УДК 004.4'23, 004.4'24 DOI: 10.14529/cmse150302

АВТОМАТИЗИРОВАННОЕ ПРЕОБРАЗОВАНИЕ ФОРТРАН-ПРОГРАММ, НЕОБХОДИМОЕ ДЛЯ ИХ ЭФФЕКТИВНОГО РАСПАРАЛЛЕЛИВАНИЯ С ПОМОЩЬЮ СИСТЕМЫ САПФОР[1]

Н.А. Катаев, А.А. Буланов

Автоматическое отображение последовательных программ на вычислительные системы с распределенной памятью может потребовать предварительного преобразования программ, ориентированного на данный класс систем. Использование системы САПФОР для распараллеливания прикладных программ позволило выделить преобразования, выполнение которых может быть автоматизировано. В статье представлены преобразования, повышающие возможность эффективного распараллеливания программ за счет устранения причин, препятствующих распараллеливанию циклов. Выполнение данных преобразований позволило автоматизировать получение последовательной реализации, эффективно отображаемой на современные кластеры автоматически распараллеливающим компилятором системы, для задачи гидродинамики.

Ключевые слова: распараллеливание, автоматизация, преобразование последовательных программ, Фортран.

Введение

Основной целью распараллеливания прикладной программы является ее эффективное выполнение на современных вычислительных системах. Отличительной чертой программирования для таких систем является совместное использование различных технологий параллельного программирования. Сложность применения данных технологий подчеркивает актуальность исследований в области автоматизации распараллеливания программ.

К автоматически распараллеливающим компиляторам можно отнести PLUTO [1], Par4all [2], Parallware [3], задать опции для автоматического распараллеливания можно при использовании компиляторов Intel. Основным языком программирования является C. Язык Fortran поддерживается в компиляторах Par4all (только Fortran 77) и Intel. Следует отметить, что компиляторы Intel и Parallware являются коммерческими. Распараллеливание ориентировано на системы с общей памятью, компилятор Par4all также обеспечивают распараллеливание для графических ускорителей. Считается, что автоматическое распараллеливание для систем с распределенной памятью пока не достижимо [4].

К автоматизированным системам распараллеливания относятся САПФОР [5], Intel Parallel Studio [6], Parawise [7], ДВОР [8]. Особенностью системы САПФОР является использование автоматически распараллеливающего компилятора – взаимодействие с пользователем осуществляется на этапе подготовки последовательной программы к автоматическому распараллеливанию. Пользователь, во-первых, предоставляет системе информацию о свойствах программы, которую не удалось получить автоматическими

[1] Статья рекомендована к публикации программным комитетом Международной научной конференции «Параллельные вычислительные технологии – 2015».

средствами анализа (статическими и динамическими). Во-вторых, он участвует в преобразовании последовательной программы в последовательную программу с целью устранения проблем, выявленных системой и мешающих автоматическому распараллеливанию.

Входным языком системы САПФОР является Fortran, результат распараллеливания представляет собой программу на языке Fortran OpenMP, Fortran DVM/OpenMP или Fortran DVMH. Модель DVMH [9] является расширением модели DVM [10] и обеспечивает распараллеливание программы на кластер с ускорителями. Основными компонентами САПФОР являются анализаторы последовательных программ (статические и динамические), блоки преобразования последовательных программ в параллельные программы (эксперты), диалоговая оболочка для взаимодействия с пользователем, генератор кода, создающий на основе принятых экспертом решений параллельную версию программы.

Целью данной работы является автоматизация преобразований последовательной программы, выполняемых на этапе подготовки ее к автоматическому распараллеливанию экспертом системы САПФОР.

Статья организована следующим образом. В первом разделе описаны три подхода к преобразованию программ, используемых в системе САПФОР. Во втором разделе рассмотрены примеры преобразований последовательных программ. Указанные преобразования могут быть выполнены системой САПФОР автоматически с целью устранения проблем, препятствующих эффективному распараллеливанию программы. В третьем разделе описан итерационный процесс распараллеливания двумерной задачи «Каверна» о течении несжимаемой жидкости или слабосжимаемого газа около прямоугольной выемки. Каждая итерация включает в себя попытку автоматического построения вариантов распараллеливания программы и автоматическое применение преобразований, устраняющих в последовательной программе проблемы, мешающие ее эффективному распараллеливанию. В четвертом разделе кратко рассмотрена программная реализация нового компонента системы САПФОР – преобразователя. В заключении суммируются результаты проведенного исследования и рассматриваются направления дальнейших работ.

1. Подходы к преобразованию программ

Большинство компиляторов использует заранее определенные последовательности фаз для оптимизации и распараллеливания программ. Данные последовательности могут сильно различаться в зависимости от цели проводимых оптимизаций: по памяти, по времени выполнения на одном процессоре или на нескольких. При распараллеливании необходимо учитывать архитектуры вычислительных систем, на которых предполагается выполнение программы. При этом оптимальность таких последовательностей сильно зависит от прикладной программы, а поиск наилучшей последовательности для конкретной программы сильно затруднен из-за огромных размеров пространства возможных оптимизационных последовательностей [11].

Распараллеливание в системе САПФОР представляет собой итерационный процесс, это обеспечивает возможность выбора трансформаций для решения проблем, мешающих эффективному распараллеливанию последовательной программы экспертом системы

САПФОР, по мере их возникновения. Эксперт системы выявляет особенности программы, влияющие на качество ее распараллеливания (см. пример на рис. 1).

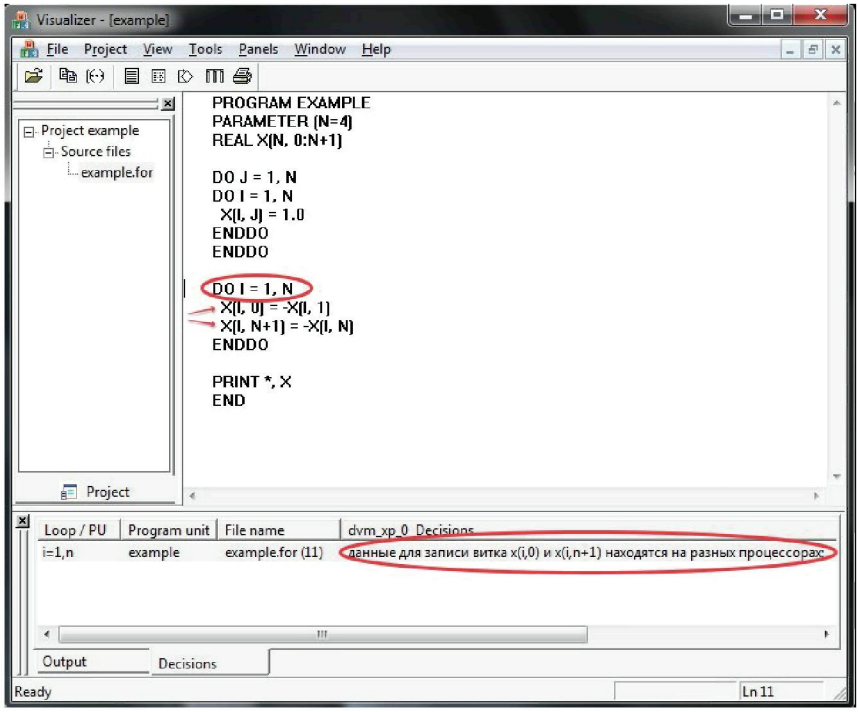

Рис. 1. Пример проблемы, мешающей эффективному отображению программы на параллельные компьютеры с распределенной памятью

Предлагается три подхода к преобразованию последовательной программы.

Первый подход состоит в автоматическом выборе способа устранения проблем. Решение о возможности устранения совокупности проблем принимается на основе удовлетворения каждой проблемы некоторому шаблону, описывающему ее решение. Процесс устранения проблем сводится к применению выбранных шаблонов. Каждая проблема рассматривается независимо от других, и в том случае если решения двух проблем конфликтуют между собой, необходимо участие пользователя для разрешения конфликта.

Второй подход состоит в активном вовлечении пользователя в процесс выбора решений выявленных проблем. Система предоставляет пользователю набор элементарных преобразований, для которых возможно автоматическое выполнение (например, разворачивание цикла, разделение цикла на несколько циклов, подстановку переменных и др.). Пользователь выбирает преобразование и указывает требующиеся характеристики, описывающие данное преобразование. Перед осуществлением преобразования система проверяет его допустимость. Допускается принудительное выполнения преобразования, если пользователь уверен в его допустимости, а требование консервативности средств анализа ограничивает его выполнение.

Если ни один из подходов не может быть применен, пользователь может вручную преобразовать исходный код программы, руководствуясь описаниями проблем, подготовленными экспертом системы САПФОР.

2. Автоматический выбор и применение шаблона

Использование системы САПФОР для распараллеливания прикладных программ [12–14]. позволило выделить проблемы, препятствующие эффективному распараллеливанию, для которых возможен автоматический поиск решения.

Одним из типов проблем является наличие в последовательной программе гнезд циклов с зависимостями по данным между различными итерациями циклов. В качестве шаблона решения проблем данного типа применяется разбиение гнезда циклов. Используемый алгоритм основан на алгоритме, предложенном в [15]. Чтобы цикл удовлетворял требованиям применимости алгоритма, системой САПФОР может выполняться автоматическое переупорядочивание операторов в цикле. В операторах, распределяемых по разным циклам, могут использоваться общие данные, вычисляемые внутри разделяемого цикла. В этом случае требующиеся вычисления либо будут продублированы в каждом из циклов, либо в программу будут добавлены вспомогательные массивы, в которых перед выполнением распределенных по циклам операторов будут размещены требующиеся данные.

Разбиение гнезда циклов также применяется для устранения проблемы, вызванной записью значений в разные элементы одного массива на одной итерации цикла (см. пример на рис. 1). Данная проблема характерна для отображения программ на системы с распределенной памятью. Перед выполнением вычислений данные должны быть распределены по процессорам таким образом, чтобы максимально снизить количество обменов. При параллельном выполнении итераций цикла процессорами системы необходимо, чтобы каждый процессор обладал необходимой ему частью распределенных данных. Для программы из примера на рис. 1 невозможно эффективно распределить данные и распараллелить выделенный цикл, несмотря на отсутствие в нем зависимостей по данным. На это указывает описание проблемы, полученное от эксперта. Результат успешного распараллеливания преобразованной программы приведен на рис. 2.

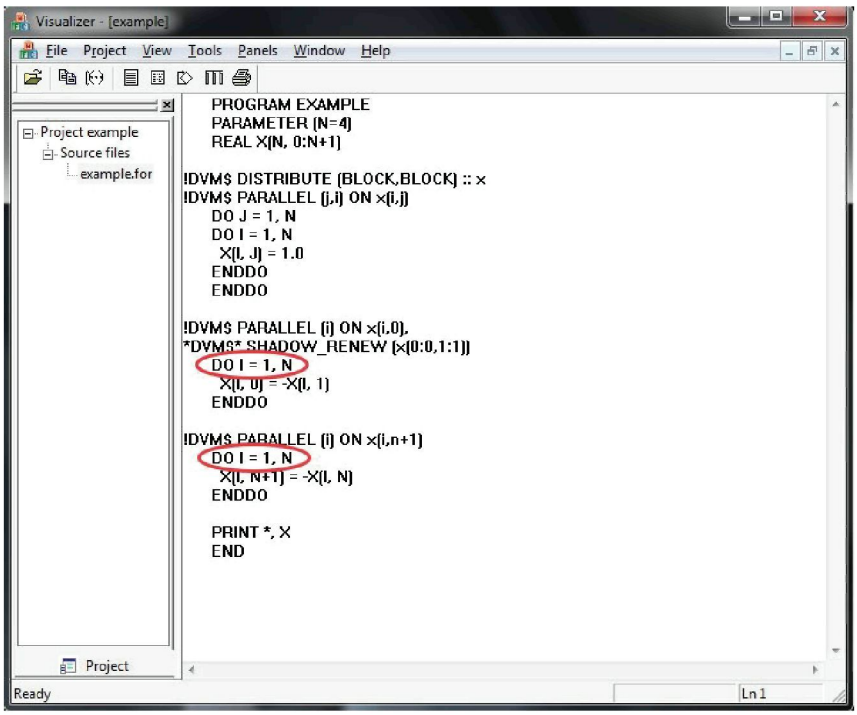

Рис. 2. Распараллеливание программы из примера на рис. 1 после разбиения цикла

Разные массивы, к элементам которых выполняется доступ на одной итерации цикла, должны быть соответствующим образом выравнены. Это накладывает ограничения на возможные варианты распределения массивов по процессорам вычислительной системы. В случае если эксперту системы не удается выполнить автоматическое распределение данных, чтобы увеличить количество возможных вариантов распределения, выполняется максимально возможное разбиение циклов программы. В один цикл попадают операторы, в которых выполняется запись в одни и те же элементы одного и того же массива. После распараллеливания программы, выполняется объединение циклов на основе информации о выравнивании массивов и распределении вычислений, полученной от эксперта.

3. Распараллеливание прикладной задачи

Предложенный подход к автоматизации получения последовательной программы, эффективно отображаемой на кластер, был протестирован на двумерной задаче Каверна о течении несжимаемой жидкости или слабосжимаемого газа около прямоугольной выемки [16]. Ручное преобразование данной задачи описано в [12].

Автоматическое преобразование потребовало проведения 6 итераций.

Запуск эксперта системы САПФОР на исходной не преобразованной программе выявил следующую проблему: «надо изменить оператор ввода-вывода на строке 520». В результате преобразования оператор вывода распределенного массива был заменен на оператор вывода скалярных переменных, содержащих предварительно скопированные элементы массива.

Следующий запуск эксперта выявил:

- четыре проблемы вида *«данные для записи витка ro(i,0) и ro(i, ny1) находятся на разных процессорах»*. В данном случае в указанных циклах не было зависимостей по данным, но на одной итерации выполнялась запись в разные элементы одного массива, указанные экспертом. Данная проблема аналогична проблеме, приведенной на рис. 1. При проведении преобразований было обнаружено, что в каждом из циклов аналогичным образом организована запись в пять разных массивов. Так как на момент преобразований не была доступна информация о выравнивании данных массивов, каждый цикл был разбит максимальным образом на десять циклов.
- пять проблем вида *«данные для записи витка ro1(:, j) находятся на разных процессорах»*. Данная проблема указывает, что на итерации цикла записывается целиком измерение массива и свидетельствует о том, что не был распараллелен один из циклов гнезда. Причиной этого является не тесная вложенность циклов. В результате преобразования во внутренний цикл был внесен инвариант, расположенный между заголовками циклов.

Следующий запуск эксперта выявил четыре проблемы вида «данные для записи витка ro1(i, j) и ro(i,j+1) находятся на разных процессорах». В указанных циклах были зависимости по данным, которые удалось устранить введением четырех временных массивов (по одному для каждого цикла) и разбиением каждого из циклов на два.

Следующий запуск эксперта выявил четыре проблемы вида «часть гнезда, данные для записи витка tmp1(i+1,j) и e1(i,j) находятся на разных процессорах». Данные проблемы указывали на циклы из предыдущего запуска. Причиной их возникновение стало

использование введенных временных массивов. Для решения данной проблемы необходимо разбить указанные циклы. Вынесение временного массива в отдельный цикл не решает данную проблему, так как в этом случае должен быть заведен новый временный массив с такой же структурой и операциями записи данных. Вынесению второго массива препятствовали зависимости по данным, которые удалось устранить введением еще четырех временных массивов (по одному для каждого цикла). Затем каждый из четырех циклов был разбит на два.

Следующий запуск эксперта выявил две проблемы: «часть гнезда, измерение 1 массива ro1 не распределено». Причиной возникновения данных проблем является вызов внутри цикла оператора вывода данных. Устранить данную проблему невозможно, но она не влияет на распараллеливание остальной части программы. В результате данного запуска программа была успешно распараллелена.

В результате разбиения циклов было добавлено 36+4+4=44 новых цикла. На следующем шаге количество циклов было уменьшено за счет объединения циклов, выполненного на основе информации о выравнивании массивов и распределении вычислений, полученной от эксперта. Количество добавленных циклов было сокращено до 8.

4. Программная реализация

Выбор подходящих шаблонов и выполнение определяемых ими преобразований реализован в виде отдельного компонента – преобразователя системы САПФОР. Преобразователь взаимодействует с другими компонентами и пользователем системы двумя способами.

Таблица

Пример специальных комментариев применяемых в системы САПФОР

Специальный комментарий	**Использование**
!PRG INDEPENDENT (<variable-name >)	Позволяет указать на отсутствие межитерационных зависимостей по данным в цикле
!PRG PRIVATE (<variable-name>) !PRG FIRST_PRIVATE (<variable-name>) !PRG LAST_PRIVATE (<variable-name>)	Позволяют указать приватные (приватные по входу или по выходу) переменные в цикле
!PRG FLOW|ANTI|OUTPUT <from> <to> [<dist>]	Позволяет указать наличие зависимости по данным между двумя операторами и расстояние указанной зависимости
!PRG REDUCTION (<variable-name> (<op>) [, <array-name>, <index-number>] <op> ::= SUM|PRODUCT|MAX|MIN|AND|OR|EQV|NEQV|MAXLOC|MINLOC	Позволяет указать редукционные переменные для цикла. Опциональный параметр <array-name> используется в случае редукции MAXLOC или MINLOC

Во-первых, через базу данных (внутреннее представление) системы САПФОР.

Во-вторых, через специальные комментарии с префиксом !PRG, добавляемые в исходный код программы. Данные комментарии позволяют задавать дополнительные свойства программы, которые не удалось выявить анализаторам системы САПФОР, или уточнить описание проблемы, возникшей при распараллеливании и выделенной экспер-

том системы САПФОР, для более эффективного подбора шаблона ее решения. Пример комментариев, задаваемых перед циклами программы, приведен в таблице.

Преобразователь написан на языке программирования Си++ с использованием библиотеки Sage++ [17] для выполнения преобразований исходного кода программы.

Заключение

Эффективное автоматическое распараллеливание последовательных программ, во многих случаях, невозможно без выполнения их преобразования. Выполнение в процессе распараллеливания заранее фиксированной последовательности фаз анализа и преобразований, не учитывает особенности каждой конкретной программы. Проведенное таким образом автоматическое распараллеливание часто оказывается неэффективным. Итерационный процесс распараллеливания программ, поддерживаемый системой автоматизированного распараллеливания САПФОР, позволяет применять только те преобразования, которые необходимы для устранения проблем, препятствующих распараллеливанию.

Выполняемые перед распараллеливанием преобразования не выводят программу из класса последовательных программ, оставляя ее понятной для пользователя, детально незнакомого с особенностями параллельного программирования. Выполнение преобразований может выполняться в автоматическом или полуавтоматическом режимах.

Реализация данных возможностей в системе САПФОР позволила автоматизировать получение последовательных реализаций программ, эффективно отображаемых на современные кластеры автоматически распараллеливающим компилятором системы САПФОР, и была проверена на прикладной задаче моделирующей течение несжимаемой жидкости или слабосжимаемого газа около прямоугольной выемки.

Дальнейшие исследования будут направлены на расширение множества автоматически выполняемых преобразований и совместное использование в системе САПФОР трех описанных подходов к преобразованию программ: автоматическое устранение проблем, мешающих распараллеливанию программы, автоматизированное выполнение преобразований под руководством пользователя и полностью ручное преобразование программы.

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 14-01-00109 а.

Литература

1. PLUTO – An automatic parallelizer and locality optimizer for multicores. URL: http://pluto-compiler.sourceforge.net/ (дата обращения 28.11.2014).
2. Par4All – Par4All 1.4.5 documentation. URL: http://www.par4all.org/ (дата обращения 28.11.2014).
3. Source-to-Source Parallelizing Compiler – Appentra. URL: http://www.appentra.com/ (дата обращения 28.11.2014).
4. Бахтин, В.А. Автоматическое распрраллеливание последовательных программ для многоядерных кластеров. / В.А. Бахтин, М.С. Клинов, В.А. Крюков, Н.В. Поддерюгина // Научный сервис в сети Интернет: суперкомпьютерные центры и задачи: Труды Международной научной конференции (Новороссийск, 20–25 сентября 2010 г.). — М.: Изд-во МГУ, 2010. — С. 12–15.

5. Бахтин, В.А. Диалог с программистом в системе автоматизации распараллеливания САПФОР. / В.А. Бахтин, И.Г. Бородич, Н.А. Катаев, М.С. Клинов, Н.В. Ковалева, В.А. Крюков, Н.В. Поддерюгина // Вестник Нижегородского университета им. Н.И. Лобачевского. — 2012 — № 5 (2). — С. 242–245.
6. Intel Parallel Studio. URL: http://software.intel.com/en-us/intel-parallel-studio-home (дата обращения 28.11.2014) .
7. Automatic Parallelization for Multi-processor/Multi-cores systems URL: http://www.parallelsp.com/ (дата обращения 28.11.2014).
8. Юрушкин, М.В. Диалоговый высокоуровневый оптимизирующий распараллеливатель (ДВОР). / М.В. Юрушкин, В.В. Петренко, Б.Я Штейнберг, Е.В. Алымова, А.А, Абрамов, А.П. Баглий, С.А. Гуда, Д.В. Дубров, Е.Н. Кравченко, Р.И. Морылев, З.Я. Нис, С.В. Полуян, И.С. Скиба, В.Н. Шаповалов., О.Б. Штейнберг, Р.Б. Штейнберг // Научный сервис в сети Интернет: суперкомпьютерные центры и задачи: Труды Международной научной конференции (Новороссийск, 20 – 25 сентября 2010 г.). — М.: Изд-во МГУ, 2010. — С. 71–75.
9. Бахтин, В.А. Расширение DVM-модели параллельного программирования для кластеров с гетерогенными узлами. / В.А. Бахтин, М.С. Клинов, В.А. Крюков, Н.В. Поддерюгина, М.Н. Притула, Ю.Л. Сазанов // Вестник Южно-Уральского государственного университета. Серия «Математическое моделирование и программирование». — 2012 — № 18 (277), вып. 12. — С 82–92.
10. Коновалов, Н.А Fortran DVM — язык разработки мобильных параллельных программ. / Н.А. Коновалов, В.А. Крюков, С.Н. Михайлов, А.А. Погребцов // Программирование. — 1995. — № 1. — С. 49–54.
11. Almagor, L. Finding effective compilation sequences. / L. Almagor, K.D. Cooper, A. Grosul, T.J. Harvey, S.W. Reeves, D. Subramanian, L. Torczon, T. Waterman. // Proceedings of the ACM SIGPLAN/SIGBED 2004 Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES'04) (Washington, DC, 11–13 June 2004). — New York: NY, USA. — Vol. 39, Issue 7. — P. 231–239. DOI: 10.1145/997163.997196.
12. Бахтин, В.А. Распараллеливание с помощью DVM-системы некоторых приложений гидродинамики для кластеров с графическими процессорами. / В.А. Бахтин, И.Г. Бородич, Н.А. Катаев, М.С. Клинов, В.А. Крюков, Н.В. Поддерюгина, М.Н. Притула, Ю.Л. Сазанов // Научный сервис в сети Интернет: поиск новых решений: Труды Международной суперкомпьютерной конференции (Новороссийск, 17–22 сентября 2012 г.). — М.: Изд-во МГУ, 2012. — С. 444–450.
13. Катаев, Н.А. Преобразования последовательных программ при их распараллеливании с помощью системы САПФОР. / Н.А. Катаев, М.С. Клинов, Н.В, Поддерюгина. // Параллельные вычислительные технологии (ПаВТ'2013): Труды международной научной конференции (Челябинск, 1–5 апреля 2013 г.). — Челябинск: Издательский центр ЮУрГУ, 2013. — С. 387–393.
14. Бахтин, В.А Автоматическое отображение программ на языке Фортран на кластеры с графическими процессорами. / В.А. Бахтин, М.С. Клинов, А.С. Колганов, В.А. Крюков, Н.В. Поддерюгина, М.Н. Притула // Вестник Южно-Уральского государственного университета. Серия «Вычислительная математика и информатика». — 2014. — Т. 3, № 3. — С. 86–96. DOI: 10.14529/cmse140305.

15. Штейнберг, Б.Я. Разбиение циклов для исполнения на суперкомпьютере с архитектурой перестраиваемого конвейера. / Б.Я. Штейнберг // Искусственный интеллект. — 2002. — № 3. — С. 331–338.
16. Давыдов, А.А. Моделирование течений несжимаемой жидкости и слабосжимаемого газа на многоядерных гибридных вычислительных системах. / А.А Давыдов, Б.Н. Четверушкин, Е.В. Шильников // Вычислительная математика и математическая физика. — 2010. — Т. 50, № 12. — С. 2275–2284.
17. pC++/Sage++ Home Page. URL: http://www.extreme.indiana.edu/sage/ (дата обращения: 01.12.2012).

Катаев Никита Андреевич, научный сотрудник, Институт прикладной математики им. М.В. Келдыша РАН (Москва, Российская Федерация), kataev_nik@mail.ru.

Буланов Артем Андреевич, студент, факультет Вычислительной Математики и Кибернетики, Московский государственный университет им. М.В. Ломоносова (Москва, Российская Федерация), bulanov.artyom@gmail.com.

Поступила в редакцию 10 апреля 2015 г.

Bulletin of the South Ural State University
Series "Computational Mathematics and Software Engineering"
2015, vol. 4, no. 3, pp. 13–23

DOI: 10.14529/cmse150302

AUTOMATED TRANSFORMATION OF FORTRAN PROGRAMS ESSENTIAL FOR THEIR EFFICIENT PARALLELIZATION THROUGH SAPFOR SYSTEM

N.A. Kataev, Keldysh Institute of Applied Mathematics Russian Academy of Sciences (Moscow, Russian Federation) kataev_nik@mail.ru,

A.A. Bulanov, Lomonosov Moscow State University (Moscow, Russian Federation) bulanov.artyom@gmail.com

Automatic mapping of sequential programs on height performance computers with distributed memory may require preliminary transformation of programs oriented to this class of systems. Using the SAPFOR system for parallelization of applications allowed to explore transformations which can be executed in an automated way. The paper presents transformations that increase the possibility of efficient parallelization of programs by eliminating the reasons that prohibit the parallel execution of loops. Implementation of these transformations allowed to automate the obtainment of serial hydrodynamic program, which may be efficiently mapped on modern clusters by the automatically parallelizing compiler included in the SAPFOR.

Keywords: parallelization, automation, transformation of serial programs, Fortran.

References

1. PLUTO – An automatic parallelizer and locality optimizer for multicores. URL: http://pluto-compiler.sourceforge.net/ (accessed: 28.11.2014).
2. Par4All – Par4All 1.4.5 documentation. URL: http://www.par4all.org/ (accessed: 28.11.2014).
3. Source-to-Source Parallelizing Compiler – Appentra. URL: http://www.appentra.com/ (accessed: 28.11.2014).
4. Bakhtin V.A, Klinov M.S., Krukov V.A., Podderugina N.V. Avtomaticheskoe rasprrallelivanie posledovatel'nyh programm dlja mnogojadernyh klasterov [Automatic parallelization of sequential programs for multi-core clusters.]. Nauchnyj servis v seti Internet: superkomp'juternye centry i zadachi: Trudy Mezhdunarodnoj nauchnoj konferencii (Novorossijsk, 20 – 25 sentjabrja 2010 g.) [Scientific service on the Internet: supercomputer centers and tasks: Proceedings of the International Scientific Conference (Novorossijsk, Russia, September, 20 – 25, 2010)]. — Moscow, Moscow University Press, 2010. P. 12–15.
5. Bahtin V.A., Borodich I.G., Kataev N.A., Klinov M.S., Kovaleva N.V., Krukov V.A., Podderugina N.V. Dialog s programmistom v sisteme avtomatizacii rasparallelivanija SAPFOR [Interaction with the programmer in the system for automation parallelization SAPFOR]. Vestnik Nizhegorodskogo universiteta im. N.I. Lobachevskogo [Vestnik of Lobachevsky State University of Nizhni Novgorod]. 2012 No. 5 (2). P. 242–245.
6. Intel Parallel Studio. URL: http://software.intel.com/en-us/intel-parallel-studio-home (accessed: 28.11.2014).
7. Automatic Parallelization for Multi-processor/Multi-cores systems URL: http://www.parallelsp.com/ (accessed: 28.11.2014).
8. Jurushkin M.V., Petrenko V.V., Shtejnberg B.Ja., Alymova E.V.,Abramov A.A., Baglij A.P., Guda S.A., Dubrov D.V., Kravchenko E.N., Morylev R.I., Nis Z.Ya., Polujan S.V., Skiba I.S., Shapovalov V.N., Shtejnberg O.B., Shtejnberg R.B. Dialogovyj vysokourovnevyj optimizirujushhij rasparallelivatel' (DVOR) [Interactive High-level Optimizing Parallelizer (IHOP). Nauchnyj servis v seti Internet: superkomp'juternye centry i zadachi: Trudy Mezhdunarodnoj nauchnoj konferencii (Novorossijsk, 20–25 sentjabrja 2010 g.) [Scientific service on the Internet: supercomputer centers and tasks: Proceedings of the International Scientific Conference (Novorossijsk, Russia, September, 20–25, 2010)]. — Moscow, Moscow University Press, 2010. P. 71–75.
9. Bakhtin V.A, Klinov M.S., Krukov V.A., Podderugina N.V., Pritula M.N., Sazanov Yu.L. Rasshirenie DVM-modeli parallel'nogo programmirovaniya dlya klasterov s geterogennymi uzlami [Extension of the DVM-model of parallel programming for clusters with heterogeneous nodesExtension of the DVM-model of parallel programming for clusters with heterogeneous nodes]. Vestnik Yuzho-Uralskogo gosudarstvennogo universiteta. Seriya "Matematicheskoe modelirovanie i programmirovanie" [Bulletin of South Ural State University. Series: Mathematical Modeling, Programming & Computer Software]. 2012 No. 18 (277), Issue 12. P 82–92.
10. Konovalov N.A., Krukov V.A, Mikhajlov S.N., Pogrebtsov A.A. Fortan DVM: a Language for Portable Parallel Program Development // Programming and Computer Software. 1995. Vol. 21, No. 1. P. 35–38.
11. Almagor L., Cooper K.D., Grosul A., Harvey T.J., Reeves S.W., Subramanian D., Torczon L., Waterman T. Finding effective compilation sequences // Proceedings of the

ACM SIGPLAN/SIGBED 2004 Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES'04) (Washington, DC, 11–13 June 2004). New York: NY, USA. Vol. 39, Issue 7. P. 231–239. DOI: 10.1145/997163.997196.

12. Bakhtin V.A, Borodich I.G, Kataev N.A., Klinov M.S., Krukov V.A., Podderugina N.V., Pritula M.N., Sazanov Yu.L. Rasparallelivanie s pomoshh'yu DVM-sistemy nekotoryx prilozhenij gidrodinamiki dlya klasterov s graficheskimi processorami [Parallelization using the DVM-system some applications hydrodynamics for clusters with GPUs]. Nauchnyj servis v seti Internet: poisk novyx reshenij: Trudy Mezhdunarodnoj nauchnoj konferencii (Novorossijsk, 17–22 sentjabrja 2012) [Scientific service on the Internet: search for new solutions: Proceedings of the International Scientific Conference (Novorossijsk, Russia, September, 17–22, 2012)]. — Moscow, Moscow University Press, 2012. P. 444–450.
13. Kataev N.A., Klinov M.S., Podderugina N.V. Preobrazovaniya posledovatel'nyx programm pri ix rasparallelivanii s pomoshh'yu sistemy SAPFOR [Transformations of sequential programs during programs parallelization with the help of SAPFOR.]. Parallelnye vychislitelnye tekhnologii (PaVT'2010): Trudy mezhdunarodnoj nauchnoj konferentsii (Chelyabinsk, 1–5 aprelya 2013) [Parallel Computational Technologies (PCT'2013): Proceedings of the International Scientific Conference (Chelyabinsk, Russia, April, 1–5, 2013)]. Chelyabinsk, Publishing of the South Ural State University, 2013. P. 387–393.
14. Bahtin V.A., Klinov M.S., Kolganov A.S., Krukov V.A., Podderugina N.V., Pritula M.N. Avtomaticheskoe otobrazhenie programm na jazyke Fortran na klastery s graficheskimi processorami [Automatic mapping of Fortran programs on clusters with accelerators]. Vestnik Yuzho-Uralskogo gosudarstvennogo universiteta. Seriya: "Vychislitel'naja matematika i informatika" [Bulletin of South Ural State University. Series: Computational Mathematics and Software Engineering]. 2014. Vol. 3, No. 3. P. 86–96. DOI: 10.14529/cmse140305.
15. Shtejnberg B.Ja. Razbienie ciklov dlja ispolnenija na superkomp'jutere s arhitekturoj perestraivaemogo konvejera [Splitting of loops for execution on a supercomputer with the architecture of configurable pipeline]. Iskusstvennyj intellekt [Artificial intelligence]. 2002. No. 3. P. 331–338.
16. Davydov A.A., Chetverushkin B.N., Shilnikov E.V. Modelirovanie techenij neszhimaemoj zhidkosti i slaboszhimaemogo gaza na mnogojadernyh gibridnyh vychislitel'nyh sistemah [Simulating flows of incompressible and weakly compressible fluids on multicore hybrid computer systems]. // Vychislitel'naja matematika i matematicheskaja fizika [Computational Mathematics and Mathematical Physics]. 2010, Vol. 50, No. 12. P. 2157–2165.
17. pC++/Sage++ Home Page. URL: http://www.extreme.indiana.edu/sage/ (accessed: 01.12.2012).

Received April 10, 2015.