УДК 681.3.06

ОБ ОДНОМ ПОДХОДЕ К ВИЗУАЛЬНОЙ РАЗРАБОТКЕ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ

© 2012 г.

В.П. Гергель, Е.А. Козинов

Нижегородский госуниверситет им. Н.И. Лобачевского

Evgeniy.Kozinov@gmail.com

Поступила в редакцию 20.12.2011

Одно из направлений снижения сложности разработки параллельных программ – использование сред *визуального* анализа и проектирования программного обеспечения. В статье приведено описание новой разрабатываемой визуальной среды Визограф-1. Рассматриваются основные понятия и возможности среды, даются примеры применения.

Ключевые слова: визуальные средства разработки параллельных программ, параллельное программирование, высокопроизводительные вычисления.

Введение

На протяжении многих лет одной из наиболее острых проблем эффективного использования потенциала компьютерных технологий является трудоемкость разработки программного обеспечения (ПО). Широкий спектр практических приложений, для решения которых разрабатываются программы, разнообразие компьютерных систем и их непрерывное совершенствование, существенное повышение сложности применяемых алгоритмов обработки – все это вместе взятое чрезвычайно затрудняет разработку ПО. В отличие от компьютерного оборудования, производительность которого практически удваивается каждые 18 месяцев (широко известное утверждение Мура), технологии разработки программ совершенствуются крайне медленно – к сожалению, закон Мура для ПО не действует.

Для параллельных вычислительных систем проблема трудоемкости разработки становится критической. Дополнительные сложности вызывает относительная новизна области параллельного программирования, наличие качественно новых эффектов при выполнении вычислительных процессов (тупики, гонки потоков, избыточная синхронизация и т.п.). Особую остроту проблемы создает необходимость оперативной массовой разработки параллельных программ в силу коренного перехода компьютерной техники на использование многоядерных процессоров.

Одно из активно развиваемых направлений, уменьшающих сложность – создание сред визуального проектирования и анализа параллельных программ.

Существующие среды визуального проектирования параллельных программ можно разделить на несколько групп в соответствии с типом их графических нотаций. Так, можно выделить следующие графические представления параллельных программ:

- отображение программ в виде графов функциональных зависимостей;
- отображение программ в виде сетей асинхронно выполняемых последовательностей задач;
- отображение программ через представление графов потоков данных.

В первой группе сред визуального проектирования, как правило, есть два типа элементов – функции, отображающие логику программы, и аргументы, составляющие данные программы. Из функций строится дерево, отображающее логику разрабатываемой программы. Характерным примером графической среды, использующей функциональное представления программ, является система «Пифагор» [1].

Во второй группе в качестве графических элементов используются один или несколько типов «задач»-вершин, содержащих части логики разрабатываемого алгоритма. «Задачи» соединены между собой направленными дугами для отображения последовательности выполнения частей алгоритмов. Представителями данной группы визуальных сред являются системы «CODE» [2] и «ГРАФ ПЛЮС» [3]. Иногда в средах этой категории используются дополнительные графические элементы, как это сделано в системе «HeNCE» [4], в которой есть специальные графические элементы для представления условного, реплицируемого, циклического и конвейерного исполнения «задач».

В последней группе в качестве основных графических элементов также используются «задачи» и дуги, но их интерпретация меняется. Дуги в данной группе отображают передаваемые данные. Визуальные среды из данной группы могут отображать параллельный алгоритм с разной степенью детализации. При малой детализации «задачи» описывают всю работу для параллельно работающего вычислительного элемента, а дуги просто отображают возможность передачи данных в одном из направлений. Примером таких сред являются системы «TREPPER» [5] и «GRAPNEL» [6]. При большей детализации «задачи» отображают часть алгоритма, не привязанную к вычислительным элементам, а каждая дуга отображает передачу ровно одного элемента данных. В качестве примера таких сред можно привести системы «PGRAPH» [7] и «ЯГСПП» [8].

Далее в статье рассматривается новая разработанная авторами визуальная среда проектирования и анализа параллельных программ, которая также может быть отнесена к последней группе вышеупомянутых систем.

1. Основы системы визуального представления вычислительных схем

Для представления разрабатываемой системы визуальной разработки параллельных программ Визограф-1 будет использоваться двухэтапная схема изложения материала. При первом – начальном – уровне рассмотрения будет дана общая характеристика системы Визограф-1. Далее – на втором этапе – будет представлено расширенное описание системы. Для пояснения излагаемого материала будет использоваться пример разработки параллельной программы для выполнения матрично-векторного умножения.

При начальном рассмотрении системы Визограф-1 можно выделить три основных типа элементов для визуального представления вычислительных схем параллельных программ:

- **«модуль»** обеспечивает представление отдельных блоков параллельной программы;
- **«параметр»** используется для визуального представления входных и выходных параметров модулей;
- **«линкер»** применяется для установления соответствия (связывания) между параметрами модулей.

1. Модуль. В графическом виде элемент «модуль» изображается в виде скругленного прямоугольника (рис. 1), содержащего название модуля. Каждый модуль соответствует некоторой реализации определенного алгоритма или его части и может быть представлен, например, в виде программного модуля dll-библиотеки. Модуль может быть выполнен после задания значений его входных параметров, результатом выполнения модуля является вычисление значений его выходных параметров. Модуль может содержать набор свойств, конкретизирующих условия его применения.

Система Визограф-1 может содержать наборы (библиотеки) модулей. В рамках системы для этих библиотек обеспечивается возможность добавления и исключении модулей.

2. Параметр. Визуальные элементы «параметры» представляются в виде небольших окружностей, располагаемых на вертикальных границах модулей (рис. 1), и используются для представления входных и выходных данных модулей. Входные параметры должны быть заданы перед выполнением модуля. При этом предполагается, что модуль может быть выполнен в любой момент времени после задания его входных параметров. После выполнения модуля выходные параметры считаются определенными и могут быть использованы для задания входных параметров других модулей.

Параметры могут использоваться только в связке с модулями. При описании модуля в обязательном порядке должны указываться его параметры.

Параметры именуются и обладают свойствами, в которых могут быть указаны дополнительные характеристики параметров (тип, значения по умолчанию и т.п.).

3. Линкер. Визуальные элементы «линкеры» изображаются в виде стрелок, направленных от выходных параметров к входным параметрам модулей (рис. 1). Именно линкеры позволяют сконструировать визуальную схему алгоритма программы как графа из модулей, соединенных линкерами.

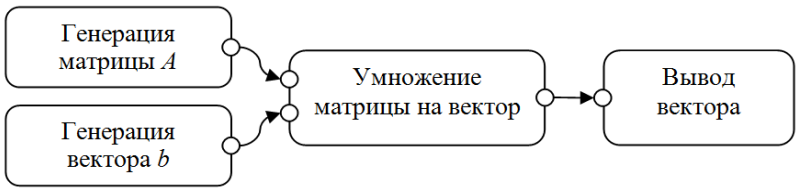

Рис. 1. Визуальная схема умножения матрицы на вектор (генерация начальных данных – выполнение операции умножения – вывод результата)

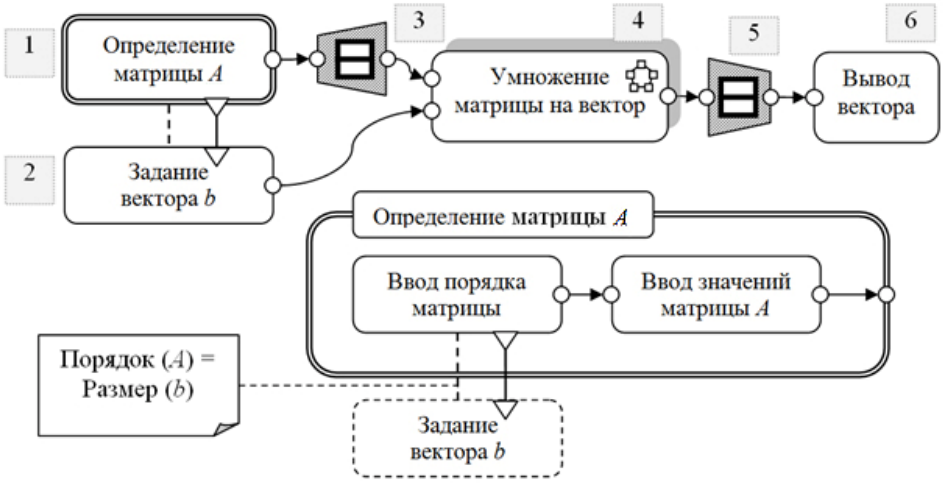

Рис. 2. Расширенная визуальная схема умножения матрицы на вектор

Рассмотренного набора визуальных элементов – модулей, параметров, линкеров – оказывается достаточно для визуального конструирования вычислительных схем без циклов и условных выражений. Пример такой простой схемы приведен на рис. 1.

2. Расширенные возможности системы визуального представления вычислительных схем

После представленного краткого введения рассмотрим основные возможности системы Визограф-1. Они позволяют строить сложные вычислительные схемы, по которым могут быть организованы параллельные вычисления на многопроцессорных системах.

1. *Многоуровневое представление вычислительных схем.* Модули быть атомарными и составными. Атомарному модулю соответствует определенный программный модуль, для задания составного модуля (*макромодуля*) должна быть задана его описывающая вычислительная схема. Макромодуль отображается на схеме в виде скругленного прямоугольника с двойной рамкой (рис. 2).

Использование макромодулей обеспечивает возможность многоуровневого представления вычислительных схем. Использование иерархии позволяет сохранить компактность представления каждого уровня при разработке вычислительных схем практически любой сложности. На рис. 2 показан пример такого многоуровневого представления визуальной схемы вычислительного алгоритма. На этом примере блок определения матрицы (блок 1) представлен в виде макромодуля, который на следующем уровне декомпозиции описывается как последовательность модуля ввода порядка матрицы и модуля непосредственного ввода значений матрицы.

2. *Многократное использование модулей.* Модули могут дублироваться (копироваться) для организации вычислений при наличии наборов однотипных действий. Так, например, операция матрично-векторного умножения может быть представлена через набор операций скалярного произведения векторов (рис. 2).

При визуальном представлении семейство экземпляров модуля (*мультимодуль*) изображается в виде прямоугольника с тенью для верхней и правой границ модуля (см. рис. 2, блок 4). Количество создаваемых экземпляров может быть указано через параметры свойств модуля.

Введение механизма дублирования модулей закладывает основы для параллельного выполнения разрабатываемых вычислительных схем, т.к. экземпляры мультимодуля (при их информационной независимости) могут быть выполнены параллельно на разных процессорах (ядрах) вычислительной системы.

3. *Декомпозиция и агрегация данных.* Данные для обработки в мультимодуле, как правило, разделяются между экземплярами модуля. Так, например, матрица при параллельном выполнении матрично-векторного умножения может быть разделена на полосы (наборы) строк, которые затем обрабатываются параллельно независимо друг от друга. Как итог, результаты, получаемые отдельными экземплярами модулей, могут являться частичными и в этом случае должны быть далее агрегированы.

Для представления операций по разделению и объединению данных в системе Визограф-1 вводятся визуальные элементы «**Разделитель**» (сплиттер) и «**Агрегатор**» данных.

Сплиттер представляет на визуальной схеме операции разделения данных и изображается пиктограммой в виде усеченного треугольника, направленного справа налево (рис. 2, блок 3). Система Визограф-1 поддерживает некоторые типовые операции разделения данных в виде вертикального (горизонтального) ленточного или блочного представления прямоугольных структур данных (матриц). Как правило, схема разделения данных должна быть согласована с топологией множественного модуля, для которого производится разделение данных. Схема используемого способа разделения данных показывается в виде пиктограммы на изображении сплиттера.

Агрегатор представляет операции объединения данных и изображается пиктограммой в виде усеченного треугольника, направленного слева направо (рис. 2, блок 5). В системе Визограф-1 поддерживается ряд типовых операций объединения данных в виде вертикальной (горизонтальной) или блочной конкатенации получаемых данных и тех или иных способов редуцирования (поэлементного сложения, поиска максимума или минимума и т.п.). Схема используемого способа объединения данных показывается в виде пиктограммы на изображении агрегатора.

4. *Указание параметрической связности модулей.* Во многих случаях между элементами визуальных схем могут существовать параметрические зависимости. Так, например, при матрично-векторном умножении порядок матрицы должен совпадать с размером вектора, при матричном умножении количество столбцов первой матрицы должно соответствовать количеству строк второй матрицы и т.д. Данные зависимости могут фиксироваться в свойствах элементов и визуализироваться пунктирными соединительными линиями при помощи команд контекстного меню[1)] (рис. 2, блоки 1 и 2).

5. *Определение информационного взаимодействия модулей.* Модули в ходе своего выполнения могут обмениваться получаемыми данными. Для представления информационного взаимодействия вводится понятие *информационного порта*. Порты изображаются пиктограммами в виде треугольников, располагаемых на горизонтальных границах модулей (рис. 2, блоки 1 и 2). Одна из вершин треугольника используется для представления направленности передачи данных: треугольник, направленный внутрь модуля, обозначает порт для приема данных, и, соответственно, треугольник, направленный от модуля, обозначает порт для передачи данных (см. рис. 2). Информационное взаимодействие на рис. 2 соответствует передаче введенного в модуле 1 значения порядка матрицы в модуль 2 для определения размера вектора.

Для связывания портов, как и в случае параметров, используются линкеры.

Передача данных между портами может осуществляться в синхронном и асинхронном[2)] режимах. Для визуального выделения более сложного – асинхронного – способа взаимодействия при показе линкеров используются более утолщенные линии для соединения портов.

Информационное взаимодействие экземпляров мультимодулей имеет, как правило, некоторый регулярный характер. Получаемый в результате такого взаимодействия граф (*топология*) представляется на изображении мультимодуля при помощи соответствующей пиктограммы. В системе Визограф-1 поддерживаются топологии «полный граф», «линейка», «кольцо», «решетка» и др.

3. Пример решения системы линейных уравнений

Рассмотрим возможные способы применения системы Визограф при разработке параллельной программы для более сложной задачи – решения системы линейных уравнений. Для решения системы линейных уравнений существует большое количество методов [9]. Принято методы решения систем линейных уравнений разделять на две большие группы – прямые и итерационные. Прямые методы дают точное решение, а итерационные на каждой итерации решение уточняют. У каждого из подходов существуют свои достоинства и недостатки. Наиболее известный среди прямых методов – метод Гаусса. Один из широко известных итерационных методов – метод сопряженных градиентов.

Пусть вначале пользователь системы решил воспользоваться прямым методом решения системы линейных уравнений. В этом случае для реализации метод Гаусса принято разделять на прямой и обратный ход. Данный подход легко реализуется в системе Визограф-1 (см. рис. 3, «Решение системы линейных уравнений (прямой метод)»).

[1)]Постоянная визуализация связности является нецелесообразной, т.к. наличие дополнительный линий может усложнить понимание визуальных схем.

[2)]При асинхронном способе выполнение передачи данных осуществляется без блокировки вычислений, что позволяет снизить временные задержки для информационного взаимодействия.

Рис. 3. Применение системы Визограф для решения системы линейных уравнений

Итерационный метод для реализации требует дополнительной подготовки – генерации начального приближения. Для многих итерационных методов выбор начального приближения является отдельной задачей, так как от выбора часто зависит качество работы методов. Логично вынести данную часть алгоритма в отдельный модуль. Далее можно реализовать метод сопряженных градиентов (см. рис. 3, «Решение системы линейных уравнений (итерационный метод)»).

Важно отметить, что благодаря многоуровневому представлению вычислительных схем пользователь Визограф-1 может в любой момент заменить часть построенного алгоритма, и при этом общая работа алгоритма останется неизменной. Также все разработанные однажды модули могут быть использованы в последующих алгоритмах. Например, в методе сопряженных градиентов необходимо перемножать матрицу на вектор, для этого можно воспользоваться схемой, уже рассмотренной на рис. 2.

Заключение

На текущий момент существует большое количество способов представления параллельных программ. Данная статья рассматривает один из них – представление параллельных программ с помощью набора визуальных схем. В статье рассмотрены основные подходы, используемые в существующих визуальных схемах, и предложен новый набор визуальных схем.

Предлагаемые визуальные схемы позволяют описывать сложные и масштабируемые параллельные алгоритмы в компактном, простом для понимания виде. Для уменьшения сложности разработки параллельных программ важно разделить разработку на этапы. Для поэтапной разработки в визуальных схемах в предложенном подходе поддерживается возможность многоуровневого представления вычислительных схем. Все разработанные модули накапливаются в системе и могут быть многократно использованы. Компактность схем параллельных программ обусловлена наличием ряда введенных компонент декомпозиции и агрегации данных, а также возможностью дублирования модулей.

В дальнейшем планируется расширение системы Визограф-1 для поддержки средств визуального представления данных, автоматизации построения вычислительных схем и для повышения эффективности алгоритмов генерации кода по визуальным схемам.

Список литературы

1. Легалов А.И., Казаков Ф.А., Кузьмин Д.А., Привалихин Д.В. Функциональная модель параллельных вычислений и язык программирования «Пифагор» //URL: http://www.softcraft.ru/parallel/fpp/fppcontent.shtml (дата обращения: 27.06.2010).

2. Browne J.C., Dongarra J., Hyder S.I. et al. Experiences with CODE and HeNCE in Visual Programming for Parallel Computing // IEEE Parallel and Distributed Technology. B. 1994. V. 3. No. 1. P. 75–83.

3. Востокин С.В. Графический метод проектирования параллельных программ с использованием асинхронной событийной модели вычислений // Вестн. Самарского гос. техн. ун-та. Сер. физико-математические науки. 2004. № 30. С. 178–183.

4. Beguelin A. et al. Graphical Development Tools for Network-Based Concurremt Supercomputing // Proc. Supercomputing ‘91, Albuquerque, NM. B. 1994. P. 435–444.

5. Scheidler C., Schäfers L. TRAPPER: A Graphical Programming Environment for Industrial High-Performance Applications // PARLE '93 Parallel Architectures and Languages Europe. Lecture Notes in Computer Science B. 1993. V. 694. P. 403–413.

6. Kacsuk P., Cunha J.C., Dózsa G., Lourenço J., Fadgyas T., Antão T. A graphical development and debugging environment for parallel programs // Parallel Computing. February 1997. V. 22. P. 1747–1770.

7. Жидченко В.В. Программный комплекс моделирования и анализа алгоритмов параллельных вычислений: Дис. к.т.н.: 05.13.18 / Жидченко В.В. [Место защиты: Самарский гос. аэрокосм. ун-т им. С.П. Королева]. Самара, 2007. 189 с.: ил. РГБ ОД, 61 07-5/5304.

8. Котляров Д.В., Кутепов В.П., Маланин В.Н., Панков Н.А. Программные средства поддержки выполнения граф-схемных программ для кластерных систем // Технологии Microsoft в теории и практике программирования: Материалы конф. Н. Новгород, 2006. С.156–159.

9. Самарский А.А., Гулин А.В. Численные методы. М.: Наука, 1989. 432 с.

AN APPROACH TO VISUAL PARALLEL PROGRAM DEVELOPMENT

V.P. Gergel, E.A. Kozinov

The use of environments of visual parallel program analysis and development is one of the ways to reduce the complexity of parallel programming. We describe a new visual environment Visiograph-1, which is under development now. The main concepts and environment potentialities are considered, application examples are presented.

Keywords: visual tools for parallel software development, parallel programming, high performance computing.