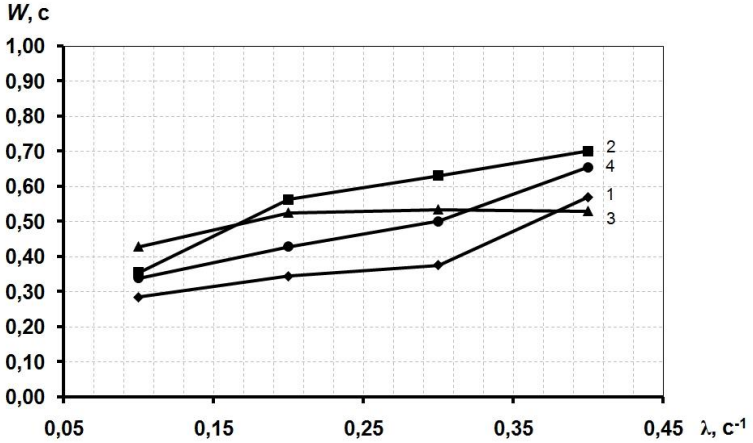

Рис. 4. Зависимости среднего времени диспетчеризации задач
от интенсивности λ потока задач

По графикам (рис. 4) можно определить, что время диспетчеризации задач незначительно по сравнению с суммарным временем обслуживания и в среднем не превосходит одной секунды (для рассматриваемой системы). Среднее время диспетчеризации задач не изменяется существенно как с ростом интенсивности потока поступления задач, так и с увеличением среднего ранга параллельных задач.

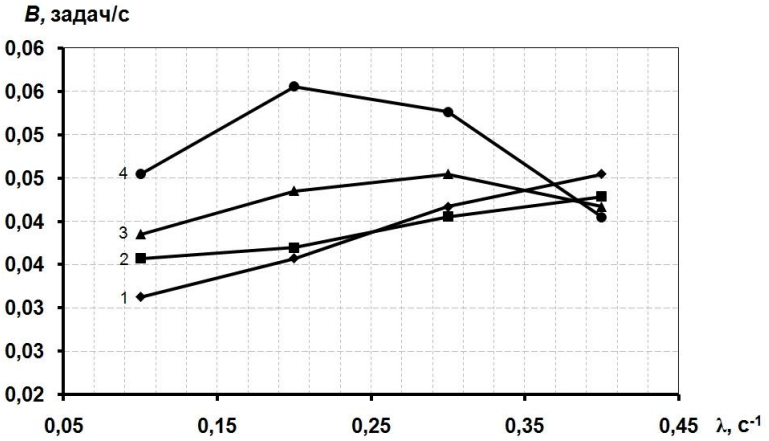

Рис. 5. Зависимости пропускной способности системы от интенсивности λ потока задач

Пропускная способность системы (рис. 5) растет при увеличении интенсивности потока поступления задач. При больших значениях интенсивности происходит снижение пропускной способности системы, что связано с образованием очередей на обслуживание в системе пакетной обработки. С увеличением среднего ранга это влияние интенсивности на пропускную способность системы усиливается.

Заключение. Централизованные системы диспетчеризации большемасштабных распределенных ВС характеризуются вычислительной сложностью поиска требуемых ресурсов. Децентрализованная диспетчеризация параллельных программ в пространственно-распределенных вычислительных и GRID-системы позволяет повысить их живучесть. Кроме того, применение централизованных систем диспетчеризации в большемасштабных ВС ограничено вычислительной сложностью поиска требуемых ресурсов.

Результаты исследования созданного инструментария децентрализованной диспетчеризации параллельных MPI-программ на мультикластерной ВС показали, что среднее время обслуживания задач и при децентрализованной, и при централизованной диспетчеризации сопоставимы. Время диспетчеризации достаточно мало по сравнению со временем выполнения задач.

Литература

1. Хорошевский В. Г. Архитектура вычислительных систем: Учеб. Пособие для вузов. – М.: Изд-во МГТУ им. Н. Э. Баумана, 2008.
2. R. Montero, E. Huedo, I. Llorente. Grid Resource Selection for Opportunistic Job Migration // 9th International Euro-Par Conference. – 2003. – V. 2790. – P. 366-373.
3. E. Huedo, R. Montero, I. Llorente. A Framework for Adaptive Execution on Grids // Software – Practice and Experience (SPE). – 2004. – V. 34 – P. 631-651
4. W. Xiaohui , D. Zhaohui , Y. Shutao. CSF4: A WSRF Compliant Meta-Scheduler // In Proc. of World Congress in Computer Science Computer Engineering, and Applied Computing. – 2006. – P. 61-67.
5. R. Buyya , D. Abramson , J. Giddy. Nimrod/G: An architecture for a resource management and scheduling system in a global computational Grid // Proceedings of the 4th International Conference on High Performance Computing in Asia-Pacific Region. – 2000. – P. 283-289.

6. J. Frey, T. Tannenbaum, M. Livny, I. Foster, S. Tuecke. Condor-G: A Computation Management Agent for Multi-Institutional Grids // Cluster Computing. – 2001. – V. 5. – P. 237-246.
7. P. Andreetto, S. Borgia, A. Dorigo. Practical approaches to grid workload and resource management in the EGEE project // In CHEP ’04: Proceedings of the Conference on Computing in High Energy and Nuclear Physics. – 2004. – V. 2. – P. 899-902.
8. E. Caron, V. Garonne, A. Tsaregorodtsev. Evaluation of Meta-scheduler Architectures and Task Assignment Policies for High Throughput Computing // Technical report. Institut National de Recherche en Informatique et en Automatique. – 2005.
9. Пазников А.А. Средства децентрализованной диспетчеризации задач в распределенных вычислительных системах // Материалы XLVII Международной научной студенческой конференции «Студент и научно-технический прогресс». – Новосибирск: НГУ, 2009.– С. 210.