Раздел III. Распределенные вычисления и системы

УДК 004.4’2+004.89 DOI 10.18522/2311-3103-2016-11-6575

А.Г. Феоктистов, Р.О. Костромин

РАЗРАБОТКА И ПРИМЕНЕНИЕ ПРЕДМЕТНО-ОРИЕНТИРОВАННЫХ МУЛЬТИАГЕНТНЫХ СИСТЕМ УПРАВЛЕНИЯ РАСПРЕДЕЛЕННЫМИ ВЫЧИСЛЕНИЯМИ*

В настоящее время эффективное управление масштабируемыми приложениями для решения больших фундаментальных и прикладных задач в гетерогенной распределенной вычислительной среде является нетривиальной проблемой. Перспективным подходом для ее решения является применение мультиагентных систем. Сегодня существует широкий спектр инструментариев для построения мультиагентных систем различного назначения. Однако известные инструменты не обладают всеми необходимыми средствами для автоматизации создания агентов, а также не обеспечивают представление знаний о предметной области решаемых задач и программно-аппаратной инфраструктуре в создаваемой мультиагентной системе. Целью нашего исследования является разработка методов и средств для решения этой проблемы. В статье проводится сравнительный анализ ряда известных мультиагентных систем для управления масштабируемыми приложениями в распределенной вычислительной среде. Рассматриваются инструментальные средства для построения мультиагентных систем. Обосновывается выбор системы JADE в качестве базового инструментария. Формулируются принципы организации мультиагентной системы для управления масштабируемыми приложениями. Эффективная работа системы основывается на комплексном использовании вычислительных, схемных и продукционных знаний, а также знаний о программно-аппаратной инфраструктуре среды и административных политиках в ее узлах. Эти знания представляются в виде концептуальной модели среды. Предлагается методика организации мультиагентных систем, базирующаяся на восходящем подходе к проектированию подобных систем с использованием методов и средств синтеза абстрактных программ, отражающих поведения агентов, а также генерации на их основе программного кода. Разработан инструментарий для организации предметно-ориентированных мультиагентных систем. Данный инструментарий, обеспечивающий возможность использования знаний о вычислительной среде, является интеллектуальной надстройкой над средствами системы JADE, существенно расширяющей возможности этой системы. Примеры решения задач параметрического синтеза линейного регулятора динамического объекта и выполнимости булевых ограничений с помощью масштабируемых приложений под управлением мультиагентной системы, разработанной в соответствии с предложенными принципами организации таких систем, показывают масштабируемость и эффективность распределенных вычислений.

Масштабируемое приложение; управление распределенными вычислениями; мультиагентная система; инструментальные средства.

* Исследование выполнено при финансовой поддержке РФФИ, проекты № 15-29-07955-офи_м и № 16-07-00931-а, а также при частичной финансовой поддержке Совета по грантам Президента Российской Федерации для государственной поддержки ведущих научных школ Российской Федерации (НШ-8081.2016.9).

A.G. Feoktistov, R.O. Kostromin

DEVELOPMENT AND APPLICATION OF SUBJECT-ORIENTED MULTI-AGENT SYSTEMS FOR DISTRIBUTED COMPUTING MANAGEMENT

Nowadays, the effective management of scalable applications for solving large fundamental and applied tasks in a heterogeneous distributed computing environment is a non-trivial problem. The promising approach to solving this problem is the use of multi-agent systems. Today, there is a wide range of frameworks for creating the multi-agent systems for various purposes. However, the known frameworks do not have all the necessary tools to automate the development of agents. They also do not provide a representation of knowledge about the subject domain of solved tasks and software/hardware infrastructure in the created multi-agent system. The aim of our study is to develop methods and tools to solve these problems. In the paper, we represent a comparative analysis of a number of the well-known multi-agent systems for the management of such applications. We also give a brief overview of frameworks for developing multi-agent systems and prove the selection of the system JADE as a basic tool. Our contribution is multifold. We formulate the principles of multi-agent systems for the management of scalable applications. The effective operation of the system is based on the integrated use of the computational, schematic and production knowledge as well as knowledge about the software/hardware infrastructure of the environment and administrative policies in its nodes. This knowledge is presented in the form of a conceptual model of the computing environment. We propose a methodology for creating of multi-agent systems. It is based on a bottom-up approach to the design of such systems. This approach uses the methods and tools for the synthesis of abstract programs which reflect the behavior of agents. These methods and tools are used to generate the code of agents. The developed tool for creation of subject-oriented multi-agent systems is the intelligent superstructure over tools of the system JADE. They significantly extend the capabilities of this system. Examples of solving problems of parametric synthesis of linear regulator for dynamic object and SAT-problems using scalable applications under the management of multi-agent system show the scalability and efficiency of distributed computing. This system is developed in accordance with the proposed principles of the organization of such systems.

Scalable application; distributed computing management; multi-agent system; development tools.

Введение. Развитие высокопроизводительных вычислений влечет за собой новые проблемы, связанные с решением больших научных задач для различных предметных областей в гетерогенных распределенных вычислительных средах (ГРВС), например, в Grid-системах или облачных инфраструктурах [1]. Специалисты в области параллельных и распределенных систем уделяют особое внимание масштабируемости приложений, эффективное управление которыми в ГРВС остается нетривиальной задачей. Использование мультиагентных технологий применительно к этой задаче показывает хорошие результаты [2, 3]. Существует широкий набор инструментов для построения мультиагентных систем (МАС), многие из которых успешно применяются на практике [4]. Однако автоматизация создания агентов и представление знаний о предметной области и программно-аппаратной инфраструктуре в МАС является сложной проблемой.

Целью нашего исследования является разработка методов и средств для ее решения.

МАС для управления вычислениями. В настоящее время ряд МАС разрабатывается и успешно применяется для управления распределенными вычислениями [5, 6]. В табл. 1 представлен сравнительный анализ известных МАС, поддерживающих масштабируемость приложений применительно к модели заданий с мягкими критериями качества обслуживания. Рассматриваются следующие функциональные возможности систем:

- наличие встроенных агентов (c_1);
- обеспечение высокоуровневых инструментов разработки пользовательских агентов, включая их конфигурирование и настройку на работу с предметно-ориентированными знаниями (c_2);

- формирование процедурной постановки задачи и построение схемы ее решения (c_3);
- формулирование непроцедурной постановки задачи и автоматический синтез схемы ее решения (c_4);
- распределение ресурсов (c_5);
- мониторинг среды (c_6);
- поддержка модели Агент-как-Сервис (c_7);
- моделирование агентами поведения других агентов с целью повышения качества принятия решений при их взаимодействии (c_8);
- управление в рамках локальной распределенной системы (например, вычислительного кластера) с целью оптимизации распределения ресурсов с учетом всех запросов пользовательских приложений (c_9);
- управление в рамках глобальной распределенной системы (например, Grid-системы) с целью оптимизации распределения ресурсов с учетом всех запросов пользовательских приложений (c_{10});
- управление на уровне приложений с целью оптимизации распределения ресурсов для конкретного приложения (c_{11});
- комбинированное управление, обеспечивающее планирование вычислений и распределение ресурсов на локальном и глобальном уровнях, а также на уровне приложений на основе единой информационной структуре (c_{12});
- централизованный алгоритм взаимодействия агентов (c_{13});
- децентрализованный алгоритм взаимодействия агентов (c_{14});
- обеспечение повышенной надежности обмена сообщениями между агентами за счет использования системы логического времени (c_{15});
- применение агентами экономических механизмов управления (c_{16});
- возможность исполнения нескольких ролей одним агентом (c_{17});
- обучение агентов (c_{18}).

Таблица 1

Функциональные возможности МАС

Система	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9/c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}	c_{16}	c_{17}	c_{18}
Condor-G [7]	+	–	+	–	+	+	–	–	–	+	–	+	–	–	+	–	–
GridSolve [8]	+	–	+	–	+	+	–	–	–	+	–	+	–	–	–	–	–
AppLes [9]	+	–	+	–	+	+	–	–	–	+	–	+	–	–	–	–	–
MAGE [10]	+	+	+	–	+	+	+	+	–	+	–	–	+	–	–	–	+
MAAG [11]	+	–	+	–	+	+	–	–	–	+	–	–	+	–	–	–	–
Singh [12]	+	–	+	–	+	+	+	–	+	–	–	+	–	–	+	–	–
MAC [13–15]	+	+	+	+	+	+	+	+	–	–	+		+	+	+	+	+

Результаты сравнения показывают достоинства и недостатки каждой из представленных систем. МАС [13–15] относится к промежуточному программному обеспечению и не поддерживает ряд функций, являющихся типовыми для традиционных систем управления прохождением заданий и некоторых систем, представленных в табл. 1. Тем не менее, комбинированное управление в этой МАС в сочетании с агентным моделированием представляет более гибкие возможности по сравнению с другими аналогичными системами. Алгоритмы взаимодействия агентов, обладающие всеми характеристиками c_{14}-c_{18}, обеспечивают надежную и эффективную работу системы.

Инструментальные средства построения МАС. Важными функциональными характеристиками инструментариев для разработки МАС являются [16]: использование общепринятой совокупности правил (стандарта) взаимодействия агентов, наличие открытого исходного кода, кроссплатформенность, масштабируемость разрабатываемых систем, а также наличие документации к ним и поддержка сообществом разработчиков. Сегодня известен широкий спектр агентных платформ и средств для создания агентов, среди которых популярными являются системы AgentBuilder, AgentScape, Cougaar, CybelePro, EMERALD, GAMA, JADE, Jadex и MaDKit [4].

Исходя из сравнительного анализа, проведенного в [4], можно сделать вывод о том, что наиболее пригодными для разработки МАС являются следующие инструментальные средства: EMERALD, GAMA, JADE и Jadex. Они поддерживаются и развиваются, соответствуют стандартам, являются кроссплатформенными, имеют открытый исходный код, являются применимыми к решению разнообразных задач. Однако, система GAMA больше подходит для агентного моделирования. Системы EMERALD, JADE и Jadex в целом близки по сравниваемым характеристикам, но при этом популярность у EMERALD достаточно низкая. Это связано с тем, что одним из важных критериев является возможность работы агентов на мобильных платформах – таким требованиям удовлетворяют только JADE и Jadex. Обе эти платформы чаще остальных используются для построения МАС [16].

Общим недостатком рассмотренных систем является слабая привязка к предметной области решаемой задачи при разработке МАС.

Принципы организации МАС. Предлагаемая иерархическая структура МАС может включать два или более уровней агентов. На каждом уровне агенты играют различные роли и соответственно выполняют различные функции. Роли агентов носят постоянный или временный характер и возникают в дискретные моменты времени в связи необходимостью организации коллективного взаимодействия в процессе решения задачи, поставленной пользователем Grid. Уровни иерархии агентов отличаются объемом их знаний – агенты более высокого уровня иерархии обладают большим объемом знаний по сравнению с агентами более низкого уровня иерархии и, кроме того, обращаются к агентам ниже лежащих уровней с запросами на получение локальных знаний этих агентов. На каждом уровне иерархии агенты могут объединяться в виртуальные сообщества, кооперироваться и конкурировать в рамках этих сообществ.

МАС включает агентов постановки задачи, планирования вычислений, мониторинга и распределения ресурсов, классификации, конкретизации и выполнения заданий, а также агентов параметрической настройки алгоритмов функционирования вышеперечисленных агентов на основе имитационного моделирования. Для агентов разработана библиотека оригинальных «встроенных» алгоритмов: имитационного моделирования и адаптивного обучения (параметрической настройки алгоритмов функционирования) агентов; алгоритмов взаимодействия агентов. Алгоритмы рассмотрены в [13–15].

Концептуальная модель. Эффективная работа МАС основывается на комплексном использовании вычислительных знаний о программных модулях для решения задач в предметных областях и работы с объектами ГРВС, схемных знаний о модульной структуре модели и алгоритмов, продукционных знаний для поддержки принятия решений по выбору оптимальных алгоритмов в зависимости от состояния среды, а также знаний о программно-аппаратной инфраструктуре ГРВС и административных политиках в ее узлах. Эти знания представляются в виде концептуальной модели ГРВС [13], являющейся частным случаем семантической сети.

Пусть Z, F и M – это множество параметров, операций и программных модулей модели. Модули являются элементом вычислительных знаний. Параметры, операции и их взаимосвязи отражают схемные знания. Операции из F определяют отношения вычислимости на множестве параметров Z. Каждой операции $f_i \in F$ соответсвует модуль $m_j \in M$, где $i \in \overline{1,n_f}$, $j \in \overline{1,n_m}$, n_f – число операций, n_m – число модулей. Один модуль может реализовывать несколько операций. Спецификация модуля включает следующую информацию: язык программирования, тип и семантику входных, выходных и транзитных параметров, способы передачи параметров и обработки нестандартных ситуаций, модуль представления, требуемый компилятор и другие сведения.

С каждой операцией f_i связано два множества параметров $Z_i^{in}, Z_i^{out} \subset Z$. Множество Z_i^{in} определяет параметры, значения которых необходимо задать, чтобы получить значения параметров, представленных множеством Z_i^{out}. Множества Z_i^{in} и Z_i^{out} представляют соответственно множества входных и выходных параметров модуля m_j, реализующего операцию f_i. Различаются базовые и составные операции. Составные операции могут включать базовые операции, а также конструкции для организации ветвления и итерации.

Постановки задач могут формулироваться в полной или сокращенной (процедурной или непроцедурной) форме. По сформулированной постановке задачи строится схема ее решения (абстрактная программа) на основе методов статического, динамического или статико-динамического планирования вычислений [17]. Оператор статико-динамического планирования может включаться в составную операцию. В общем случае в модели может существовать множество S эквивалентных схем решения задачи. Схема $s \in S$ определяет, какие операции и в какой последовательности должны быть выполнены для решения задачи. Полная постановка задачи s_f, совпадающая со схемой s, определяется структурой $s_f = < F_s, X_0, Y_0 >$, где $F_s \subset F$ – множество операций, которые нужно выполнить для решения задачи, $X_0 \subset Z$ – множество исходных параметров, значения которых заданы, $Y_0 \subset Z$ – множество целевых параметров, значения которых нужно вычислить.

В множествах параметров и операций модели введем соответственно подмножества системных параметров, отражающих характеристики объектов ГРВС, и операций, представляющих алгоритмы планирования вычислений, мониторинга и распределения ресурсов, моделирования вычислительных процессов и других действий в ГРВС. В процессе эксплуатации среды значения системных параметров (характеристики ее объектов) определяются системой метамониторинга [15].

Агентная модель. Функционирование агентов осуществляется с использованием парадигмы конечно-автоматного программирования. Модель агента представлена структурой $M^{agent} = < sm^p, \left\{ sm_{i,j}^c : i \in \overline{1,n_{vc}}, j \in \overline{1,n_{rol}} \right\}, MES >$, где sm^p – родительский автомат, $sm_{i,j}^c$ – дочерние автоматы, n_{vc} – число виртуальных сообществ, в которых состоит агент, n_{rol} – число ролей, которые может играть агент, MES – множество сообщений агента. Основной функцией родитель-

ского автомата sm^p является создание дочернего автомата $sm^c_{i,j}$ при каждом включении агента в новое виртуальное сообщество, где *i* и *j* – это соответственно номера виртуального сообщества и роли агента.

Модель родительского автомата представлена структурой $sm^p =< STS^p, sts^p_0, ACT^p, h^p, GV >$, где STS^p – множество состояний родительского автомата, $sts^p_0 \in STS^p$ – начальное состояние родительского автомата, $ACT^p \subset F$ – множество действий родительского автомата, $h^p \in F$ – логическая функция, определяющая условия переходов родительского автомата, GV – множество глобальных переменных родительского автомата, доступных дочерним автоматам. При создании агента все схемы выполнения действий, представляющие собой абстрактные программы, генерируются на языке программирования Java.

Модель дочернего автомата представлена структурой $sm^c_{i,j} =< STS^c_j, sts^c_{j,0}, ACT^c_j, h^c_j, SLT_i >$, где STS^c_o – множество состояний дочернего автомата, $sts^c_{j,0} \in STS^c_j$ – начальное состояние дочернего автомата, $ACT^c_j \subset F$ – множество действий дочернего автомата, $h^c_j \in F$ – логическая функция, определяющая условия переходов дочернего автомата, SLT_i – система логического времени *i*-го сообщества.

Дочерние автоматы разных агентов, входящих в одно виртуальное сообщество, взаимодействуют путем обмена сообщениями, передаваемыми через родительские автоматы. Автоматы, являющиеся потомками одного и того же родительского автомата, обмениваются информацией об использовании общих ресурсов агента через глобальные переменные родительского автомата и агентную базу знаний.

Пусть $L \subset Z$ – множество логических параметров, I – множество индексов действий из множества $ACT = ACT^p \cup ACT^c_1 \cup ... \cup ACT^c_{n_{rol}}$. Функции, определяющие условия переходов родительского и дочернего автоматов определены как $h^p : L \to i$ и $h^c_j : L \to i$, $i \in I$, $j \in \overline{1, n_{rol}}$.

При вступлении агента в *i*-е виртуальное сообщество для соответствующего дочернего автомата создается система логического времени SLT_i, определяемая структурой $SLT_i =< T, T_m, g_t, g_m, g_r >$, где T – область значений логического времени, T_m – область значений временных маркеров датировки сообщений, $T_m \subseteq T$, g_t, g_m и g_r – функции датировки событий автомата, маркировки сообщений и сравнения значений логического времени, $\forall i \in \overline{1, n_{vc}}$. Система логического времени использует векторные часы, в которых число компонент вектора времени равно числу агентов виртуального сообщества, и обеспечивает отношение частичного порядка на множестве событий виртуального сообщества с учетом их обусловленности. Применение функции датировки сообщений обеспечивает их обработку в установленной логической последовательности, а не в произвольном порядке поступления их в общий пул.

Методы автоматизации создания агентов. Мы используем инструментарий JADE [18] для программной реализации МАС. Инструментарий включает набор стандартных агентов для управления конструируемыми агентами и библиотеку стандартных классов для Java-программ. JADE применяется для организации агентной платформы, которая в общем случае включает: один или несколько вы-

числительных узлов; стандартные агенты Agent Management System, Directory Facilitator и Remote Monitoring Agent; агенты создаваемой МАС. Стандартные агенты создаются автоматически при запуске агентной платформы.

Конструирование агентов МАС в JADE является весьма трудоемкой, рутинной работой, требующей высокой программистской квалификации и погружения во все тонкости представления и использования предметно-ориентированных знаний агентом. Предложенный в статье подход к автоматизации конструирования агента МАС на базе стандартных классов JADE позволяет существенно упростить этот процесс.

Нами разработан инструментарий для организации предметно-ориентированной МАС, включающий конструктор модели системы и генератор агентов. В конструкторе разработчик МАС выполняет концептуальное моделирование – описывает новые объекты: агенты, их роли, виртуальные сообщества и отношения, а также состояния, функции и графы переходов автоматов. В качестве состояния агента мы используем состояние-действие – последовательность системных операций, выполняемых над полем системных параметров модели. Фрагмент такой модели, описывающей объекты, необходимые для построения графов переходов агентов, представлен на рис. 1. Здесь G – множество графов переходов, A – множество агентов, VC – множество виртуальных сообществ агентов, R – множество ролей агентов и $STS = STS^p \cup STS_1^c \cup ... \cup STS_{n_{rol}}^c$ – множество состояния автоматов. На рис. 1 отношения между объектами обозначены $o_1 - o_8$. Отношение o_4 представляет взаимосвязь состояний с операциями, реализующими функции переходов. Отношение o_5 представляет взаимосвязь состояний с остальными системными операциями. База знаний агента создается на основе фрагмента модели ГРВС, рассмотренного выше.

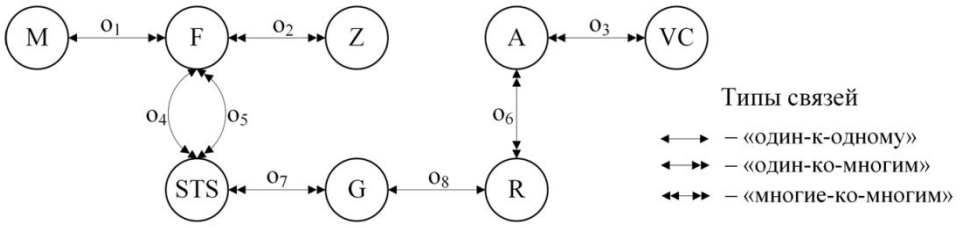

Рис. 1. Фрагмент описания концептуальной модели

С помощью генератора разработчик МАС производит синтез абстрактной программы, специфицирующей поведение агента. Ее построение производится путем статического (на основе процедурной и непроцедурной постановок задач) или динамического планирования вычислений. Далее разработчик выполняет формирование программного кода агента на языке Java. Генератор использует граф переходов состояний агента, базу знаний, библиотеку стандартных классов JADE и библиотеку оригинальных алгоритмов функционирования агентов. Генерация программного кода агента осуществляется в рамках каркасного подхода к конструированию программ (рис. 2). Для реализации дополнительных методов стандартных классов JADE, представляющих функции (операции) агентов МАС, применяется библиотека «встроенных» алгоритмов, оформленных в виде модулей.

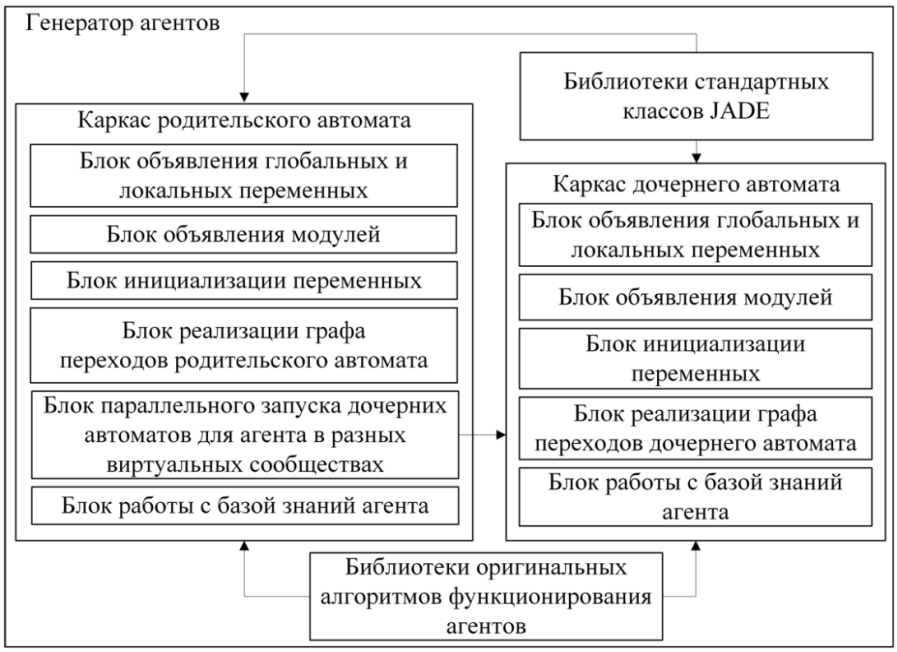

Рис. 2. Генерация программного кода агента

Экспериментальные исследования. Примеры решения задач параметрического синтеза линейного регулятора динамического объекта [19] и выполнимости булевых ограничений [20] с помощью приложений под управлением МАС, разработанной в соответствии сформулированным принципам организации таких систем, показали масштабируемость и эффективность распределенных вычислений.

Эксперименты проводились в ГРВС, организованной на базе ресурсов Иркутского суперкомпьютерного центра (http://hpc.icc.ru), с использованием до 3000 ядер.

Заключение. Рассматриваемые инструментальные средства обеспечивают эффективность процесса организации и применения МАС. Предлагаемая методика организации МАС базируется на восходящем подходе к проектированию подобных систем с использованием методов и средств синтеза абстрактных программ поведения агентов по концептуальной модели ГРВС и генерации на их основе программного кода.

Научная новизна представленных результатов исследований заключается в разработке новых методов создания МАС, интегрирующих методы концептуального и автоматного программирования, а также в развитии модели ГРВС. Применение представленного в статье инструментария обеспечивает в отличие от известных инструментов более высокий уровень планирования действий агентов МАС, их проблемную ориентацию, а также частичную автоматизацию конструирования агентов.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Qureshi M.B., Dehnavi M.M., Min-Allah N., Qureshi M.S., Hussain H., Rentifis I., Tziritas N., Loukopoulos T., Khan S.U., Xu C.Z., Zomaya A.Y.* Survey on Grid Resource Allocation Mechanisms // Journal of Grid Computing. – 2014. – Vol. 12, No. 2. – P. 399-441.
2. *Talia D.* Cloud Computing and Software Agents: Towards Cloud Intelligent Services // Proceedings of the 12th Workshop on Objects and Agent. – 2011. – P. 2-6.

3. *Leitao P., Inden U., Ruckemann C.-P.* Parallelising Multi-agent Systems for High Performance Computing // Proceedings of the 3rd International Conference on Advanced Communications and Computation. – 2013. – P. 1-6.
4. *Kravari K., Bassiliades N.* A Survey of Agent Platforms // Journal of Artificial Societies and Social Simulation. – 2015. – Vol. 18, No. 1. – P. 1-18.
5. *Kumar A., Toussaint M., Zilberstein S.* Scalable Multiagent Planning Using Probabilistic Inference // Proceedings of the 22nd International Joint Conference on Artificial Intelligence. – 2011. – P. 2140-2146.
6. *Amato A., Venticinque S.* A Distributed Agent-Based Decision Support for Cloud Brokering // Scalable Computing: Practice and Experience. – 2014. – Vol. 15, No. 1. – P. 65-78.
7. *Frey J., Tannenbaum T., Foster I., Livny M., Tuecke S.* Condor-G: A Computation Management Agent for Multi-Institutional Grids // Journal of Cluster Computing. – 2002. – Vol. 5. – P. 237-246.
8. *YarKhan A., Dongarra J., Seymour K*. GridSolve: The Evolution of a Network Enabled Solver // Grid-based problem solving environments. – 2007. – P. 215-224.
9. *Laxmi CH.V.T.E.V., Somasundaram K.* Application Level Scheduling (AppLeS) in Grid with Quality of Service (QoS) // International Journal of Grid Computing and Applications. – 2014. – Vol. 5, No. 2. – P. 1-10.
10. *Shi Z.* Advanced Artificial Intelligence. – Hackensack: World scientific, 2011. – 624 с.
11. *Rezaee A., Rahmani A.M., Parsa S., Adabi S.* A Multi-Agent Architecture for QoS Support in Grid Environment // Journal of Computer Science. – 2008. – Vol. 4, No. 3. – P. 225-231.
12. *Singh A., Malhotra M.* Agent Based Framework for Scalability in Cloud Computing // International Journal of Computer Science and Engineering Technology. – 2012. – Vol. 3, No. 4. – P. 41-45.
13. *Bogdanova V.G., Bychkov I.V., Korsukov A.S., Oparin G.A., Feoktistov A.G.* Multiagent Approach to Controlling Distributed Computing in a Cluster Grid System // Journal of Computer and Systems Sciences International. – 2014. – Vol. 53, No. 5. – P. 713-722.
14. *Bychkov I.V., Oparin G.A., Feoktistov A.G., Bogdanova V.G., Pashinin A.A.* Service-oriented Multiagent Control of Distributed Computations // Automation and Remote Control. – 2015. – Vol. 76, No. 11. – P. 2000-2010.
15. *Bychkov I.V., Oparin G.A., Feoktistov A.G., Sidorov I.A., Bogdanova V.G., Gorsky S.A.* Multiagent Control of Computational Systems on the Basis of Meta-Monitoring and Imitational Simulation // Optoelectronics, Instrumentation and Data Processing. – 2016. – Vol. 52, No. 2. – P. 107-112.
16. *Unland R., Klusch M., Calisti M.* Software Agent-Based Applications, Platforms and Development Kits. – Birkhauser Verlag, 2005. – 455 p.
17. *Опарин Г.А., Феоктистов Д.Г.* Планирование схем решения задач в инструментальном комплексе САТУРН/ПЗ // Компьютерная логика, алгебра и интеллектное управление: Труды Всероссийской школы. – Иркутск: Изд-во ИрВЦ СО РАН, 1994. – Т. 1. – С. 5-13.
18. *Bellifemine F., Bergenti F., Caire G., Poggi A.* Jade: a Java Agent Development Framework // Multiagent Systems, Artificial Societies, And Simulated Organizations: MultiAgent Programming / Eds. A. Bordini, M. Dastani, J. Dix and A. El Fallax Seghrouchni. – 2006. – Vol. 15. – P. 125-147.
19. *Бычков И.В., Опарин Г.А., Феоктистов А.Г., Богданова В.Г., Сидоров И.А., Пашинин А.А.* Мультиагентный подход к управлению сервис-ориентированными высокопроизводительными вычислениями // Вестник компьютерных и информационных технологий. – 2016. – № 9. – С. 35-41.
20. *Бычков И.В., Опарин Г.А., Богданова В.Г., Горский С.А., Пашинин А.А.* Мультиагентная технология автоматизации параллельного решения булевых уравнений в распределенной вычислительной среде // Вычислительные технологии. – 2016. – Т. 21, № 3. –С. 5-17.

REFERENCES

1. *Qureshi M.B., Dehnavi M.M., Min-Allah N., Qureshi M.S., Hussain H., Rentifis I., Tziritas N., Loukopoulos T., Khan S.U., Xu C.Z., Zomaya A.Y.* Survey on Grid Resource Allocation Mechanisms, *Journal of Grid Computing*, 2014, Vol. 12, No. 2, pp. 399-441.
2. *Talia D.* Cloud Computing and Software Agents: Towards Cloud Intelligent Services, *Proceedings of the 12th Workshop on Objects and Agent*, 2011, pp. 2-6.

3. *Leitao P., Inden U., Ruckemann C.-P.* Parallelising Multi-agent Systems for High Performance Computing, *Proceedings of the 3rd International Conference on Advanced Communications and Computation*, 2013, pp. 1-6.
4. *Kravari K., Bassiliades N.* A Survey of Agent Platforms, *Journal of Artificial Societies and Social Simulation*, 2015, Vol. 18, No. 1, pp. 1-18.
5. *Kumar A., Toussaint M., Zilberstein S.* Scalable Multiagent Planning Using Probabilistic Inference, *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, 2011. pp. 2140-2146.
6. *Amato A., Venticinque S.* A Distributed Agent-Based Decision Support for Cloud Brokering, *Scalable Computing: Practice and Experience*, 2014, Vol. 15, No. 1, pp. 65-78.
7. *Frey J., Tannenbaum T., Foster I., Livny M., Tuecke S.* Condor-G: A Computation Management Agent for Multi-Institutional Grids, *Journal of Cluster Computing*, 2002, Vol. 5, pp. 237-246.
8. *YarKhan A., Dongarra J., Seymour K.* GridSolve: The Evolution of a Network Enabled Solver, *Grid-based problem solving environments*, 2007, pp. 215-224.
9. *Laxmi CH.V.T.E.V., Somasundaram K.* Application Level Scheduling (AppLeS) in Grid with Quality of Service (QoS), *International Journal of Grid Computing and Applications*, 2014, Vol. 5, No. 2, pp. 1-10.
10. *Shi Z.* Advanced Artificial Intelligence, *Hackensack: World scientific*, 2011, 624 p.
11. *Rezaee A., Rahmani A.M., Parsa S., Adabi S.* A Multi-Agent Architecture for QoS Support in Grid Environment, *Journal of Computer Science*, 2008, Vol. 4, No. 3, pp. 225-231.
12. *Singh A., Malhotra M.* Agent Based Framework for Scalability in Cloud Computing, *International Journal of Computer Science and Engineering Technology*, 2012, Vol. 3, No. 4, pp. 41-45.
13. *Bogdanova V.G., Bychkov I.V., Korsukov A.S., Oparin G.A., Feoktistov A.G.* Multiagent Approach to Controlling Distributed Computing in a Cluster Grid System, *Journal of Computer and Systems Sciences International*, 2014, Vol. 53, No. 5, pp. 713-722.
14. *Bychkov I.V., Oparin G.A., Feoktistov A.G., Bogdanova V.G., Pashinin A.A.* Service-oriented Multiagent Control of Distributed Computations, *Automation and Remote Control*, 2015, Vol. 76, No. 11, pp. 2000-2010.
15. *Bychkov I.V., Oparin G.A., Feoktistov A.G., Sidorov I.A., Bogdanova V.G., Gorsky S.A.* Multiagent Control of Computational Systems on the Basis of Meta-Monitoring and Imitational Simulation, *Optoelectronics, Instrumentation and Data Processing*, 2016, Vol. 52, No. 2, pp. 107-112.
16. *Unland R., Klusch M., Calisti M.* Software Agent-Based Applications, *Platforms and Development Kits*. Birkhauser Verlag, 2005, 455 p.
17. *Oparin G.A., Feoktistov D.G.* Planirovanie skhem resheniya zadach v instrumental'nom komplekse SATURN/PZ [Planning of Problem Solving Schemes in Framework SATURN/PZ], *Komp'yuternaya logika, algebra i intellektnoe upravlenie: Trudy Vserossiyskoy shkoly* [Computer logic, algebra and intelligence control: Proceedings of the National school]. Irkutsk: Izd-vo IrVTs SO RAN, 1994, Vol. 1, pp. 5-13.
18. *Bellifemine F., Bergenti F., Caire G., Poggi A.* Jade: a Java Agent Development Framework, *Multiagent Systems, Artificial Societies, And Simulated Organizations: MultiAgent Programming*, Eds. A. Bordini, M. Dastani, J. Dix and A. El Fallax Seghrouchni, 2006, Vol. 15, pp. 125-147.
19. *Bychkov I.V., Oparin G.A., Feoktistov A.G., Bogdanova V.G., Sidorov I.A., Pashinin A.A.* Mul'tiagentnyy podkhod k upravleniyu servis-orientirovannymi vysokoproizvoditel'nymi vychisleniyami [Multiagent approach to control service-oriented high performance computing], *Vestnik komp'yuternykh i informatsionnykh tekhnologiy* [Journal of Computer and Information Technology], 2016, No. 9, pp. 35-41.
20. *Bychkov I.V., Oparin G.A., Bogdanova V.G., Gorskiy S.A., Pashinin A.A.* Mul'tiagentnaya tekhnologiya avtomatizatsii parallel'nogo resheniya bulevykh uravneniy v raspredelennoy vychislitel'noy srede [A multiagent technology of automation for parallel solution of Boolean equations in a distributed computing environment], *Vychislitel'nye tekhnologii* [Computational technologies], 2016, Vol. 21, No. 3, pp. 5-17.

Статью рекомендовал к опубликованию д.ф.-м.н. Ф.И. Иванов.

Феоктистов Александр Геннадьевич – Институт динамики систем и теории управления им. В.М. Матросова СО РАН; e-mail: agf65@yandex.ru; 664033, г. Иркутск, ул. Лермонтова, 134; тел.: +79247116704; с.н.с.; к.т.н.; доцент.

Костромин Роман Олегович – e-mail: romang70055@gmail.com; тел.: +79041150109; аспирант.

Feoktistov Aleksandr Gennadyevich – Matrosov Institute for System Dynamics and Control Theory of SB RAS; e-mail: agf65@yandex.ru; 134, Lermontov street, Irkutsk, Russia; phone: +79247116704; senior research officer; cand. of eng. sc.; associate professor.

Kostromin Roman Olegovich – e-mail: romang70055@gmail.com; phone: +79041150109; post-graduate student.

УДК 004.272 DOI 10.18522/2311-3103-2016-11-7587

М.Г. Курносов

АНАЛИЗ МАСШТАБИРУЕМОСТИ АЛГОРИТМОВ КОЛЛЕКТИВНЫХ ОБМЕНОВ НА РАСПРЕДЕЛЕННЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ*

При разработке параллельных программ для вычислительных систем (ВС) с массовым параллелизмом значительное место по частоте использования и приходящемуся на них суммарному времени выполнения занимают коллективные операции обменов информацией (групповые, глобальные, collective communications). В них участвуют все ветви параллельной программы: трансляционная передача («один-всем», one-to-all broadcast/scatter), коллекторный прием («все-одному», all-to-one gather/reduce), трансляционно-циклический обмен («каждый-всем», all-to-all gather/reduce). Для реализации каждой коллективной операции, имеется множество алгоритмов, из которого необходимо выбрать оптимальный – обеспечивающий минимум времени выполнения операции. Для такого сравнительного анализа в моделях параллельных вычислений (BSP – bulk synchronous parallel, Дж. Хокни, LogP, LogGP, PLogP) строят аналитические оценки времени выполнения алгоритмов, как функции от параметров системы и коллективной операции: числа процессоров, показателей производительности каналов связи, размеров передаваемых сообщений. В данной работе для коллективной операции корневой редукции (all-to-one reduce) в модели параллельных вычислений LogP построены аналитические выражения (оценки) времени выполнения алгоритмов ее реализации. В отличие от известных работ, выражения построены для общих и частных (особых) случаев значений параметров вычислительной системы и коллективных операций. Для учета копирования сообщений в памяти вычислительных узлов модель LogP расширена дополнительным параметром λ – время, требуемое на копирование одного байта в памяти вычислительного узла. На примере алгоритма -параллельных цепочек продемонстрирован подход к построению оптимальных в модели LogP алгоритмов коллективных операций. Предложенный оптимизированный алгоритм -параллельных цепочек реализован в стандарте MPI. Результаты экспериментов на вычислительных кластерах с сетями связи стандарта MPI подтверждают полученные теоретические результаты, в частности рекомендации относительно выбор числа k цепочек. Выбор конкретной математической модели параллельных вычислений обусловлен спецификой алгоритма и целевой ВС. Например, если алгоритм реализует группировку сообщений в пакеты больших размеров, то целесообразно использовать модель LogGP, которая в явном виде учитывает издержки на передачу сообщений больших размеров.

Коллективные обмены; глобальные обмены; LogP; MPI; параллельное программирование; вычислительные системы.

* Работа выполнена при поддержке РФФИ (проекты 15-37-20113, 15-07-00653).