УДК 004.942+004.021

ЯЗЫК СПЕЦИФИКАЦИИ ВЫЧИСЛИТЕЛЬНЫХ МОДЕЛЕЙ В МАСШТАБИРУЕМЫХ ПАКЕТАХ ПРИКЛАДНЫХ ПРОГРАММ

Феоктистов А.Г., Горский С.А.

ФБГУН «Институт динамики систем и теории управления им. В.М. Матросова» СО РАН, Иркутск, e-mail: agf65@yandex.ru

Построение вычислительной модели в масштабируемом пакете прикладных программ, определяющей возможности пакета при решении определенного класса сложных задач в высокопроизводительной вычислительной системе, является нетривиальной проблемой. Предполагается, что схема решения задачи может быть представлена в виде композиции программных модулей. Для спецификации модели и формулировки постановок задач предлагается использовать специальный декларативный язык. В процессе трансляции спецификации модели и постановки задачи выявляются все информационно-логические связи между объектами модели и строится схема решения задачи, которая далее выполняется в режиме интерпретации. Использование вычислительной модели обеспечивает возможность формирования потока заданий для вычислительной системы и его эффективную динамическую декомпозицию в зависимости от состояния ресурсов системы и процесса решения задачи.

Ключевые слова: распределенные вычисления, вычислительная модель, язык спецификации

SPECIFICATION LANGUAGE OF COMPUTATIONAL MODELS IN SCALABLE APPLIED SOFTWARE PACKAGES

Feoktistov A.G., Gorsky S.A.

Matrosov Institute for System Dynamics and Control Theory of Siberian Branch of Russian Academy of Sciences, Irkutsk, e-mail: agf65@yandex.ru

Designing a computational model in a scalable applied software package, which describes the package possibilities for solving a certain class of complex problems in high-performance computing system, is non-trivial problem. It is assumed that a problem solving scheme is presented as a composition of program modules. A special declarative language is proposed for specifying the computational model and problem formulations. During translating the specification of the computational model and problem formulations, all information-logical connections between computational model objects are identified and a problem solving scheme is constructed. Later this scheme is executed in interpreted mode. The computational model use provides the possibility of forming a job flow for a computer system and effective dynamic decomposition of the flow, depending on the loading system resources and state of the problem solving process.

Keywords: distributed computing, computational model, specification language

Анализ тенденций развития технологий распределенных вычислений [8] позволяет сделать следующие выводы: во-первых, необходимо решение проблемы интеграции проблемно-ориентированных и сервис-ориентированных систем; во-вторых, эффективное решение этой проблемы непосредственно связано с интеллектуализацией промежуточного ПО, позволяющего динамически интегрировать распределенные разнородные ресурсы в виртуальную исполнительную среду и предоставляющего пользователям возможности прозрачного использования этой среды. Это обуславливает возвращение интереса к пакетной проблематике, но вместе с тем требует реконструировать старые и определить новые средства и методы организации вычислений в пакетах прикладных программ (ППП) как системах модульного программирования.

Совершенствование технологий параллельного программирования привело к существенным изменениям архитектуры современных ППП (рис. 1), разрабатываемых для работы в распределенной вычислительной среде (РВС). Такая среда состоит из вычислительных узлов, объединенных коммуникационной сетью. Вычислительный узел представляет собой программно-аппаратный ресурс, включающий модуль оперативной памяти, один или несколько процессоров, жесткий диск и системное программное обеспечение (совокупность программных средств, поддерживающих функционирование узла в РВС). Один из вычислительных узлов назначается главным узлом и наделяется функциями управления заданиями пользователей и ресурсами РВС. Как следствие, у пользователей ППП возникают новые требования к средствам описания и обработки схем решения задач.

В общем случае архитектура ППП должна включать вычислительные ресурсы, в которых будут выполняться модули пакета; промежуточное программное обеспечение (ПО) для поддержки взаимодействия между функциональным и системным наполнением ППП и вычис-

лительными ресурсами; средства виртуализации компонентов ППП, обеспечивающие открытый, прозрачный доступ пользователей к этим компонентам.

Можно выделить два основных подхода к организации функционального наполнения пакета и решению прикладных задач в РВС. В рамках первого подхода все вычислительные модули (прикладные программы пакета) находятся в одном узле РВС. По схеме решения задачи производится синтез целевой параллельной программы с последующей ее компиляцией и запуском на распределенной вычислительной установке. При втором подходе вычислительные модули размещаются в разных узлах РВС. Схема решения задачи выполняется в режиме интерпретации. Второй подход позволяет повысить отказоустойчивость вычислительного процесса и устранить необходимость постоянной перекомпиляции управляющей программы пакета.

круг потенциальных пользователей таких приложений достаточно ограничен, что обусловлено сложностью их освоения и применения прикладными специалистами для решения своих задач.

В частности, обеспечение эффективного масштабирования [2] потоков заданий, порождаемых ППП в РВС, является в настоящее время одним из фундаментальных и практически важных направлений исследований по организации проблемно-ориентированных РВС. Необходимость создания масштабируемых ППП возникает при выполнении многовариантных расчетов, решении широкого спектра переборных задач и многих других. Предполагается, что масштабируемое приложение включает набор прикладных программ для параллельного решения задачи с помощью различных вычислительных единиц (например, ядер) разнородных узлов РВС и порождает комбинированный поток, объединяющий

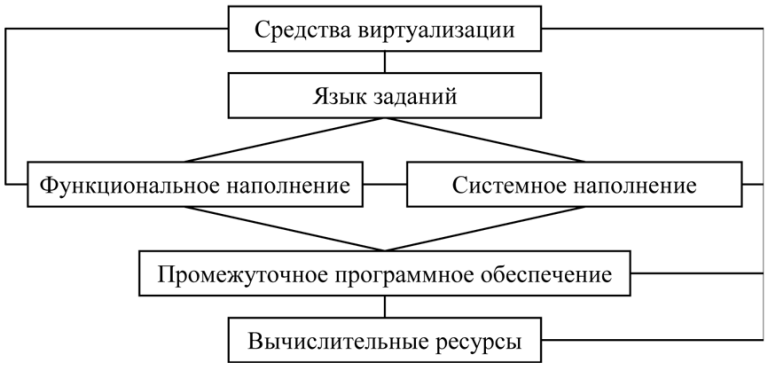


Рис. 1. Структура современного ППП

Наличие развитого базового ПО, реализующего технологии организации расчетов в РВС, и эффективных алгоритмов управления вычислениями [4, 5, 7] является основой для массовых параллельных и распределенных приложений. Однако анализ методов и средств организации такого рода вычислений выявил ряд проблем научно-технического характера: разработка распределенных и параллельных приложений выполняется зачастую «точечно», в привязке к узкому классу задач из конкретной предметной области; создаваемые приложения плохо интегрируются, вследствие использования различного промежуточного ПО, форматов данных и протоколов их передачи, а также использования разных моделей программирования приложений, планирования вычислительных процессов и загрузки вычислительных ресурсов; задания для этих прикладных программ. При этом вычислительная нагрузка, связанная с решением задачи, распределяется между вычислительными единицами разнородных узлов РВС, а время выполнения заданий комбинированного потока уменьшается обратно пропорционально числу используемых вычислительных единиц с учетом их производительности в составе конкретного узла РВС. Создание системы управления комбинированными потоками заданий для ППП является нетривиальной и весьма актуальной проблемой.

Обзор текущего состояния

В настоящее время известен целый ряд систем [9, 10] управления потоками заданий, упоминаемых в зарубежной литературе как Workflow Management Systems. В их числе такие системы, как UNICORE Workflow

System, Taverna, Pegasus, Triana, Askalon, Kepler, GWES и Karajan. Перечисленные системы имеют как общие, так и уникальные возможности формирования потоков заданий и разработки научных приложений.

Общие возможности систем управления потоками заданий реализуются, как правило, в рамках двух основных компонентов: подсистемы текстового и/или графического описания потоков заданий (построения схем выполнения потоков заданий); подсистемы выполнения потоков заданий, мониторинга и распределения ресурсов, управления данными.

вычислимости между двумя подмножествами параметров предметной области (возможность вычисления искомых значений параметров первого подмножества, когда известны значения параметров второго подмножества), модуль – программная реализация операций, и постановка задачи. Данное описание включает: спецификации объектов предметной области и постановок задач. Все информационно-логические связи между объектами предметной области выявляются и учитываются в дальнейшем на этапах формирования и анализа модели предметной области. В качестве входного языка

Элементы входного языка описания модели предметной области ППП

<module> <name>Имя_модуля</name> <parameters>Входные_параметры> Выходные_параметры</parameters> <signature>Команда_запуска</signature> <cores>Число_ядер</cores> <walltime>Время_останова</walltime> <run_mode>Режим_запуска</run_mode> </module>	<operation> <name>Имя_операции</name> <parameters>Входные_параметры> Выходные_параметры</parameters> <run_condition>Условие_запуска</run_condition> <while_flag>Признак_повторения</while_flag> <module_name>Имя_модуля</module_name> <split_condition>Условие_расщепления</split_condition> <task_name>Имя_задачи</task_name> </operation>
<parameter> <name>Имя_параметра</name> <extention>Расширение</extention> <list>Число_элементов</list> </parameter>	<task> <name>Имя задачи</name> <parametrs>Входные_параметры> Выходные_параметры</parametrs> <operations>Список_операций</operations> </task>

Схема выполнения потока заданий может включать управляющие конструкции ветвления и цикла. Однако анализ реальных возможностей рассматриваемых систем применительно к широкому спектру вычислительных крупномасштабных задач разных классов показывает, что существует ряд актуальных вопросов [9], связанных с формированием и обработкой масштабируемых потоков заданий, в том числе вопросы разработки высокоуровневых языков и инструментариев спецификации потоков заданий для высокопроизводительных вычислительных систем, адаптивного управления такими потоками заданий и обеспечения надежности процессов их выполнения.

Язык спецификации вычислительной модели

В описании модели предметной области ППП выделяются три концептуально обособленных слоя знаний – вычислительный, схемный и продукционный, над которыми формируются постановки задач, а также выделяются следующие основные типы объектов: параметр – значимая величина предметной области, операция – отношение

при описании модели предметной области ППП в инструментальном комплексе (ИК) ORLANDO [3] используется расширяемый язык разметки XML. В таблице приведены его элементы для спецификации основных объектов предметной области ППП.

Элемент module предназначен для спецификации объектов вычислительного слоя знаний. Данная спецификация включает информацию об исполняемом программном модуле, входных и выходных параметрах модуля, а также инструкции для запуска и выполнения модуля. Вложенные элементы name, parameters, signature, cores, walltime и run_mode содержат соответственно следующие данные: идентификатор модуля (Имя_модуля); списки формальных входных и выходных параметров модуля (Входные_параметры > Выходные_параметры); команду запуска модуля (Команда_запуска), включающую имя исполняемой программы и опции ее запуска, в том числе регулярные выражения, определяющие структуры данных для параметров модуля и способы их использования; максимально допустимое время выполнения модуля в часах (Время_останова); режим (Режим_запуска), определяющий, запускается

MPI-программа или нет. Функциональное наполнение ППП, реализующее вычислительный слой знаний, включает библиотеку модулей для решения задач в узлах с различными вычислительными характеристиками, библиотеку модулей для препроцессорной и постпроцессорной обработки данных, библиотеку модулей, реализующих предметно-ориентированные процессы декомпозиции задач на подзадачи. Здесь и ниже список объектов (параметров или операций) представляет собой список идентификаторов объектов, разделенных запятыми.

Элемент parameter используется для спецификации объектов схемного уровня знаний – параметров предметной области ППП. Множество параметров включает два вида: базовый параметр и параметр-список. В вычислительной системе базовый параметр представлен произвольным файлом данных неопределенной структуры. Параметр-список формируется на основе некоторого базового параметра, предназначается для представления множества вариантов значений (файлов данных) этого базового параметра и их параллельной обработки. Под параллельной обработкой параметра-списка подразумевается следующее. Пусть x и y – параметры-списки, а IPN_i и $OUTP_i$ – соответственно множества входных и выходных параметров операции o_i, предназначенной для параллельной обработки этих параметров-списков, $x \in INP_i$ и $y \in OUTP_i$. Интерпретация операции o_i выполняется следующим образом: создается r экземпляров (r – число элементов параметров-списков x и y) операции o_i, j-му экземпляру операции o_{ij} передается j-й элемент параметра-списка x; результат присваивается j-му элементу параметра-списка y. В процессе вычислений параметры-списки могут обрабатываться и как неделимые структуры данных. Вложенные элементы name, extention и list содержат соответственно следующие данные: идентификатор параметра (Имя_параметра); расширение файла данных, представляющего параметр (Расширение); число элементов параметра (Число_элементов) – указывается только для парметров-списков.

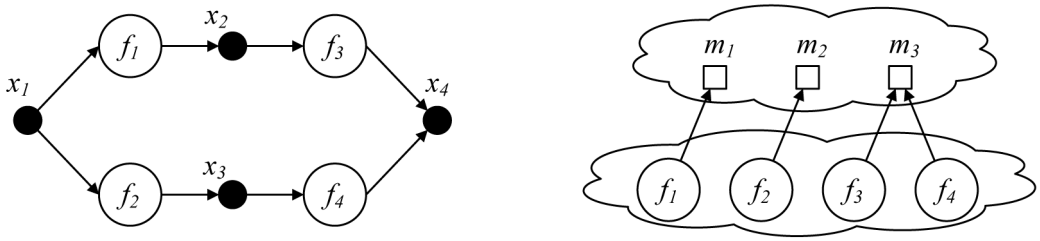
Элемент operation служит для спецификации объектов схемного уровня знаний – операций предметной области ППП. Вложенные элементы name, parameters, run_condition, while_flag, module_name, split_condition и task_name содержат соответственно следующие данные: идентификатор операции (Имя_операции); списки входных и выходных параметров операции (Входные_параметры > Выходные_параметры); логический предикат (Условие_запуска), определяющий необходимость запуска операции, все значения входных параметров которой известны; признак повторного запуска операции (Признак_повторения); идентификатор модуля, реализующего операцию (Имя_модуля); логический предикат (Условие_расщепления), определяющий необходимость динамической декомпозиции подзадачи, решаемой с помощью операции; новая постановка задачи для динамической декомпозиции (Имя_задачи).

Реализация динамической декомпозиции подзадачи обеспечивает возможность порождения комбинированных потоков заданий для РВС и эффективного использования ее разнородных ресурсов. Комбинированные потоки заданий возникают при решении многих задач (например, в задаче поиска всех решений булевых уравнений, являющейся фундаментальной проблемой в математической логике и теории вычислений), обладающих рядом специфических свойств. В их числе высокая вычислительная сложность, необходимость использования высокопроизводительных вычислительных систем, наличие ряда решателей, реализованных для различных вычислительных архитектур и платформ.

Вложенные элементы run_condition и split_condition элемента operation реализуют знания продукционного слоя. В качестве логических предикатов используются функции, заданные на множестве параметров предметной области и возвращающие значения из множества {0, 1}.

Элемент task служит для спецификации постановки задачи, формулируемой в процедурной или непроцедурной форме. Процедурная форма постановки задачи определяется списком операций, которые нужно выполнить для решения задачи. В непроцедурной форме постановка задачи имеет вид «по заданным значениям параметров $x_1, x_2, \ldots, x_n$ вычислить значение параметров $y_1, y_2, \ldots, y_m$». *В* случае непроцедурной постановки задачи список операций определяется автоматически путем планирования на модели предметной области ППП. Вложенные элементы name, parameters и operations содержат соответственно следующие данные: идентификатор постановки задачи (Имя_задачи); списки входных и выходных параметров постановки задачи (Входные_параметры > Выходные_параметры); список операций (Список_операций), необходимых для решения задачи.

Проверка корректности и целостности описания предметной области ППП выполняется на этапах его трансляции с входного языка ИК ORLANDO в базу данных. Средства виртуализации взаимодействия пользователя с масштабируемыми ППП на основе сервис-ориентированного подхода представлены в [1].



а) информационный граф *б) отношение между операциями и модулями предметной области*

Рис. 2. Связи между объектами вычислительного и схемного слоев знаний

Связи между объектами вычислительного и схемного слоев знаний

На рис. 2 изображены двудольный ориентированный граф (рис. 2, а), отражающий информационно-логические связи между параметрами и операциями, а также отношение (рис. 2, б) между операциями и модулями. Отметим, что в данном примере операции f_3 и f_4 реализованы одним и тем же модулем m_3. Такая возможность позволяет отразить семантику используемых данных (параметр x_2 качественно отличается от параметра x_3, вследствие применения к x_1 различных модулей). Двудольный ориентированный граф, изображенный на рис. 1, а, также отражает неявный параллелизм операций предметной области (например, операции f_1 и f_2 могут быть выполнены одновременно), заложенный в ее описании.

Пусть на модели предметной области, приведенной на рис. 1, сформулирована следующая постановка задачи: «зная x_1, вычислить x_4». Для этой постановки задачи планировщиком ИК ORLANDO будут найдены два плана решения задачи: f_1,f_3 и f_2,f_4. Выбор плана решения задачи может быть осуществлен либо непосредственно самим пользователем ППП, либо планировщиком ИК ORLANDO в соответствии с заданными пользователем критериями оценки эффективности вычислений, такими как общее время решения задачи, длина плана и другими характеристиками вычислительного процесса [6].

Поливариантность модели предметной области ППП позволяет адаптировать процесс решения задачи к разнородным ресурсам РВС и, тем самым, обеспечивает масштабируемость вычислений.

Заключение

Компактный, но вместе с тем достаточно выразительный, входной язык ИК ORLANDO позволяет пользователю лаконично в декларативном стиле описывать предметные области сложной динамической структуры и выявлять в них внутренний параллелизм алгоритмов решения задач, что обеспечивает масштабируемость ППП в РВС.

Исследование выполнено при финансовой поддержке РФФИ, проекты № 15-29-07955-офи_м и № 16-07-00931-а.

Список литературы

1. Богданова В.Г., Горский С.А., Пашинин А.А. Сервис-ориентированные инструментальные средства для решения задач булевой выполнимости // Фундаментальные исследования. – 2015. – № 2–6. – С. 1151–1156.

2. Гергель В.П., Линев А.В. Проблемы и перспективы достижения экзафлопного уровня производительности суперкомпьютерных систем // Вестник Нижегородского университета им. Н.И. Лобачевского. – 2012. – № 3–1. – С. 189–198.

3. Опарин Г.А., Феоктистов А.Г., Новопашин А.П., Горский С.А. Инструментальный комплекс ORLANDO TOOLS // Программные продукты и системы. – 2007. – № 4. – С. 63–65.

4. Топорков В.В., Емельянов Д.М. Экономическая модель планирования и справедливого разделения ресурсов в распределенных вычислениях // Программирование. – 2014. – Т. 40, № 1. – С. 54–65.

5. Топорков В.В., Емельянов Д.М., Потехин П.А. Формирование и планирование пакетов заданий в распределенных вычислительных средах // Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика. – 2015. – Т. 4, № 2. – С. 44–57.

6. Феоктистов А.Г. Управление сложной системой на основе методологии многокритериального выбора управляющих воздействий // Фундаментальные исследования. – 2015. – № 9–1. – С. 82–86.

7. Фраленко В.П., Агроник А.Ю. Средства, методы и алгоритмы эффективного распараллеливания вычислительной нагрузки в гетерогенных средах // Программные системы: теория и приложения. – 2015. – № 3. – С. 73–92.

8. Шамакина А.В. Обзор технологий распределенных вычислений // Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика. – 2014. – Т. 3, № 3. – С. 51–85.

9. Talia D. Workflow Systems for Science: Concepts and Tools // ISRN Software Engineering Vol. 2013. URL: http://dx.doi.org/10.1155/2013/404525 (дата обращения: 27.04.2016).

10. Tao J., Kolodziej J., Ranjan R., Jayaraman P., Buyya R. A Note on New Trends in Data-Aware Scheduling and Resource Provisioning in Modern HPC Systems // Future Generation Computer Systems. – 2015. – Vol. 51. – P. 45–46.