Раздел 5 | Автоматика. Энергетика. Управление

УДК 621.3.083

Кластерные системы для организации параллельных вычислений

Т.Л. ТЕН, *д.т.н., профессор,*
Г.Д. КОГАЙ, *к.т.н., профессор,*
Н.И. ТОМИЛОВА, *к.т.н., ст. преподаватель,*
Карагандинский государственный технический университет, кафедра ВТиПО

Ключевые слова: *параллельное вычисление, кластерные системы, управление распределением, коммуникация данных, процессор.*

Параллелизм – это возможность одновременного выполнения нескольких арифметико-логических или служебных операций. На стадии постановки задачи параллелизм не определен, он появляется только после выбора метода вычислений. В зависимости от характера этого метода параллелизм алгоритма, то есть количество одновременно выполняемых операций, может меняться в довольно больших пределах.

Параллелизм используемой ЭВМ также меняется в широких пределах и зависит в первую очередь от числа процессоров, способа размещения данных, методов коммутации и способов синхронизации процессов. Язык программирования является средством переноса параллелизма алгоритма на параллелизм ЭВМ, и тип языка может в сильной степени влиять на результат переноса.

Для сравнения параллельных алгоритмов необходимо уметь оценивать степень параллелизма. Одновременное выполнение операций возможно, если они не зависят друг от друга по данным или управлению.

Разработка алгоритмов (в особенности методов параллельных вычислений) для решения сложных научно-технических задач часто представляет собой значительную проблему. С учетом высказанных предположений последующие действия для определения эффективных способов организации параллельных вычислений могут состоять в следующем:

– выполнить анализ имеющихся вычислительных схем и осуществить их разделение (декомпозицию) на части (подзадачи), которые могут быть реализованы в значительной степени независимо друг от друга;

– выделить для сформированного набора подзадач информационные взаимодействия, которые должны осуществляться в ходе решения исходной поставленной задачи;

– определить необходимую (или доступную) для решения задачи вычислительную систему и выполнить распределение имеющего набора подзадач между процессорами системы.

При самом общем рассмотрении понятно, что объем вычислений для каждого используемого процессора должен быть примерно одинаков, – это позволит обеспечить равномерную вычислительную загрузку (балансировку) процессоров. Кроме того, также понятно, что распределение подзадач между процессорами должно быть выполнено таким образом, чтобы

количество информационных связей (коммуникационных взаимодействий) между подзадачами было минимальным.

После выполнения всех перечисленных этапов проектирования можно оценить эффективность разрабатываемых параллельных методов: для этого обычно определяются значения показателей качества порождаемых параллельных вычислений (ускорение, эффективность, масштабируемость). По результатам проведенного анализа может оказаться необходимым повторение отдельных (в предельном случае всех) этапов разработки – следует отметить, что возврат к предшествующим шагам разработки может происходить на любой стадии проектирования параллельных вычислительных схем.

Чтобы применить получаемый в конечном итоге параллельный метод, необходимо выполнить разработку программ для решения сформированного набора подзадач и разместить разработанные программы по процессорам в соответствии с выбранной схемой распределения подзадач. Для проведения вычислений программы запускаются на выполнение (программы на стадии выполнения обычно именуются процессами), для реализации информационных взаимодействий программы должны иметь в своем распоряжении средства обмена данными (каналы передачи сообщений).

Следует отметить, что каждый процессор обычно выделяется для решения единственной подзадачи, однако при наличии большого количества подзадач или использовании ограниченного числа процессоров это правило может не соблюдаться и в результате на процессорах может выполняться одновременно несколько программ (процессов). В частности, при разработке и начальной проверке параллельной программы для выполнения всех процессов может использоваться один процессор (при расположении на одном процессоре процессы выполняются в режиме разделения времени).

Параллельное программирование представляет дополнительные источники сложности, необходимо явно управлять работой тысяч процессоров, координировать миллионы межпроцессорных взаимодействий. Для того чтобы решить задачу на кластере, необходимо распределить вычисления между процессорами системы так, чтобы каждый процессор был занят решением части задачи. Кроме того, желательно, чтобы как можно меньший объем данных пересылался между процессорами, поскольку коммуникации значительно более медленные операции, чем вычисления.

Часто возникают конфликты между степенью распараллеливания и объемом коммуникаций, то есть чем между большим числом процессоров распределена задача, тем больший объем данных необходимо пересылать между ними. Среда параллельного программирования должна обеспечивать адекватное управление распределением и коммуникациями данных.

Основные модели параллельного программирования, их абстракции в параллельном программировании.

Процесс/канал (Process/Channel). В этой модели программы состоят из одного или более процессов, распределенных по процессорам. Процессы выполняются одновременно, их число может измениться в течение времени выполнения программы. Процессы обмениваются данными через каналы, которые представляют собой однонаправленные коммуникационные линии, соединяющие только два процесса. Каналы можно создавать и удалять.

Модель процесс/канал характеризуется следующими свойствами:

– Параллельное вычисление состоит из одного или более одновременно исполняющихся процессов, число которых может изменяться в течение времени выполнения программы.

– Процесс – это последовательная программа с локальными данными. Процесс имеет входные и выходные порты, которые служат интерфейсом к среде процесса.

– В дополнение к обычным операциям процесс может выполнять следующие действия: послать сообщение через выходной порт, получить сообщение из входного порта, создать новый процесс и завершить процесс.

– Посылающаяся операция асинхронная – она завершается сразу, не ожидая того, когда данные будут получены. Получающаяся операция синхронная: она блокирует процесс до момента поступления сообщения.

– Пары из входного и выходного портов соединяются очередями сообщений, называемыми каналами (channels). Каналы можно создавать и удалять. Ссылки на каналы (порты) можно включать в сообщения, так что связность может измениться динамически.

– Процессы можно распределять по физическим процессорам произвольным способами, причем используемое отображение (распределение) не воздействует на семантику программы. В частности, множество процессов можно отобразить на одиночный процессор.

Обмен сообщениями (Message Passing). В этой модели программы, возможно, различные, написанные на традиционном последовательном языке, исполняются процессорами компьютера. Каждая программа имеет доступ к памяти исполняющего ее процессора. Программы обмениваются между собой данными, используя подпрограммы приема/передачи данных некоторой коммуникационной системы. Программы, использующие обмен сообщениями, могут выполняться только на MIMD – компьютерах.

Модель «обмен сообщениями» не накладывает ограничений ни на динамическое создание процессов, ни на выполнение нескольких процессов одним процессором, ни на использование разных программ для разных процессов. На практике сложилось так, что большинство систем обмена сообщениями при запуске параллельной программы создает фиксированное число идентичных процессов и не позволяет создавать и разрушать процессы в течение работы программы.

В таких системах каждый процесс выполняет одну и ту же программу (параметризованную относительно идентификатора либо процесса, либо процессора), но работает с разными данными, поэтому о таких системах говорят, что они реализуют SPMD (single program multiple data – одна программа много данных) модель

программирования. SPMD-модель приемлема и достаточно удобна для широкого диапазона приложений параллельного программирования, но она затрудняет разработку некоторых типов параллельных алгоритмов.

Параллелизм данных (Data Parallel). В этой модели единственная программа задает распределение данных между всеми процессорами компьютера и операции над ними. Распределяемыми данными обычно являются массивы. Как правило, языки программирования, поддерживающие данную модель, допускают операции над массивами, позволяют использовать в выражениях целые массивы, вырезки из массивов. Распараллеливание операций над массивами, циклов обработки массивов позволяет увеличить производительность программы. Компилятор отвечает за генерацию кода, осуществляющего распределение элементов массивов и вычислений между процессорами. Каждый процессор отвечает за то подмножество элементов массива, которое расположено в его локальной памяти.

Следовательно, компиляторы языков с параллелизмом данных часто требуют, чтобы программист предоставил информацию относительно того, как данные должны быть распределены между процессорами, другими словами, как программа должна быть разбита на процессы. Компилятор транслирует программу с параллелизмом данных в SPMD программу, генерируя коммуникационный код автоматически.

Общая память (Shared Memory). В этой модели все процессы совместно используют общее адресное пространство. Процессы асинхронно обращаются к общей памяти как с запросами на чтение, так и с запросами на запись, что создает проблемы при выборе момента, когда можно будет поместить данные в память, когда можно будет удалить их. Для управления доступом к общей памяти используются стандартные механизмы синхронизации – семафоры и блокировки процессов.

В модели программирования с общей памятью все процессы совместно используют общее адресное пространство, к которому они асинхронно обращаются с запросами на чтение и запись. В таких моделях для управления доступом к общей памяти используются всевозможные механизмы синхронизации типа семафоров и блокировок процессов.

Преимущество этой модели с точки зрения программирования состоит в том, что понятие собственности данных (монопольного владения данными) отсутствует, следовательно, не нужно явно задавать обмен данными между производителями и потребителями.

СПИСОК ЛИТЕРАТУРЫ

1. Малашонок Г.И., Аветисян А.И., Валеев Ю.Д., Зуев М.С. Параллельные алгоритмы компьютерной алгебры. М., 2004. 376 с.
2. Воеводин Вл.В., Капитонова А.П. Методы описания и классификации вычислительных систем. М.: Изд-во МГУ, 1994. 413 с.
3. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. М.: Изд-во МГУ, 2002. 342 с.

УДК 656.13.05

Структурная модель интегрированной системы менеджмента качества автотранспортных услуг

Ж.К. МУСТАФИН, *ст. преподаватель СГУ им. Шакарима*

Ключевые слова: *бизнес-моделирование, автотранспортные услуги, структурная модель, интегрированная система, менеджмент качества, автотранспортные услуги, метод экспертных оценок, коэффициент ранговой корреляции Спирмена, взаимооценка компетентности экспертов, матрица корреляций, нормированные балльные оценки, системы технического обслуживания автомобилей.*

Первым этапом структурно-функционального бизнес-моделирования в системе автотранспортных услуг необходимо выделить основные и вспомогательные бизнес-процессы.

В качестве инструмента расстановки приоритетов среди всего перечня процессов нами выбран метод экспертных оценок. При этом учитывалось, что приоритетные процессы должны отвечать следующим характеристикам [1, 2]:

– оказывать наибольшее влияние;

– быть максимально эффективными в целевых радикальных улучшениях;

– легко подвергаться улучшению.

Как известно, метод экспертных оценок используется для получения решений в слабо формализованных задачах, в которых накоплен достаточно большой объем информации и носителями информации являются специалисты, выступающие в роли экспертов. Отбор экспертов осуществлялся на основе следующих критериев [2]:

- компетентности;

- отсутствия личной заинтересованности в результате экспертизы;

- креативности (широта познаний);