УДК 330.4:330.46

ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ И СОЗДАНИЕ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ

В.И. КЛИМЕНКО

Кубанский государственный технологический университет,
350002, Российская Федерация, г. Краснодар, ул. Московская, 2;
электронная почта: lerik.ru_26@mail.ru

Статья посвящена параллельным вычислениям и инструментам, помогающим распараллелить задачу для наиболее эффективного ее использования на вычислительных системах. Актуальность данной проблемы бесспорна. В современном обществе область параллельных вычислений до сих пор не получила широкого применения в силу необходимости построения принципиально новых алгоритмов, решающих поставленные задачи. Развитие многопроцессорной техники связано с разработкой технологий параллельного программирования, используемых как для универсальных, так и для конкретных архитектур ЭВМ. При составлении программных средств особое внимание уделяется построению архитектуры программы, в силу этого изучение алгоритмов распараллеливания и знание области применения каждого из них для повышения эффективности их выполнения на многопроцессорных ЭВМ становится злободневной темой. В связи с этим исследование, отражённое в рецензируемой статье, своевременно и актуально.

Ключевые слова: параллельные вычисления, уровни распараллеливания, параллелизм данных, циклические и ациклические участки распараллеливания.

В современном мире существует большое количество задач, решение которых требует использования огромных вычислительных мощностей, не всегда доступных для вычислительных систем, и задач подобного рода становится все больше и больше [1-9]. Так же возрастают и требования к точности и скорости решения прежних задач, поэтому вопросы создания и использования более мощных компьютеров являются актуальными на сегодняшний день и в ближайшем будущем. Однако технологические возможности повышения быстродействия процессоров ограничены, поэтому становится все более востребованным создание параллельных вычислительных систем, которые позволяют одновременную реализацию ряда вычислительных процессов, связанных с решением одной задачи. Но и данный метод имеет свои недостатки, которые не позволяют получить ускорение вычислений во столько раз, сколько использовано вычислительных модулей (процессоров или компьютеров) для данного решения, да и не всякую задачу можно подвергнуть

распараллеливанию. Рассмотрим в статье положительные и отрицательные стороны использования и создания параллельных программ.

В первую очередь дадим определение такому понятию как параллельные вычисления. Параллельные вычисления – это способ выполнения компьютерных вычислений, при котором программы разрабатываются как набор взаимодействующих вычислительных процессов, работающих параллельно, то есть одновременно. Задачей параллельных вычислений является получение параллельного алгоритма и управление реализацией параллелизма с использованием многопроцессорной вычислительной техники для достижения наибольшей эффективности. Под эффективностью следует понимать максимальную загруженность процессоров, т.е. отсутствие простоя каждого элемента вычислительной системы, однако на практике это нереализуемо. Вследствие этого эффективность использования многопроцессорных вычислительных систем в большей степени зависит от качества прикладных параллельных программ [9].

Для использования суперкомпьютеров подразумевалось создание единой программы, для выполнения которой будут задействованы все ресурсы системы. Но данная задача не всегда может быть выполнена или ее реализация будет нецелесообразной в силу недостаточности выполнения простого разбиения программы на параллельные ветви. Чтобы данное решение стало обладать преимуществом в скорости и эффективности работы необходимо, чтобы все ветви программы выполняли примерно одинаковый объем вычислений, так же важно исключить простои из-за ожидания данных, передачи управления и возникновения конфликтов в виду использования общих ресурсов [9]. Выполнение данных принципов при разработке программы позволит получить идеальную параллельную программу.

Для получения параллельного алгоритма можно преобразовать уже существующий последовательный алгоритм путем его распараллеливания или создать новый параллельный алгоритм.

Существуют различные приемы распараллеливания того или иного участка программы, поэтому их классифицируют на ациклические и циклические, для каждого из которых характерны свои способы распараллеливания.

Алгоритм распараллеливания ациклических участков программы состоит из четырех этапов:

– построение графа зависимостей по данным;
– построение ярусно-параллельной формы (ЯПФ) программы;
– составление по ЯПФ параллельной программы;
– отображение полученной программы на архитектуру.

На первом этапе строится граф зависимостей по данным между операторами программы, который показывает зависимость между ними. Различают информационную, логическую и конкурентоспособную зависимость. Следует отметить, что граф можно строить на разных уровнях – от отдельных конструкций до крупных программных блоков [8].

Информационная зависимость выражается в необходимости получения некоторого «вырабатывающего» значения предшествующим оператором для выполнения данного, при условии, что путь от входного оператора программы, проходящий через оба этих оператора, существует. При логической зависимости предшествующий оператор выполняет функцию регулирования выполнения данного оператора, выступая своеобразным «распознавателем». Если же операторы «вырабатывают» одну и ту же переменную, то такую зависимость называют конкуренционной. Каждая из зависимостей отражается на графике стрелками.

При построении данного графа целесообразно выделять в нем подграфы для уменьшения масштаба. Это позволит сделать его более наглядным.

Затем строится ярусно-параллельная форма программы (ЯПФ), на первый уровень которой помещаются операторы, не содержащие входящих в них стрелок зависимостей, а каждый последующий уровень состоит из операторов, имеющих входные стрелки только от предыдущих ярусов.

Из алгоритма построения ЯПФ можно сделать вывод, что все операторы, которые находятся на одном ярусе, могут выполнятся параллельно, причем каждый ярус показывает, что к моменту его вычисления должны быть закончены все вычисления на предыдущих ярусах.

Затем по ЯПФ происходит составление параллельной программы с учетом цели выполнения ее на определенной вычислительной машине, с дальнейшим ее отображением на архитектуру параллельной вычислительной системы.

Во многих алгоритмах количество операций для вычислений, производящихся в цикле, довольно велико по отношению ко всему числу операций. Поэтому задача распараллеливания циклических участков программы является весьма значимой.

При рассмотрении задачи распараллеливания циклов принято говорить в терминах пространства итераций, под которым подразумевается множество целочисленных векторов. Задача формулируется следующим образом: необходимо разбить множество целочисленных векторов на такие подмножества, одновременное вычисление тела цикла в каждом из которых может быть выполнено с сохранением информационных связей исходного цикла [8].

Распараллеливание циклов реализуется компиляторами, векторизующим или распараллеливающим в зависимости от вида вычислительной системы.

В зависимости от типа областей подмножеств выделяют следующие методы распараллеливания циклов:

– метод параллелепипедов;
– метод гиперплоскостей;
– метод пирамид.

Методы используются только в случае выполнения ряда ограничений на операторы тела цикла и в свою очередь зависят от типа ЭВМ.

Целью любой работы, связанной с параллельным программированием, является рассмотрение взаимосвязей между структурой математического

алгоритма и архитектурой многопроцессорной вычислительной системы, которые называются уровнями декомпозиции исходной задачи. Между этими уровнями не существует четкой границы и довольно сложно какую-то конкретную технологию распараллеливания отнести к одному из них.

В большинстве случаев распараллеливание на уровне задач является самым простым и эффективным. Его применяют в таких случаях, когда решаемая задача состоит из независимых отдельных подзадач. Однако данный метод не всегда применим в силу невозможности деления некоторых задач на более мелкие. В таких случаях применяют иные уровни параллелизма [7].

Параллелизм данных – это применении одной и той же операции к множеству элементов данных. Этот вид параллелизма широко применяется при решении задач численного моделирования. В таких задачах область вычислений представлена в виде ячеек и таких ячеек может быть миллионы и миллиарды, над каждой из которых должна быть выполнена одна и та же операция. Здесь модель параллелизма по данным крайне удобна, так как позволяет загрузить каждое ядро, выделив ему определенный набор ячеек. Вся счетная область делится на геометрические объекты и ячейки, вошедшие каждую из этих областей, отдаются на обработку определенному ядру. В математической физике данный вид параллелизма называют геометрическим параллелизмом.

Следующий уровень – это распараллеливание отдельных процедур и алгоритмов. На данный уровень можно отнести алгоритмы параллельной сортировки, умножение матриц, решение системы линейных уравнений. На этом уровне отдают предпочтение использованию технологии параллельного программирования, как ***OpenMP***, при которой программа начинается как единственный поток выполнения, называемый начальным потоком, который при встрече с параллельной конструкцией создает новую группу потоков, состоящую из себя и дополнительных потоков. Легкость данного метода состоит в том, что разработчик не создает новую параллельную программу, а просто добавляет к тексту последовательной программы ***OpenMP***-директивы.

Наиболее низкий уровень параллелизма, осуществляемый на уровне параллельной обработки процессором нескольких инструкций. На данном уровне находится пакетная обработка нескольких элементов данных одной командой процессора. Этот тип параллельности иногда выделяют в еще более глубокий уровень, получивший название параллелизм на уровне битов.

Несмотря на успехи в применении мультипроцессорных систем, имеют место рассуждения об их низкой эффективности. Несбалансированность вычислительной работы одна из причин уменьшения производительности при распараллеливании с увеличением числа вычислительных модулей программы.

При разработке параллельных программ для достижения высокой эффективности требуется многократные изменения программы для получения наилучшей схемы ее распараллеливания. Успешность данного решения определяется простотой модификации программы.

ЛИТЕРАТУРА

1. Попова О.Б. База данных раздела «Вычислительные системы» курса лекций «Вычислительные системы, сети и телекоммуникации» // Программы для ЭВМ. Базы данных. Топологии интегральных микросхем. 2015. №9. С. 88.

2. Попова О.Б. База данных раздела «Телекоммуникации» курса лекций «Вычислительные системы, сети и телекоммуникации» // Программы для ЭВМ. Базы данных. Топологии интегральных микросхем. 2015. №9. С. 72.

3. Попова О.Б. База данных раздела «Сети» курса лекций «Вычислительные системы, сети и телекоммуникации» // Программы для ЭВМ. Базы данных. Топологии интегральных микросхем. 2015. №9. С. 87.

4. Головашкин Д. Л.Современные методы и алгоритмы решения сложных задач на суперкомпьютерах: Электронное учебное пособие, Самара, 2010. – 104 с.

5. Бурова И. Г., Демьянович Ю. К. Алгоритмы параллельных вычислений и программирование: Курс лекций, Санкт-Петербургский государственный университет, 2007. – 207 с.

6. Документация по языку C#, http://msdn.microsoft.com

7. Знакомство с уровнями распараллеливания, http://habrahabr.ru

8. Параллельные вычисления (базовый курс), http://bigor.bmstu.ru

9. Андрей Карпов. Введение в проблематику разработки параллельных программ, http://www.viva64.com

REFERENCES

1. Popova O.B. Baza dannykh razdela «Vychislitelnye sistemy» kursa lektsiy «Vychislitelnye sistemy, seti i telekommunikatsii» // Programmy dlya EVM. Bazy dannykh. Topologii integralnykh mikroskhem. 2015. №9. S. 88.

2. Popova O.B. Baza dannykh razdela «Telekommunikatsii» kursa lektsiy «Vychislitelnye sistemy, seti i telekommunikatsii» // Programmy dlya EVM. Bazy dannykh. Topologii integralnykh mikroskhem. 2015. №9. S. 72.

3. Popova O.B. Baza dannykh razdela «Seti» kursa lektsiy «Vychislitelnye sistemy, seti i telekommunikatsii» // Programmy dlya EVM. Bazy dannykh. Topologii integralnykh mikroskhem. 2015. №9. S. 87.

4. Golovashkin D. L.Sovremennye metody i algoritmy resheniya slozhnykh zadach na superkompyuterakh: Elektronnoe uchebnoe posobie, Samara, 2010.-104 c.

5. Burova I. G., Demyanovich Yu. K. Algoritmy parallelnykh vychisleniy i programmirovanie: Kurs lektsiy, Sankt-Peterburgskiy gosudarstvennyy universitet, 2007. – 207 s.

6. Dokumentatsiya po yazyku S#, http://msdn.microsoft.com

7. Znakomstvo s urovnyami rasparallelivaniya, http://habrahabr.ru

8. Parallelnye vychisleniya (bazovyy kurs), http://bigor.bmstu.ru

9. Andrey Karpov. Vvedenie v problematiku razrabotki parallelnykh programm, http://www.viva64.com

PARALLEL CALCULATIONS AND CREATION OF PARALLEL PROGRAMS

V. I. KLIMENKO

Kuban State Technological University,
2, Moskovskaya st., Krasnodar, Russian Federation, 350002,
e-mail: lerik.ru_26@mail.ru

Article is devoted to the parallel calculations and tools helping to parallelize a task for its most effective use on computing systems. Relevance of this problem is indisputable. In modern society the field of parallel calculations wasn't widely used owing to need of creation of essentially new algorithms solving objectives yet. Development of the multiprocessor equipment is connected with development of the technologies of parallel programming used both for universal and for specific architecture of the computer. In case of creation of software special attention is paid to creation of architecture of the program, owing to this fact studying of algorithms of parallelization and knowledge of a scope of each of them for increase in efficiency of their accomplishment on the multiprocessor computers becomes a topic of the day. With respect thereto the research reflected in the reviewed article, timely and urgent.
Key words: parallel calculations, parallelization levels, overlapping of data, cyclic and acyclic sites of parallelization.