Об инструментальной поддержке архитектурно-независимого параллельного программирования

Легалов Александр Иванович
Сибирский федеральный университет, д.т.н., профессор
Непомнящий Олег Владимирович
Сибирский федеральный университет, к.т.н., доцент

Аннотация

Представлен проект по созданию языковых и инструментальных средств, обеспечивающих поддержку процесса разработки архитектурно-независимых параллельных программ. Описаны основные этапы анализа, преобразования и выполнения архитектурно-независимых параллельных программ. Рассмотрены особенности предлагаемого подхода и предложены пути его реализации.

Ключевые слова: Архитектурно-независимое параллельное программирование; функционально-потоковая парадигма параллельного программирования; языки программирования; преобразование параллельных программ; верификация и отладка параллельных программ.

1. Введение

В настоящее время разработка параллельных программ жестко увязана с архитектурами конкретных параллельных вычислительных систем (ПВС), что ведет к необходимости их практически полного переписывания при переносе с одной ПВС на другую. Это обусловлено специфическими особенностями различных архитектур, которые сильно отличаются друг от друга. В качестве примера «несовместимых» ПВС можно привести: кластеры, многоядерные системы с общей памятью, графические ускорители, системы на перепрограммируемых логических интегральных схемах (ПЛИС). При осуществлении переноса приходится не только переписывать программы, но и практически заново проводить их верификацию и отладку, изучать особенности новой системы. Постоянное изменение архитектур ПВС еще больше усугубляет проблемы и повышает трудоемкость процесса разработки параллельного программного обеспечения. В связи с этим актуальной является задача создания инструментальных и языковых средств позволяющих разрабатывать, отлаживать и выполнять программы, без привязки к конкретной архитектуре. Дальнейшее отображение написанных и отлаженных программ на конкретную параллельную архитектуру может осуществляться в автоматическом или интерактивном режиме с сохранением ранее достигнутой логической корректности.

Для решения поставленной задачи предложен подход к разработке архитектурно-независимых параллельных программ, опирающийся на функционально-потоковую парадигму, основная идея которого заключается в использовании языка программирования, напрямую не связанного с использованием вычислительных ресурсов. Предполагается, что вычислительные ресурсы, выделяемые для выполнения программы, являются неограниченными (бесконечными). Разработанная архитектурно-независимая параллельная программа отлаживается и верифицируется с использованием соответствующих инструментальных средств. Отсутствие ресурсных зависимостей позволяет упростить и формализовать этот процесс. Дальнейший перенос на реальную архитектуру связан с ручным или полуавтоматическим преобразованием за счет сжатия максимального параллелизма решаемой задачи и приведения его к требуемой

архитектуре ПВС. Один и тот же архитектурно-независимый параллельный код можно использовать как общую основу архитектурно-зависимых программ различных ПВС.

2. Особенности реализации системы функционально-потокового параллельного программирования

Исходя из этой базовой концепции, были сформулированы требования к модели параллельных вычислений, языковым и инструментальным средствам, подходам к тестированию, отладке, верификации и выполнению программ. На текущий момент разработаны следующие компоненты:

– функционально-потоковая модель параллельных вычислений [1], поддерживающая концепцию неограниченных вычислительных ресурсов и использующая управление вычислениями по готовности данных (dataflow);

– язык функционально-потокового параллельного (ФПП) программирования, реализующий концепции, заложенные в модель вычислений и обеспечивающий написание архитектурно-независимых параллельных программ [1] (при разработке языка предложены новые подходы, которые позволяют создавать программы со свойствами, отсутствующими в других системах программирования);

– транслятор с функционального языка и дополнительные системные утилиты, предназначенные для порождения выходного представления, используемого во время оптимизации, верификации, отладки и выполнения функционально-потоковых параллельных программ [2];

– событийный процессор, предназначенный для выполнения ФПП программ, представленных порожденным выходным представлением [3];

– средства отладки ФПП программ [4].

Для повышения архитектурной независимости разработано специальное внутреннее представление данных формируемых при трансляции исходных текстов ФПП программы. Разработана четырехслойная модель, используемая на этапе выполнения, которая содержит следующие слои, имеющие соответствующее представление на этапе выполнения программы [2].

1. Реверсивный информационный граф (РИГ) задает зависимости между выполняемыми функциями. В отличие от информационного графа программы, связи между узлами РИГ направлены от приемника информации к источникам. Это позволяет во время выполнения использовать их в качестве ссылок на данные, которые хранятся в точках получения результатов. РИГ используется также для оптимизации исходного представления программы и ее формальной верификации.

2. Управляющий граф (УГ) описывает передачу управления между функциями. Изначально данный граф формируется по РИГ в соответствии с принципом готовности данных. Однако в дальнейшем возможны его оптимизационные преобразования, приводящие к использованию других стратегий управления вычислениями.

3. Автоматный слой определяет особенности срабатывания вершин УГ, в каждой из которых находится свой автомат, получающий сигналы о выполнении тех или иных операций и принимающий решение о запуске функции обработки данных.

4. Слой данных обеспечивает хранение констант и промежуточных данных, формируемых в ходе выполнения программы.

Выполнение функционально-потоковых параллельных программ обеспечивается событийной машиной, состоящей из множества событийных процессоров. Каждый событийный процессор выполняет только одну функцию,

используя для управления вычислениями управляющие сигналы, определяемые УГ [4].

Для отладки функционально-потоковых параллельных программ разработан специальный отладчик, поддерживающий разнообразные режимы отладки [3].

3. Развитие системы функционально-потокового параллельного программирования

Вместе с тем, создание базовых инструментов и их демонстрационное использование не позволяет приступить к практическому использованию концепции архитектурно-независимого параллельного программирования на основе функционально-потоковой парадигмы. Необходимо дальнейшее развитие языковых и инструментальных средств функционально-потокового параллельного программирования. Для этого предполагается решить следующие задачи.

1. Интегрировать средства отладки ФПП программ в событийный процессор, обеспечивающий их исполнение. При этом предусмотреть возможность проведения отладки в различных предусмотренных режимах. Предоставить программисту графический интерфейс пользователя, обеспечивающий эффективную работу в режиме отладки ФПП программ.

2. Доработать язык программирования, а также инструментальные средства, обеспечивающие трансляцию и выполнение ФПП программ, чтобы иметь возможность подключения библиотек функций из внешних репозиториев, размещаемых на удаленных вычислительных узлах и доступных через Интернет. Реализовать режим сборки всех функций и выполнения полученной программы на рабочем месте пользователя. На основе реализованного решения провести анализ возможности выполнения собранной программы или ее отдельных фрагментов на удаленных вычислительных узлах в качестве внешних сервисов.

3. Добавить в язык ФПП программирования дополнительные возможности: альфа-функции, доступ к внешним репозиториям, контроль данных на входе и выходе из функций в виде постусловий и предусловий. Расширить систему пользовательских типов, включив в нее возможность разработки собственных функций приведения типов и доступа к полям сложных типов по символическим именам.

4. Разработать инструментальные средства, обеспечивающие поддержку процесса анализа корректности ФПП программ с использованием методов формальной верификации.

5. Разработать библиотеку функций, поддерживающие проведение математических вычислений с использованием языка ФПП программирования.

6. Разработать методы и алгоритмы оптимизации информационного графа ФПП программы, описывающего основные отношения между данными и операторами, а также управляющего графа, задающего порядок выполнения операторов. Осуществить программную реализацию рассмотренных алгоритмов и интегрировать их с другими инструментальными средствами.

7. Провести анализ архитектурных решений, обеспечивающих одновременное функционирование множества событийных процессоров. Разработать программную систему, поддерживающую параллельное выполнение ФПП программ множеством событийных процессоров в многопроцессорной вычислительной системе с общей памятью.

8. Проанализировать возможность использования языка ФПП программирования в качестве инструмента верхнего уровня для программирования

ПЛИС. Предложить методы преобразования ФПП программ в программы, пригодные для реализации в топологии ПЛИС. Разработать методы преобразования ФПП программ в языки, используемые для описания топологии интегральных схем.

Работа выполняется в рамках ФЦП НПП № 14.A18.21.0396 «Инструментальная поддержка архитектурно-независимой разработки параллельных программ на основе функционально-потоковой парадигмы параллельного программирования».

Литература

1. Легалов А.И. Функциональный язык для создания архитектурно-независимых параллельных программ. – Вычислительные технологии, № 1 (10), 2005. С. 71-89.

2. 139. Легалов А.И., Савченко Г.В., Васильев В.С. Событийная модель вычислений, поддерживающая выполнение функционально-потоковых параллельных программ. / Системы. Методы. Технологии. № 1 (13). - 2012. - С. 113-119.

3. Редькин А.В., Легалов А.И. Событийное управление выполнением функционально-потоковых параллельных программ. / Научный вестник НГТУ, № 3 (32). – 2008. – С. 111-120.

4. 128. Удалова Ю.В. Методы отладки и верификации функционально-потоковых параллельных программ / Ю.В. Удалова, А.И. Легалов, Н.Ю. Сиротинина // Журнал Сибирского федерального университета. Серия «Техника и технологии». Апрель 2011 (том 4, номер 2) – С. 213-224.