МОДЕЛЬ ВЫЧИСЛИТЕЛЬНЫХ РЕСУРСОВ С НЕОДНОРОДНОЙ КОММУТАЦИЕЙ

Е.А. Гнедов

КГТУ, Красноярск

Введение

Использование параллельных вычислительных систем дает неоценимые преимущества при решении больших вычислительных задач, но при этом возникает целый ряд трудностей, связанных, с одной стороны, с разработкой параллельных алгоритмов, а с другой – с управлением ресурсами параллельной вычислительной системы (ПВС).

Следует заметить, что общий характер проблем, сопровождающих развитие параллельных вычислений, в целом выглядит таким же, каким он был и во времена последовательных. Только для параллельных вычислений они проявляются в более острой форме во многом из-за большей сложности самой предметной области.

Ситуация обостряется наличием многообразия в различных параллельных архитектурах, что в свою очередь породило множество методов написания параллельных программ. Во многом это было обусловлено тем, что языки программирования непосредственно разрабатывались под особенности ПВС. Такое программирование, несмотря на эффективное решение задач в рамках конкретной архитектуры, привело к проблемам при переносе программного обеспечения. Однако появление универсальных языков параллельного программирования тоже мало способствовало улучшению ситуации в сфере как разработки алгоритмов, так и управления ресурсами ПВС, так как универсальность в данном случае оказалась весьма многогранной.

Исходя из того, что в настоящее время наблюдается постоянное повышение мощности параллельных вычислительных систем, существуют ПВС, состоящие из десятков тысяч процессоров, можно сделать выводы о том, что актуальной задачей является разработка новых методов управления вычислительными ресурсами и данными этих ПВС.

Используемые подходы, в основном, ориентированы на статическое распределение ресурсов ПВС, что при повышении размерности решаемой задачи ведет к несбалансированности вычислений и повышению нагрузки на отдельные элементы ПВС. Положение усугубляется тем, что разрабатываемые вычислительные системы используют различные внутренние методы обмена данными между параллельными процессами [1].

На данный момент можно выделить два основных направления в управлении ресурсами параллельных вычислительных систем: статическое

управление и динамическое управление ресурсами. В основе статического распределения ресурсов лежит модель, реализующая выполнение параллельных процессов на заранее выделенных ресурсах ПВС, набор которых в течение времени выполнения программы остается постоянным. При этом наблюдаются следующие основные негативные факторы: во-первых, ограниченная масштабируемость программы, а именно, при изменении характеристик ПВС в сторону увеличения или уменьшения доступных ресурсов приходится с большой долей вероятности изменять программу. Во-вторых, управление ресурсами ПВС находится в руках разработчика и именно от него в большей степени зависит, будет ли эффективно выполняться программа на доступных ресурсах ПВС или большую часть времени процессы будут простаивать в ожидании данных.

Более гибкое управление ресурсами параллельных вычислительных систем было бы возможно при использовании динамического управления ими. Однако в настоящее время эти методы не обеспечивают эффективного управления (загрузки и балансировки нагрузки). Следует также отметить отсутствие эффективных и обобщенных моделей динамического распределения ресурсов, что во многом объясняется разнообразием механизмов порождения вычислительных процессов и методов обмена данными между ними.

Методы реализации параллелизма

Современные параллельные вычислительные системы являются симбиозом различных комбинаций параллельных архитектур, используемых при взаимодействии процессов. Множество таких архитектур можно разделить на два класса: параллельные архитектуры и псевдопараллельные архитектуры. Последний вид на уровне процессора использует последовательное выполнение, а пользователю предоставляется параллельная виртуальная архитектура.

Рассмотрим методы управления параллельными процессами и технологию обмена данными в этих классах архитектур:

1. Параллельные вычислительные архитектуры:

Конвейерные процессоры

Специализированные процессоры, выполняющие последовательность определенных действий над множеством данных. Со стороны процессора данные исполняются параллельно, но со стороны пользователя происходит последовательное выполнение программы.

Многоядерность (CoreMultiProcessor – CMP-процессор)

Принцип многоядерной архитектуры – размещение двух или более «исполняющих» ядер в одном процессоре. CMP-процессор вставляется в один процессорный разъем, но операционная система компьютера воспринимает каждое из его исполняющих ядер как отдельный логический про-

цессор, обладающий всеми соответствующими исполняющими ресурсами. Разделяя вычислительную работу, выполняемую в традиционных микропроцессорах одним ядром, между несколькими исполнительными ядрами, многоядерный процессор может выполнять больше работы в единицу времени.

Кластеры

Независимые вычислительные процессоры, имеющие свою память, объединенные с помощью достаточно медленных, по сравнению с обменом через общую память, коммутационных каналов (обычно – Ethernet).

2. Псевдопараллельные вычислительные архитектуры:

Мультипрограммный режим

Поддерживаемый операционной системой механизм управления разделением доступа выполняющихся процессов к процессорному времени путем переключения их контекста.

Многопоточность (Multithreads)

Более легкий вариант мультипрограммного режима. Части кода процесса могут выполняться как отдельные процессы в рамках контекста основного процесса. Доступ к процессорному времени так же, как и в мультипрограммном режиме, квантуется.

Взаимодействие процессов в различных архитектурах может происходить с помощью различных методов: последовательная передача данных (конвейерная обработка данных), взаимодействие через общую память (мультипрограммный или многоядерный режимы) или обмен сообщениями (кластеры) и т.д.

Различные методы управления процессами, а также взаимодействия между процессами в различных параллельных архитектурах и тем более совместное их использование в вычислительных системах приводят либо к созданию управляющих механизмов для какого-то одного вида архитектуры, либо к разработке и реализации таких методов управления, которые бы выбирали наилучший вариант управления ресурсами из возможно допустимых, и решали такие актуальные задачи, как платформонезависимость, масштабируемость, балансировка нагрузки, анализ процесса выполнения и т.д., зачастую возникающие и являющиеся препятствиями, усложняющими разработку параллельных программ при использовании методов статического управления ресурсами параллельных вычислительных систем.

В данной работе делается попытка унифицировать и представить механизмы обмена данными между процессами с единой точки зрения, а именно предложить модель управления ресурсами ПВС, в основании которой лежат два предположения: первое – все узлы однородны, второе – узлы, связанные коммутационными каналами различной пропускной способности.

Описание модели

Для того чтобы сделать структуру параллельной вычислительной системы более прозрачной для разработчика, т.е. предложить взаимодействие не с физической ПВС, которая может состоять из различных параллельных архитектур и иметь множество методов для обмена данными между процессами, тем самым, сводя разработку к решению сразу двух задач, а именно: к разработке алгоритма и к разработке стратегии выделения необходимых ресурсов ПВС, а предложить взаимодействие с обобщенной или унифицированной моделью параллельной вычислительной системы, которая, с одной стороны, позволяла бы разрабатывать платформонезависимые программы и, с другой стороны, являлась бы эффективным механизмом по управлению и распределению ресурсов ПВС и поддержанию различных параллельных архитектур и методов обмена данными.

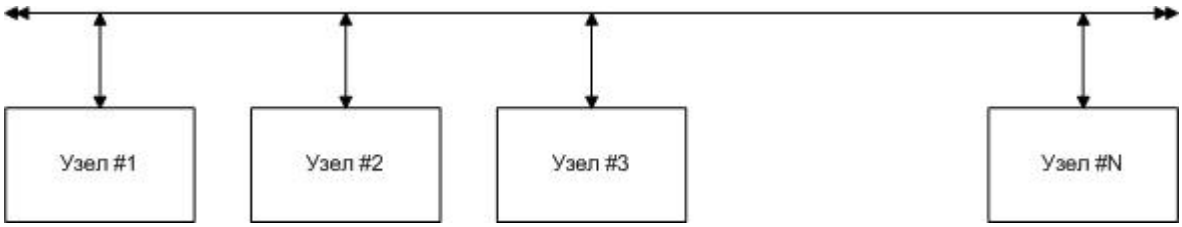

Рис. 1. Обобщенная модель ПВС с однородными узлами

Обобщенная модель параллельной вычислительной системы (рис. 1), которая в нашем случае базируется на однородной ПВС (например, кластер, который с большой долей вероятности будет однородным), основывается на предположении о том, что процессорные узлы являются однородными и взаимодействуют между собой по неоднородным коммутационным каналам.

Каждый вид каналов ассоциируется с тем или иным методом физического обмена данными. К различным видам неоднородных каналов относятся следующие методы взаимодействия процессов: последовательное взаимодействие между вызывающей и вызываемой процедурой, взаимодействие между независимыми потоками в рамках мультипрограммной среды, взаимодействие между параллельно выполняемыми процессами через общую память и т.д.

В конечном итоге обобщенная параллельная вычислительная система с обозначенными коммутационными каналами представлена на рис. 2, где физический узел №1 может, например, поддерживать параллелизм на уровне ядра процессора, следовательно, его можно представить как два ресурса, которые имеют более производительный коммутационный канал

для обмена данными, т.е. для программы это будут разнозначные узлы, но доступ к ним осуществляется по более быстрому каналу передачи данных.

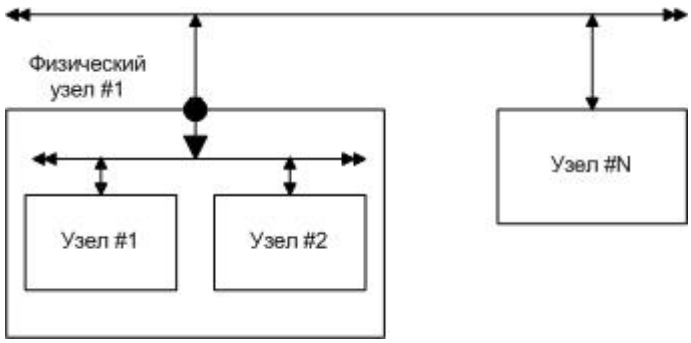

Рис. 2. Модель ПВС с неоднородными
коммутационными каналами

Выбор той или иной стратегии управления ресурсами ПВС зависит от нескольких факторов:

1) пропускной способности коммутационных каналов;
2) доступных ресурсов ПВС;
3) характеристики параллельного алгоритма.

Каждый вид коммутационных каналов вносит свои задержки в процесс обмена данными между процессами, эти задержки можно классифицировать следующим образом:

1. Задержки на узле параллельной вычислительной системы.
2. Задержки при взаимодействии между узлами (в контексте обобщенной модели, т.е. взаимодействие не только между процессорными узлами, а обмен данными по любым коммутационным каналам) ПВС.

К задержкам первого вида относятся следующие: выполнение кода функции, порождение контекста процесса, загрузка данных для обработки, ожидание момента запуска.

Задержки взаимодействия между узлами в основном связаны с физической пропускной способностью каналов, если речь идет об обмене с помощью Ethernet или внутренними задержками на процессорном узле, которые связаны, например, с синхронизацией доступа к общей памяти или ожидания переключения контекста процесса в мультипрограммной среде.

В дополнение к этому следует также отметить наличие задержки на принятие решения управляющим процессом о запуске того или иного процесса и миграции данных с одного узла на другой.

Процесс выполнения параллельной программы разбивается на несколько этапов, а именно: первый этап – оценка ресурсов ПВС и оценка

информационного графа на предмет определения степени используемого параллелизма, второй – принятие решения об использовании той или иной стратегии управления ресурсами ПВС и третий – непосредственный запуск программы, в ходе которого происходит выполнение функций, а также принятие решения о следовании выбранной стратегии, либо принятия иной, в случае, если состояние системы изменилось настолько, что далее невозможно эффективное исполнение программы.

В момент запуска программы каждый узел параллельной вычислительной системы содержит в себе каждую функцию и все начальные данные, и становится понятно, что стратегия управления – это есть не что иное, как метод обхода информационного графа и принятие решения о том, на каком узле будут выполняться те или иные функции и необходимо ли для этого передавать какие-нибудь данные.

Из всего вышесказанного мы приходим к выводу, что использование обобщенного представления различных физических методов обмена данными посредством одинаково устроенных виртуальных каналов с различными временными параметрами, вносящими задержки при обмене данными, позволяет обобщить процесс анализа методов динамического выделения вычислительных ресурсов для программ, не учитывающих на этапе разработки методов выделения вычислительных ресурсов.

Заключение

Предложенная модель в полной мере описывает взаимодействие виртуальной параллельной вычислительной системы, реализующей модель максимального параллелизма и реальной ПВС с ограниченными ресурсами.

В результате мы получаем новый подход в разработке параллельных программ – написание алгоритмов в машинно-независимой форме с последующей их адаптацией к конкретной вычислительной среде путем сжатия максимального параллелизма с учетом налагаемых ресурсных и функциональных ограничений.

Учитывая все вышесказанное, мы можем проводить анализ алгоритмов с точки зрения эффективности их выполнения на параллельных вычислительных системах, а также проводить исследования в сфере управления ресурсами ПВС в различных ситуациях, вызываемых постоянными изменениями в состоянии ресурсов вычислительной системы.

Данный подход предполагается использовать для динамического распределения ресурсов в программах, написанных на функционально-потоковом языке параллельного программирования Пифагор [2], оценки эффективности моделей управления распределением вычислительных ресурсов параллельных вычислительных систем, а также создания имитационного комплекса для проведения исследований взаимодействия парал-

лельных процессов, исполняемых в условиях ограниченных ресурсов различных по архитектуре параллельных вычислительных систем.

Литература

1. Воеводин В. В. Математические проблемы параллельных вычислений // Сб. тр. Всероссийской конф. «Научный сервис в сети Интернет: технологии распределенных вычислений». Новороссийск, 19-24 сентября 2005 г. – М.: Изд-во МГУ, 2005. – С. 3−8.
2. Легалов А. И., Казаков Ф. А., Кузьмин Д. А., Привалихин Д. В. Функциональная модель параллельных вычислений и язык программирования «Пифагор». http://softcraft.ru/parallel/fpp/fppcontent.shtml