ПЛАНИРОВАНИЕ ВЫПОЛНЕНИЯ НА ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ ГРАФ-СХЕМНЫХ ПАРАЛЛЕЛЬНЫХ ПОТОКОВЫХ ПРОГРАММ

В.П. Кутепов, С.Н. Макарьевский

ВВЕДЕНИЕ

Целью статьи является рассмотрение алгоритмов статического планирования выполнения граф-схемных программ на вычислительных системах (ВС) [1]. Статическое планирование предполагает такое начальное распределение фрагментов параллельной программы на узлы ВС, при котором минимизируется время ее выполнения. Обычно это решается путем достижения баланса между суммарными временами выполнения размещенных фрагментов на узлах и минимизации времени межкомпьютерных обменов.

В отличие от статического планирования процессов динамическое планирование позволяет достигать большего эффекта за счет использования более «тонких» механизмов перераспределения процессов параллельной программы во время ее выполнения. Использование эффективных алгоритмов динамического управления процессами может, в свою очередь, до 50% сокращать время выполнения параллельной программы и существенно повысить эффективность использования ресурсов, в частности, загруженность узлов ВС [2,3].

Только комплексное решение проблемы планирования процессов и управления загруженностью кластера с возможностью статического и динамического планирования может дать необходимый эффект. Именно такой подход используется для решения этой проблемы в созданной на кафедре прикладной математики МЭИ системе граф-схемного потокового параллельного программирования [1].

ОСОБЕННОСТИ ГРАФ-СХЕМНОГО ПОТОКОВОГО ПРОГРАММИРОВАНИЯ

Граф-схемная параллельная программа (ГСПП) представляется в виде пары <ГС,I>, где ГС – граф-схема, I – интерпретация.

Граф-схема позволяет визуально представлять строящуюся из отдельных компонентов (модулей) программу решения задачи (рис. 1). Интерпретация сопоставляет каждому модулю множество подпрограмм, а информационным связям (ИС) между модулями – типы данных, передаваемых между подпрограммами модулей в процессе выполнения ГСПП.

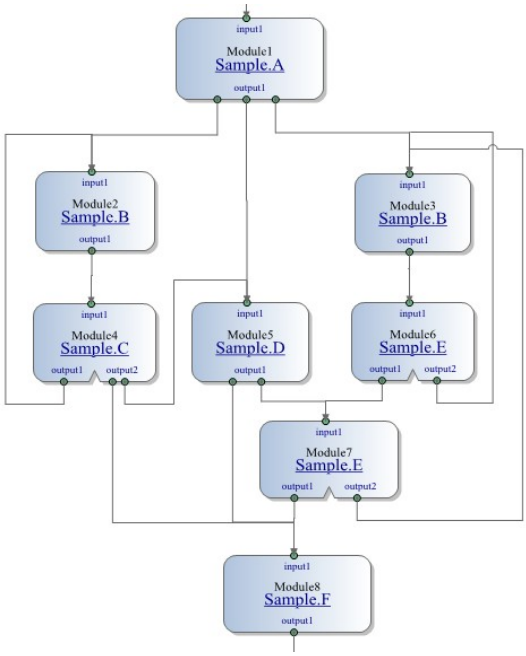

Рис. 1 Пример граф-схемной параллельной программы

Все входы и выходы модулей строго типизированы и разделены на группы, которые называются соответственно конъюнктивными группами входов (КГВх) и конъюнктивными группами выходов (КГВых) и отражают структуру потоков данных, передаваемых между подпрограммами модулей. Каждой КГВх модуля однозначно сопоставляется подпрограмма. Кроме того, в процессе выполнения подпрограмме передается дополнительно целочисленный параметр – тег. Он служит для идентификации данных, передаваемых между модулями ГСПП в процессе его выполнения, и закреплен за конкретной копией данных. Механизм тегирования данных (приписывания тега данным) позволяет обеспечивать однозначность отношения «входные данные – результат» при параллельном выполнении ГСПП.

Выполнение ГСПП описывается последовательностью состояний, каждое из которых есть множество процессов, индуцируемых при запуске подпрограмм модулей. При выполнении ГСПП каждой ИС однозначно сопоставляется буфер, в котором хранятся асинхронно поступающие на входы КГВх модуля данные. Если на всех входах какой-либо из КГВх модуля есть данные, помеченные одним и тем же тегом, то индуцируется новый процесс, предполагающий применение подпрограммы, сопоставленной КГВх к собранным данным с указанным тегом.

МОДЕЛЬНОЕ ПРЕДСТАВЛЕНИЕ ГРАФ-СХЕМНЫХ ПРОГРАММ ПРИ РЕШЕНИИ ЗАДАЧИ СТАТИЧЕСКОГО ПЛАНИРОВАНИЯ

Особенностью ГСПП является графическое их представление, строго определяющее связи по данным, которые передаются между подпрограммами модулей программы при ее выполнении (рис. 1). Поскольку эти связи при выполнении ГСПП выполняют роль буферов данных, в качестве модельного ее представления, используемого для статического планирования, предлагается использовать взвешенные мультиграфы, в которых вершины соответствуют подпрограммам ГСПП, а дуги – передаваемым данным. При этом веса вершин – это вычислительная сложность соответствующих подпрограмм (например, среднее время выполнения подпрограммы), а веса дуг – интенсивность передаваемых данных от выхода одной подпрограммы к входу другой (возможно, этой же) подпрограммы.

Взвешенный мультиграф, соответствующий ГСПП на рис. 1, показан на рис. 2.

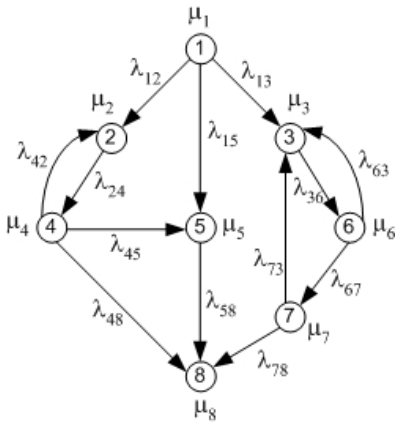

Рис. 2 Взвешенный мультиграф
граф-схемной программы

На этом рисунке μ_i – приведенные к быстродействию процессора интенсивности выполнения i-ой программы на процессоре компьютера (в измерении оп/сек), а λ_{ij} – приведенные к быстродействию коммуникаций интенсивности передаваемых между i-ой и j-ой подпрограммами данных в процессе выполнения параллельной программы. Эти интенсивности либо программист определяет сам, либо они являются результатом многократных прогонов программы.

Задача статического планирования параллельного выполнения ГСПП на заданной ВС или в вычислительной среде GRID может быть сформулирована как задача оптимальной декомпозиции ГСПП на подграфы и такого их распределения на вычислительные узлы, чтобы достигалось минимальное время ее выполнения.

Мы будем рассматривать две ситуации. В первой известны только структурные характеристики ГСПП, полученные из граф-схемы, с целью произвести оптимальную декомпозицию ГСПП. Во второй к структурным характеристикам добавлены динамические данные о выполнении ГСПП.

Пусть N – число узлов ВС.

В первом случае мы рассматриваем следующие постановки задачи:

- Разрезать ГС на N фрагментов таким образом, чтобы при условии минимума межфрагментных связей достигалось минимальное среднеквадратичное отклонение по количеству подпрограмм во фрагментах.

- Разрезать ГС на *N* фрагментов таким образом, чтобы при условии минимального среднеквадратичного отклонения по количеству подпрограмм во фрагментах достигался минимум межфрагментных связей.

Примеры на рис. 3 иллюстрируют разрезания ГС на два фрагмента (F_1 и F_2).

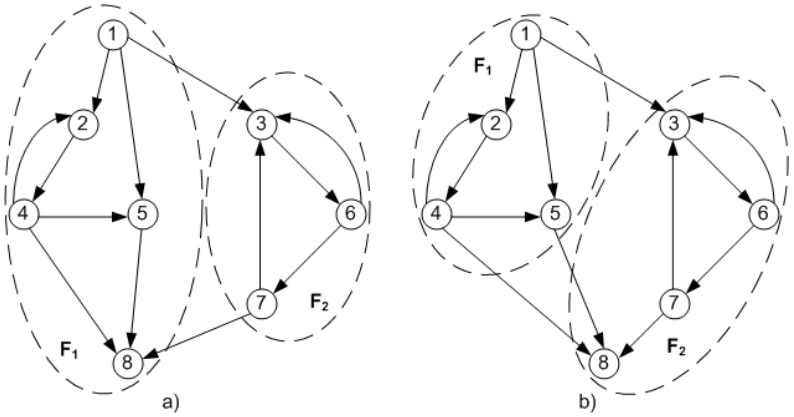

Рис. 3 Примеры разрезания граф-схемы на два фрагмента,соответствующие первой (a) и второй задаче (b)

К обеим формулировкам задач может быть добавлено дополнительное условие: циклические участки ГС должны иметь минимальное число разрезанных связей.

В случае если известны вычислительные сложности подпрограмм (например, их время выполнения) и интенсивности передачи данных между подпрограммами, приведенные выше задачи могут быть переформулированы, учитывая эту информацию:

- Разрезать ГС на *N* фрагментов таким образом, чтобы при условии минимума суммарной интенсивности передачи данных между фрагментами достигалось минимальное среднеквадратичное отклонение по сложности всех подпрограмм во фрагментах.
- Разрезать ГС на *N* фрагментов таким образом, чтобы при условии минимального среднеквадратичного отклонения по сложности всех подпрограмм во фрагментах достигался минимум суммарной интенсивности передачи данных между фрагментами.

Как и в предыдущем случае, может быть учтено дополнительное условие минимизации числа разрезанных дуг циклических участков ГС. Для вычислительных сред GRID минимизация времени обменов между фрагментами программы более предпочтительна из-за временных затрат, расходуемых на обмены данными между узлами через интернет-коммуникации.

АЛГОРИТМЫ СТАТИЧЕСКОГО ПЛАНИРОВАНИЯ

Для получения точного решения каждой из поставленных задач необходимо использовать переборные алгоритмы (NP-полная задача). Как показали эксперименты, при числе вершин, превышающем 20, затрачиваемое время на получение результата очень велико. Выходом из этого положения является использование приближенных алгоритмов.

Приближенные алгоритмы для решения подобных задач уже рассматривались ранее [4], но, как правило, каждый из них применялся на определенном узком классе графов. Графы, являющиеся модельным представлением граф-схемных потоковых программ, обладают следующими особенностями: направленность дуг, возможность существования циклов, в общем случае непланарность. В настоящий момент разрабатываются специальные алгоритмы, учитывающие специфику рассматриваемого класса графов и основанные на стягивании сильно связных групп вершин.

ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИЕ АЛГОРИТМОВ СТАТИЧЕСКОГО ПЛАНИРОВАНИЯ

Исследование разработанных точных переборных алгоритмов статического планирования параллельного выполнения граф-схемных программ осуществлялось на кластере МЭИ со следующими характеристиками: 16 узлов, каждый из которых содержит два двуядерных процессора AMD Opteron с частотой 2.2 ГГц, пропускная способность коммуникаций – 1 Гбит/с. Исследование эффективности алгоритмов производилось с помощью синтетических граф-схемных программ с 12 и 18 модулями и таким же числом подпрограмм.

Далее будут использованы следующие сокращения:

Распределение 1 – минимизация среднеквадратичного отклонения по количеству подпрограмм во фрагментах при условии минимума межфрагментных связей.

Распределение 2 – минимизация межфрагментных связей при условии минимального среднеквадратичного отклонения по количеству подпрограмм во фрагментах.

Распределение 3 – минимизация среднеквадратичного отклонения по сложности всех подпрограмм при условии минимума суммарной интенсивности передачи данных между фрагментами.

Распределение 4 – минимизация суммарной интенсивности передачи данных между фрагментами при условии минимального среднеквадратичного отклонения по сложности всех подпрограмм во фрагментах.

Проведенные эксперименты показали, что:

- Распределение 1 позволяет уменьшить время выполнения приложения на 5-34% по сравнению со случайным распределением модулей по узлам ВС.
- Распределение 2 позволяет уменьшить время выполнения приложения на 10-23% по сравнению со случайным распределением модулей по узлам ВС.
- Распределение 3 позволяет уменьшить время выполнения приложения на 2-31% по сравнению со случайным распределением модулей по узлам ВС.
- Распределение 4 позволяет уменьшить время выполнения приложения на 13-28% по сравнению со случайным распределением модулей по узлам ВС.

ЗАКЛЮЧЕНИЕ

Проведенные исследования показали, что предложенные точные алгоритмы достаточно эффективны и позволяют существенно сокращать время выполнения программы (до 30%). В то же время экспоненциальная сложность этих алгоритмов ограничивает применение их для разрезания граф-схем. В ближайшее время планируется провести исследование приближенных алгоритмов распределения модулей ГСПП на узлы ВС.

ЛИТЕРАТУРА:

1. Д.В. Котляров, В.П. Кутепов, М.А. Осипов. Граф-схемное потоковое параллельное программирование и его реализация на кластерных системах. М: Из-во РАН, Теория и системы управления, 2005, №1.
2. В.П. Кутепов. Об интеллектуальных компьютерах и больших компьютерных системах нового поколения. Теория и системы управления, 1996, №5.
3. V.P. Kutepov. Scheduling parallel processes and load balancing in large-scale computing systems. 2007 International Symposium on Distributed Computing and Applications to Business, Engineering and Science, August 14-17, 2007, YiChang, HuBei, China.
4. Bradford L. Chamberlain. Graph partitioning algorithms for distributing workloads of parallel computations. Technical Report TR-98-10-03, Univ. of Washington, Dept. of Computer Science & Engineering, October 1998.