

УПРАВЛЕНИЕ ПАРАЛЛЕЛЬНЫМИ ЗАДАНИЯМИ В ГРИД С ПОМОЩЬЮ МЕТОДА ОПЕРЕЖАЮЩЕГО ПЛАНИРОВАНИЯ*

Коваленко В.Н., Семячкин Д.А.

Институт прикладной математики им. М.В.Келдыша РАН;
Россия, 125047, Москва, Миусская пл. 4; тел. (495)250-79-01,

kvn@keldysh.ru, dms@keldysh.ru

1. Введение. Основная сложность управления параллельными (многопроцессорными) заданиями в грид состоит в необходимости накопления и затем гарантированного синхронного выделения ресурсов в нескольких кластерах (коаллокация ресурсов): это предотвращает зависание заданий, которое является следствием фрагментации ресурсного пула. В настоящей работе рассматривается решение этой задачи с помощью метода опережающего планирования [1], разработанного в ИПМ им. М.В. Келдыша РАН, ранее примененного для управления однопроцессорными заданиями в грид [2]. Метод обладает свойствами, заключающимися в построении плана распределения ресурсов на некоторый период времени вперед и гарантии выполнения этого плана (детерминированность планирования). Это позволяет использовать его для обслуживания параллельных заданий в условиях совместного использования ресурсов с их владельцами за счет использования механизма предварительного резервирования.

В работе дается формальная постановка задачи коаллокации (п.3). В отличие от локальных систем планирование в грид должно учитывать не только уже выполняющиеся в кластерах задания, но и задания, находящиеся в локальных очередях. Поэтому формализация основана на представлении плана в виде множества временных слотов, где каждый слот соответствует началу и концу некоторого кластерного задания, а планирование заключается в подборе слотов, удовлетворяющих ресурсному запросу задания (с учетом их стоимости).

В этих условиях алгоритм обратного заполнения Backfill [3], применяемый в локальных системах для параллельных заданий, не дает линейных оценок сложности от количества слотов (п.4). Основным результатом работы — построение алгоритма с такой сложностью (п.5). Алгоритм основан на совместном использовании двух представлений плана, в одном из которых слоты упорядочены по возрастанию времени начала, а во втором — по времени конца. Рассмотрены также модификации алгоритма для обеспечения самого раннего завершения задания, учитывающие различную производительность процессоров, и проведено их сравнение с различными вариантами алгоритма Backfill.

2. Существующие методы управления заданиями. Согласно классификации, приведенной в работе [4], современные методы управления заданиями делятся на два

* Работа выполнена при поддержке фонда РФФИ (грант № 05-01-00626-а), программы Президента РФ для ведущих научных школ (проект 2006-РП-112.0/001/417) и программы фундаментальных исследований Президиума РАН «Разработка фундаментальных основ создания научной распределённой информационно-вычислительной среды на основе технологий GRID».

класса. К первому классу принадлежат методы, основанные на использовании очередей заданий (Queueing), традиционно применяющиеся в кластерных системах управления заданиями, таких как PBS/Torque [5], NQE/NQS, LoadLeveler [6] и другие. Примером системы управления заданиями в грид, использующей эти методы, является брокер ресурсов WMS [7], входящий в состав программного обеспечения gLite [8], которое используется в наиболее крупном на сегодняшний день инфраструктурном грид-проекте EGEE (Enabling Grids for E-sciencE) [9]. Основной принцип работы методов этого класса заключается в выделении ресурсов для стоящих в очереди заданий в порядке их приоритетов, исходя из текущего состояния ресурсов. Накопление ресурсов, необходимых для запуска параллельного задания, может быть осуществлено с помощью блокировки свободных ресурсов, что приводит к их простоям в течение неопределенного времени. Механизм предварительного резервирования трудно применить для обеспечения параллельному заданию гарантированного синхронного выделения ресурсов в нескольких кластерах из-за отсутствия информации о времени образования требуемого заданию объема ресурсов. Таким образом, использование методов, основанных на использовании очередей, в грид неэффективно, а основанным на них системам присущи довольно жесткие ограничения по применению: они не способны исключить такие нежелательные эффекты, как непредсказуемость времени обработки заданий, а также задержка обработки в ситуациях, когда имеются простаивающие ресурсы. Существенным недостатком этих систем является невозможность планирования параллельных заданий с синхронным выделением ресурсов в нескольких кластерах.

Второй класс опирается на методы, использующие для распределения ресурсов между заданиями полноценный план на будущее или расписание (Planning). С его помощью автоматически обеспечивается накопление ресурсов и естественным образом реализуется механизм предварительного резервирования ресурсов. Эти свойства позволяют использовать методы этого класса для обслуживания параллельных заданий в грид. В настоящее время существует небольшое количество систем, основанных на этих методах. К их числу относятся, например, планировщик Maui [10], применяющийся в локальных системах, или система Computing Center Software (CCS) [11], предназначенная для диспетчеризации заданий грид с одним уровнем управления — ресурсы отчуждаются от владельцев и целиком отдаются в грид, а все задания распределяются только с глобального уровня. Представляется, что такой подход не может претендовать на универсальность, однако может быть полезен в специальных условиях применения, например, когда владельцы сами не используют свои ресурсы, а выступают в роли профессиональных провайдеров.

К этому классу может быть отнесен и рассматриваемый в работе метод опережающего планирования, разработанный для актуальной формы организации ресурсов грид, предполагающей их использование совместно с владельцами, когда ресурсы не отчуждаются от владельцев, а задания поступают не только с глобального, но и с локального уровня. Ресурсы грид организованы в кластеры (кластеризованные ресурсы), а объектами планирования являются параллельные задания, ресурсы для которых могут выделяться в нескольких кластерах грид. Достоинство этого способа организации ресурсов состоит в том, что грид может образовываться динамически, на некоторое время, требуемое для решения конкретной задачи, а для его создания не

требуется формирования специальной ресурсной базы: достаточно лишь вовлечь уже имеющиеся ресурсы, не полностью загружаемые своими владельцами. Однако, с точки зрения планирования, неотчуждаемость ресурсов усложняет ситуацию, так как задания, поступающие с локального уровня, не могут контролироваться планировщиком, но должны учитываться при распределении заданий в грид.

3. Формальная постановка задачи коаллокации в грид с неотчуждаемыми ресурсами. Для формализации рассматриваемой задачи используется экономический подход, согласно которому процесс распределения ресурсов организуется на основе моделей рынка: владельцы ресурсов выступают в качестве продавцов, пользователи — в качестве покупателей этих ресурсов. Реализуя эту модель, предлагается использовать следующие соглашения по разделению ресурсов и способы их поддержки при планировании.

Первое соглашение касается разделения ресурсов между пользователями, основная идея которого заключается в том, что пользователь должен иметь возможность управлять скоростью получения ресурсов при планировании. Для этого им устанавливается плата за выполнение каждого задания в рамках выделенного ему бюджета. При увеличении платы за задание, шансы выполнить его быстрее повышаются, однако при этом ограничение бюджета пользователя не позволяет ему монополизировать весь ресурсный пул.

В условиях поступления двух потоков заданий — локального и глобального, планировщику необходимо каким-то образом разделять ресурсы между этими потоками. Для этого предлагается второе соглашение о разделении ресурсов с владельцами, позволяющее глобальному заданию занять ресурсы при условии, что назначенная пользователем плата за выполнение задания не меньше их стоимости. Это соглашение позволяет осуществлять динамическую балансировку потоков заданий, обеспечивая конкуренцию глобальных и локальных заданий в кластере, и является ключевым для грида с неотчуждаемыми ресурсами. Его принципиальной особенностью является то, что стоимость ресурсов меняется во времени и определяется не только их объемами, но и вычисляется с учетом приоритетов локальных заданий, которые могут занять те же ресурсы.

Наличие двух потоков заданий является важным отличием грида с неотчуждаемыми ресурсами от локальных систем и грида с отчуждаемыми ресурсами. План в этом случае содержит не только выполняющиеся в кластерах задания, но и задания, находящиеся в локальных очередях. Поэтому его можно представить в виде множества временных слотов вида: {процессорный узел, время начала, время конца, стоимость ресурса}, где каждый слот соответствует началу и концу некоторого задания или свободному участку плана. Изобразить план можно в виде временной развертки состояний ресурсов: для каждого ресурса на оси времени наносятся точки, соответствующие началу и концу слотов, которые представляются прямоугольниками со сторонами, равными времени занятия ресурса и его объему (рис. 1).

Согласно представленной экономической модели глобальные задания могут размещаться не только на свободных ресурсах, но и на ресурсах, на которые претендуют локальные задания, при условии, что плата за эти задания не ниже стоимости ресурсов. Таким образом, планирование параллельных заданий заключается

в подборе удовлетворяющих ресурсному запросу задания слотов и подходящих по стоимости, которые можно синхронно выделить в количестве, необходимом заданию.

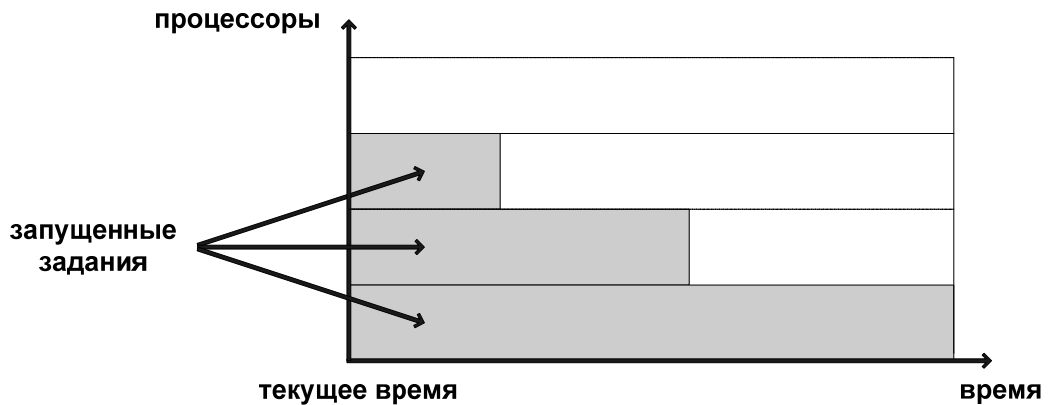


Рис. 1. Изображение плана распределения ресурсов на будущее, характерное для локальных систем.

4. Подбор слотов. Процесс подбора слотов для задания состоит в определении наилучшего (согласно некоторому критерию) времени, начиная с которого существуют слоты (или части слотов), удовлетворяющие ресурсному запросу задания подходящие по стоимости, длины не менее длины задания, в количестве, равном требуемому заданию количеству процессорных узлов. Рассмотрим алгоритмы, решающие эту задачу.

4.1. Алгоритм Backfill для локальных систем или грид с отчуждаемыми ресурсами. В кластерных системах, ближайшего аналога грид среди архитектур распределенного компьютеринга, для подбора слотов применяются различные варианты алгоритма обратного заполнения Backfill, изначально разработанного для больших многопроцессорных систем (MPP) типа IBM SP2. В качестве критерия выбора слотов в этих алгоритмах обычно выступает время старта задания: слоты подбираются так, чтобы запустить задания как можно раньше.

В работе [2] рассмотрен вариант этого алгоритма для случая, характерного для локальных систем, а также для грид с отчуждаемыми ресурсами, когда в начале очередного шага планирования расписание состоит только из запущенных заданий и не содержит запланированных заданий (резервирований), то есть находится в начальном состоянии. В терминах слотов это означает, что каждому процессорному узлу соответствует один единственный свободный слот, начало которого совпадает со временем ожидаемого завершения задания, выполняющегося на процессорном узле (рис. 1). Это ограничение позволяет определить в каждой точке, соответствующей началу слота, сколько процессорных узлов будет свободно, начиная с этого времени. Для этого производится перебор слотов, упорядоченных по возрастанию времени начала. Далее осуществляется подбор слотов для всех заданий из очереди уже с учетом этой информации.

Алгоритм позволяет подобрать слоты для одного задания с линейной сложностью от количества слотов, равного общему количеству заданий (запущенных и

запланированных). Это обеспечивается несколькими проходами по списку слотов, количество элементов в котором прямо пропорционально количеству заданий в системе.

4.2. Алгоритм Backfill для грид с неотчуждаемыми ресурсами. В случае с грид с неотчуждаемыми ресурсами, ситуация усложняется по двум причинам.

Во-первых, расписание в начале очередного шага планирования не находится в начальном состоянии — оно содержит как запущенные задания, так и задания, запланированные в локальных очередях. Представляя его в виде множества слотов, получаем, что каждому процессорному узлу может соответствовать несколько слотов, относящихся к разным промежуткам времени. Причиной этого является то, что теперь в расписании отражены не только ожидаемые времена освобождения запущенных заданий, но и времена начал и концов резервирований, сделанных кластерными планировщиками под стоящие в локальных очередях задания (рис. 2). Тогда при подборе слотов для задания необходимо в каждой точке на временной оси, соответствующей ожидаемому времени завершения задания, проверять, не запланирован ли на это время старт одного или нескольких заданий, находящихся в локальных очередях. В предположении, что на каждом процессорном узле запланировано как минимум одно локальное задание, сложность определения, сколько процессорных узлов и в какое время будет свободно, получается квадратичной от общего количества слотов.

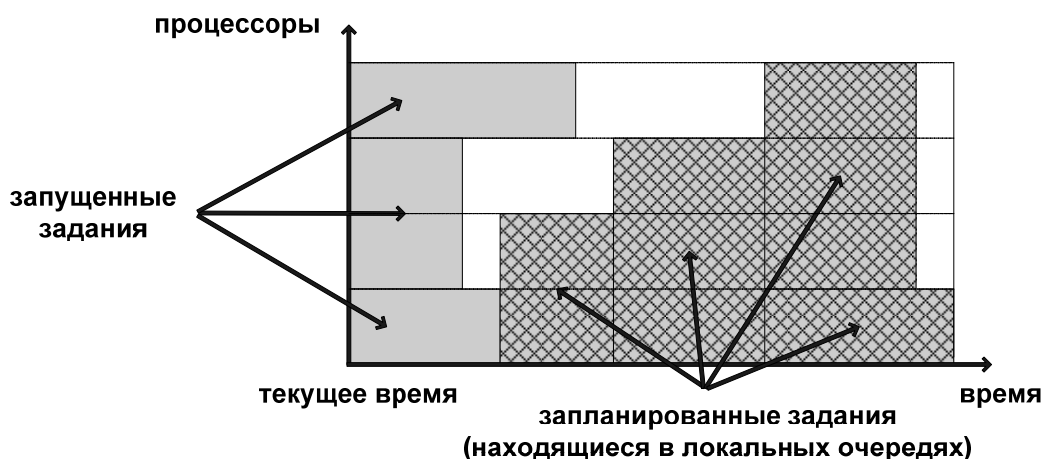


Рис. 2. Изображение плана распределения ресурсов на будущее, характерное для грид с неотчуждаемыми ресурсами.

Во-вторых, принимая во внимание соглашение о разделении ресурсов с владельцами, слоты, соответствующие резервированиям под локальные задания, могут быть использованы под размещение глобальных заданий, только в случае, если плата за эти задания окажется выше стоимости слотов. Ввиду того, что плата за разные задания может различаться, набор слотов, на которых они могут разместиться, также может быть свой для каждого задания. Следовательно, использовать информацию о свободных слотах для планирования всех заданий из очереди, не представляется

возможным — ее нужно получать для каждого задания. Учитывая, что сложность этой операции является квадратичной, получается, что рассматриваемая модификация алгоритма Backfill, в условиях грид с неотчуждаемыми ресурсами, работает с квадратичной сложностью от количества слотов. Кроме того, алгоритм Backfill, как было упомянуто выше, не учитывает различную производительность процессоров: слоты подбираются так, чтобы запустить задание как можно раньше. В условиях грид (при большом числе объектов — узлов и заданий) существенна разработка алгоритма подбора слотов минимальной сложности, кроме того, представляется более эффективным подбирать слоты так, чтобы задания завершались как можно раньше.

4.3. Подбор слотов для задания как оптимизационная задача. В работе [12] приведен алгоритм, основанный на экономических моделях, позволяющий подбирать слоты для заданий, исходя из критерия, определенного пользователем в виде целевой линейной функции. С помощью этой функции можно указать, например, что слоты необходимо подбирать так, чтобы задание завершилось как можно раньше. Кроме того, данный подход позволяет пользователю задавать сразу несколько интересующих его критериев, объединяя их в виде аддитивной функции с определяемыми весами. Задача подбора слотов в этом случае сводится к максимизации линейной функции на ограниченном множестве слотов. Недостаток этого метода заключается в том, что в общем случае используемая целевая функция не является монотонной, поэтому для определения ее максимума на множестве всех слотов алгоритм должен перебрать все возможные комбинации слотов требуемого заданию объема, количество которых может быть велико. Для сокращения времени работы этого алгоритма в работе предложено применять эвристические методы, однако в этом случае получаемое решение уже не является наилучшим.

5. Предлагаемый алгоритм подбора слотов для заданий в грид с неотчуждаемыми ресурсами. Для подбора слотов для параллельного задания нами разработан алгоритм линейной сложности от количества слотов, позволяющий заданиям завершаться как можно раньше. Алгоритм основан на совместном использовании двух представлений плана распределения ресурсов на некоторый период времени вперед в виде связанных списков, в одном из которых слоты упорядочены по возрастанию времени начала, а во втором — по возрастанию времени конца, и работает в два этапа. На первом этапе осуществляется последовательный проход по первому списку, в процессе которого подсчитывается количество слотов, которые могут быть использованы для запуска задания. Параллельно проходит второй список, движение по которому позволяет исключить слоты, время действия которых истекло. Как только набирается достаточное количество слотов для запуска задания, первый этап работы алгоритма завершается, и на выходе получается время, начиная с которого для задания можно синхронно выделить требуемое количество слотов. На втором этапе совершается еще один проход по первому списку с целью определения слотов для запуска задания в найденное время. После этого задание размещается на этих слотах, а сами слоты считаются использованными и удаляются из обоих списков. Таким образом, для подбора слотов для одного задания необходимо совершить два прохода по плану распределения ресурсов на первом этапе работы алгоритма и один проход на втором, следовательно, итоговая сложность разработанного алгоритма является линейной от общего количества слотов.

Для обеспечения самого раннего завершения задания разработана модификация этого алгоритма, учитывающая различную производительность процессорных узлов. Для этого вводится градация производительностей процессоров $G = \{P_1, P_2, \dots, P_K\}$, где $P_1 \geq P_2 \geq \dots \geq P_K$, где K — константа, равная, например, 10. Пользовательская оценка времени выполнения задания (T), сделанная для эталонного процессора ($P = 1$), пересчитывается для производительностей из G : $T^* = T/P_i$, $i = 1, \dots, K$. После этого алгоритм подбора слотов, используя пересчитанную оценку времени выполнения задания, выполняется для каждой группы процессоров с производительностью $P \geq P_i$, $i = 1, \dots, K$, таким образом, в первую очередь делается попытка подобрать слоты на более производительных процессорах. В результате модифицированный алгоритм также имеет линейную сложность.

6. Заключение. В работе представлен подход к решению задачи управления параллельными (многопроцессорными) заданиями в грид, предотвращающий их зависание.

В рамках этого подхода получены следующие результаты.

- Формализована задача коаллокации в грид с неотчуждаемыми ресурсами.
- Построен алгоритм линейной сложности для подбора ресурсов для запуска параллельного задания в грид с неотчуждаемыми ресурсами.
- Предложена модификация алгоритма линейной сложности, учитывающая различную производительность процессорных узлов.

СПИСОК ЛИТЕРАТУРЫ

- [1] Коваленко В.Н., Коваленко Е.И., Корягин Д.А., Любимский Э.З. (2005) Метод опережающего планирования для грид // Препринт № 112, Москва: ИПМ РАН, с. 1–33
- [2] Коваленко В.Н., Коваленко Е.И., Шорин О.Н. (2005) Разработка диспетчера заданий грид, основанного на опережающем планировании // Препринт № 133, Москва: ИПМ РАН, с. 1–28
- [3] D.G.Feitelson and A.M.Weil (1998) Utilization and Predictability in Scheduling the IBM SP2 with Backfilling // In Proceedings of IPPS/SPDP 1998, p. 542–546. IEEE Computer Society
- [4] Matthias Hovestadt, Odej Kao, Axel Keller, and Achim Streit. Scheduling in HPC Resource Management Systems: Queuing vs. Planning. <http://www.uni-paderborn.de/pc2/papers/files/409.pdf>
- [5] Portable Batch System. <http://www.openpbs.org>
- [6] LoadLeveler <http://www.mhpcc.edu/training/workshop/loadleveler/MAIN.html>
- [7] Workload Management System User and Reference Guide, <https://edms.cern.ch/file/572489/1/WMS-guide.pdf>
- [8] gLite — Lightweight Middleware for Grid Computing. <http://www.glite.org>
- [9] Проект Enabling Grids for E-Science (EGEE), <http://www.eu-egee.org>
- [10] Maui scheduler. <http://www.supercluster.org/maui>

- [11] A. Keller and A. Reinefeld. Anatomy of a Resource Management System for HPC Clusters. In Annual Review of Scalable Computing, vol. 3, Singapore University Press, pages 1–31, 2001
- [12] C. Ernemann, V. Hamscher, and R. Yahyapour. Economic scheduling in Grid computing. In D. Feitelson and L. Rudolph, editors, Job Scheduling Strategies for Parallel Processing (Proceedings of the Eighth International JSSPP Workshop; LNCS #2537), pages 129–152. Springer-Verlag, 2002