результаты. Это обусловлено тем, что для таких задач эффект «узкого» места оригинальной схемы не достигается.

Список литературы

1. Стронгин Р.Г. Численные методы в многоэкстремальных задачах. – М.: Наука, 1978.

2. Strongin R.G., Sergeyev Y.D. Global Optimization with non-convex constraints: Sequential and parallel algorithms. Kluwer Academic Publishers. – Dordrecht, 2000.

3. Городецкий С.Ю., Гришагин В.А. Нелинейное программирование и многоэкстремальная оптимизация. – Н. Новгород: Изд-во Нижегород. гос. ун-та, 2007.

4. Sergeyev Y.D., Grishagin V.A. Parallel asynchronous global search and the nested optimization scheme. // J. Comput. Anal. Appl. – 2001. – Vol. 3, No. 2. – P. 123–145.

5. Гергель В.П. Теория и практика параллельных вычислений. – М.: Интернет-университет информационных технологий; БИНОМ. Лаборатория знаний, 2007.

6. Гергель А.В., Гнатюк Д.В. Адаптивные параллельные алгоритмы для многомерной многоэкстремальной оптимизации // Высокопроизводительные параллельные вычисления на кластерных системах: материалы конф. – Н. Новогород, 2009. С. 92–96.

В.П. Гергель, Д.В. Ахматнуров, А.В. Калачев, М.С. Царев

Нижегородский государственный университет им. Н.И. Лобачевского

УЧЕБНО-ИССЛЕДОВАТЕЛЬСКАЯ БИБЛИОТЕКА ПАРАЛЛЕЛЬНЫХ МЕТОДОВ ДЛЯ СИСТЕМ С РАЗДЕЛЕННОЙ ОБЩЕЙ ПАМЯТЬЮ

На протяжении многих лет одной из наиболее острых проблем эффективного использования потенциала компьютерной техники является трудоемкость разработки программного обеспечения. Сложность разработки программного обеспечения

связана с большим разнообразием компьютерных систем и их непрерывным совершенствованием. При появлении новых вычислительных систем часто приходится разрабатывать заново большое количество алгоритмов. К сожалению, в отличие от компьютерного оборудования технологии разработки программ совершенствуются существенно медленнее. При разработке параллельных программ следует учитывать особые типы возможных ошибок, такие как тупики и гонки данных. Данные типы ошибок часто требуют больших временных затрат на выявление и отладку. Следовательно, для параллельных вычислительных систем проблема трудоемкости разработки программ особенно критична. Одним из выходов из столь сложной ситуации может быть создание языков параллельного программирования.

В настоящее время для разработки параллельных программ наиболее распространено использование технологий OpenMP и MPI. Позволяя в целом создавать высокоэффективные параллельные программы, данные технологии, являются средствами низкоуровневой разработки – фактически, эти технологии являются «ассемблерами» параллельного программирования. В связи с этим многими исследователями предпринимаются попытки создания новых – параллельных – языков программирования, которые помогли бы снизить трудоемкость разработки параллельных программ. В этом направлении представляет интерес модель вычислительных систем с разделенной глобальной адресуемой памятью (*partitionedglobaladdress*space, PGAS). В рамках данной модели память многопроцессорной системы образуется из памяти отдельных процессоров и является глобально-адресуемой.

В работе представлен ряд языков параллельного программирования, созданных на основе модели PGAS. К данной группе языков относятся Chapel и X10, разработанные в рамках программы HPCS (High Productivity Computing Systems), и язык Co-Array Fortran (CAF). Далее дается краткая характеристика этих языков, рассматривается разработанная библиотека параллельных методов ParaLibPGAS и приводятся результаты экспериментов.

Общая характеристика исследуемых языков программирования. Chapel – это новый язык параллельного программирования, разрабатываемый компанией Cray (http://chapel.cray.com). Основу модели вычислительной системы в языке представляет понятие *locale*, являющееся абстракцией вычислительного элемента системы. Locale содержит обрабатывающее устройство и память. Задачи, исполняемые на locale, имеют однородный доступ к локальной памяти, доступ к памяти других locale происходит более медленно.

X10 разрабатывается при поддержке компании IBM и находится сейчас в стадии активного развития (http://x10-lang.org). Синтаксис языка X10 создан на основе языка программирования Java. Для представления отдельного вычислительного узла высокопроизводительной системы в X10 используется понятие *place*. Place может рассматриваться как абстракция мультипроцессора с общей памятью (SMP-узла). Выполняемая последовательность действий в X10-программе определяется как активность (*activity*), выполняемая исполнителем. На одном исполнителе могут выполняться одновременно несколько активностей.

Co-Array Fotran(CAF) – расширение языка Fortran 95/2003 для параллельного программирования. CAF разработан Robert Numrich и John Reid в 1990-х годах (http://www.co-array.org). В качестве основной задачи, поставленной при разработке языка, было внесение минимально возможных расширений Fortran, достаточных для организации параллельных вычислений. В основе подхода схема SPMD (Single-Program-Multiple-Data). В результате разрабатывается один экземпляр CAF-программы, который далее копируется необходимое количество раз. Копии CAF-программы могут выполняться параллельно и обрабатывают свои «локальные» данные. Данные, к которым требуется доступ из разных копий CAF-программы, должны быть представлены в виде *co-arrays*-массивов. Передача данных между копиями программы осуществляется только при помощи явно указываемых действий.

Библиотека параллельных методов. Учебно-исследовательская библиотека параллельных методов ParaLibPGAS разработана для исследования эффективности языков про-

граммирования на основе модели PGAS (Chapel, X10, CAF). Библиотека содержит параллельные алгоритмы для решения широкого класса задач вычислительной математики (матричная алгебра, решение дифференциальных уравнений в частных производных и др.).Архитектура библиотеки представлена на рис. 1.

Для реализации интерфейса и служебных модулей библиотеки был использован язык Java. Каждый исполняемый файл на уровне библиотеке «обёртывается» в специально созданный класс, который реализует интерфейс для запуска алгоритма на исполнения. Преимущество данного подхода заключается в том, что класс-обёртка знает все ограничения, которые накладываются на метод и на входные данные и может проверить корректность этих данных ещё до запуска алгоритма на исполнение.

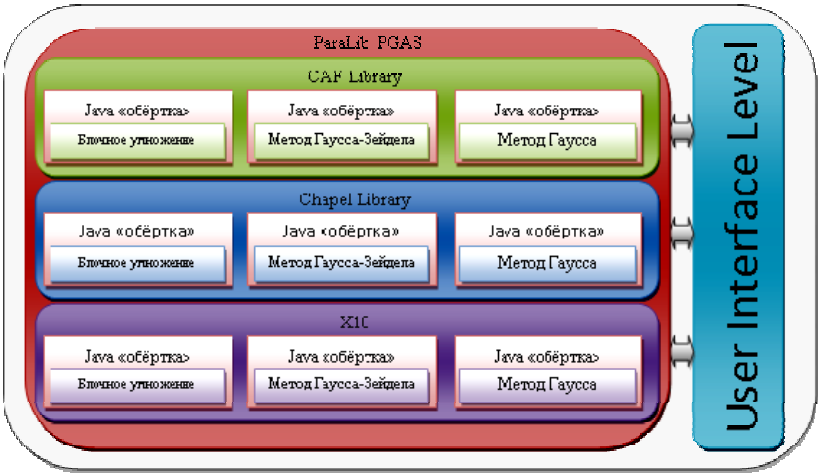

Рис. 1. Архитектура библиотеки ParaLibPGAS

Все методы для одного языка объединяются в классы библиотеки, которые предоставляют набор функций для работы с методами. Данный подход позволяет обеспечить расширяемость данной системы. Для добавления нового метода (или даже языка) необходимо добавить один новый класс в исходный код, всё остальное система будет делать автоматически (рис. 2).

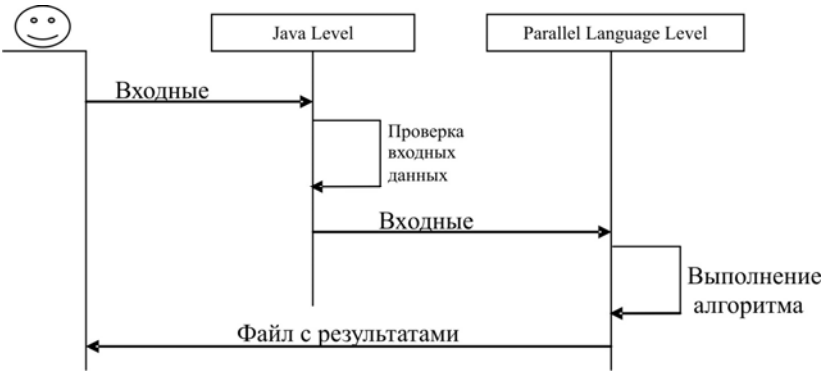

Рис. 2. Общая схема выполнения метода библиотеки

Результаты вычислительных экспериментов. Вычислительные эксперименты проводились на компьютере с процессором Intel Core 2 Quad CPU Q6850 2,40 GHz и оперативной памятью объемом 4 гигабайта под управлением ОС OpenSuse 11.4. Для компиляции программ используются компиляторы Chapel 1.1 для программ на Chapel и компилятор g95 для программ на CAF.

Вычислительные эксперименты заключались в сравнении последовательных версий алгоритмов матричного перемножения и решении систем дифференциальных уравнений с параллельными версиями этих же алгоритмов, написанных с помощью новых языков.

На рис. 3 представлены результаты вычислительного эксперимента матричного перемножения. Программы, разработанные на C++ и распараллеленные с помощью OpenMP, сравниваются с программами, разработанными на Chapel. Сравнение производится при проведении вычислений с использованием 2, 3 и 4 потоков.

На рис. 4 показаны результаты выполнения блочного параллельного алгоритма Гаусса-Зейделя для языка CAF (4 и 16 блоков).

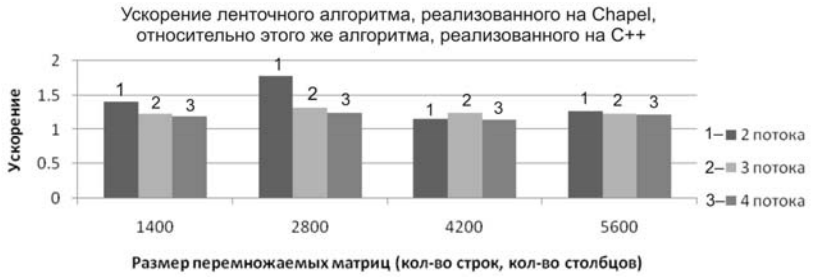

Рис. 3. Результаты экспериментов для языка Chapel

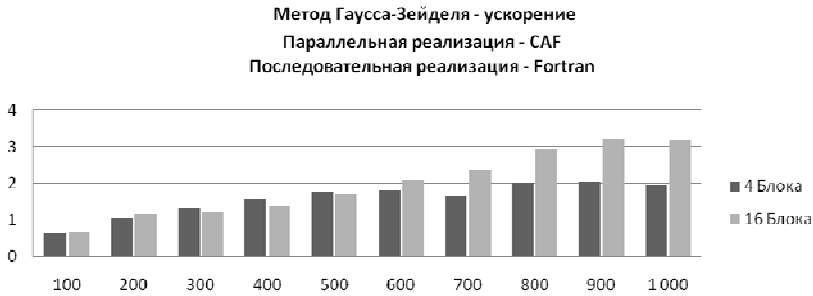

Рис. 4. Результаты экспериментов для языка CAF

В ходе выполнения работы были изучены новые языки параллельного программирования, разработана библиотека параллельных методов ParaLibPGAS, проведены вычислительные эксперименты. Результаты экспериментов показывают достаточную эффективность параллельных программ, разработанных с использовнием новых языков.

Планируемые в дальнейшем работы связаны с расширением состава библиотеки ParaLibPGAS, проведением более расширенного набора вычислительных экспериментов, сопоставлением трудоемкости разработки параллельных программ с использованием разных языков параллельного программирования.

Работа выполнена в лаборатории Intel-ННГУ «Информационные технологии» (ITLab).

Список литературы

1. Гергель В.П. Теория и практика параллельных вычислений. – М.: Интернет-университет информационных технологий; БИНОМ. Лаборатория знаний, 2007. – 424 с.
2. Сайт алгоритмического языка Chapel – http://chapel.cray.com.
3. Сайт алгоритмического языкаX10 – http://x10-lang.org.
4. Сайт алгоритмического языкаCAF – http://www.co-array. org.

В.П. Гергель, А.А. Сиднев

Нижегородский государственный университет им. Н.И. Лобачевского

О ПРИМЕНЕНИИ МАКРОМОДУЛЬНОЙ ТЕХНОЛОГИИ РАЗРАБОТКИ ПРОГРАММ

Разработка программного обеспечения является трудоемкой профессиональной деятельностью. Для упрощения разработки программного обеспечения разработчики выполняют декомпозицию задачи на более мелкие и простые подзадачи, организуя их в отдельные вычислительные блоки (модули). Разработанные модули могут быть использованы в дальнейшем повторно, оформляясь в виде отдельных библиотек. Модульный подход разработки программного обеспечения является на данный момент общепризнанным подходом.

Одним из недостатков такого подхода является отсутствие стандартов на интерфейсы модулей. Разработчик библиотеки сам определяет удобные ему структуры хранения данных и интерфейсы функций, которые их обрабатывают. В результате каждая библиотека получается уникальной и возникает сложность, связанная с заменой используемых библиотек.

Другая проблема при использовании библиотек – проблема выбора наиболее оптимальной библиотеки под текущие задачи проекта. Для решения широкого круга задач существует достаточно много библиотек. Каждая библиотека может иметь