ИНТЕЛЛЕКТУАЛЬНОЕ УПРАВЛЕНИЕ КОМПЛЕКСОМ ПРИКЛАДНЫХ ВЫЧИСЛИТЕЛЬНЫХ ПАКЕТОВ В СЕРВИСНО-ОРИЕНТИРОВАННОЙ СРЕДЕ

С.В. Марьин

В настоящее время значительной популярностью для решения задач научного моделирования пользуется концепция PSE (Problem Solving Environment – проблемно-ориентированная оболочка). PSE — это программный комплекс, предоставляющий пользователю все необходимые для решения определенного класса задач вычислительные средства: методы решения задач конкретной предметной области, способы выбора методов, а также способы добавления новых методов решений [1, 2]. Применение этой концепции обеспечивает централизованный пользовательский доступ к ряду проблемно-ориентированных приложений, предназначенных для решения задач заданной предметной области. Принципиальной особенностью PSE по сравнению с традиционными формами организации научных пакетов (препроцессор — солвер — постпроцессор) является возможность пользователю самостоятельно формировать процесс решения своей задачи из существующих расчетных модулей непосредственно в терминах предметной области.

Для освобождения пользователя от необходимости знания особенностей использования каждого из применяемых программных средств, форматов данных и технических особенностей реализации алгоритмов предметной области, а также от решения задачи оптимального использования доступных вычислительных ресурсов предлагается построение интеллектуальной системы управления набором предметно-ориентированных пакетов, берущей на себя ответственность за все низкоуровневое управление пакетами. В работе рассматриваются особенности построения интеллектуальной системы, основанной на экспертных знаниях, которая позволит осуществить автоматическое управление предметно-ориентированными программными компонентами, входящими в состав высокопроизводительного программного комплекса, отвечая при этом критериям эффективности, предъявляемым к типовым высокопроизводительным научным приложениям.

Такая система, опираясь на совокупность экспертных знаний как в предметной области, так и в области информационных технологий, способна построить решение поставленной задачи, сформулированной в терминах предметной области. С одной стороны, при этом учитываются технологические особенности реализации численных алгоритмов в доступных предметно-ориентированных пакетах. С другой стороны, на основании информации об актуальном состоянии вычислительной программно-аппаратной платформы, интеллектуальная система управления может построить оптимальную схему запуска высокопроизводительных сервисов, обеспечивающих доступ к отдельным вычислительным платформам.

Как предметно-ориентированное, так и технологическое ядро знаний экспертной системы может быть представлено в форме онтологической структуры, описывающей основные понятия и объекты а) предметной области; б) информационно-технического пространства в котором функционирует программный комплекс. Слой классов онтологии определяется в данном случае как граф $O = \langle C, R \rangle$, где C — множество классов, R — множество абстрактных отношений, связывающих классы. Аналогично, слой индивидов онтологии определяется как граф $\widetilde{O} = \langle \widetilde{C}, \widetilde{R} \rangle$, где $\widetilde{C}$ — множество индивидов, а $\widetilde{R}$ — множество отношений между индивидами. При этом для каждого элемента слоя индивидов определены:

а) отношение генерализации $gn^{(C)} : \widetilde{C} \to C$, $gn^{(R)} : \widetilde{R} \to R$, определяющие связь индивидов и связей между ними с соответствующими классами и связями классов;

б) «сторожевые» условия (guard condition) определяющие применимость элементов в данных условиях $gc^{(C)}(F) : \widetilde{C} \to \{0,1\}$, $gc^{(R)}(F) : \widetilde{R} \to \{0,1\}$, где F – множество активных фактов, определенных для текущей задачи;

в) функция критериальной оценки $k^{(C)}(F) : \left\{\tilde{c} \in \widetilde{C} \mid gc^{(C)}(\tilde{c}) = 1\right\} \to \Psi^{(C)}$, $k^{(R)}(F) : \left\{\tilde{r} \in \tilde{R} \mid gc^{(R)}(\tilde{r}) = 1\right\} \to \Psi^{(R)}$, где $\Psi^{(C)}$ и $\Psi^{(R)}$ соответственно, пространство критериев оценки индивидов и отношений между ними.

Следует отметить, что в общем случае, сторожевые условия могут рассматриваться как один из критериев оценки. Тем не менее, в данной работе они выделены в отдельный класс в целях упрощения работы критериальной оценки, оперирующей сокращенными множествами $\widetilde{C}$ и $\widetilde{R}$.

В процессе реализации логического вывода на базе описанной онтологической структуры управляющая система осуществляет фильтрацию вариантов запуска прикладных пакетов, реализующих те или иные вычислительные методы. При этом, результаты фильтрации подвергаются ранжированию по системе критериев качества, основой для вычисления которых являются а) априорные экспертные знания онтологической структуры; б) совокупность активных фактов, полученных в процессе анализа текущего состояния вычислительной системы и характеристик поступившего от пользователя описания задачи. Результаты

фильтрации и ранжирования, в зависимости от соотношения полученных критериев могут быть использованы, как для автоматического определения субоптимального (в силу его априорности) способа запуска вычислительных модулей, так и в качестве смысловой оценки для предоставления выбора конечному пользователю. Примером системы критериев может служить следующая совокупность оценок:

- время исполнения (как критерий эффективности использования вычислительных ресурсов) — определяется на основании совокупности моделей производительности;
- точность решения — определяется на основании совокупности экспертных оценок используемых алгоритмов и их реализаций;
- надежность исполнения — определяется исходя из доли успешных запусков предлагаемого набора пакетов в прошлом.

В качестве примера реализации такой интеллектуальной управляющей системы может рассматриваться управляющая система высокопроизводительного распределенного программного комплекса HPC-NASIS [3]. Комплекс включает в себя 17 прикладных вычислительных сервисов, из которых он позволяет строить цепочку (последовательность) запусков; при этом он изолирует от пользователя реализации механизмов подготовки (форматирования) данных, выбора вычислителя, запуска задачи и мониторинга ее исполнения. Комплекс HPC-NASIS является многоуровневой распределенной системой, позволяющей запускать вычислительные компоненты на ресурсах Грид и (или) на удаленных кластерах в рамках модели метакомпьютинга. Комплекс включает в себя уровни: взаимодействия с пользователем, управления вычислениями и пакетной обработки заданий на кластере (или запуска сервисов в Грид).

Реализацией уровня управления вычислениями является разработанная в процессе создания комплекса сервисно-ориентированная распределенная среда управления вычислительными ресурсами и прикладными вычислительными пакетами (управляющая среда, подсистема управления). Требования к такой подсистеме проистекают от требований к комплексу в целом:

- выбор кластера для запуска;
- формирование входного файла для конкретного вычислительного пакета по параметрам, составленным в терминах предметной области;
- предоставление сервисных «оберток» для вычислительных пакетов, их запуск;
- мониторинг процесса выполнения задачи;
- постобработка выходных данных, если она необходима (например, для другого пакета в цепочке, либо для визуализации).

Управляющая среда опирается на сервисно-ориентированную архитектуру (service-oriented architecture, SOA), в рамках которой каждый из программных модулей (компонентов композитного приложения), доступных для использования представляется вычислительным сервисом, функционирующим на определенной программно-аппаратной платформе (кластере или Грид-инфраструктуре). Таким образом, обеспечивается унификация доступа к изначально разнородным предметметно-ориентированным модулям и платформам исполнения. Среда реализована на платформе Microsoft .Net Framework; компоненты, представляющие сервисную обертку вычислительных пакетов на Linux-кластерах, работают под управлением платформы Mono.

Управляющая среда получает от взаимодействующей с пользователем части системы абстрактную последовательность вычислений, то есть последовательность вычислительных пакетов и набор параметров, соответствующий описанию поставленной задачи в терминах предметной области. Подсистема управления при получении новой последовательности вычислений формирует входной файл на основе полученных параметров, выбирает кластер в соответствии с применяемыми моделями запуска и запускает вычислительный пакет на исполнение. Для оптимизации выбора кластера используется набор моделей производительности, которые позволяют оценивать время счета заданного пакета на каждом кластере с учетом параметров задачи.

Предлагаемый подход к построению интеллектуальных систем управления высокопроизводительными программными комплексами с композитной архитектурой позволяет осуществить динамическое построение и модификацию схемы запуска вычислительных сервисов, входящих в состав комплекса, а также проконтролировать ее исполнение. При этом система управления принимает решение на основании экспертных знаний и набора активных фактов, построенного с учетом текущего состояния высокопроизводительного программного комплекса. Принципы, лежащие в основе описанного подхода, были успешно использованы в процессе разработки системы управления высокопроизводительного программного комплекса HPC-NASIS.

ЛИТЕРАТУРА:

1. Rice J. R., Boisvert R. F. From Scientific Software Libraries to Problem-Solving Environments // IEEE Computational Science & Engineering. 1996. Vol. 3, N 3. P. 44-53.
2. Бухановский А. В., Ковальчук С. В., Марьин С. В. Интеллектуальные высокопроизводительные программные комплексы моделирования сложных систем: концепция, архитектура и примеры реализации // Известия вузов. Приборостроение. 2009. Т. 52, № 10. С. 5-24.

3. Васильев В. Н. и др. Высокопроизводительный программный комплекс моделирования атомно-молекулярных наноразмерных систем // Науч.-технич. вестн. СПбГУ ИТМО. Технологии высокопроизводительных вычислений и компьютерного моделирования. 2008. Вып. 54. С. 3-12.