УДК 681.324

М.А. Волк, Т.В. Филимончук, Р.Н. Гридель

Харьковский национальный университет радиоэлектроники, Харьков

МЕТОДЫ РАСПРЕДЕЛЕНИЯ РЕСУРСОВ ДЛЯ GRID-СИСТЕМ

В работе представлен анализ методов распределения ресурсов, обсуждается возможность их использования в Grid-системах и приведена классификация с точки зрения областей применения методов для различных заданий и конфигураций вычислительных ресурсов.

Ключевые слова: *методы распределения, многокритериальная оптимизация, приоритет заданий, неотчуждаемые ресурсы.*

Анализ предметной области

В настоящее время создается инфраструктура, которая обеспечивает надежный, устойчивый и недорогой доступ к высокопроизводительным вычислительным ресурсам. Перед научными организациями (Украинская академия наук, министерство образования Украины) ставится задача объединения национальных, региональных и тематических Grid-разработок в единую Grid-инфраструктуру, которая использовалась бы для поддержки научных исследований. Одной из задач, которая находится в рамках этих работ, является задача распределения вычислительных ресурсов между заданиями, поступающими на выполнение.

Для эффективного использования выделенных ресурсов необходимо решать задачу планирования. В данном случае под планированием понимается распределение поступающих на вход заданий на имеющиеся ресурсы. Целью планирования является определение эффективности распределения заданий при использовании различных критериев. Процесс планирования осуществляется динамически (в процессе поступления заданий на вход) специальной программой, которая носит название планировщик (брокер) [1]. Планировщик управляет потоком заданий, поступающим от пользователей, и производит их распределение на имеющиеся ресурсы с помощью информации, которую предоставляет пользователь (описание требований для корректного выполнения заданий – язык JDL, JSDL, TSDL, ClassAds). Планировщик должен удовлетворить все требования пользователей и оптимизировать общее время выполнения задания, а также стоимость используемых ресурсов.

Любое задание, поступающее в Grid систему на выполнение, описывается с помощью специального языка. Для планировщика важными параметрами задания являются те, которые определяют требования к ресурсам. Возникает вопрос, какой из параметров является главным, т.е. по какому параметру осуществлять оптимизацию. В настоящее время существует ряд задач, которые необходимо анализировать не только по одному параметру, что в свою очередь накладывает трудности для процесса распределения. В таких случаях рационально использовать методы решения многокритериальных задач оптимизации. Применение этих методов подразумевает объединение ряда критериев в единый критерий. Объединение критериев, как правило, производится эвристическими методами, поэтому с математической точки зрения не существует идеального решения таких задач, каждый из них имеет преимущества и недостатки. Чаще всего для решения многокритериальных задач оптимизации используют обобщенный (интегральный) критерий. Его суть заключается в том, что частные критерии $F_i(X), i = \overline{1,n}$ каким-либо образом объединяются в один интегральный критерий $F(X) = \Phi(F_1(X), F_2(X), ..., F_n(X))$, а затем находится максимум или минимум данного критерия. В зависимости от того, каким образом частные критерии объединяются в обобщенный критерий, различают следующие виды критериев: аддитивный, мультипликативный, максиминный (минимаксный).

Частные критерии часто имеют различную физическую природу и разную размерность, следовательно, суммировать их некорректно. Поэтому числовые значения частных критериев делятся на некоторые нормирующие делители. Размерности самих частных критериев и соответствующих нормирующих делителей одинаковы, следовательно, обобщенный аддитивный критерий получается безразмерной величиной.

Преимуществом такого подхода является то, что всегда удается определить единственный оптимальный вариант решения. Однако присутствуют и недостатки:

– трудности (субъективизм) в определении весовых коэффициентов;

– аддитивный критерий не вытекает из объектной роли частных критериев и поэтому выступает как формальный математический прием;

– в аддитивном критерии происходит взаимная компенсация частных критериев, т.е. уменьшение

одного из них может быть компенсировано увеличением другого критерия.

После того как получен аддитивный коэффициент, для распределения заданий можно использовать методы, которые позволят выбрать наилучший вариант ресурса для запуска задания. Рассмотрим следующие методы планирования: FCFS, HPF, RR, EDF, FB, Backfill, а также метод критического пути и обоснуем рациональность использования их в Grid системах.

Методы планирования

Очень часто при осуществлении планирования у пользователя имеются списки заданий, которые упорядочены по убыванию приоритетов. Списки заданий могут быть представлены либо в статическом виде (составлены заранее), либо в динамическом виде (списки обновляются по мере исполнения заданий, прихода новых). Поэтому построение расписаний на основе списка заданий в данном случае является актуальным.

Метод критического пути является одним из самых популярных. Его работа заключается в последовательном распределении заданий по имеющимся ресурсам. Ресурс для выполнения задания выбирается по заданному критерию. Данный метод позволяет построить расписание и оценить временные затраты, которые возникнут при распределении задач на конкретный ресурс. Данный метод можно использовать в распределенных системах только с точки зрения заданий, при условии, что известно время выполнения каждой задачи из задания. С его помощью можно осуществлять предварительный расчет времени окончания задания, что позволяет принять решение пользователю о дальнейших действиях (например, запустить задание на выполнение в данной конфигурации или изменить ряд параметров ресурсов для уменьшения времени выполнения задания). В связи с тем, что пользователь не всегда может оценить время выполнения задач из задания, использование этого метода в Grid-системах ограничено. В случае, если эта информация известна (например, в случае повторяющихся экспериментов), применение этого метода дает хороший результат.

Простейшим методом планирования является метод First-Come First-Served (FCFS), планирование заданий выполнятся в той последовательности, в которой они поступили. Преимуществом данного метода является простота его реализации, однако он имеет и ряд недостатков. Среднее время ожидания и среднее время выполнения задания зависят от порядка расположения заданий в очереди [2]. Если в начале очереди находятся задания, которые требуют для выполнения большого количества ресурсов (а они в это время отсутствует), то может возникнуть ситуация, при которой остальные задания, требующие меньший объем ресурсов будут простаивать, даже при условии, что для их выполнения имеется заданный объем ресурсов.

При методе планирования Highest Priority First (HPF) ресурс предоставляется на выполнение тому заданию, которое имеет наивысший приоритет. При использовании данного метода может возникнуть ситуация, когда высокоприоритетные задания могут захватить ресурсы на длительное время, вследствие чего задания с низким приоритетом могут не получить ресурсы, т.е. каждый раз когда возникнет ситуация и требуемый ресурс освобождается – приходит задание с более высоким приоритетом и забирает данный ресурс. Также может возникнуть ситуация, когда для выполнения высокоприоритетного задания не имеется требуемого ресурса, оно стоит в очереди и ожидает освобождение ресурса, однако для менее приоритетного задания ресурс имеется, но оно не может занять его из-за того, что впереди него стоит задание с более высоким приоритетом. Следовательно возникает ситуация нерационального использования ресурсов. Также следует не забывать, что при использовании метода планирования HPF необходимо задать параметры, которые лягут в основу формирования приоритета. Иногда имеет смысл выбирать первым самое короткое задание SJF (shortest job first) [2].

Классический метод обратного заполнения BackFill широко применяется на практике для больших многопроцессорных систем. Изначально данный метод был разработан для массивно-параллельных систем типа IBM SP2. Идея метода заключается в следующем: ресурсы выделяются заданиям не непосредственно в момент освобождения, а заблаговременно. Планировщик строит план распределения ресурсов, т.е. строится расписание запусков заданий. Преимуществом метода является то, что он гарантирует получение ресурсов высокоприоритетными заданиями в минимально возможное время и в то же время допускает нарушение порядка очереди, позволяя низкоприоритетным заданиям занимать оставшиеся неиспользованными ресурсы, что способствует повышению коэффициента общей загрузки ресурсов. Метод BackFill позволяет реализовать механизм предварительного резервирования ресурсов.

Существует две модификации метода BackFill: агрессивное и консервативное резервирование. При использовании консервативного варианта ресурсы резервируются для всех заданий из очереди, однако они используются менее эффективно, чем при агрессивном резервировании. Это связано с тем, что при агрессивном резервировании ресурсы резервируются только под самые приоритетные задания из очереди. В случае невозможности немедленного запуска задания метод зарезервирует требуемый

ресурс, следовательно, задания из очереди не смогут отодвинуть процесс запуска самого приоритетного задания, т.к. они либо закончатся, либо будут использовать другие ресурсы. Недостатком данной модификации является то, что могут быть отодвинуты задания, которые не участвовали в процессе резервирования. Согласно результатам моделирования на стандартных трассах [3] существенных задержек запуска заданий в агрессивном варианте метода не происходит. Поэтому на практике консервативный вариант метода обратного заполнения используется реже агрессивного.

Точные методы распределения

Существует ряд математических методов, которые возможно использовать для распределения ресурсов. Можно классифицировать такие методы по способу предоставления решения: точные и приближенные. Ряд точных методов представляют переборные методы, метод ветвей и границ, методы динамического программирования, метод множителей Лагранжа, методы предварительного расширения и последовательного сужения, а также методы линейного программирования. В процессе анализа был сделан вывод, что некоторые из них нельзя использовать для работы в Grid-системах. Это связано с тем, что для своей работы такие методы требуют конкретные значения параметров, которые получить на определенном этапе невозможно. Остановимся более подробно на методах, отобранных в качестве претендентов на использование в Grid.

При выборе переборных методов множество решений, как правило, конечно [3]. Поэтому, при полном переборе всех возможных решений задачи можно найти наилучшее из решений. Недостатком данного метода является огромное число рассматриваемых вариантов. Этот метод очень прост и эффективен для задач с малым числом входных данных (или когда имеется возможность свести множество входных параметров к одному).

Если допустимое множество решений конкретной задачи является конечным, то с помощью метода полного перебора можно прийти к оптимальному решению. При увеличении числа параметров задачи увеличивается число допустимых решений и увеличивается время поиска оптимального решения, и в данном случае метод перебора не эффективен. Ускорить процесс поиска можно за счет отбрасывания подмножества заведомо неоптимальных решений. Эта модернизация нашла отражение в методе ветвей и границ [5]. Так как данный метод является разновидностью метода полного перебора, то ему присущи все его недостатки. В связи с тем, что при использовании данного метода имеется возможность отбрасывать заведомо ложные решения, сокращается время его работы. Как и метод полного перебора метод ветвей и границ является конечным методом (результат будет получен обязательно).

Динамическое программирование – это ряд методов, основанных на пошаговом решении задачи, когда на каждом шаге из множества допустимых решений выбирается одно, которое оптимизирует целевую функцию [4]. Метод динамического программирования рационально использовать в том случае, если задача может быть разбита на одинаковые подзадачи. Каждая подзадача решается только один раз, после чего ее решение заносится в таблицу. Использование таблиц позволяет сэкономить время расчета и вычислительные мощности, т.е. не происходит решение одних и тех же задач многократно. В основе динамического программирования лежит рекуррентное соотношение, которое связывает между собой оптимальные значения целевых функций подзадач. Преимуществами данного метода является то, что он позволяет найти глобальное оптимальное решение, а уравнение Беллмана удобно для программирования. Ограничением же данного метода является размерность задачи, это связано с тем, что приходится хранить результаты оптимизации всех этапов.

Линейное программирование [6] – это ряд методов, ориентируемых на решение задач об экстремумах линейных функций на множествах n-мерного векторного пространства, задаваемых системами линейных уравнений и неравенств.

При решении задачи распределения поступивших заданий на имеющиеся ресурсы накладывается обязательное ограничение: решение должно быть целочисленным. Также можно предположить, что в случае выбора ресурса может быть только два решения – ресурс занят (1) или ресурс свободен (0). Задачи целочисленного программирования, в которых допустимы значения 0 или 1, называются задачами целочисленного программирования с булевыми переменными. Эффективным для решения таких задач является алгоритм на основе ранговый подхода [7]. Ранговый подход основан на разбиении n-мерного единичного куба на n рангов, каждый вектор которого имеет одинаковое число единиц и нулей. К достоинствам такого метода можно отнести простоту и наглядность представления решения задачи, его полиномиальную сложность, возможность реализации на параллельных вычислительных структурах.

Симплекс-метод – это один из инструментов для решения задачи линейного программирования. Недостатком этого метода является экспоненциальная сложность, но он позволяет за конечное число итераций найти оптимальное решение. В качестве ограничений могут быть использованы как равенства, так и неравенства. Вследствие того, что в рассматриваемом нами случае используются неравен-

ства типа ≤ 1, то данный метод является эффективным, т.к. в решение всегда является целочисленным.

Приближенные методы распределения

В качестве приближенных методов распределения можно выделить эвристические, итерационные и стохастические. Эвристические методы представлены жадными алгоритмами, нейронными сетями и эволюционными вычислениями.

Жадный алгоритм [4] – это простой и быстрый алгоритм, который на каждом шаге делает локально оптимальный выбор и в дальнейшем этот выбор не отменяется. Если глобальная оптимальность алгоритма имеет место практически всегда, его обычно предпочитают другим методам оптимизации, таким как динамическое программирование. Жадные алгоритмы целесообразно использовать на начальном этапе, т.е. при формировании первичного плана. Этот тип алгоритмов подходит для решения широкого класса задач. Одним из свойств жадного алгоритма является то, что с его помощью удается найти оптимальное решение, однако это бывает не всегда.

Эволюционные алгоритмы [8] – это эвристические алгоритмы поиска, которые используются для решения задач оптимизации путем последовательного варьирования искомых параметров. Они основаны на принципах естественного отбора и является разновидностью эволюционных вычислений. В настоящее время существует несколько типов эволюционных алгоритмов: алгоритмы эволюционных стратегий, алгоритмы эволюционного программирования, генетические алгоритмы, алгоритмы генетического программирования.

С помощью эволюционных алгоритмов не всегда можно найти оптимальное решение, т.е. построить план распределения заданий, однако имеется вероятность того, что предложенный алгоритм найдет достаточно хорошее решение. Использование этого класса алгоритмов целесообразно в том случае, когда способ точного решения задачи не известен, либо когда способ для точного решения существует, однако он очень сложен в реализации и требует больших затрат времени, денег, больших вычислительных мощностей. Данный класс алгоритмов легко реализуются и способен решать широкий круг задач, т.к. в эволюционных алгоритмах используется аналогия с развитием живых организмов (от простых форм к более сложным), то они приблизились к тому, чтобы стать общим методом решения задач оптимизации.

В настоящее время широкое применение получили нейронные сети, которые успешно применяются в самых различных областях: медицине, бизнесе, физике и т.д. Они используются там, где возникает необходимость решать задачи управления, классификации, так как позволяют воспроизводить чрезвычайно сложные зависимости. Нейронные сети нелинейны, их можно использовать в тех задачах, где линейная аппроксимация неудовлетворительна и имеющиеся линейные модели работают плохо. Задачи линейного программирования очень часто сталкиваются с "проклятием размерности", которое не позволяет моделировать линейные зависимости в случае большого числа переменных. Для нейронных сетей это не проблема. Также нейронные сети являются обучаемыми. С помощью пользователя для нейронной сети подбираются данные, а далее запускается алгоритм обучения [9].

Использование нейронных сетей в Grid-системах при распределении заданий по имеющимся ресурсам имеет положительные стороны. Так как они являются обучаемыми, то в ходе их работы накапливаются статистические данные, которые в дальнейшем могут быть использованы для распределения. Нет необходимости каждый раз при поступлении задания заново осуществлять процесс прогнозирования и тратить дополнительное время на получение промежуточных данных – нужно просто взять имеющиеся. После обучения сеть способна предсказать будущее значение некой последовательности на основе нескольких предыдущих значений и/или каких-то существующих в настоящий момент факторов. Достоинством использования нейронных сетей является высокая скорость и точность получения результата, однако присутствует и недостаток – это сложность анализа, программной и аппаратной реализации.

Выводы

Проведенный анализ позволяет сделать вывод о том, что для распределения поступивших заданий на имеющиеся вычислительные ресурсы, в Grid-системах можно использовать ряд методов, закрепив каждый из них за отдельным участком системы.

Планировщик выбирает задание для распределения из очереди заданий (каждое задание содержит в себе N задач). На данном этапе возможен один из двух вариантов: выбор задания по приоритету (HFP), либо выбор задания по мере его поступления (FCFS). Далее выбранное задание рассматривается с точки зрения требований, заложенных пользователем (формируется область допустимых решений выбранного задания, содержащая K ресурсов, т.е. производится отсечение ресурсов заведомо непригодных для решения).

Планировщик осуществляет распределения задач из задания по выбранным ресурсам, ориентируясь на высчитанные ранее аддитивные коэффициенты ресурсов (сведение многопараметрической задачи к задаче с одним параметром). При распределении может возникнуть следующие ситуации: K=N – все задачи получают в свое пользование ресурс и

планировщик выбирает следующие задание из очереди, K<N – все ресурсы занимаются задачами, оставшиеся задачи ставятся в очередь и ожидают освобождения ресурса (при условии, что задачи не связаны между собой), K>N – ресурсов больше, чем задач, перед планировщиком стоит дилемма о выборе ресурсов, которые обеспечат выполнение задачи за более короткое время. Выбор «наилучшего ресурса» осуществляется при решении задачи о назначении, которая относится к классу задач линейного программирования. После решения задачи о назначении за ресурсами будет закреплена задача на определенное время, требуемое для ее выполнения.

Увеличить загруженность системы можно, используя жадные алгоритмы. Для каждого ресурса в определенный момент времени появляется очередь заданий. Иногда имеется смысл просчитать варианты выполнения заданий не в той последовательности, в которой они поступили в очередь, а с точки зрения более рационального использования ресурса. В данном случае рационально воспользоваться жадными алгоритмами, методом Backfill, использовать эволюционное вычисление или нейронные сети.

Часто используются Grid-системы с неотчуждаемыми ресурсами, и в ходе работы возникает ситуация о переделе очереди заданий вследствие того, что поставщик ресурсов использует данный ресурс для решения своей локальной задачи. В данном случае планировщик заново распределяет задачи, однако общее время выполнения задания изменяется (время рассчитывается с помощью метода критического пути) и может возникнуть ситуация, превышения лимита времени выполнения задания. Чтобы избежать таких ситуаций рационально использование перераспределения очередности выполнения задач.

В настоящее время не существует универсального по эффективности метода, который мог бы осуществить распределение для любого класса задач и совокупности доступных ресурсов, и возникает необходимость учитывать свойства конкретного класса задач, для которых данный метод разрабатывается. Задача планирования относится к классу NP-полных задач [4] и для ее решения не существует полиноминальных алгоритмов. Однако существование таких задач заставляет искать пути, с помощью которых возможно преодолеть данные сложности. Можно выделить следующие направления: нахождение решений с помощью эвристических методов, улучшение переборных методов за счет введения ряда ограничений или условий, динамическое программирование, которое использует для нахождения решения таблицы, сформированные ранее (размер таблиц находится в экспоненциальной зависимости от размерности задачи), сбор статистических данных о выполненных заданиях (можно использовать для обучения нейронной сети).

Список литературы

1. Аветисян А.И. Эвристики распределения задач для брокера ресурсов Grid / А.И. Аветисян, С.С. Гайсарян, Д.А. Грушин, Н.Н. Кузюрин, А.В. Шокуров // Труды Института системного программирования. – 2004. – Т. 5. – С. 269-280.

2. Коваленко В.Н. Управление параллельными заданиями в гриде с неотчуждаемыми ресурсами / В.Н. Коваленко, Е.И. Коваленко, Д.А. Корягин, Д.А Семячкин // Препринт №63. – М.: ИПМ РАН. – 2007. – С. 1-28.

3. Mu'alem A.W. and Feitelson D. G.. Utilization, predictability, workloads, and user runtime estimates in scheduling the IBM SP2 with backfilling. IEEE Trans. Parallel & Distributed Syst. 12(6), 2001. – Pp. 529-543.

4. Кормен Т. Алгоритмы: построение и анализ / Т. Кормен, Ч. Лейзерсон, Р. Ривест. – М.: МЦНМО, 1999. – 960 с., 263 ил.

5. Nemhauster G.L. Branch-and-bound and Parallel Computation: a Historical Note / G.L. Nemhauster, E.A. Pruul, R.A. Rushmeier // Oper. Res. Let., 7, 65-69, 1988.

6. Хемди А. Таха Введение в исследование операций, 7-е издание.: Пер. с англ. – М.: Издательский дом «Вильямс», 2005. – 912 с.: ил.

7. Пономаренко В.С. Методы и модели планирования ресурсов в GRID-системах. Монография / В.С. Пономаренко, С.В. Листровой, С.В. Минухин, С.В. Знахур. – Х.: ВД „ІНЖЕК", 2008. – 408 с.

8. Michalewicz Z. Genetic Algorithms + Data Structures = Evolution Programs (3rd Edn.) // New York, Springer-Verlag, 1996.

9. Назаров А.В. Нейросетевые алгоритмы прогнозирования и оптимизации систем / А.В. Назаров, А.И. Лоскутов. – Наука и техника, 2003. – 384 с.

Поступила в редколлегию 9.02.2009

Рецензент: д-р техн. наук, проф. С.Г. Удовенко, Харьковский национальный университет радиоэлектроники, Харьков.

МЕТОДИ РОЗПОДІЛЕННЯ РЕСУРСІВ ДЛЯ GRID-СИСТЕМ

М.О. Волк, Т.В. Філімончук, Р.М. Грідель

У роботі розглянуто аналіз методів розподілу ресурсів, обговорюється можливість їх використання в Grid-системах і приведена класифікація з точки зору сфер застосування методів для різноманітних завдань і конфігурацій обчислювальних ресурсів.

Ключові слова: *методи розподілу, багатокритерійна оптимізація, пріоритет завдань, невідчужувані ресурси.*

METHODS OF ALLOCATION OF RESOURCES FOR THE GRID-SYSTEMS

M.A. Volk, T.V. Filimonchuk, R.N. Gredel

The analysis of resources allocation methods is presented, discuss of their use into the Grid-systems, classification is resulted from the view point of methods application field for different tasks and calculable resources configurations.

Keywords: *distributing methods, multicriterion optimization, priority of tasks, inalienable resources.*