ВЕСТНИК

ЮЖНО-УРАЛЬСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА

2016
Т. 5, № 4

ISSN 2305-9052

СЕРИЯ

«ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА И ИНФОРМАТИКА»

Решением ВАК включен в Перечень научных изданий, в которых должны быть опубликованы результаты диссертаций на соискание ученых степеней кандидата и доктора наук

Учредитель — Федеральное государственное автономное образовательное учреждение высшего образования «Южно-Уральский государственный университет» (национальный исследовательский университет)

Тематика журнала:

- Вычислительная математика и численные методы
- Математическое программирование
- Распознавание образов
- Вычислительные методы линейной алгебры
- Решение обратных и некорректно поставленных задач
- Доказательные вычисления
- Численное решение дифференциальных и интегральных уравнений
- Исследование операций
- Теория игр
- Теория аппроксимации
- Информатика
- Математическое и программное обеспечение высокопроизводительных вычислительных систем
- Системное программирование
- Перспективные многопроцессорные архитектуры
- Облачные вычисления
- Технология программирования
- Машинная графика
- Интернет-технологии
- Системы электронного обучения
- Технологии обработки баз данных и знаний
- Интеллектуальный анализ данных

BULLETIN
OF THE SOUTH URAL STATE UNIVERSITY

2016
vol. 5, no. 4

SERIES
"COMPUTATIONAL MATHEMATICS AND SOFTWARE ENGINEERING"

ISSN 2305-9052

Vestnik Yuzhno-Ural'skogo Gosudarstvennogo Universiteta.
Seriya "Vychislitel'naya Matematika i Informatika"

South Ural State University

The scope of the journal:

- Numerical analysis and methods
- Mathematical optimization
- Pattern recognition
- Numerical methods of linear algebra
- Reverse and ill-posed problems solution
- Computer-assisted proofs
- Numerical solutions of differential and integral equations
- Operations research
- Game theory
- Approximation theory
- Computer science
- High performance computing
- System software
- Advanced multiprocessor architectures
- Cloud computing
- Software engineering
- Computer graphics
- Internet technologies
- E-learning
- Database processing
- Data mining

Содержание

Вычислительная математика

РЕАЛИЗАЦИЯ ИТЕРАЦИОННЫХ МЕТОДОВ РЕШЕНИЯ СИСТЕМ ЛИНЕЙНЫХ УРАВНЕНИЙ В ЗАДАЧАХ МАТЕМАТИЧЕСКОЙ ФИЗИКИ НА РЕКОНФИГУРИРУЕМЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ
И.И. Левин, А.И. Дордопуло, А.В. Пелипец 5

Дискретная математика и математическая кибернетика

ПРИМЕНЕНИЕ МАТЕМАТИЧЕСКОГО МОДЕЛИРОВАНИЯ ДЛЯ ВЫБОРА ИНВЕСТИЦИОННОЙ ПРОГРАММЫ ПРЕДПРИЯТИЯ
А.В. Панюков, Е.Н. Козина 19

Информатика, вычислительная техника и управление

SUPERCOMPUTER APPLICATION INTEGRAL CHARACTERISTICS ANALYSIS FOR THE WHOLE QUEUED JOB COLLECTION OF LARGE-SCALE HPC SYSTEMS
D.A. Nikitenko, V.V. Voevodin, A.M. Teplov, S.A. Zhumatiy, Vad.V. Voevodin, K.S. Stefanov, P.A. Shvets 32

ОПТИМИЗАЦИЯ ОБНАРУЖЕНИЯ КОНФЛИКТОВ В ПАРАЛЛЕЛЬНЫХ ПРОГРАММАХ С ТРАНЗАКЦИОННОЙ ПАМЯТЬЮ
И.И. Кулагин, М.Г. Курносов 46

ОБЗОР АДАПТИВНЫХ МОДЕЛЕЙ ЭЛЕКТРОННОГО ОБУЧЕНИЯ
Н.С. Силкина, Л.Б. Соколинский 61

Суперкомпьютерное моделирование

ЧИСЛЕННОЕ ГИДРОДИНАМИЧЕСКОЕ МОДЕЛИРОВАНИЕ АСТРОФИЗИЧЕСКИХ ТЕЧЕНИЙ НА ГИБРИДНЫХ СУПЕРЭВМ, ОСНАЩЕННЫХ УСКОРИТЕЛЯМИ INTEL XEON PHI
И.М. Куликов, И.Г. Черных, Э.И. Воробьев, А.В. Снытников, Д.В. Винс, А.А. Московский, А.Б. Шмелёв, В.А. Протасов, А.А. Серенко, В.Е. Ненашев, В.А. Вшивков, А.С. Родионов, Б.М. Глинский, А.В. Тутуков 77

Персоналии

АНАТОЛИЙ ВАСИЛЬЕВИЧ ПАНЮКОВ (К 65-ЛЕТИЮ СО ДНЯ РОЖДЕНИЯ)
В.И. Дударева 98

Contents

Computational Mathematics

IMPLEMENTATION OF ITERATION METHODS FOR SOLUTION OF LINEAR EQUATION SYSTEMS IN PROBLEMS OF MATHEMATICAL PHYSICS ON RECONFIGURABLE COMPUTER SYSTEMS
I.I. Levin, A.I. Dordopulo, A.V. Pelipets 5

Discrete Mathematics and Mathematical Cybernetics

APPLICATION OF MATHEMATICAL MODELLING FOR CHOOSING COMPANY'S INVESTMENT PROGRAM
A.V. Panyukov, E.N. Kozina 19

Computer Science, Engineering and Control

SUPERCOMPUTER APPLICATION INTEGRAL CHARACTERISTICS ANALYSIS FOR THE WHOLE QUEUED JOB COLLECTION OF LARGE-SCALE HPC SYSTEMS
D.A. Nikitenko, V.V. Voevodin, A.M. Teplov, S.A. Zhumatiy, Vad.V. Voevodin, K.S. Stefanov, P.A. Shvets 32

OPTIMIZATION OF CONFLICT DETECTION IN PARALLEL PROGRAMS WITH TRANSACTIONAL MEMORY
I.I. Kulagin, M.G. Kurnosov 46

SURVEY OF ADAPTIVE E-LEARNING MODELS
N.S Silkina, L.B. Sokolinsky 61

Supercomputer Simulation

NUMERICAL HYDRODYNAMICS SIMULATION OF ASTROPHYSICAL FLOWS AT INTEL XEON PHI SUPERCOMPUTERS
I.M. Kulikov, I.G. Chernykh, E.I. Vorobyov, A.V. Snytnikov, D.V. Weins, A.A. Moskovsky, A.B. Shmelev, V.A. Protasov, A.A. Serenko, V.E. Nenashev, V.A. Vshivkov, A.S. Rodionov, B.M. Glinsky, A.V. Tutukov 77

Personalities

ANATOLY V. PANYUKOV (TO THE 65TH ANNIVERSARY)
V.I. Dudareva 98

УДК 004.382.2 DOI: 10.14529/cmse160401

РЕАЛИЗАЦИЯ ИТЕРАЦИОННЫХ МЕТОДОВ РЕШЕНИЯ СИСТЕМ ЛИНЕЙНЫХ УРАВНЕНИЙ В ЗАДАЧАХ МАТЕМАТИЧЕСКОЙ ФИЗИКИ НА РЕКОНФИГУРИРУЕМЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ

© 2016 г. И.И. Левин, А.И. Дордопуло, А.В. Пелипец

Научно-исследовательский институт многопроцессорных вычислительных систем Южного федерального университета (Таганрог, Российская Федерация) (347928 Таганрог, ул. Чехова, 2, ГСП-284, г. Таганрог, Ростовская область), E-mail: levin@mvs.tsure.ru, scorpio@mvs.tsure.ru, pelipets@mail.ru

Поступила в редакцию: 12.07.2016

В статье рассматриваются характерные особенности реализации итерационных методов решения систем линейных уравнений в задачах математической физики на параллельных вычислительных системах, которыми являются геометрическая декомпозиция расчетной области и распараллеливание данных внутри последовательно выполняемых процессором итераций с интенсивным информационным обменом между процессорами и памятью. Стандартные методы реализации итерационных методов решения систем линейных уравнений при множественных транзакциях с памятью и межпроцессорных обменах, существенно снижающих реальную производительность, дополнительно требуют от вычислительной системы наличия большого числа физических линий связи для реализации сложных топологий и иерархических схем хранения данных. Выходом является использование многопроцессорных систем с реконфигурируемой архитектурой, позволяющее адаптировать свою архитектуру под структуру итерационных алгоритмов математической физики путем распараллеливания по итерациям. В статье рассмотрена реализация метода Якоби для решения краевой задачи Дирихле для уравнения Лапласа на реконфигурируемой вычислительной системе, на примере которой показано сокращение количества внешних каналов обмена как одного из наиболее критических ресурсов реконфигурируемой вычислительной системы.

Ключевые слова: реконфигурируемые вычислительные системы, программируемые логические интегральные схемы, численные методы математической физики, распараллеливание по итерациям, вычислительный конвейер.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Левин И.И., Дордопуло А.И., Пелипец А.В. Реализация итерационных методов решения систем линейных уравнений в задачах математической физики на реконфигурируемых вычислительных системах // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2016. Т. 5, № 4. С. 5–18. DOI: 10.14529/cmse160401.

Введение

В процессе численного решении модельных уравнений математической физики часто возникает необходимость нахождения неизвестных в системах линейных алгебраических уравнений (СЛАУ) высокого порядка (для числа уравнений $n > 1000$). В данных условиях прямые методы, такие, как метод исключений Гаусса, могут быть неэффективны из-за высоких требований к оперативной памяти, большой вычислительной трудоемкости и накопления ошибок округления [1]. Поэтому предпочтительными становятся итерационные методы, которые позволяют найти приближенное решение СЛАУ за конечное число итераций, количество которых зависит от требуемой точности.

Параллельная форма реализации итерационных методов решения СЛАУ основана на концепции геометрического параллелизма, при котором исходная расчетная область, описывающая состояние физической среды в определенных точках пространства, разбивается по одному или двум пространственным направлениям с последующим распределением подобластей между процессорами. Ключевой особенностью реализаций итерационных методов на параллельных многопроцессорных системах является то, что итерации выполняются последовательно, а распараллеливание задачи происходит по данным на уровне вычислений внутри итерации [2]. При этом каждый процессор обрабатывает свою часть области в течение всех итераций, обмениваясь результатами с другими процессорами и с внешней оперативной памятью.

Однако необходимость интенсивных взаимодействий типа процессор-процессор и процессор-память сопряжена с большими накладными расходами, порой сводящими на нет весь эффект от распараллеливания. Особенно актуальной эта проблема становится по мере увеличения масштабов многопроцессорных систем и количества компонентов, входящих в них [3].

На параллельных вычислительных системах с «жесткой» архитектурой способы решения данной проблемы, в основном, сводятся к выбору подходящей топологии и организации системы хранения данных в виде сложной иерархической структуры. Это, в свою очередь, порождает проблему дефицита каналов памяти для обеспечения доступа к различным уровням иерархии.

Для решения указанных проблем активно проводятся исследования, связанные с поиском способов ускорения решения СЛАУ итерационными методами на высокопроизводительных системах. Работа [4] описывает применение гибридной версии технологий параллельного программирования OpenMP–MPI для сокращения накладных расходов вычислений в задачах механики жидкости методом конечных элементов. В статье [5] делается обзор опыта применения графических ускорителей для решения итерационными методами разреженных систем линейных уравнений, приводится сравнение эффективности реализаций алгоритмов на GPU NVIDIA TESLA с CPU Intel Xeon, использующим специализированные библиотеки линейной алгебры Intel MKL. Публикация [6] посвящена асинхронным итерационным алгоритмам, позволяющим уменьшить коммуникационные расходы по сравнению с другими алгоритмами, за счет устранения простоя узлов из-за ожидания синхрособщений. Приведенные экспериментальные результаты показывают, что модифицированный релаксационный метод Саусвелла уменьшает затраты на коммуникации по сравнению с другими асинхронными итерационными методами и классическим синхронным методом Якоби. Множество работ касаются использования гетерогенных вычислительных архитектур в решении разреженных СЛАУ итерационными методами. В частности, вопросам построения адаптированных итерационных алгоритмов, эффективно распределяющих вычислительную нагрузку в системах CPU-GPU, посвящены статьи [7–9].

В настоящее время альтернативой традиционным многопроцессорным архитектурам при решении задач математической физики все чаще выступают реконфигурируемые вычислительные системы (РВС), построенные на основе программируемых логических интегральных схем (ПЛИС) [10]. Здесь также проблема дефицита внешних каналов обмена является весьма актуальной, поскольку удельное количество пользовательских выводов относительно логической емкости кристалла снижается в каждом поколении

ПЛИС [11–13]. Применение различных методов распараллеливания по данным лишь усугубляет эту проблему.

«Гибкая» архитектура РВС позволяет, не меняя исходный алгоритм задачи, адаптировать вычислитель под ее информационную структуру, что дает возможность организовать для матричных задач не только геометрическое распараллеливание (по данным), но и временное (по итерациям). Это ведет к существенному сокращению количества внешних каналов обмена, являющихся одним из наиболее критических ресурсов реконфигурируемой вычислительной системы.

Целью данной работы является разработка и исследование алгоритма решения на РВС систем линейных алгебраических уравнений методом Якоби для матриц специального вида в задачах математической физики. В разделе 1 настоящей статьи дается описание преобразования краевой задачи Дирихле для уравнения Лапласа для решения ее в структурно-процедурной форме. В разделе 2 анализируются практические результаты решения на РВС краевой задачи Дирихле для уравнения Лапласа, приводится сравнение полученных результатов с реализацией данной задачи на параллельной кластерной системе. В заключении обсуждаются перспективы применения реконфигурируемых систем для решения систем линейных уравнений итерационными методами в задачах математической физики и указываются направления дальнейших исследований.

1. Преобразование в структурно-процедурную форму краевой задачи Дирихле для уравнения Лапласа

Рассмотрим в качестве примера решение задачи Дирихле для уравнения Лапласа, которое используется для описания различных стационарных (установившихся) физических процессов [14]. Пусть требуется решить задачу Дирихле для уравнения Лапласа, имеющего общий вид $\Delta u(x,y,z) = 0$, которое задано в двумерном пространстве дифференциальной формой

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0, \tag{1}$$

где $u(x,y)$ – функция с граничными условиями первого рода, значение которой необходимо вычислить.

Тогда в результате дискретизации уравнения (1) методом конечных разностей на квадратной координатной сетке ϖ согласно пятиточечному шаблону получим разностную (алгебраическую) форму

$$\frac{u_{i+1,j} - 2u_{ij} + u_{i-1,j}}{h^2} + \frac{u_{i,j+1} - 2u_{ij} + u_{i,j-1}}{h^2} = 0, \tag{2}$$

где $i,j = 1, 2, ..., N\text{-}1$;

h – постоянный шаг сетки, $hN = 1$.

Если построить сетку с постоянным шагом $h = \frac{1}{10}$, то поскольку порядок системы равен $n = (N\text{-}1)^2$, получим систему из 81 уравнения, которую можно последовательно решить методом Якоби с помощью формулы общего вида

$$u_{ij}^{(k+1)} = \frac{1}{4}(u_{i+1,j}^{k} + u_{i-1,j}^{k} + u_{i,j+1}^{k} + u_{i,j-1}^{k}), \tag{3}$$

где k – номер итерации.

Общая формула (3) эквивалентна системе линейных уравнений

$$\begin{cases} u_{11}^{k+1} = \frac{1}{4}\left(u_{21}^{k} + u_{01}^{k} + u_{12}^{k} + u_{10}^{k}\right) \\ u_{12}^{k+1} = \frac{1}{4}\left(u_{22}^{k} + u_{02}^{k} + u_{13}^{k} + u_{11}^{k}\right) \\ \dots \\ u_{99}^{k+1} = \frac{1}{4}\left(u_{10,9}^{k} + u_{89}^{k} + u_{9,10}^{k} + u_{98}^{k}\right) \end{cases} \quad (4)$$

Исходя из того, что значения функции $u(x,y)$ на границе рассматриваемой области известны заранее, система (4) может быть преобразована к виду

$$\begin{cases} u_{11}^{k+1} = \gamma\left(u_{21}^{k} + c_{01} + u_{12}^{k} + c_{10}\right) \\ u_{12}^{k+1} = \gamma\left(u_{22}^{k} + c_{02} + u_{13}^{k} + u_{11}^{k}\right) \\ \dots \\ u_{99}^{k+1} = \gamma\left(c_{10,9} + u_{89}^{k} + c_{9,10} + u_{98}^{k}\right) \end{cases}, \quad (5)$$

где c_{ij} – известные граничные значения функции;

γ – константное значение множителя, $\gamma = \frac{1}{4}$.

Итерационных процесс вычислений завершается при выполнении условия

$$| u_{ij}^{k+1} - u_{ij}^{k} | \le \varepsilon, \quad (6)$$

где ε – это требуемая точность.

Система (5) позволяет отобразить процесс вычислений неизвестных u_{ij} на k+1-ой итерации в виде полного информационного графа β, представленного в параллельном виде на рис. 1. Здесь информационные вершины соответствуют множеству исходных данных и результатов, операционные вершины изображают операции с данными, а дуги показывают направления информационного обмена между вершинами.

Особенности граничных условий первого рода определяют количество и порядок информационных дуг, соединяющих информационные вершины-источники с операционными вершинами-приемниками, а именно: адресные вершины $\langle u_{11}^{k}$, $u_{1,N-1}^{k}$, $u_{N-1,1}^{k}$, $u_{N-1,N-1}^{k}\rangle$, соответствующие точкам, примыкающим к границе в углах области решения, соединяются с операционными вершинами-приемниками двумя дугами.

Адресные вершины u_{ij}^{k}, соответствующие точкам, примыкающим к границе области (исключая угловые точки $\langle u_{11}^{k}$, $u_{1,N-1}^{k}$, $u_{N-1,1}^{k}$, $u_{N-1,N-1}^{k}\rangle$), соединяются с операционными вершинами-приемниками тремя дугами. Адресные вершины u_{ij}^{k}, соответствующие остальным точкам внутри области, имеют по четыре исходящих дуги. Адресные вершины c_{ij}, соответствующие граничным точкам области решения (заданные значения функции), имеют по одной исходящей дуге.

Схема информационных зависимостей вершин, соответствующих узлам сетки внутри области решения, проиллюстрирована на рис. 2.

В пределах одной итерации информационный граф параллельной формы данной задачи можно рассматривать как кортеж информационно-независимых подграфов $\langle \alpha^k_1, \alpha^k_2, \ldots, \alpha^k_n \rangle$. Информационная независимость подграфов α_i определяется отсутствием дуг, соединяющих операционные вершины двух или более подграфов в кортеже.

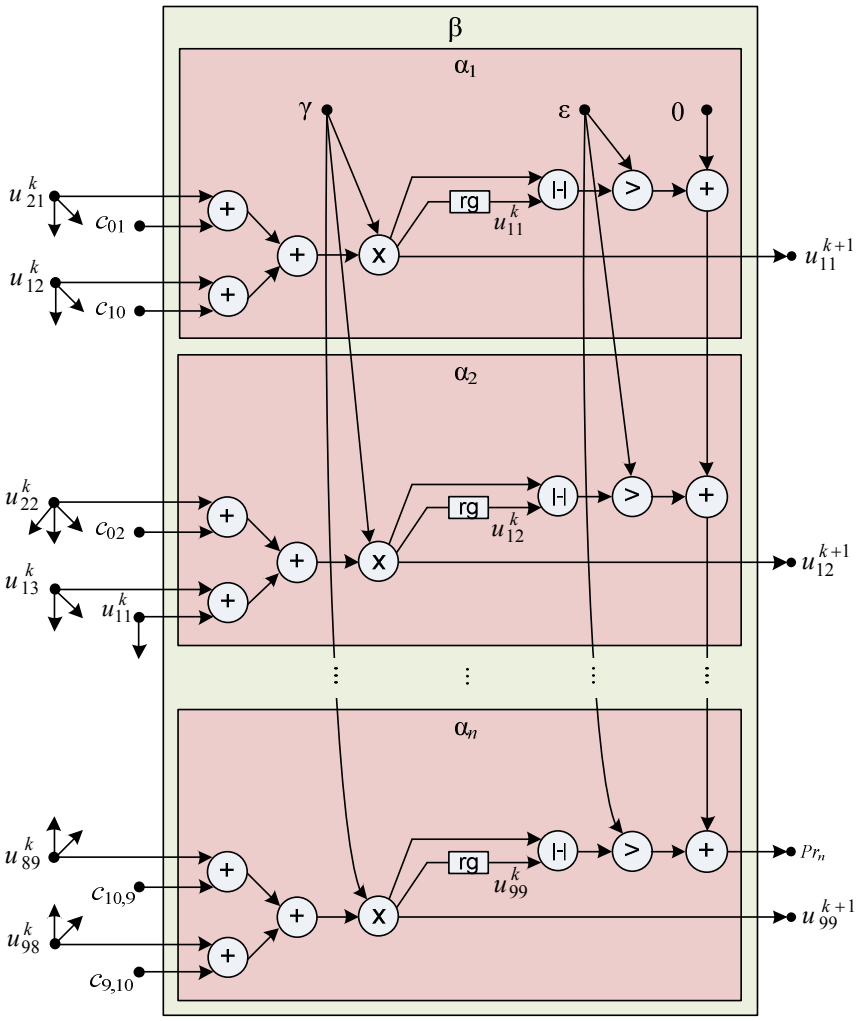


Рис. 1. Параллельная форма метода Якоби решения уравнения Лапласа

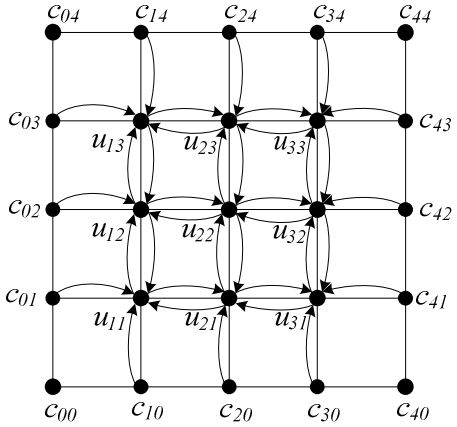


Рис. 2. Схема информационных зависимостей вершин, соответствующих узлам сетки с шагом $h = \frac{1}{4}$

В соответствии с пятиточечным шаблоном каждый из n изоморфных информационных подграфов α_i на $(k+1)$-й итерации реализует одно умножение и три сложения операндов для получения на выходе очередного узлового значения функции u_{ij}^{k+1}. Кроме этого, операции модуля разности, сравнения и суммирования реализуют процедуру проверки условия сходимости (6), результатом которой является предикат Pr_n. Управляющая программа останавливает вычислительный процесс при обнаружении предиката, значение которого говорит о выполненном условии сходимости.

Общее число операционных вершин полного графа β параллельной формы для решения n систем уравнений будет составлять $7n$. Если бы количество итераций, которые необходимо пройти до завершения процесса вычислений, было известно заранее (например, m), то алгоритм мог быть представлен в абсолютно-параллельной форме с числом базовых подграфов, равным $n \cdot m$. Но поскольку необходимое количество итераций изначально не определено и зависит от выполнения условий сходимости, то число информационных подграфов в параллельной форме равно размерности матрицы n, а количество реализованных ими итераций может быть произвольным. Промежуточные результаты итерации k, являющиеся исходными данными для итерации $(k+1)$, должны записываться и считываться из внешней оперативной памяти.

Количество аппаратного ресурса для реализации параллельной формы задачи рассчитывается по формуле

$$A = n \cdot N(\alpha_i), \tag{7}$$

где $N(a_i)$ – количество ресурса, необходимого для реализации базового информационного подграфа a_i.

Реализация параллельной формы метода Якоби (рис. 1) возможна при наличии в вычислительной системе количества процессоров, достаточного для параллельной обработки всех узлов координатной сетки ϖ. Из (7) следует, что структурная реализация параллельной формы задачи требует наличия $7n$ процессорных элементов и $5n$ внешних каналов обмена. Для систем уравнений больших размерностей необходимый ресурс может превосходить параллельные возможности аппаратуры. В этом случае необходимо перейти от параллельной к структурно–процедурной (кадровой) форме вычислений и провести редукцию производительности путем сокращения числа подграфов, операционных и адресных вершин. При переходе к кадровой форме в результате векторизации подмножества информационных вершин представляются в виде адресных вершин–массивов G, функция которых состоит в генерации потока адресов обращения к памяти, где хранятся массивы обрабатываемых данных (результатов), а также вывод операндов в вычислительные устройства.

Пусть L – общий объем аппаратного ресурса вычислительной системы, такой, что $L<A$. Тогда реализация структурно–процедурной кадровой формы задачи может быть осуществлена путем редукции производительности по числу базовых подграфов с коэффициентом

$$r = \left]\frac{A}{L}\right[. \tag{8}$$

Если аппаратного ресурса вычислительной системы достаточно для структурной реализации не более, чем r базовых подграфов α_i, то кадровая форма задачи может быть получена путем сокращения полного графа β в $\frac{n}{r}$ раз. Степень редукции производи-

тельности по подграфам [15] определяется возможностью структурной реализации определенного числа базовых подграфов в теле кадра. Предельным случаем редукции по подграфам является кадр Q, состоящий из одного базового подграфа α (рис. 3). При этом вычислительный процесс расчета узловых значений функции на текущей итерации $(k+1)$ будет представлять собой циклическое (от 1 до n) выполнение кадра с получением на каждом шаге цикла очередного приближения u_{ij}^{k+1}.

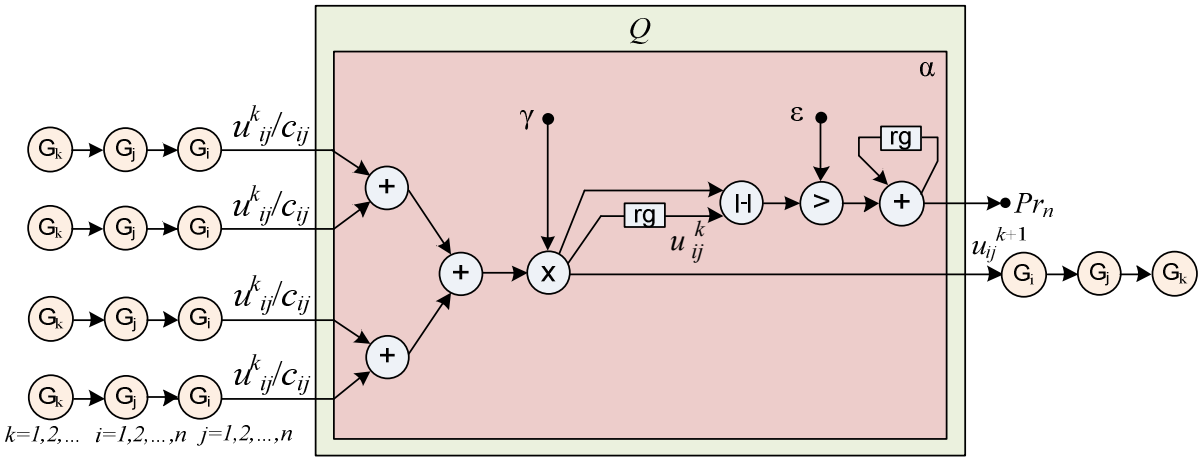


Рис. 3. Кадровая форма метода Якоби решения уравнения Лапласа, состоящая из одного базового подграфа

Особенностью данного кадра является то, что выходом адресных вершин–массивов G являются как вычисленные на предыдущей итерации узловые значения функции u_{ij}^k, так и граничные значения c_{ij}, представляющие собой константы. И константы, и найденные приближения сеточной функции хранятся в одном массиве внешней оперативной памяти, в котором обновляются только значения u_{ij}^{k+1}, вычисленные на текущей итерации.

Дальнейшая редукция производительности путем сокращения функциональных устройств базового подграфа α_i может быть осуществлена путем замены трех вершин суммирования значений функции на одну, где сложения будут производиться последовательно. Это позволит не только сократить аппаратные затраты на функциональные устройства, но и уменьшить количество внешних коммуникационных каналов. В данном случае число операционных вершин базового подграфа было сокращено с семи до пяти, а количество внешних каналов уменьшилось вдвое – с шести до трех (рис. 4).

Особенностью разностной аппроксимации оператора Лапласа на пятиточечном шаблоне является то, что каждое уравнение системы (кроме точек, примыкающих к граничным областям) содержит пять неизвестных, используемых для нахождения приближенных значений u_{ij} функции $u(x, y)$ в узлах двумерной сетки (x_i,y_j). Поэтому обратная связь накапливающего сумматора в базовом подграфе содержит четыре регистра rg для хранения промежуточных результатов суммирования. При аппроксимации на шаблонах, содержащих большее число точек, количество регистров в обратной связи накапливающего сумматора должно быть соответственно увеличено.

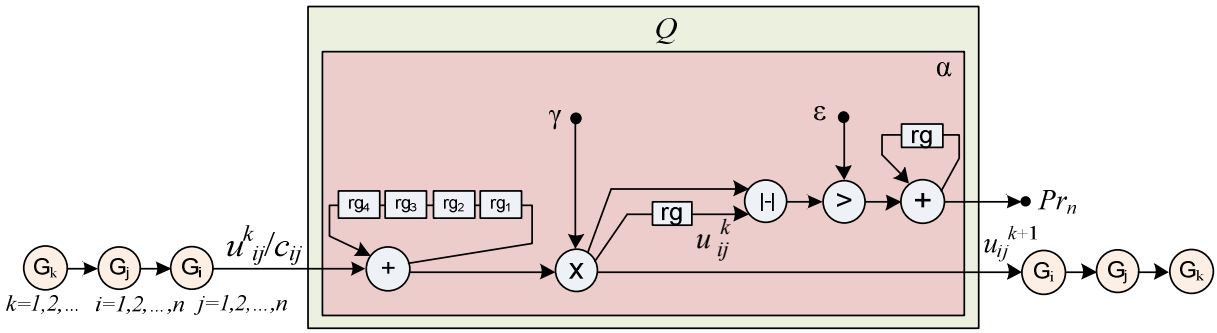


Рис. 4. Кадровая форма метода Якоби решения уравнения Лапласа с предельно сокращенным числом базовых подграфов и операционных вершин

Кадровая форма задачи может состоять как из нескольких, так и из одного предельно редуцированного базового подграфа. Выбор реализации, состоящей из одного базового подграфа, главным образом обусловлен коммуникационными ресурсами вычислительной системы и возможностью построения конвейерной структуры с минимально возможным числом внешних и внутренних информационных каналов.

При параллельно–конвейерной организации вычислительного процесса базовый подграф, соответствующий одной ступени конвейера, будет иметь вид, представленный на рис. 5.

Каждая ступень конвейера реализует функционально законченный фрагмент P, общее количество которых упорядочено в кортеж $\langle P^1, P^2, \ldots, P^k \rangle$ по числу итераций k.

На каждой итерации $(k+1)$ на вход ступени поступает поток операндов, представляющих собой константные значения функции c_{ij} на границе области и ее приближения u_{ij}^k, полученные на предыдущей итерации. Выходом ступени является поток транзитных данных c_{ij} и рассчитанных значений u_{ij}^{k+1}, которые поступают на вход следующей ступени для вычисления узловых значений функции на следующей итерации.

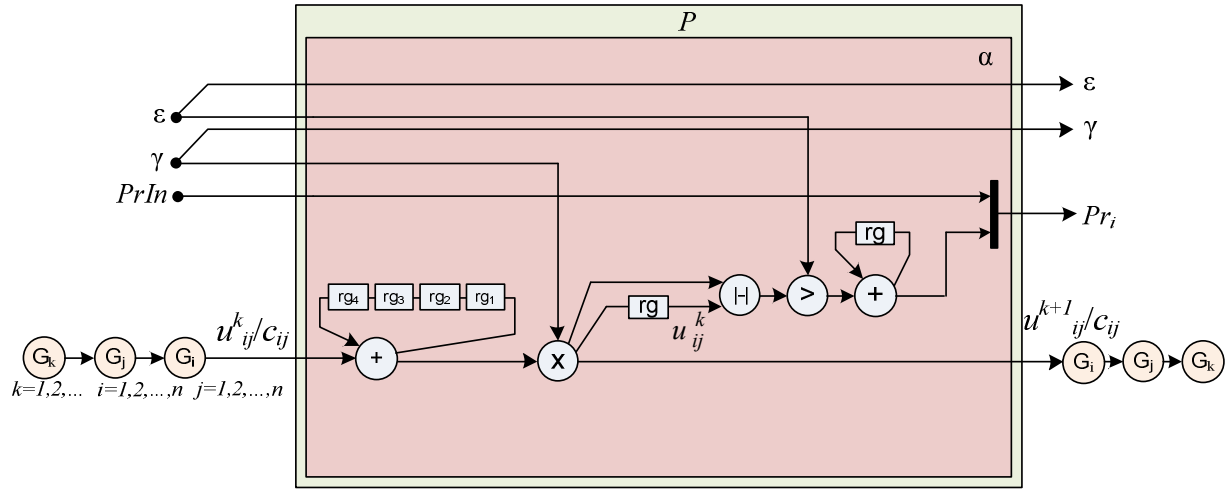


Рис. 5. Базовый подграф метода Якоби решения уравнения Лапласа, соответствующий одной ступени вычислительного конвейера

Помимо информационных дуг, связанных с поступлением и выдачей вычисляемых приближений функции, структура базового подграфа должна быть дополнена связями, осуществляющими транзит констант ε, γ в следующую ступень конвейера. Также в структуру подграфа вводится дополнительный вход $PrIn_i$, на который подается значе-

ние предиката предыдущей ступени. Выходной предикат Pr_i, формируемый внутри подграфа, получается в результате мультиплексирования потока предикатов предыдущих ступеней и собственного вычисленного предиката.

Таким образом, в вычислительном конвейере, представленном на рис. 6, достигается параллелизм решения задачи на уровне итераций.

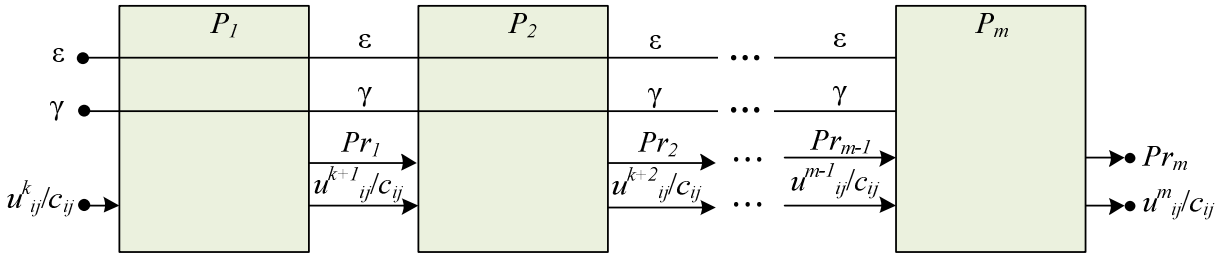


Рис. 6. Параллельно–конвейерная форма метода Якоби решения уравнения Лапласа с применением распараллеливания по итерациям

2. Практические результаты решения на РВС краевой задачи Дирихле для уравнения Лапласа

Численное решение задачи Дирихле для уравнения Лапласа методом Якоби было реализовано авторами на базовом вычислительном модуле «Тайгета» [16]. В результате аппроксимации 5–точечной разностной схемы на квадратной сетке с шагом $h = 0{,}01$ была получена СЛАУ размерностью $n = 10^4$. При размещении в одной ПЛИС XC7VX485T–1FFG1761 ($48{\cdot}10^6$ эквивалентных вентилей, 400 МГц, 192 элементарных процессора стандарта IEEE-754) базового модуля 59 ступеней конвейера время решения СЛАУ составило 0,22 с. Для решения этой задачи на персональном компьютере (Intel Core i5–3570K, 3.4 ГГц, 8 Гб ОЗУ) при использовании одного ядра процессора потребовалось 4,61 с. Таким образом, на одной ПЛИС по сравнению с ПК было получено ускорение вычислений примерно в 21 раз.

Всего базовый модуль «Тайгета» содержит 32 ПЛИС, что позволяет разместить в едином решающем поле вычислительный конвейер, состоящий из 1888 ступеней.

Масштабирование алгоритма продемонстрировало увеличение скорости решения задачи при увеличении числа используемых кристаллов ПЛИС. На графиках на рис. 7 и 8 представлены полученные экспериментальные результаты для времени решения задачи и ускорения по сравнению с одним ядром процессора Intel Core i5–3570K, 3.4 ГГц для количества ПЛИС от 1 до 32.

Результаты вычислительных экспериментов для параллельной реализации на кластере МВС–1000/ИВМ алгоритма численного решения методом Якоби краевой задачи на кластерной многопроцессорной вычислительной системе для числа процессов от 1 до 22 приведены в работе [17]. Согласно полученным в [17] экспериментальным данным, стабильное увеличение ускорения при решении задачи для всех рассматриваемых в [17] вариантов реализации коммуникаций и способа декомпозиции вычислительной области наблюдалось вплоть до использования 22–х процессоров, после чего ускорение решения задачи приобрело неустойчивый характер по причине преобладания времени межпроцессорных обменов над вычислениями.

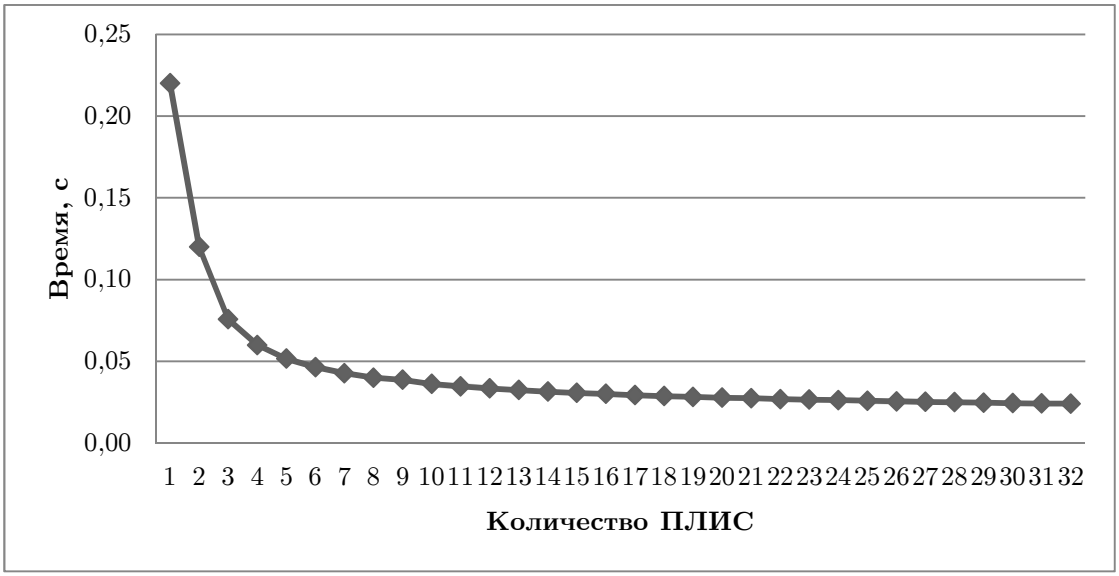


Рис. 7. Метод Якоби решения уравнения Лапласа. Зависимость времени вычислений от числа используемых для решения задачи ПЛИС

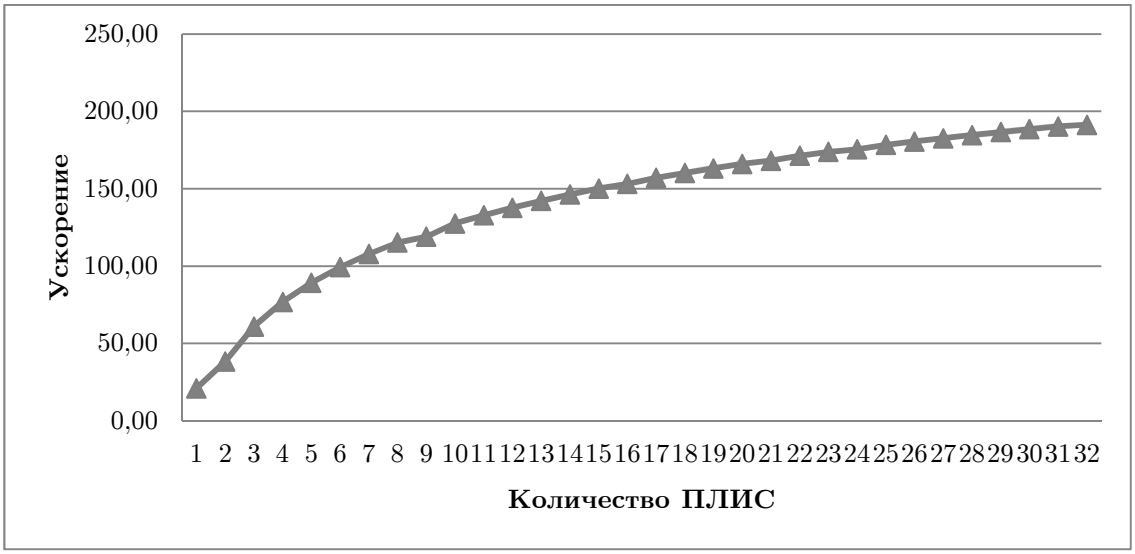


Рис. 8. Метод Якоби решения уравнения Лапласа. Изменение ускорения в зависимости от числа используемых для решения задачи ПЛИС

Согласно данным, приведенным на рис.7 и 8 при использовании ПЛИС для численного решения задачи Дирихле для уравнения Лапласа методом Якоби наблюдается устойчивый рост ускорения решения задачи при увеличении используемого вычислительного ресурса до 32 кристаллов ПЛИС включительно. Параллельно–конвейерная форма реализации матричных задач минимизирует использование внешних выводов, но остается требовательной к остальным вычислительным ресурсам системы – тактовой частоте, объему логики и количеству внутрикристальной памяти ПЛИС. Использование ПЛИС для решения матричных задач позволяет существенно увеличить производительность, в том числе за счет использования кристаллов новых семейств. Так, для каждого следующего семейства ПЛИС Xilinx, начиная с семейства Virtex-4, наблюдается не менее чем 25% рост тактовых частот и, как правило, двукратное увеличение логической

емкости ПЛИС. Ресурс встроенной блочной памяти в каждой серии растет еще большими темпами: реализованное на 16 нм техпроцессе семейство Virtex UltraScale+ имеет в четыре раза больше памяти, чем в предыдущей серии Virtex–7 с нормами 28 нм. Поскольку объем внутрикристальной памяти имеет критическое значение для успешной реализации задач решения больших СЛАУ, прогресс этого показателя делает использование ПЛИС весьма перспективным.

Заключение

Предложенный конвейерно-параллельный способ решения систем линейных уравнений в задачах математической физики может быть применен и к другим итерационным методам, представимым в потоковом виде, среди них: метод Гаусса—Зейделя, попеременно-треугольный метод и так далее. Реализация этого способа сегодня осуществима на ПЛИС и построенных на их основе реконфигурируемых системах, что говорит в пользу широкого использования программируемых архитектур в высокопроизводительных вычислениях.

Применение метода распараллеливания по итерациям позволяет существенно сократить количество внешних каналов обмена по сравнению с методами распараллеливания по данным. Поскольку стандартные методы и средства в настоящее время не могут в полной мере решить проблему падения реальной производительности в ряде задач с интенсивным доступом к данным, имеет смысл использовать альтернативные многопроцессорные архитектуры с «гибкой» реализацией различных форм распараллеливания.

Среди направлений дальнейших исследований можно выделить поиск эффективных алгоритмов решения на РВС разреженных СЛАУ итерационными методами с различного вида предобуславливанием (на основе неполного LU–разложения, на основе аппроксимации обратной матрицы, на основе алгебраического многосеточного метода и так далее). В настоящее время остается недостаточно изученным вопрос о соотношении времени, затрачиваемого на построение матрицы предобуславливателя, и последующего процесса нахождения решения с сокращенным количеством итераций.

Работа выполнена при частичной финансовой поддержке Южного научного центра Российской академии наук по теме№0256-2015-0080 (00–16–15) государственного задания в рамках программы I.5П «Проблемы создания высокопроизводительных, распределенных и облачных систем и технологий. Интеллектуальные информационные технологии и системы».

Литература

1. Самарский А.А., Гулин А.В. Численные методы математической физики: учебное пособие. 2-е изд. М.: Научный мир, 2003. 316 с.
2. Гергель В.П. Теория и практика параллельных вычислений. М.: БИНОМ, 2007. 424 с.
3. Asanovic K. et al. A view of the parallel computing landscape // Communications of the ACM. 2009. Vol. 52, No. 10. P. 56–67.
4. Guo X. et al. Developing a scalable hybrid MPI/OpenMP unstructured finite element model // Computers & Fluids. 2015. Vol. 110. P. 227–234.

5. Li R., Saad Y. GPU-accelerated preconditioned iterative linear solvers // The Journal of Supercomputing. 2013. Vol. 63, No. 2. P. 443–466.
6. Wolfson-Pou J., Chow E. Reducing Communication in Distributed Asynchronous Iterative Methods // Procedia Computer Science. 2016. Vol. 80. P. 1906–1916.
7. Liu X., Zhong Z., Xu K. A hybrid solution method for CFD applications on GPU-accelerated hybrid HPC platforms // Future Generation Computer Systems. 2016. Vol. 56. P. 759–765.
8. Deng L. et al. CPU/GPU Computing for AN Implicit Multi-Block Compressible Navier-Stokes Solver on Heterogeneous Platform // International Journal of Modern Physics: Conference Series. World Scientific Publishing Company. 2016. Vol. 42. P. 163–166.
9. Stroia I. et al. GPU accelerated geometric multigrid method: Comparison with preconditioned conjugate gradient // High Performance Extreme Computing Conference (HPEC) IEEE. 2015. P. 1–6.
10. Каляев И.А. Левин И.И., Семерников Е.А., Шмойлов В.И. Реконфигурируемые мультиконвейерные вычислительные структуры. Ростов-на-Дону: Изд-во ЮНЦ РАН, 2008. 320 с.
11. Virtex-6 Family Overview. Xilinx, 2015. URL: http://www.xilinx.com/support/documentation/data_sheets/ds150.pdf (дата обращения: 16.06.2016).
12. Series FPGAs Overview. Xilinx, 2015. URL: http://www.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf (дата обращения: 16.06.2016).
13. UltraScale Architecture and Product Overview. Xilinx, 2016. URL: http://www.xilinx.com/support/documentation/data_sheets/ds890-ultrascale-overview.pdf (дата обращения: 16.06.2016).
14. Тихонов А.Н. Самарский А.А. Уравнения математической физики. М.: Наука, 1972. 736 с.
15. Левин И.И., Сорокин Д.А., Мельников А.К., Дордопуло А.И. Решение задач с существенно-переменной интенсивностью потоков данных на реконфигурируемых вычислительных системах // Вестник компьютерных и информационных технологий. М.: Машиностроение, 2012. № 2. С. 49–56.
16. Левин И.И., Дордопуло А.И., Каляев И.А., Доронченко Ю.И., Раскладкин М.К.. Современные и перспективные высокопроизводительные вычислительные системы с реконфигурируемой архитектурой // Вестник ЮУрГУ. Серия «Вычислительная математика и информатика». 2015. Т. 4, № 3. С. 24–39.
17. Карепова Е. Д., Шайдуров В. В., Вдовенко М. С. Параллельные реализации метода конечных элементов для краевой задачи для уравнений мелкой воды // Вестник ЮУрГУ. Серия «Математическое моделирование и программирование». 2009. Вып. 3, № 17(150). С. 73–85.

DOI: 10.14529/cmse160401

IMPLEMENTATION OF ITERATION METHODS FOR SOLUTION OF LINEAR EQUATION SYSTEMS IN PROBLEMS OF MATHEMATICAL PHYSICS ON RECONFIGURABLE COMPUTER SYSTEMS

© 2016 I.I. Levin, A.I. Dordopulo, A.V. Pelipets

Academician A.V. Kalyaev SRI Multiprocessor Computer System at Southern Federal University (2 Chekhov st., GSP-284, Taganrog, 347928, Russia)

E-mail: levin@mvs.tsure.ru, scorpio@mvs.tsure.ru, pelipets@mail.ru

Received: 12.07.2016

In the paper we consider unique features of implementation of iteration methods for solution of linear equation systems in problems of mathematical physics on parallel computer systems, such as geometric decomposition of the computational domain and data parallelization in sequentially performed iterations with intensive data exchange between processors and memory. Standard methods of implementation of iteration methods of solution of linear equation systems with multiple exchanges with memory and between processors, which considerably reduce the performance, require a big number of communication channels in the computer system for implementation of complex topologies and hierarchic schemes of data storage. The solution of this problem is use of multiprocessor systems with reconfigurable architecture which allow adaptation of their architecture to the structure of iteration algorithms of mathematical physics owing to iteration parallelization. In this paper we analyze implementation of the Jacobi method for the Dirichlet problem for the Laplace equation on a reconfigurable computer system. This implementation is an example which illustrates reduction of the number of external data exchange channels, which are the most critical resource of the reconfigurable computer system.

Keywords: reconfigurable computer systems, FPGAs, numerical methods of mathematical physics, iteration parallelization, computational pipeline.

FOR CITATION

Levin I.I., Dordopulo A.I., Pelipets A.V. Implementation of Iteration Methods for Solution of Linear Equation Systems in Problems of Mathematical Physics on Reconfigurable Computer Systems. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. 2016. vol. 5, no. 4. pp. 5–18. (in Russian) DOI: 10.14529/cmse160401.

References

1. Samarskij A.A., Gulin A.V. *Chislennye metody matematicheskoj fiziki* [Numerical Methods of Mathematical Physics]. 2 ed. M: Nauchnyj mir, 2003. 316 p.
2. Gergel' V.P. *Teorija i praktika parallel'nyh vychislenij* [Theory and Practice of Parallel Calculations]. M.: BINOM, 2007. 424 p.
3. Asanovic K. et al. A View of the Parallel Computing Landscape. *Communications of the ACM*, 2009. vol. 52, no. 10. pp. 56–67. DOI: 10.1145/1562764.1562783.
4. Guo X. et al. Developing a Scalable Hybrid MPI/OpenMP Unstructured Finite Element Model. *Computers & Fluids.* 2015. vol. 110. pp. 227–234. DOI: 10.1016/j.compfluid.2014.09.007.
5. Li R., Saad Y. GPU-accelerated Preconditioned Iterative Linear Solvers. *The Journal of Supercomputing.* 2013. vol. 63, no. 2. pp. 443–466. DOI: 10.1007/s11227-012-0825-3.

6. Wolfson-Pou J., Chow E. Reducing Communication in Distributed Asynchronous Iterative Methods. *Procedia Computer Science*. 2016. vol. 80. pp. 1906–1916. DOI: 10.1016/j.procs.2016.05.501.
7. Liu X., Zhong Z., Xu K. A Hybrid Solution Method for CFD Applications on GPU-accelerated Hybrid HPC Platforms. *Future Generation Computer Systems*. 2016. vol. 56. pp. 759–765. DOI: 10.1016/j.future.2015.08.002.
8. Deng L. et al. CPU/GPU Computing for AN Implicit Multi-Block Compressible Navier-Stokes Solver on Heterogeneous Platform. *International Journal of Modern Physics: Conference Series. World Scientific Publishing Company*. 2016. vol. 42. pp. 163–166.
9. Stroia I. et al. GPU Accelerated Geometric Multigrid Method: Comparison with Preconditioned Conjugate Gradient. *High Performance Extreme Computing Conference (HPEC)*. IEEE, 2015. pp. 1–6. DOI: 10.1109/HPEC.2015.7322480.
10. Kaljaev I.A., Levin I.I., Semernikov E.A., Shmojlov V.I. *Rekonfiguriruemye mul'tikonvejernye vychislitel'nye struktury* [Reconfigurable Multipipeline Computing Structures]. Rostov-on-Don: ed. SSC RAS, 2008. 320 p.
11. *Virtex-6 Family Overview*. Xilinx, 2015. Available at: http://www.xilinx.com/support/documentation/data_sheets/ds150.pdf (accessed: 16.06.2016).
12. *7 Series FPGAs Overview*. Xilinx, 2015. Available at: http://www.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf (accessed: 16.06.2016).
13. *UltraScale Architecture and Product Overview*. Xilinx, 2016. Available at: http://www.xilinx.com/support/documentation/data_sheets/ds890-ultrascale-overview.pdf (accessed: 16.06.2016).
14. Tihonov A.N. Samarskij A.A. *Uravnenija matematicheskoj fiziki* [Equations of Mathematical Physics]. Moscow: Nauka, 1972. 736 p.
15. Levin I.I., Sorokin D.A., Mel'nikov A.K., Dordopulo A.I. The Decision of Tasks with Essential and Variable Intensity of Data Streams on Reconfigurable Computing Systems. *Vestnik komp'juternyh i informacionnyh tehnologij* [Bulletin of Computer and Information Technologies]. Moscow: Mashinostroenie, 2012. no. 2. pp. 49–56. (in Russian)
16. Levin I.I., Dordopulo A.I., Kaljaev I.A., Doronchenko Y.I., Rasklakin M.K. Modern and Next-generation High-performance Computer Systems with Reconfigurable Architecture. *Vestnik YuUrGU. Seriya «Vychislitel'naja matematika i informatika»* [Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering]. 2015. vol. 4, no. 3. pp. 24–39. DOI: 10.14529/cmse150303. (in Russian)
17. Karepova E.D., SHajdurov V.V., Vdovenko M.S. Parallel Realization of a Method of Final Elements for a Regional Task for the Equations of Small Water. *Vestnik YuUrGU. Seriya «Matematicheskoe modelirovanie i programmirovanie»* [Bulletin of the South Ural State University. Series: Mathematical Modeling, Programming & Computer Software]. 2009. vol. 3, no. 17(150). pp. 73–85. (in Russian)

УДК 519.86 DOI: 10.14529/cmse160402

ПРИМЕНЕНИЕ МАТЕМАТИЧЕСКОГО МОДЕЛИРОВАНИЯ ДЛЯ ВЫБОРА ИНВЕСТИЦИОННОЙ ПРОГРАММЫ ПРЕДПРИЯТИЯ

© 2016 г. А.В. Панюков, Е.Н. Козина

Южно-Уральский государственный университет
(454080 Челябинск, пр. им. В.И. Ленина, д. 76)
E-mail: paniukovav@susu.ac.ru, kozinaen@susu.ac.ru

Поступила в редакцию: 18.08.2016

В статье представлены три экономико-математические модели для формирования инвестиционной программы предприятия: (1) на основе принципа гарантированного результата (т.е. принципа максимина); (2) на основе принципа максимизации ожидаемого в дисконтированного дохода при заданном ограничении сверху на его дисперсию; (3) на основе принципа максимизации ожидаемого в дисконтированного дохода при ограничении сверху вероятности его недостижимости. Последние две модели дают не гарантированные, а средние оценки доходности в уловиях риска и неопределенности. Решения предложенных задач позволяют дать системную оценку инвестиционной привлекательности предприятия, которую можно использовать при выборе эффективного инвестиционного портфеля с учетом склонности к риску лица принимающего решение.

Ключевые слова: инвестиционная программа, чистый дисконтированный доход, риск, дисперсия, вероятность, стохастическое программирование.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Панюков А.В., Козина Е.Н. Применение математического моделирования для выбора инвестиционной программы предприятия // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2016. Т. 5, № 4. С. 19–31. DOI: 10.14529/cmse160402.

Введение

В настоящее время имеется множество моделей выбоа инвестиционной политики. На финансовых рынках наиболее известной является модель Марковица-Тобина управления портфелем ценных бумаг [1, 2], позволяющая с приемлемым риском максимизировать ожидаемую доходность. Основным элементом управления является периодическая диверсификация портфеля.

Для реализации стратегических целей предприятия также необходимо эффективное управление его инвестиционной деятельностью, с целью наиболее эффективной реализации возможных проектов, приносящих с минимальным риском максимальный финансовый результат в условиях ограниченности инвестиционных ресурсов и неопределенности их объемов [3–9]. В данном случае инвестиции являются долгосрочными, а инвестиционная программа по своей сути является стратегическим планом развития предприятия [6–9]. Она рассчитывается на определенное время и включает список различных проектов инвестирования, в котором детально указаны объемы финансовых вложений.

В настоящее время министерством экономики Российской Федерации выработаны рекомендации по оценке инвестиционных проектов предприятий [10]. В [7–9] показано, что

чистый дисконтированный доход (ЧДД) является агрегированным показателем эффективности как отдельного проекта, так и всей инвестиционной программы. На основе ЧДД легко вычисляются производные показатели эффективности: (1) сроки окупаемости инвестиций, (2) норма прибыли на капитал, (3) разность между суммой доходов и инвестиционными издержками (единовременными затратами) за весь срок использования инвестиционного проекта, (4) приведенные затрат на производство продукции и др. В бизнес-плане отдельного проекта инвестирования должна быть четко обоснована экономическая целесообразность и описаны все действия по его реализации [11–13]. При этом вычисляют потоки доходов/расходов и ЧДД для всех возможных расчетных периодов начала реализации.

Целью работы является разработка математических моделей определения оптимальной инвестиционной программы предприятия, т.е. нахождение порядка выполнения множества независимых проектов, для которых построены оценки в соответствии с [10]. Для построения инвестиционной программы известны эвристические алгоритмы, в которых заложена система предпочтений их разработчиков [7, 14]. В отличие от указанных методов более разумным представляется выбор оптимальной инвестиционной программы предприятия осуществлять из множества эффективных инвестиционных программ, каждая из которых: (1) обеспечивает максимальный ожидаемый ЧДД для некоторого уровня риска возможности выполнимости программы; (2) обеспечивает минимальный риск возможности выполнить программу для некоторого значения ожидаемого ЧДД. В качестве меры риска, используется либо дисперсия ЧДД , по аналогии с подходом Марковица-Тобина [1, 2] при формирования портфеля ценных бумаг [15, 16], либо вероятность недостижимости желаемого среднего значения ЧДД [17, 18].

Построение множества эффективных инвестиционных программ (т.е. множества Парето в пространстве критериев «риск»-«ЧДД») позволит выбрать эффективную инвестиционную программу с учетом склонности к риску лица принимающего решение. В работе предложены математические модели, позволяющие строить такое множество эффективных инвестиционных программ.

Статья состоит из четырех разделов, заключения и списка литературы (19 назв.). В разделе 1 вводятся используемые обозначения и основные соотношения. В разделе 2 построена математическая модель, реализующая принцип гарантированного результата (принцип максимина). Поиск эффективного портфеля в условиях неопределенности и риска [2] рассмотрен в разделе 3. Предложены еще две математические модели основанные на различных определениях понятия «риск»: (a) дисперсия ЧДД , (b) вероятность недостижимости ожидаемого ЧДД. В разделе 4 рассмотрен модельный пример задачи и найдены численные решения для различных вариантов построения инвестиционной программы предприятия. В заключении подведены итоги исследования.

1. Общая постановка задачи

В качестве основного критерия, в соответствии с которым будет формироваться оптимальная инвестиционная программа предприятия, будет использован чистый дисконтированный доход по инвестиционной программе в целом [11, 13].

Пусть

$P = \{p_1, p_2, \ldots, p_n\}$ – множество из n инвестиционных проектов, которые могут быть включены в состав инвестиционной программы;

$L = \{l_1, l_2, \ldots, l_n\}$ – множество продолжительностей реализации инвестиционных проектов (в расчетных периодах);
m – горизонт планирования (число расчетных периодов);
$R = \{r_0, r_1, \ldots, r_{m-1}\}$ – фиксированные финансовые ресурсы или объемы финансирования инвестиционной программы предприятия по расчетным периодам.

Каждый из инвестиционных проектов p_j, $j = 1, 2, \ldots, n$ может быть охарактеризован двумя показателями:
C_{js} – величина чистого дисконтированного дохода, приведенного к моменту начала реализации проекта p_j, если он был начат в период s.
I_{jsi} – потребность в финансировании инвестиционного проекта j в расчетный период i от начала реализации инвестиционной программы, при условии, что он будет начат в период s.

Показатели доходов и расходов являются прогнозируемыми величинами и зависят от ряда факторов. Поэтому целесообразно считать C_{js} и I_{jst} случайными величинами. На основе ретроспективного анализа для каждого проекта p_j, $j = 1, 2, \ldots, n$ и для всех расчетных периодов $i, s = 1, 2, \ldots m$ получены интервальные оценки чистого дисконтированного дохода $[\underline{C}_{js}, \overline{C}_{js}]$, потребностей $[\underline{I}_{jst}, \overline{I}_{jst}]$, и финансовых ресурсов предприятия $[\underline{r}_i, \overline{r}_i]$.

Введем булевы переменные

$$x_{js} = \begin{cases} 1, & \text{начало реализации проекта } p_j \text{ в расчетный период } s, \\ 0, & \text{начало реализации проекта } p_j \text{ отличается от расчетного периода } s. \end{cases} \tag{1}$$

Под реализуемым подмножеством инвестиционных проектов из заданного множества P будем понимать такое подмножество проектов, которое может быть профинансировано в рамках доступных финансовых ресурсов по периодам финансирования. Под чистым дисконтированным доходом инвестиционной программы будем понимать сумму чистых дисконтированных доходов проектов, включенных в инвестиционную программу [11], [13]. Поскольку реализация инвестиционного проекта P_j может начаться не позже чем в период $m - l_j$, то должно выполняться следующее условие:

$$\sum_{s=0}^{m-l_j} x_{js} \le 1, \quad j = 1, 2, \ldots, n. \tag{2}$$

Условие реализуемости инвестиционной программы может быть записано в виде

$$\sum_{j=1}^{n} \sum_{s=0}^{i} I_{jsi} x_{js} \le r_i, \quad i = 0, 1, 2, \ldots, m-1. \tag{3}$$

Определим чистый дисконтированный доход (ЧДД) инвестиционной программы. Рассмотрим инвестиционный проект p_j, который может быть включен в инвестиционную программу. С учетом (2) ЧДД от включения в программу инвестиционного проекта p_j равен

$$C_j = \sum_{s=0}^{m-l_j} C_{js} x_{js}. \tag{4}$$

ЧДД всей инвестиционной программы равен

$$C = \sum_{j=1}^{n} C_j = \sum_{j=1}^{n} \sum_{s=0}^{m-l_j} C_{js} x_{js}. \tag{5}$$

2. Максиминная стратегия

Применение осторожной стратегии, направленной на получение максимального гарантированного ЧДД сводится к решению задачи

$$\underline{C}(x) = \sum_{j=1}^{n} \sum_{s=0}^{m-l_j} \underline{C}_{js} x_{js} \to \max_{x \in D}, \tag{6}$$

где $x = \{x_{js} : j = 1, 2, \ldots, n,\ s = 0, 1, 2, \ldots m - l_j\}$, допустимое множество D удовлетворяет ограничениям

$$\sum_{j=1}^{n} \sum_{s=0}^{i} \overline{I}_{jsi} x_{js} \le \underline{r}_i, \quad i = 0, 1, 2, \ldots, m-1; \tag{7}$$

$$\sum_{s=0}^{m-l_j} x_{js} \le 1, \quad j = 1, 2, \ldots, n; \tag{8}$$

$$x_{js} \in \{0, 1\}, \quad j = 1, 2, \ldots, n, \quad s = 0, 1, .., m - l_j. \tag{9}$$

Гарантированность оптимального значения задачи (6)-(9) обусловлена применением нижних оценок $\underline{C}_{js}$ ЧДД и объемов финансирования $\underline{r}_{js}$ по расчетным периодам и верхних оценок $\overline{I}_{jst}$ потребностей финансирования.

Задача (6)-(9) представляет задачу булева линейного программирования с неотрицательной матрицей условий, поэтому может быть решена псевдополиномиальным алгоритмом на основе динамического программирования.

3. Построение эффективных инвестиционных программ в условиях риска

Будем искать оптимальную инвестиционную программу предприятия в множестве эффективных инвестиционных программ (т.е. в множестве Парето в пространстве критериев «риск»-«ЧДД»). Использование интеллектуальных систем поддержки принятия решений позволит на основе выявленной системы предпочтений лица принимающего решение выбрать наиболее подходящую инвестиционную программу.

3.1. Использование дисперсии ЧДД в качестве меры риска

В качестве ожидаемого чистого дисконтированного дохода инвестиционной программы будем использовать его математическое ожидание, а в качестве меры риска — дисперсию.

Полагая, что чистый дисконтированный доход от каждого проекта $p_j \in P$ равномерно распределен в интервале $[\underline{C}_{js}, \overline{C}_{js}]$ для любого $s = 1, 2, \ldots, l_j$, находим математическое ожидание чистого дисконтированного дохода

$$\mathbf{E}\{C(x)\} = \mathbf{E}\left\{\sum_{j=1}^{n} \sum_{s=0}^{m-l_j} C_{js} x_{js}\right\} = \sum_{j=1}^{n} \sum_{s=0}^{m-l_j} (x_{js} \cdot \mathbf{E}\{C_{js}\}) = \sum_{j=1}^{n} \sum_{s=0}^{m-l_j} \left(\frac{\overline{C}_{js} + \underline{C}_{js}}{2} \cdot x_{js}\right).$$

Учитывая булев характер переменных x и независимость между чистыми дисконтированными доходами от различных проектов, находим дисперсию чистого дисконтированного

дохода

$$\mathbf{D}\{C(x)\} = \mathbf{D}\left\{\sum_{j=1}^{n}\sum_{s=0}^{m-l_j} C_{js}x_{js}\right\} = \sum_{j=1}^{n}\sum_{s=0}^{m-l_j}(x_{js}\cdot\mathbf{D}\{C_{js}\}) = \sum_{j=1}^{n}\sum_{s=0}^{m-l_j}\left(\frac{(\overline{C}_{js}-\underline{C}_{js})^2}{12}\cdot x_{js}\right).$$

Таким образом, построение эффективной инвестиционной программы в условиях риска сведено к задачам

$$\mathbf{E}\{C(x)\} = \sum_{j=1}^{n}\sum_{s=0}^{m-l_j}\left(\frac{\overline{C}_{js}+\underline{C}_{js}}{2}\cdot x_{js}\right) \to \max_{x\in D:\,\mathbf{D}\{C(x)\}\le d}, \tag{10}$$

$$\mathbf{D}\{C(x)\} = \sum_{j=1}^{n}\sum_{s=0}^{m-l_j}\left(\frac{(\overline{C}_{js}-\underline{C}_{js})^2}{12}\cdot x_{js}\right) \to \min_{x\in D:\,\mathbf{E}\{C(x)\}\ge e}, \tag{11}$$

где $x = \{x_{js} : j = 1,2,\ldots,n,\ s = 0,1,2,\ldots m-l_j\}$, допустимое множество D удовлетворяет ограничениям (7)-(9), d и e — допустимые уровни дисперсии и математического ожидания соответственно.

Задачи (10) и (11), как и задача (6)-(9), представляют задачи булева линейного программирования с неотрицательной матрицей условий, поэтому могут быть решены псевдополиномиальным алгоритмом на основе динамического программирования.

3.2. Использование вероятности недостижимости заданного ЧДД в качестве меры риска

Пусть C — заданный уровень ЧДД. Рассмотрим вероятность достижения заданного уровня

$$\mathbf{P}\{C(x) \geqslant C\} = \mathbf{P}\left\{\sum_{j=1}^{n}\sum_{s=0}^{m-l_j} C_{js}x_{js} \geqslant C\right\} \tag{12}$$

С целью использования методов [19] приведения задач с вероятностными критериями к детерминированному виду введем в рассмотрение события

$$E_j:\ \sum_{s=0}^{m-l_j} C_{js}x_{js} = y_j,\ j = 1,2,\ldots,n,\ \sum_{j=1}^{n} y_j \geqslant C,$$

состоящие в том, что доход от проекта p_j будет не меньше величины y_j. Учитывая (8) и (9), имеем

$$\mathbf{P}\{E_j\} = \sum_{s=0}^{m-l_j} x_{js}\mathbf{P}\{C_{js} \geqslant y_j\} = \sum_{s=0}^{m-l_j}\left(x_{js}\frac{\overline{C}_{js}-y_j}{\overline{C}_{js}-\underline{C}_{js}}\right).$$

Отсюда, учитывая независимость ЧДД различных проектов, имеем

$$\mathbf{P}\{C(x) \geqslant C\} = \prod_{j=1}^{n}\mathbf{P}\{E_j\} = \prod_{j=1}^{n}\sum_{s=0}^{m-l_j}\left(x_{js}\frac{\overline{C}_{js}-y_j}{\overline{C}_{js}-\underline{C}_{js}}\right).$$

В дальнейшем вместо задачи максимизации вероятности $\mathbf{P}\{\cdot\}$ будем рассматривать задачу максимизации ее логарифма, т.к. в силу монотонности логарифмической функции опти-

мальные решения обеих задач совпадают. Имеем

$$\ln \mathbf{P}\left\{C(x) \geqslant C\right\} = \sum_{j=1}^{n} \ln \left(\sum_{s=0}^{m-l_j} \left(x_{js} \frac{\overline{C}_{js} - y_j}{\overline{C}_{js} - \underline{C}_{js}} \right) \right) =$$

$$\sum_{j=1}^{n} \sum_{s=0}^{m-l_j} \left(x_{js} \ln \frac{\overline{C}_{js} - y_j}{\overline{C}_{js} - \underline{C}_{js}} \right) \cong - \sum_{j=1}^{n} \sum_{s=0}^{m-l_j} \left(x_{js} \frac{y_j - \underline{C}_{js}}{\overline{C}_{js} - \underline{C}_{js}} \right),$$

здесь второе равенство есть следствие (8) и (9), последнее равенство является следствием приближенного равенства $\ln(1-\xi) \cong -\xi$.

С другой стороны

$$\ln \mathbf{P}\left\{C(x) \geqslant C\right\} = \ln\left[1 - \mathbf{P}\left\{C(x) < C\right\}\right] \cong -\mathbf{P}\left\{C(x) < C\right\},$$

следовательно,

$$\mathbf{P}\left\{C(x) < C\right\} \cong \sum_{j=1}^{n} \sum_{s=0}^{m-l_j} \left(x_{js} \frac{y_j - \underline{C}_{js}}{\overline{C}_{js} - \underline{C}_{js}} \right). \tag{13}$$

Полученное равенство определяет вероятность недостижимости инвестиционной программой заданного значения ЧДД и в дальнейшем используется в качестве меры риска.

Введем детерминированные переменные z_{ji}, $j = 1, 2, \ldots, n$, $i = 0, 1, 2, \ldots, m-1$ и рассмотрим события $A_{ji} = \{I_{jsi} \leq z_{ji}\}$ и $\{B_i = \sum_{j=1}^{n} z_{ji} \leq r_i\}$. Событие A_{ji} состоит в том, что в расчетный период i ресурсы, требуемые проекту j, начатому в любой расчетный период $s \leq m - l_j$, не превосходят значения z_{ij}. Событие B_i состоит в том, что ресурсы, требуемые всем выполняемым в расчетный период i проектам не превосходит величины r_i. Вероятности введенных событий равны

$$\mathbf{P}\left\{I_{jsi} \leq z_{ji}\right\} = \frac{z_{ji} - \underline{I}_{jsi}}{\overline{I}_{jsi} - \underline{I}_{jsi}}, \quad \mathbf{P}\left\{ \sum_{j=1}^{n} z_{ji} \leq r_i \right\} = \frac{\overline{r}_i - \sum_{j=1}^{n} z_{ji}}{\overline{r}_i - \underline{r}_i}$$

Считая переменные z_{ji} фиксированными, найдем вероятности выполнения условий 3 реализуемости инвестиционной программы. Для любого расчетного периода $i = 0, 1, 2, \ldots, m-1$ имеем

$$\mathbf{P}\left\{ r_i(x) = \sum_{j=1}^{n} \sum_{s=0}^{i} I_{jsi} x_{js} \leq r_i \right\} = \mathbf{P}\left\{ \sum_{j=1}^{n} z_{ji} \leq r_i \right\} \cdot \prod_{j=1}^{n} \sum_{s=0}^{m-l_j} \left(x_{js} \mathbf{P}\left\{I_{jsi} \leq z_{ji}\right\} \right). \tag{14}$$

Логарифмируя равенство (14) и учитывая

$$\ln \mathbf{P}\left\{r_i(x) \leq r_i\right\} \cong -\mathbf{P}\left\{r_i(x) > r_i\right\},$$

$$\ln \mathbf{P}\left\{I_{jsi} \leq z_{ji}\right\} \cong -\mathbf{P}\left\{I_{jsi} > z_{ji}\right\} = -\frac{\overline{I}_{jsi} - z_{ji}}{\overline{I}_{jsi} - \underline{I}_{jsi}},$$

$$\ln \mathbf{P}\left\{ \sum_{j=1}^{n} z_{ji} \leq r_i \right\} \cong -\mathbf{P}\left\{ \sum_{j=1}^{n} z_{ji} > r_i \right\} = -\frac{\sum_{j=1}^{n} z_{ji} - \underline{r}_i}{\overline{r}_i - \underline{r}_i},$$

а также условия (8) и (9) получим

$$\mathbf{P}\left\{r_i(x) > r_i\right\} = \frac{\sum\limits_{j=1}^{n} z_{ji} - \underline{r}_i}{\overline{r}_i - \underline{r}_i} + \sum_{j=1}^{n}\sum_{s=0}^{m-l_j}\left(x_{js}\frac{\overline{I}_{jsi} - z_{ji}}{\overline{I}_{jsi} - \underline{I}_{jsi}}\right), \quad i = 1, 2, \ldots, m. \tag{15}$$

Равенство (15) определяет вероятность превышения инвестиционной программой предприятия требуемых ресурсов по расчетном периоду $i = 0, 1, 2, \ldots, m$.

Таким образом, если α — допустимый риск недостижимости инвестиционной программой заданного значения ЧДД, β_i — допустимый риск превышения инвестиционной программой предприятия, требуемых ресурсов по расчетном периоду $i = 0, 1, 2, \ldots, m$, то задачу нахождения максимального ожидаемого дохода можно представить в следующем виде

$$C(x, y, z) = \sum_{j=1}^{n} y_j \to \max_{x,y,z} \tag{16}$$

$$\sum_{s=0}^{m-l_j} C_{js}x_{js} = y_j, \ j = 1, 2, \ldots, n; \tag{17}$$

$$\sum_{j=1}^{n}\sum_{s=0}^{m-l_j}\left(x_{js}\frac{y_j - \underline{C}_{js}}{\overline{C}_{js} - \underline{C}_{js}}\right) \le \alpha; \tag{18}$$

$$\frac{\sum\limits_{j=1}^{n} z_{ji} - \underline{r}_i}{\overline{r}_i - \underline{r}_i} + \sum_{j=1}^{n}\sum_{s=0}^{m-l_j}\left(x_{js}\frac{\overline{I}_{jsi} - z_{ji}}{\overline{I}_{jsi} - \underline{I}_{jsi}}\right) \le \beta_i, \quad i = 1, 2, \ldots, m; \tag{19}$$

$$x_{js}z_{ji} \le \overline{I}_{jsi}, \quad j = 1, 2, \ldots, n, \ s = 0, 1, 2, \ldots, m - l_j, \ i = 1, 2, \ldots, m; \tag{20}$$

$$\sum_{s=0}^{m-l_j} x_{js} \le 1, \quad j = 1, 2, \ldots, n; \tag{21}$$

$$x_{js} \in \{0, 1\}, \quad j = 1, 2, \ldots, n, \quad s = 0, 1, .., m - l_j. \tag{22}$$

4. Пример

Рассмотрим применение изложенных выше математических моделей на примере следующей задачи. К реализации предлагается $n = 7$ инвестиционных проектов, все проекты по предварительным расчетам являются экономически эффективными. Горизонт планирования инвестиционной программы $m = 11$ расчетных периодов (с 0 по 10-й). Продолжительность реализации проектов $j = 1, 2, \ldots, 7$ одинакова и составляет $l_j = 8$ расчетных периодов, следовательно, начало любог проекта возможно только в расчетные периоды $s = 0, 1, 2, 3$.

В табл. 1 приведены интервальные оценки зависимости ЧДД проектов от времени s начала реализации. В табл. 2 приведены интервальные оценки допустимых объемов финансирования инвестиционной программы предприятия по расчетным периодам i. В табл. 3 представлены интервальные оценки I_{jsi} затрат по проектам $j = 1, 2, \ldots, 7$ в расчетный периоды $i = 0, 1, \ldots, 10$. Все допустимые риски β_i превышения инвестиционной программой

Таблица 1

Зависимости ЧДД проектов от момента s начала реализации (тыс. руб.)

Проект	$s=0$		$s=1$		$s=2$		$s=3$	
j	$\underline{C}_{j0}$	$\overline{C}_{j0}$	$\underline{C}_{j1}$	$\overline{C}_{j1}$	$\underline{C}_{j2}$	$\overline{C}_{j2}$	$\underline{C}_{j3}$	$\overline{C}_{j3}$
1	655	850	585	780	522	717	466	661
2	246	441	220	415	196	391	175	370
3	164	359	146	341	131	326	117	312
4	383	578	342	537	305	500	272	433
5	334	529	298	493	266	461	237	964
6	972	1167	867	1063	774	970	691	887
7	414	609	369	565	330	525	294	490

Таблица 2

Оценки допустимых объемов финансирования инвестиционной программы по расчетным периодам i (тыс. руб.)

i	0	1	2	3	4	5	6	7	8	9	10
$\underline{r}_i$	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800
$\overline{r}_i$	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000

Таблица 3

Затраты по по проекту $j=1,2,\ldots,7$ в расчетный период $i=0,1,\ldots,10$, $s=0,1,2,3$ (тыс. руб.)

	$i=s$		$i=1+s$		$i=2+s$		$i=3+s$		$i=4+s$		$j=5+s$		$i=6+s$		$i=7+s$	
j	$\underline{I}_{jsi}$	$\overline{I}_{jsi}$	$\underline{I}_{jsi}$	$\overline{I}_{jsi}$	$\underline{I}_{jsi}$	$\overline{I}_{jsi}$	$\underline{I}_{jsi}$	$\overline{I}_{jsi}$	$\underline{I}_{jsi}$	$\overline{I}_{jsi}$	$\underline{I}_{jsi}$	$\overline{I}_{jsi}$	$\underline{I}_{jsi}$	$\overline{I}_{jsi}$	$\underline{I}_{jsi}$	$\overline{I}_{jsi}$
1	100	120	100	119	140	150	120	132	88	100	72	80	50	58	40	50
2	300	320	300	303	300	400	80	100	80	100	80	100	90	100	90	100
3	88	99	120	144	130	155	88	100	75	80	59	60	58	60	55	60
4	400	500	680	700	199	215	140	150	140	150	140	150	140	150	140	150
5	530	600	530	600	530	600	70	80	70	80	70	80	70	80	70	80
6	680	850	680	850	390	500	390	500	350	400	180	200	180	200	180	200
7	480	500	380	385	330	380	320	350	300	330	300	330	300	330	300	330

Таблица 4

Конкурентоспособные инвестиционные программы

Проект j	Максимин $s: x_{js}=1$	Дисперсия ЧДД $s: x_{js}=1$			Вероятность недостижимости ЧДД: $s: x_{js}=1$		
		$3D_0=3174$	$1.5D_0=2116$	$D_0=1058$	$\alpha=0.1$	$\alpha=0.05$	$\alpha=0.03$
1	0	0	1	0	0	1	1
2	-	0	3	3	-	0	0
3	3	1	0	0	0	-	-
4	-	3	-	-	2	2	3
5	3	-	0	1	0	-	-
6	0	0	1	2	0	0	0
7	0	0	3	0	0	0	3
Номер	1	2	3	4	5	6	7
ЧДД	2644	3782	3297	2916	4044	3588	2944
D	$D_0=1058$	3174	2116	1058	3372	2253	1103
α	0	0.104	0.051	0.031	0.096	0.049	0.026

предприятия, требуемых ресурсов по расчетном периоду $i=0,1,2,\ldots,m$ равны допустимому риску α недостижимости инвестиционной программой заданного значения ЧДД.

Результаты решения, выполненные с применением MS Excel, представлены в табл. 4. Столбцы со второго по восьмой табл. 4 соответствуют соответствуют семи оптимальным инвестиционным программам предприятия, удовлетворяющих заданному ограничению на вид и величину риска. В них для каждого проекта $j=1,1,\ldots,n$, включенного в инвестиционную программу, указано значение $s:\ x_{js}=1$ расчетного периода, соответствующее началу его реализации.

При формировании инвестиционной программы по максиминной стратегии гарантированный ЧДД программы равен 2644, дисперсия гарантированного ЧДД $D_0=1058$.

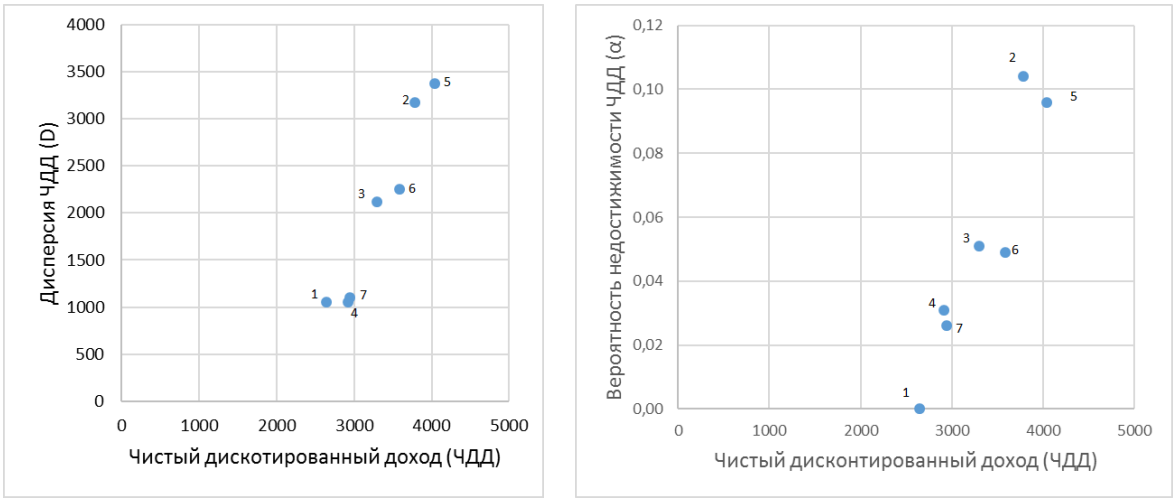
Наибольший ожидаемый ЧДД для дисперсии $D_0=1058$ равен 2916 и его реализует инвестиционная программа, отличающаяся от максиминной. Уровням допустимой дисперсии $1.5D_0$ и $3D_0$ соответствуют различные оптимальные по Парето инвестиционные программы, имеющие средний ЧДД равные 3297 и 3782 соответственно.

Увеличение уровня α вероятности недостижимости возможного ожидаемого значения ЧДД в стохастических моделях также ведет к различным инвестиционным программам с возрастающими средними ожидаемыми значениями ЧДД. Поскольку стохастическая модель (16)-(22), в отличие от модели 10)-(11), допускает риск превышения инвестиционной программой предприятия требуемых ресурсов по расчетным периодам, то возможные средние значения ожидаемых ЧДД в стохастической модели оказываются выше.

Образы всех проектов из табл.4 в системах координат «дисперсия ЧДД»-«ожидаемый ЧДД» и «вероятность недостижимости»-«ожидаемый ЧДД» представлены на рисунке. Построенные в примере инвестиционные программы предприятия являются эффективными (т.е. принадлежат множеству Парето в пространстве критериев «мера риска»-«ЧДД»).

Заключение

Рассмотренные модели оптимальной инвестиционной программы с фиксированными объемами финансирования позволяют формировать оптимальные по Парето инвестицион-



Образы проектов в различных пространствах критериев

ные программы при известном распределении финансовых средств по периодам. Представленные модификации данной модели, учитывают неопределенность финансовых ресурсов для поддержки инвестиционных проектов.

Решения соответствующих задач дают системную оценку инвестиционной привлекательности моделируемого предприятия и могут быть использованы в интеллектуальных системах поддержки выбора эффективного портфеля с учетом производных показателей эффективности: (1) сроки окупаемости инвестиций, (2) норма прибыли на капитал, (3) разность между суммой доходов и инвестиционными издержками (единовременными затратами) за весь срок использования инвестиционного проекта, (4) приведенные затрат на производство продукции и др. и склонности к риску лица принимающего решение.

Применение использованных в исследовании универсальных программных средств, таких как MS Excel, Cplex и т.п. не позволяют решать задачи большой размерности. Направление дальнейшего исследования — построение программного обеспечения, позволяющего за счет учета специфики задачи решать задачи большой размерности.

Литература

1. Кибзун А.И., Чернобровов А.И. Алгоритм решения обобщенной задачи Марковица // Автоматика и телемеханика 2011. № 2. С. 77–92.

2. Новоселов А.А. Математическое моделирование финансовых рисков: Теория измерения. Новосибирск: Наука, 2001. 102 с.

3. Dilger R.J., Gonzales O.R. SBA Small Business Investment Company Program // Journal of Current Issues in Finance, Business and Economics. 2012. Vol. 5. No. 4. P. 407–417.

4. Михалина Л.М., Голованов Е.Б. Изменение инвестиционной активности субъектов малого бизнеса в условиях коррупционной среды // Вестник Южно-Уральского государственного университета. Серия: Экономика и менеджмент. 2014. Т. 8(3). С. 41–47.

5. Schlegel D., Frank F., Britzelmaier B. Investment decisions and capital budgeting practices in German manufacturing companies. // International Journal of Business and Globalisation. 2016. Vol. 16, No. 1. P. 66–78.

6. Юзвович Л.И. Экономическая природа и роль инвестиций в национальной экономической системе // Финансы и кредит. 2010. № 9. С. 48–52.

7. Гамилова Д.А. Обоснование эффективности проведения инвестиционной политики предприятия // Инновации и инвестиции. 2010. № 2. С. 37–41.

8. Сергеева Д.П. Методы оценки эффективности инвестиционных проектов с учетом рекомендаций минэкономики // Инновационная наука. 2015. № 9. С. 201–204.

9. Желнова К.В. Анализ практики принятия решений в области инвестиционной политики // Вопросы современной экономики. 2013. № 4. С. 38–46.

10. Методические рекомендации по оценке эффективности инвестиционных проектов (утв. Минэкономики РФ, Минфином РФ, Госстроем РФ 21.06.1999 N ВК 477). URL: http://www.consultant.ru/document/cons_doc_LAW_28224/ (дата обращения: 30.07.2016).

11. Панюков А.В. Математическое моделирование экономических процессов. Москва: URSS, 2009. 191 с.

12. Panyukov A.V., Teleghin V.A. Forming of Discrete Mechanical Assembly Production Program // Journal of Computational and Engineering Mathematics. 2015. Vol. 2, No. 1. P. 57–64.

13. Афанасьев М.А. Оптимальная инвестиционная программа // Инвестиции в России. 2002. № 12. С. 50–54.

14. Виленский В.П. Об одном подходе к учету влияния неопределенности и риска на эффективность инвестиционных процессов // Экономика и математические методы. 2012. № 2. С. 29–38.

15. Глухов В.В. Математические методы и модели для менеджмента. СПб: Лань, 2005. 528 с.

16. Козина Е.Н. Разработка экономико-математической модели инвестиционной программы предприятия // Сб. трудов международной научно-практической конференции «Современные тенденции экономики, права, управления, математики: новый взгляд. Санкт-Петербургб., 01-03 декабря». СПб.: Изд-во «КультИнформПресс», 2012. С. 64–67.

17. Кибзун А.И. Задачи стохастического программирования с вероятностными критериями. М.: Физматлит, 2009. 372 с.

18. Козина Е.Н. Математическая модель инвестиционной программы предприятия // Сб. трудов XII Всероссийского совещания по проблемам управления (Москва, 16-19 июня 2014). М.: ИПУ РАН, 2014. С. 1250–1253.

19. Вишняков Б.В., Кибзун А.И. Детерминированные эквиваленты для задач стохастического программирования с вероятностными критериями // Автоматика и телемеханика. 2006. № 6. С. 126–143.

DOI: 10.14529/cmse160402

APPLICATION OF MATHEMATICAL MODELLING FOR CHOOSING COMPANY'S INVESTMENT PROGRAM

© 2016 A.V. Panyukov, E.N. Kozina

South Ural State University (pr. Lenina 76, Chelyabinsk, 454080 Russia)

E-mail: paniukovav@susu.ac.ru, kozinaen@susu.ac.ru

Received: 18.08.2016

The paper presents three economic-mathematical models for the formation of the company's investment program: (1) based on the principle of guaranteed payoff (ie maximin principle); (2) based on the principle of maximizing the average expected value under predetermined up limiting of its dispersion; (3) based on the principle of maximizing the average expected value under up limiting the probability of its inaccessibility. Proposed solutions of the problems allow us to give a system estimate of the investment attractiveness of the enterprise which can be used in selecting an effective investment portfolio based on risk appetite of decision makers.

Keywords: investment program, net present value, risk dispersion, probability, stochastic programming.

FOR CITATION

Panyukov A.V., Kozina E.N. Application of Mathematical Modelling for Choosing Company's Investment Program. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. 2016. vol. 5, no. 4. pp. 19–31. (in Russian) DOI: 10.14529/cmse160402.

References

1. Kibzun A.I., Chernobrovov A.I. Algorithm to Solve the Generalized Markowitz Problem. *Avtomatika i telemekhanika* [Automation and Remote Control]. 2011. vol. 72, no. 2. pp. 289–304. DOI: 10.1134/S0005117911020081. (in Russian)
2. Novoselov A.A. *Matematicheskoe modelirovaie finansovykh riskov: Teoriya izmereniya* [Mathematical Modeling of Financial Risks: Measurement Theory]. Novosibirsk, Nauka, 2001. 102 p.
3. Dilger R.J., Gonzales O.R. SBA Small Business Investment Company Program. *Journal of Current Issues in Finance, Business and Economics.* 2012. vol. 5, no. 4. pp. 407–417.
4. Mikhalina L.M., Golovanov E.B. Investment activity variance of small businesses in a corrupt environment. *Vestnik Yuzhno-Uralskogo gosudarstvennogo universiteta. Seriya: Ekonomika i menedzhment* [Bulletin of the South Ural State University. Series: Economics and Management]. 2014. vol. 8, no. 3. pp. 41–47. (in Russian)
5. Schlegel D., Frank F., Britzelmaier B. Investment Decisions and Capital Budgeting Practices in German Manufacturing Companies. *International Journal of Business and Globalisation.* 2016. vol. 16, no. 1. pp. 66–78. DOI: 10.1504/IJBG.2016.073626.
6. Yuzvovich L.I. Economical Nature and the Role of Investment in the National Economics. *Finansy i kredit* [The Finance and the Credit]. 2010. no. 9. pp. 48–52. (in Russian)
7. Gamilova D.A. Substantiation of Efficiency of Company's Investment Policy. *Innovatsii i investitsii* [Innovation and Investment]. 2010. no. 2. pp. 37–41. (in Russian)

8. Sergeeva D.P. Methods for Evaluating the Effectiveness of Investment Projects Taking into Account the Recommendations of the Ministry of Economy. *Innovatsionnaya nauka* [Innovative Science]. 2015. no. 9. pp. 201–204. (in Russian)

9. Zhelnova K.V. Analysis of Decision Making Practice of Investment Policy. *Voprosy sovremennoy ekonomiki* [Problems of Current Economics]. 2013. no. 4. pp. 38–46. (in Russian)

10. *Metodicheskie rekomendatsii po otsenke effektivnosti investitsionnykh proektov (utv. Minekonomiki RF, Minfinom RF, Gosstroem RF 21.06.1999. N VK 477)* [Methodical recommendations according to efficiency of investment projects (approved by RF Ministry of Economy, Ministry of Finance, the State Construction Committee of Russia 21.06.1999. N VK 477)]. Available at: http://www.consultant.ru/document/cons_doc_LAW_28224/ (accessed: 30.07.2016).

11. Panyukov A.V. *Matematicheskoe modelirovanie ekonomicheskih processov* [Mathematical Modeling of Economic Processes]. Moscow: URSS, 2009. 191 p.

12. Panyukov A.V., Teleghin V.A. Forming of Discrete Mechanical Assembly Production Program. *Journal of Computational and Engineering Mathematics.* 2015. vol. 2, no. 1. pp. 57–64. DOI: 10.14529/jcem150107.

13. Afanasiev M.A. Optimal Investment Program. *Investitsii v Rossii* [Investments into Russia]. 2002. no. 12. pp. 50–54. (in Russian)

14. Vilenskiy V.P.An Approach to the Calculation of the Influence of Uncertainty and Risk at the Effectiveness of the Investment Processes. *Ekonomika i matematicheskie metody* [Economics and Mathematical Methods]. 2012. no. 2. pp. 29–38. (in Russian)

15. Glukhov V.V. *Matematicheskie metody i modeli dlya menedzhmenta* [Mathematical Methods and Models for the Management]. St. Petersburg, Lan', 2005. 528 p.

16. Kozina E.N. The Development of Economic and Mathematical Models of the Company's Investment Program. *Sb. trudov mezhdunarodnoy nauchno-prakticheskoy konferencii «Sovremennye tendetsii ekonomiki, prava, upravleniya, matematiki: Novyi vzglyad. Sankt-Peterburg, 01-03 dekabrya»* [Coll. Proceedings of the International Scientific-Practical Conference « Modern Trends in Economics, Law, Management, Mathematics: a New Look. St. Petersburg., 01-03 December »] Sankt Petersburg, Publishing house "KultInformPress". 2012. vol. 3. pp. 64–67. (in Russian)

17. Kibzun A.I. *Zadachi stokhasticheskogo programmirovaniya s veroyatnostnymi kriteriyami* [Tasks of Stochastic Programming with Probabilistic Criteria]. Moscow, FIZMATLIT, 2009. 372 p.

18. Kozina E.N. Mathematical Model of the Investment Program of the Enterprise. *Sb. trudov XII Vserossiyskogo soveshhaniya po problemam upravleniya (VSPU-2014) Moskva, 16-19 yunya 2014* [Coll. Works of XII All-Russian Conference on Control (VSPU 2014) Moscow, June 16-19, 2014]. Moscow, Institute of Control Sciences, 2014. pp. 1250–1253. (in Russian)

19. Vishnaykov V.B., Kibzun A.I. Deterministic Equivalents for the Problems of Stochastic Programming with Probabilistic Criteria. *Automation and Remote Control.* 2006. vol. 67, no. 6. pp. 945–961. DOI: 10.1134/S0005117906060099.

DOI: 10.14529/cmse160403

SUPERCOMPUTER APPLICATION INTEGRAL CHARACTERISTICS ANALYSIS FOR THE WHOLE QUEUED JOB COLLECTION OF LARGE-SCALE HPC SYSTEMS*

© 2016 D.A. Nikitenko, V.V. Voevodin, A.M. Teplov, S.A. Zhumatiy, Vad.V. Voevodin, K.S. Stefanov, P.A. Shvets

Research Computing Center, M.V. Lomonosov Moscow State University (Leninskie Gory 1, Moscow, 119991 Russia)

E-mail: dan@parallel.ru, alex-teplov@yandex.ru, serg@parallel.ru, vadim@parallel.ru, cstef@parallel.ru, shvets.pavel.srcc@gmail.com

Received: 11.04.2016

Efficient use and high output of any supercomputer depends on a great number of factors. The problem of controlling granted resource utilization is one of those, and becomes especially noticeable in conditions of concurrent work of many user projects. It is important to provide users with detailed information on peculiarities of their executed jobs. At the same time it is important to provide project managers with detailed information on resource utilization by project members by giving access to the detailed job analysis. Unfortunately, such information is rarely available. This gap should be eliminated with our proposed approach to supercomputer application integral characteristics analysis for the whole queued job collection of large-scale HPC systems based on system monitoring data management and study, building integral job characteristics, revealing job categories and single job run peculiarities.

Keywords: supercomputer, efficiency, system monitoring, job categories, integral job characteristics, queued job collection, job queue, resource utilization control.

FOR CITATION

Nikitenko D.A., Voevodin V.V., Teplov A.M., Zhumatiy S.A., Voevodin Vad.V., Stefanov K.S., Shvets P.A. Supercomputer Application Integral Characteristics Analysis for the Whole Queued Job Collection of Large-Scale HPC Systems. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. 2016. vol. 5, no. 4. pp. 32–45. DOI: 10.14529/cmse160403.

Introduction

Securing efficient resource utilization of HPC systems is one of the most important and challenging tasks at present trends of rapid growth of scales and capabilities of modern supercomputers [1, 2]. There is a variety of approaches that are aimed at analysis of efficient utilization of certain HPC system components or systems as a whole. Some of them are based on system monitoring data analysis [3, 4]. This type of approaches sets especially strict requirements on monitoring system implementation and configuration [5], as well as for the means of data storage and access. At the same time these approaches possess a number of fundamental advantages.

*The paper was recommended for publication by the program committee of the International Scientific Conference "Parallel Computing Technologies – 2016".

First, the analyzed data reflects physical, real levels of HPC system components and appropriate resource utilization.

Second, filtering system monitoring data obtained from known set of components and period of time allows binding this data to certain jobs. Thus, allowing analyzing resource utilization history and trends by certain applications, users, projects, partitions, and so on.

Third, typically it is possible to configure monitoring systems obtaining data from the whole system in such a way that it induces acceptable overhead. This allows collecting data with a rougher granulation, when possible, but still sufficient for basic analysis of resource utilization by any and every job. To have more detailed information on certain job, of course, used monitoring system should likely support data acquisition rate reconfiguration on-the-fly for specified sensor sets and sources. If not (of course, it is much less efficient way), most monitoring systems can be started in a higher granularity mode to record certain job activity and restarted in a normal mode afterwards. There are other options available, for example, data aggregation implementation that is precise for first, say, 30 seconds of job execution (to study short jobs) that is later switches to rough mode (for longer jobs). Anyhow, there are a number of techniques available to study certain application behavior.

The existing methods and techniques that base on system monitoring data analysis, allow both analysis of dynamic characteristics of certain application runs and peculiarities of resource utilization within system partitions and systems as a whole [6]. With a project-oriented workflow, when a number of users run jobs as a part of one applied research, it is very useful to let administrator and system manager have a clear view of resource utilization distribution in the workgroup to have a possibility to influence permissions or workflow inside the workgroup to meet the granted resources limitation [7, 8]. Nevertheless there is still need for specialized tools and techniques to analyze available system monitoring data. In a point of fact, the one is needed as a valuable additional tool to the set of implemented approaches in every-day practice of MSU Supercomputer Center [9–12] — a tool for job queue analysis based on system monitoring that would allow revealing job categories, job grouping by some criteria, starting from belonging to user or project domain and other resource manager specific characteristics, to categories by levels and peculiarities of HPC system resource utilization or its combinations. As a basic technique for such grouping implementing tagging system seems to be an adequate option — assigning special tags to a job description as soon as each tag description criteria is met by job characteristics. Tagging principles are widely successfully used for categorizing and search purposes managing huge collections of data in Internet: news, videos, photos, notes, and so forth, that is quite close to the challenge that is being tackled.

The paper is organized as follows. Section 1 is devoted to job categories and tagging principles. Section 2 describes implementation. Section 3 provides examples and use cases. Conclusion section includes summary as well as future work overview.

1. Job categories and tagging

The combined analysis of system monitoring data and resource manager log data, as was already mentioned, allows binding raw system monitoring data to certain jobs. This provides means to analyze job dynamics as far as data granularity allows. To analyze the average rate of application resource utilization every dynamic characteristic can serve basis for the calculation of minimum, maximum, average and median values. These types of values are often named integral job characteristics.

When one takes a look at the whole scope of executed jobs for analysis of job queue structure, application run sequences, jobs comparative analysis and even searching for outstanding single job behavior it becomes obvious that it would be very useful to have tools that provide means for revealing job categories based on various criteria.

This functionality can be implemented by introducing special tags. Every tag is based on its own criteria, based on a single integral job characteristic or its combination, resource manager job-related information and any other available info from used data sources. For example tags can correspond to certain average rates of various resource utilization, job ownership, job duration, resource utilization specifics, special execution modes, detailed system monitoring data availability, and so forth.

The approach features means to make efficient grouping and filtration of whole job queue history collection by any improvised combination of specified tags. Driven by experience of application efficiency and scalability study based on system monitoring data analysis, the authors propose introducing the following job categories on the first stage of implementation. Tag naming is designed to give self-explanatory tag description, nevertheless, every tag must have a detailed full-format description available.

1.1. System monitoring data based categories

CPU utilization

- Tag name: avg_CPU_user LOW
  Category: Low CPU user utilization.
  Criteria: Average value of CPU_user doesn't exceed 20%.
- Tag name: avg_CPU_user HIGH
  Category: High CPU user utilization.
  Criteria: Average value of CPU_user exceeds 40%.
- Tag name: avg_CPU_idle TOO HIGH
  Category: CPU is idle for a considerable time.
  Criteria: Average CPU_idle value exceeds 25%.

Competition of processes for CPU cores

- Tag name: avg_LA LOW
  Category: User job is almost out of action, almost no utilization of CPU.
  Criteria: Average Load Average is below 1.
- Tag name: avg_LA SINGLE CORE
  Category: Only one process per node is active as an average.
  Criteria: Average Load Average is approximately 1.
- Tag name: avg_LA NORMAL
  Category: Optimal competition of processes.
  Criteria: Average Load Average is approximately equal to the number of cores per node.
- Tag name: avg_LA HYPERTHREADED
  Category: Normal process competition with hyperthreading is on.
  Criteria: Average Load Average value is approximately equal to the double number of CPU cores per node.

Floating point operations

- Tag name: avg_Flops HIGH
  Category: Intensive CPU floating point operations.

Criteria: Average value of floating point operations number exceeds 10% of theoretical CPU peak.

Interconnect activity

- Tag name: avg_IB_packages_num LOW
  Category: Low number of inter-node data transmissions.
  Criteria: Average package send rate does not exceed 103 packages per second.
- Tag name: avg_IB_packages_size TOO LOW
  Category: Small size of packages.
  Criteria: Average package send rate exceeds 103 packages per second while average data transmission rate is below 2 kilobytes per second.
- Tag name: avg_IB_speed HIGH
  Category: High data transmission intensity.
  Criteria: Average data transmission rate is over 0,2 Gigabytes per second and up to 1 Gigabytes per second.
- Tag name: avg_IB_speed TOO HIGH
  Category: Very high data transmission intensity.
  Criteria: Average data transmission rate is over 1 Gigabytes per second.

Memory utilization

- Tag name: avg_cache_L1/L3 TOO LOW
  Category: Very low efficiency of cache stack utilization.
  Criteria: Ratio of the number of L1 misses to the number of L3 misses is below 5.
- Tag name: avg_cache_L1/L3 LOW
  Category: Reduced efficiency of cache stack utilization.
  Criteria: Ratio of the number of L1 misses to the number of L3 misses is below 10.
- Tag name: avg_cache_L1/L3 HIGH
  Category: Good efficiency of cache stack utilization.
  Criteria: Ratio of the number of L1 misses to the number of L3 misses exceeds 10.
- Tag name: avg_mem/cache_L1 LOW
  Category: Reduced efficiency of cache L1 utilization.
  Criteria: Ratio of the number of total memory operations to the number of L1 misses does not exceed 15.
- Tag name: avg_memload HIGH
  Category: Intensive memory operations.
  Criteria: Average number of memory operations exceeds 109 operations per second.

1.2. Resource manager based categories

Job execution status

- Tag name: job_status COMPLETED
  Category: Job is successfully finished.
- Tag name: job_status FAILED
  Category: Job is finished with an error in program.
- Tag name: job_status CANCELED
  Category: Job was cancelled by user.
- Tag name: job_status TIMEOUT
  Category: Job was cancelled by exceeding time limit.

- Tag name: job_status NODE_FAIL
  Category: Job is finished with system error.

Job submission details

- Tag name: job_time_limit CUSTOM
  Category: Requested time limit is custom.
- Tag name: job_start_script CUSTOM
  Category: Job batch file is custom.
- Tag name: job_cores_requested FEW
  Category: Not all available CPU cores per node requested.
- Tag name: job_cores_requested SINGLE
  Category: Just a single CPU core per node requested.
- Tag name: job_MPI INTEL
  Category: MPI type used: Intel MPI.
- Tag name: job_MPI OpenMPI
  Category: MPI type used: OpenMPI.
- Tag name: job_nnodes SINGLE
  Category: Job used a single node.
- Tag name: job_nnodes FEW
  Category: Job used from 2 up to 8 nodes.
- Tag name: job_nnodes MANY
  Category: Job used 8 nodes and above.

System-dependent peculiarities and partition usage
Illustrated by the example of "Lomonosov" supercomputer partitions.

- Tag name: job_partition REGULAR4
  Category: Job allocated to REGULAR4 partition.
- Tag name: job_partition REGULAR6
  Category: Job allocated to REGULAR6 partition.
- Tag name: job_partition HDD4
  Category: Job allocated to HDD4 partition.
- Tag name: job_partition HDD6
  Category: Job allocated to HDD6 partition.
- Tag name: job_partition SMP
  Category: Job allocated to SMP partition.
- Tag name: job_partition GPU
  Category: Job allocated to GPU partition.
- Tag name: job_partition TEST
  Category: Job allocated to TEST partition.
- Tag name: job_partition GPUTEST
  Category: Job allocated to GPUTEST partition.
- Tag name: job_partition EXCEPT TEST
  Category: Job allocated to regular or high priority partition.
- Tag name: job_priority HIGH
  Category: Job allocated to partitions with a higher priority (queues reg4prio, gpu_p, dedicated6)

Matching partition specifics

- Tag name: job_accell GPU
  Category: User application uses accelerators. Accelerator type: GPU.
- Tag name: job_accel GPU UNUSED
  Category: Job is run on GPU partition, but never uses GPUs.
- Tag name: job_disks UNUSED
  Category: Job is run on HDD-equipped partition, but never uses local I/O.
- Tag name: job_disks TOO LOW
  Category: Job is run on HDD-equipped partition, but I/O rate is very low.

1.3. Other categories

Beyond the tags that can be assigned automatically, it is possible to introduce manually-set tags. This is useful when the criteria is cannot be automatically determined. There are now different manual setting options available. First, most typical, selecting the one from known tags lost or introducing new one with human-read and formal description. Second, is pushing some tags like "higher system monitoring rate for the job" via the command line when submitting a job.

This applies for instance to general job description characterizing type of data processing as it is usually known a priori or determined in the course of job behavior study by a specialist: job_behavior DATA MINING, job_behavior MASTER-SLAVE, job_behavior COMMUNICATION, job_behavior ITERATIVE, etc.

In the same manner typical anomalies encountered during analysis course can be specified: job_bug DEADLOCK, job_bug DATA RACE, etc.

It is very useful to specify if a widely used algorithm implementation or software package is used. Just in case, this can provide a great contribution to scalability and algorithms-studying projects, like AlgoWiki [13]: job_sw VASP, job_sw FIREFLY, job_sw GROMACS, etc.

If detailed reports on job efficiency analysis or issues is available, or specific standard report like JobDigest is available, it is useful to mark such a feature with another tag, for example: job_analized, job_analized JobDigest.

2. Implementation

We keep to the basis of building a tool that might be deployed at any supercomputer center with minimal efforts. We currently support Slurm [14], Cleo [15] resource managers and Ganglia [16], Collectd [17], Clustrx [18], DiMMon [5] (most promising) monitoring systems.

As for integral job characteristics derivation and tagging, PostgreSQL is used as data storage for coarsened system monitoring data and saved job information from resource managers. The saved job info is processed by JavaScript, jQuery with jQuery UI [19] and TagIt [20].

The tag can be assigned to a job only if it is already declared in tag description table. Such a table includes tag id, name, human-readable description, criteria (a specification of SQL request for automatic processing), comments, and a flag of availability that can be set only by administrator. Any user can suggest introducing a new tag, but it will be available only after administrator approval. Information on new tag author is saved in the comments attribute, added the user tag description suggestion and motivation.

All tags can be assigned in two ways: automatically and manually. Any tag set by mistake or error can be manually removed from a job.

Automatic mode. In this mode, the tags are automatically assigned:

- to all finished jobs according to SQL-based criteria regarding saved integral job characteristics data, information from resource manager and other available saved data;
- as a result of running a special script that processes whole saved job collection info.

In this mode a special attribute would indicate that the tag was set in package (automatic) mode of tag assignment.

Manual mode. Manual tag assignment is usually done by user, project manager or administrator in the following cases:

- as a result of certain job analysis (specifying algorithm implemented, etc.);
- as a result of specifying the tag via command line when submitting a job;
- any tag in a user-specific tag space (marking out important job runs as a part of the project, etc.).

In this mode an attribute addressing tag author is set, that also allows finding jobs, marked as a part of a certain project or by a certain user.

User-specific tag space consists of regular tags and custom user tags. Any manually assigned tags by a user are seen only in the scope of the project and system administrators. The members of other project see their own tag spaces and the general tag space is available for all of the users.

3. Use cases

Of course, real life use cases are very diverse. It this section we would like to share our experience of every-day usage of the proposed technique as a part of the developed tool approbation at Supercomputer Center of Moscow State University on a few examples just to give a general idea of it.

3.1. Revealing jobs, users and projects that practice inappropriate resource utilization

One of the problems of every-day practice of large-scale supercomputer center with a number of heterogeneous resources and considerable number of users concurring for the resources is a problem of inacceptable efficiency or inappropriate resource utilization. This is of a higher priority for specific limited resources, like compute nodes equipped with specialized accelerators, local disks, extra memory or other hardware and software that is critical for some applications and at the same time these nodes can still be used by applications that do not need that specific type of resources that the nodes possess. Such nodes usually have a high potential for resource-demanding specific applications and for the large systems like “Lomonosov” are usually managed as a separate partition with a special queuing options to allow submitting jobs to the appropriate partition. This is vital for projects that perform computations only due to the advantages of such partitions, so by queuing to the desired partition user get a guarantee that their application would have all necessary resources at disposal.

Nevertheless, when analyzing the whole job collection for such partitions it appears that there are numerous job runs that do not use any partition facilities benefits. Of course, sometimes algorithm peculiarities can use resources with totally different intensity, but further analysis usually shows that the majority of suspicious jobs never use any benefit of such partitions. The reasons can be different, but usually it is a shorter wait time in a queue.

This can be seen on GPU partitions with user job runs that never use GPUs. A slightly different situation is seen on HDD-equipped nodes with absent or extreme low disk usage rate

and finally, single-process application that don't benefit from multiple CPU cores can be seen almost on any partition regardless of hardware and software.

It is important to find the root cause of such applications behavior and as soon as the reason is found and changes by user or administrator are applied, the ratio of such jobs can be lowered that would immediately raise HPC system efficiency and overall throughput.

The most popular reasons are:

- Problems inside the application, program or algorithm. The user is sure that he needs resources, but in practice application doesn't utilize any or utilizes at extremely low rates.
- Problems of HPC system. The declared resources are not available on the nodes.
- Inappropriate job allocation. This can be both a mistake, and cheating for lower job waiting time.

Regardless of real reason, these job runs lead to a higher wait time for the jobs that really need specific resources.

The search for such jobs can be automated using integral job characteristics and some of introduced tags.

For example, to filter the jobs allocated to GPU partition with no usage of accelerator one can use tags job_partition GPU and job_accel GPU UNUSED at the same time. Next, one can cut off jobs allocated to the test partition as of no interest. The rest jobs that are assigned job_status COMPLETED tag probably do not need GPUs at all, as finished successfully with no registered GPU usage. At this point two options are available whether it is a mistake (user or system) or it was done by user intentionally, trying to reduce job wait time as wait time in specific partitions is sometimes less than in regular.

A very similar situation is seen for HDD-equipped nodes. Jobs that are tagged with job_partition HDD4, job_partition HDD6, job_disks UNUSED or job_disks TOO LOW can potentially be successfully executed at regular partitions. Note that there appears an option of very low resource utilization. This means that disk operations might be easily replaced with network file system operations with minimal additional overhead or even without it.

For those jobs that are tagged with job_nodes SINGLE and avg_LA SINGLE CORE or avg_LA LOW, it is quite reasonable to inquire what for it was submitted to the supercomputer. Such jobs use a single node and a single core (or just few processes per node) and can potentially run well on a desktop. Unfortunately such jobs are met very often.

Users who submit types of jobs mentioned above must be contacted to figure out the reasons of the revealed facts of inappropriate and inefficient resource utilization. The problems found should be resolved. If cheating is met or it is proved that the executed jobs do not really need HPC resources, quotas for corresponding user accounts and projects can be reduced to the extent of blocking.

Let us take a look at one of real-life examples. Figure 1 illustrates the filtered job list allocated to regular partitions with automatically avg_LA_SINGLE_CORE tag assigned. It is clearly seen that the jobs have a low LoadAverage close to 1 as filtered by the tag, at the same time having very low CPU_user. Note, that it is not a test partition and all jobs are run on a single node, grabbing 8 cores on regular4 and hdd4 partitions!

A close look at the longest job owner that was cancelled by timeout illustrates that the user always runs such single-node, even single-process jobs regardless of partitions (Figure 2).

Fig. 1. Filtered single-process jobs found in real job queue in various regular partitions

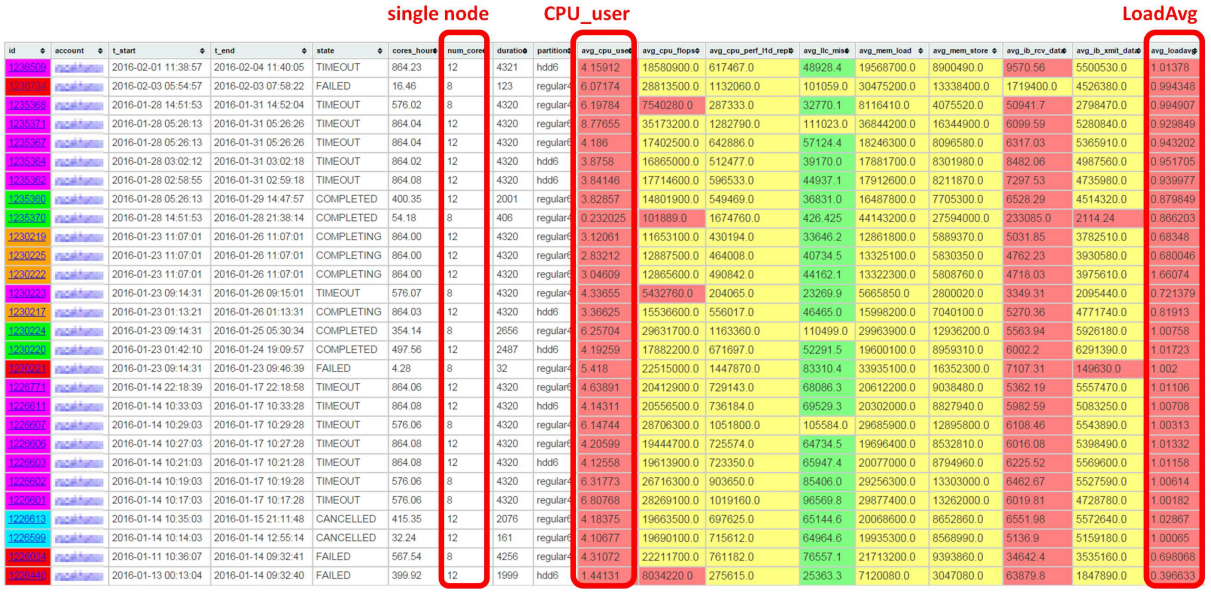


Fig. 2. Filtered single-process jobs found in real job queue in various regular partitions

3.2. Finding jobs with high Flops intensity and high efficiency

Apart from finding problem cases, there is a task of finding well-optimized jobs that utilize HPC resources with high efficiency. These jobs owners are usually experienced users with high qualification in parallel programming and fine tuning of software that secures highly efficient supercomputer load. The contact with such users is very important first and foremost to learn the techniques used and share them with novice users, contributing to FAQ/wiki sections of helpdesk and so on. The experienced users are very likely to be invited to give public lectures, make reports on seminars and join other educational activities.

As a rule, most of jobs with high floating point intensity are tagged with avg_Flops HIGH, avg_CPU_user HIGH tags and most efficient apps in terms of memory utilization are tagged

avg_cache_L1/L3 HIGH. Filtered by these criteria jobs are usually having good data locality, they are well-balanced and show high performance.

The deeper analysis of such jobs allows revealing optimal command line and compiler options for the variety of categories of standard applications and algorithm implementations. Once such a job is approved to be a well-optimized typical example of a SW package usage or algorithm implemented, a proper tag, corresponding to such a category can be set (like job_sw VASP). This provides means for the comparative analysis of similar jobs. This can also serve as a good basis for the more detailed analysis of the whole job collection and revealing inefficient applications and users that use resources inefficiently.

3.3. Finding applications with special need for large amounts of memory

Many users of supercomputer complex run applications that are resource-demanding regarding amount of available memory per process. Such applications are usually effective enough, but are often scaled down in different ways to fit available memory, for example, reducing number of MPI processes per node and so on.

Such applications are usually run on a considerable number of nodes and LoadAvg values are below the number of CPU cores per node. These jobs are usually tagged with avg_memload HIGH, and related to node and core usage tags avg_LA SINGLE CORE, job_nnodes FEW or job_nnodes MANY.

If such a job is found in the 6-cored CPU "Regular 6" partition (tagged with job_partition REGULAR6), even changing allocation to the "Regular 4" partition can be an optimization choice leading to reducing the number of idle CPU cores per node by 4 cores (2*6–2*4).

Some of such applications can also benefit from moving to hybrid MPI+OpemMP or MPI+Cilk models. If such a model cannot be applied, some of the applications can be reallocated to SMP partition with much larger amounts of memory available.

3.4. Revealing categories of issues and inefficient behavior

The accumulation of statistics and knowledge on the problems of parallel applications is one of the most important components of the HPC center job collection analysis. The ability to add tags to the analyzed jobs related to the implementation issues found is a useful feature for this purpose as well as tags corresponding to non-efficient use of computing resources, hardware problems or other features found in course of job execution characteristics analysis.

When analyzing inefficient, abnormal application behavior of a single run or of a sequence of jobs, based on the certain software package, it is often needed to contact the user, the application owner who can provide additional information on the program details: algorithm implementation used, program architecture and structure, computing model and so on up to dependencies on input data and command line options. All this information should be recorded to aid further analysis of similar applications and categories.

If any application run is being analyzed it is useful to mark and tag the used system software details. This is true first of all regarding the math libraries used, compiler and compiler options, MPI type, etc. This provides the basis for the comparative analysis of similar jobs or sequences of jobs. If differences in behavior are found, one can continue deep study on the reason origin: user application reaction, system software configuration, etc.

All widely-met issues like data race, deadlocks and so forth can be marked by special tags (job_bug DATA RACE, job_bug DEADLOCK, etc.). This can help in further analysis of other

jobs. One can compare strange program behavior to the analyzed profiles marked as having specific issues. Once a similar behavior is found, it can be a key to resolving the problems of the originally analyzed job.

Conclusion and future work

Close-future plans include implementation of the Octoshell [7, 8] module for full project-oriented workflow support and authentication, thus securing accessibility of the proposed service for any user. We expect it ready by the middle of 2016. By that time we also plan to extend supported tag set and adjust criteria for existing tags if needed.

To sum up, a user-friendly, useful and effective technique for filtration, grouping and further analysis of the whole queued job collection of large-scale HPC systems based on system monitoring and resource manager data is proposed and implemented. The developed tool is evaluated in the every-day practice of the Supercomputer Center of Lomonosov Moscow State University, providing means for effective analysis for any and every user application run. The priceless collection of information on all finished jobs is already being enriched in a 24/7 mode for several month.

The work was funded in part by the Russian Foundation for Basic Research (grants №16–07–00972A, №13–07–00786A), Russian Presidential study grant (SP-1981.2016.5), and by the Ministry of Education and Science of the Russian Federation, Agreement No. 14.607.21.0006 (unique identifier RFMEFI60714X0006).

References

1. *Top50 Supercomputers of Russia and CIS*. Available at: http://top50.supercomputers.ru/ (accessed 15.02.2016).
2. *Top500 Supercomputer Sites*. Available at: http://top500.org/ (accessed:15.02.2016).
3. Antonov A., Zhumatiy S., Nikitenko D., Stefanov K., Teplov A., Shvets P. *Analysis of Dynamic Characteristics of Job Stream on Supercomputer System Numerical Methods and Programming*. 2013. vol. 14, no. 2. pp. 104–108.
4. Safonov A., Kostenetskiy P., Borodulin K., Melekhin F. A Monitoring System for Supercomputers of SUSU. *Russian Supercomputing Days International Conference, Moscow, Russian Federation, 28-29 September, 2015, Proceedings*. CEUR Workshop Proceedings, 2015. vol. 1482. pp. 662–666.
5. Stefanov K. et al. Dynamically Reconfigurable Distributed Modular Monitoring System for Supercomputers (DiMMon). *Procedia Computer Science / Elsevier B.V.*. 2015. vol. 66. pp. 625–634. DOI: 10.1016/j.procs.2015.11.071.
6. Nikitenko D. Complex Approach to Performance Analysis of Supercomputer Systems Based on System Monitoring Data. *Numerical Methods and Programming*. 2014. vol. 15. pp. 85–97.
7. Voevodin V., Zhumatiy S., Nikitenko D. Octoshell: Large Supercomputer Complex Administration System. *Russian Supercomputing Days International Conference, Moscow, Russian Federation, 28-29 September, 2015, Proceedings*. CEUR Workshop Proceedings, 2015. vol. 1482. pp. 69–83.
8. Nikitenko D., Voevodin V., Zhumatiy S. Resolving Frontier Problems of Mastering Large-Scale Supercomputer Complexes. *Proceedings of the ACM International Conference on Computing*

Frontiers (CF'16), Como, Italy, 16-18 May, 2016. ACM New York, NY, USA, 2016. pp. 349–352. DOI: 10.1145/2903150.2903481.

9. Voevodin Vl., Antonov A., Bryzgalov P., Nikitenko D., Zhumatiy S., Sobolev S., Stefanov K., Voevodin Vad. Practice of "Lomonosov" Supercomputer. *Open Systems*. 2012. no. 7. pp. 36–39.
10. Zhumatiy S., Nikitenko D. Approach to Flexible Supercomputers Management. *International Supercomputing Conference Scientific Services & Internet: All Parallelism Edges, Novorossiysk, Russian Federation, 23-28 September, 2013, Proceedings*. MSU, 2013. pp. 296–300.
11. Voevodin Vl. Supercomputer Situational Screen. *Open Systems*. 2014. no. 3. pp. 36–39.
12. Shvets P. , Antonov A., Nikitenko D., Sobolev S., Stefanov K., Voevodin Vad., Voevodin V., Zhumatiy S. An Approach for Ensuring Reliable Functioning of a Supercomputer Based on a Formal Model. *Parallel Processing and Applied Mathematics. 11th International Conference, PPAM 2015, Krakow, Poland, September 6-9, 2015*. Springer International Publishing. vol. 9573. pp. 12–22. DOI: 10.1007/978-3-319-32149-3_2.
13. Voevodin V., Antonov A., Dongarra J. AlgoWiki: an Open Encyclopedia of Parallel Algorithmic Features. *Supercomputing Frontiers and Innovations*. 2015. vol. 2, no. 1. pp. 4–18. DOI: 10.14529/jsfi150101.
14. SLURM Workload Manager. Available at: http://slurm.schedmd.com/ (accessed: 15.02.2016).
15. Cleo Cluster Batch System. Available at: http://sourceforge.net/projects/cleo-bs/ (accessed: 15.02.2016).
16. Ganglia Monitoring System. Available at: http://ganglia.sourceforge.net/ (accessed:15.02.2016).
17. Collectd – The System Statistics Collection Daemon. Available at: https://collectd.org/ (accessed: 15.02.2016).
18. Clustrx. Available at: http://www.t-platforms.ru/products/software/clustrxproductfamily/clustrxwatch.html (accessed: 15.02.2016).
19. jQuery & jQuery UI. Available at: http://jqueryui.com/ (accessed: 15.02.2016).
20. TagIt. Available at: http://aehlke.github.io/tag-it/ (accessed 15.02.2016).

УДК 004.457, 004.382.2 DOI: 10.14529/cmse160403

ИССЛЕДОВАНИЕ ИНТЕГРАЛЬНЫХ ХАРАКТЕРИСТИК СУПЕРКОМПЬЮТЕРНЫХ ПРИЛОЖЕНИЙ ДЛЯ ВСЕГО ПОТОКА ЗАДАЧ БОЛЬШИХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

© 2016 г. Д.А. Никитенко, В.В. Воеводин, А.М. Теплов, С.А. Жуматий, Вад.В. Воеводин, К.С. Стефанов, П.А. Швец

Московский государственный университет имени М.В. Ломоносова
(119991 Москва, ул. Ленинские Горы, д. 1)
E-mail: dan@parallel.ru, alex-teplov@yandex.ru, serg@parallel.ru, vadim@parallel.ru, cstef@parallel.ru, shvets.pavel.srcc@gmail.com

Поступила в редакцию: 11.04.2016

Эффективность работы суперкомпьютерных систем зависит от множества факторов. В условиях одновременной работы множества пользователей особую роль играет контроль использования выделенных для расчетов ресурсов. Важно, чтобы в распоряжении пользователей была подробная информация о свойствах выполненных задач. В условиях групповой работы над прикладными задачами дополнительно стоит выделить необходимость контроля использования ресурсов участниками проекта руководителем работ. К сожалению, такие сведения сейчас как правило не доступны. Этот пробел призван восполнить разработанный авторами подход к получению и исследованию интегральных характеристик суперкомпьютерных приложений для всего потока задач больших суперкомпьютерных систем. В основе подхода лежит использование данных системного мониторинга, построение интегральных характеристик отдельных запусков для всего множества выполненных задач, деление их на классы, выявление особенностей запусков.

Ключевые слова: суперкомпьютер, эффективность, системный мониторинг, классы задач, интегральные характеристики задач, поток задач, контроль использования вы-числительных ресурсов.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Nikitenko D.A., Voevodin V.V., Teplov A.M., Zhumatiy S.A., Voevodin Vad.V., Stefanov K.S., Shvets P.A. Supercomputer Application Integral Characteristics Analysis for the Whole Queued Job Collection of Large-Scale HPC Systems // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2016. Т. 5, № 4. С. 32–45. DOI: 10.14529/cmse160403.

Литература

1. Top50 Supercomputers of Russia and CIS. URL: http://top50.supercomputers.ru/ (дата обращения: 15.02.2016).

2. Top500 Supercomputer sites. URL: http://top500.org/ (дата обращения: 15.02.2016).

3. Antonov A., Zhumatiy S., Nikitenko D., Stefanov K., Teplov A., Shvets P. Analysis of dynamic characteristics of job stream on supercomputer system Numerical Methods and Programming, 2013. Vol. 14, No. 2. P. 104–108.

4. Safonov A., Kostenetskiy P., Borodulin K., Melekhin F. A monitoring system for supercomputers of SUSU // Russian Supercomputing Days International Conference, Moscow, Russian Federation, 28-29 September, 2015, Proceedings. CEUR Workshop Proceedings, 2015. Vol. 1482. P. 662–666.

5. Stefanov K. et al. Dynamically Reconfigurable Distributed Modular Monitoring System for Supercomputers (DiMMon) // Procedia Computer Science / Elsevier B.V., 2015. Vol. 66. P. 625–634.

6. Nikitenko D. Complex approach to performance analysis of supercomputer systems based on system monitoring data // Numerical Methods and Programming, 2014. Vol. 15. P. 85–97.

7. Voevodin V., Zhumatiy S., Nikitenko D. Octoshell: Large Supercomputer Complex Administration System // Russian Supercomputing Days International Conference, Moscow, Russian Federation, 28-29 September, 2015, Proceedings. CEUR Workshop Proceedings, 2015. Vol. 1482. P. 69–83.

8. Nikitenko D., Voevodin V., Zhumatiy S. Resolving frontier problems of mastering large-scale supercomputer complexes // Proceedings of the ACM International Conference on Computing Frontiers (CF'16), Como, Italy, 16-18 May, 2016. ACM New York, NY, USA, 2016. P. 349–352.

9. Voevodin Vl., Antonov A., Bryzgalov P., Nikitenko D., Zhumatiy S., Sobolev S., Stefanov K., Voevodin Vad. Practice of "Lomonosov" Supercomputer // Open systems, 2012. No. 7. P. 36–39.

10. Zhumatiy S., Nikitenko D. Approach to flexible supercomputers management // International supercomputing conference Scientific Services & Internet: all parallelism edges, Novorossiysk, Russian Federation, 23-28 September, 2013, Proceedings. MSU, 2013. P. 296–300.

11. Voevodin Vl. Supercomputer situational screen // Open systems, 2014. No. 3. P. 36–39.

12. Shvets P. , Antonov A., Nikitenko D., Sobolev S., Stefanov K., Voevodin Vad., Voevodin V., Zhumatiy S. An Approach for Ensuring Reliable Functioning of a Supercomputer Based on a Formal Model // Parallel Processing and Applied Mathematics. 11th International Conference, PPAM 2015, Krakow, Poland, September 6-9, 2015. Springer International Publishing. Vol. 9573. P. 12–22.

13. Voevodin V., Antonov A., Dongarra J. AlgoWiki: an Open Encyclopedia of Parallel Algorithmic Features // Supercomputing Frontiers and Innovations, 2015. Vol. 2, No.1. P. 4–18.

14. SLURM workload manager. URL: http://slurm.schedmd.com/ (дата обращения: 15.02.2016).

15. Cleo cluster batch system. URL: http://sourceforge.net/projects/cleo-bs/ (дата обращения: 15.02.2016).

16. Ganglia Monitoring System. URL: http://ganglia.sourceforge.net/ (дата обращения: 15.02.2016).

17. Collectd – The system statistics collection daemon. URL: https://collectd.org/ (дата обращения: 15.02.2016).

18. Clustrx. URL: http://www.t-platforms.ru/products/software/clustrxproductfamily/clustrxwatch.html (дата обращения: 15.02.2016).

19. jQuery & jQuery UI. URL: http://jqueryui.com/ (дата обращения: 15.02.2016).

20. TagIt. URL: http://aehlke.github.io/tag-it/ (дата обращения: 15.02.2016).

УДК 004.272 DOI: 10.14529/cmse160404

ОПТИМИЗАЦИЯ ОБНАРУЖЕНИЯ КОНФЛИКТОВ В ПАРАЛЛЕЛЬНЫХ ПРОГРАММАХ С ТРАНЗАКЦИОННОЙ ПАМЯТЬЮ*

© 2016 г. И.И. Кулагин[1], М.Г. Курносов[2]

[1]*Сибирский государственный университет телекоммуникаций и информатики (630102 Новосибирск, ул. Кирова, д. 86),*

[2]*Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» имени В.И. Ульянова (Ленина) (197376 Санкт-Петербург, ул. Профессора Попова, д. 5)*

E-mail: ivan.i.kulagin@gmail.com, mkurnosov@gmail.com

Поступила в редакцию: 10.03.2016

В настоящее время активно развивается альтернативный подход к созданию масштабируемых и потокобезопасных параллельных программ для многопроцессорных систем с общей памятью — технология транзакционной памяти (transactional memory). Ожидается, что она войдет в стандарт языка C++17. В данной работе предложен метод оптимизации обнаружения конфликтов (конкурентного доступа потоков к общим областям памяти), возникающих при выполнении параллельных программ на базе транзакционной памяти. Реализован модуль компилятора GCC для профилирования параллельных программ и адаптивной настройки параметров реализации транзакционной памяти под программу. Эффективность метода исследована на тестовых программах из пакета STAMP.

Ключевые слова: программная транзакционная память, параллельное программирование, профилирование, компиляторы.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Кулагин И.И., Курносов М.Г. Оптимизация обнаружения конфликтов в параллельных программах с транзакционной памятью // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2016. Т. 5, № 4. С. 46–60. DOI: 10.14529/cmse160404.

Введение

Синхронизация доступа к разделяемым ресурсам является одной из важных и сложных задач при разработке параллельных алгоритмов для многопоточных программ. Используя примитивы синхронизации (семафоры, мьютексы и др.), программист должен обеспечить не только корректность программы — отсутствие взаимных блокировок (deadlock, livelock) и состояний гонок за данными (data race), но и минимизировать время ожидания доступа к критическим секциям (разделяемым ресурсам). Классические методы синхронизации, основанные на механизме блокировок, позволяют создавать в коде программы критические секции, выполнение которых возможно только одним потоком в каждый момент времени [1]. Такие примитивы синхронизации позволяют защитить участок кода программы, одновременно выполняющийся множеством потоков.

*Статья рекомендована к публикации программным комитетом Международной научной конференции «Параллельные вычислительные технологии — 2016».

Анализ многопоточных программ показывает, что при одновременном выполнении потоками одной критической секций в ней может происходить обращения по разным адресам памяти и, следовательно, возникновение состояния гонки данных маловероятно. Например, такая ситуация наблюдается, если в критическую секцию помещен код добавления элемента в хеш-таблицу (рис. 1). Если все потоки в один момент времени обратятся к функции для добавления нового элемента с хеш-кодом i, то с большой долей вероятности ключи key будут иметь разные хеш-коды и, как следствие, каждый поток будет выполнять добавление нового узла в отдельный связный список $h[i]$. Здесь блокировка мьютексом всей функции является избыточной и неэффективной.

```
function hashtable_add(h, key, value)
    lock_acquire()
    i = hash(key)
    list_add_front(h[i], key, value)
    lock_release()
end function
```

Рис. 1. Добавление пары $(key, value)$ в хеш-таблицу h

Активно развиваются два альтернативных подхода к созданию потокобезопасных и масштабируемых многопоточных программ — это *неблокирующие алгоритмы и структуры данных* (lock-free data structures) [2] и *транзакционная память* (ТП, *transactional memory*) [3, 4]. Использование неблокирующих структур данных, как правило, требует глубокой переработки многопоточных программ и совместно используемых потоками объектов в памяти [2].

Менее трудоемким и прозрачным для программиста видится использование технологии транзакционной памяти, основная идея которой заключается в защите от конкурентного доступа области памяти программы, а не участка кода, как в случае использования блокировок на базе мьютексов. Известны как программные реализации транзакционной памяти (software transactional memory — STM): LazySTM, TinySTM, GCC TM, DTMC, RSTM, STMX, STM Monad, так и аппаратные реализации в процессорах (hardware transactional memory): Intel TSX, AMD ASF, Oracle Rock, IBM POWER8, IBM PowerPC A2.

В рамках *программной транзакционной памяти* программисту предоставляются языковые конструкции или API для формирования в программе *транзакционных секций* (transactional section) — участков кода, в которых осуществляется защита совместно используемых областей памяти. Выполнение потоками таких секций осуществляется без их блокирования. На среду выполнения (runtime) ложатся задачи по контролю за корректностью выполнения транзакций. Если во время выполнения транзакции другие потоки одновременно с ней не модифицировали защищенную область памяти, то транзакция считается корректной, и она фиксируется. Если же два или более потока при выполнении транзакций обращаются к одной и той же области памяти и как минимум один из них выполняет операцию записи, то возникает *конфликт* (аналог состояния гонки данных). Для его разрешения выполнение одной или нескольких транзакций может быть либо приостановлено (до завершения конфликтующей транзакции), либо прервано, а все модифицированные ими (их потоками) области памяти приведены в исходное состояние (на момент старта транзакции) – *отмена транзакции и восстановление* (cancel and rollback).

Для того чтобы обнаруживать конфликты, runtime-система должна отслеживать попытки одновременного доступа к одной и той же области памяти. Это реализуется путем поддержки информации о состоянии защищаемых регионов памяти. Возможны два уровня гранулярности контролируемых областей: *уровень программных объектов* (object-based STM) и *уровень слов памяти* (word-based STM).

Уровень программных объектов подразумевает поддержку runtime-системой метаданных о состоянии каждого объекта программы. Например, объектов в C++-программе.

Для реализации уровня слов памяти в простейшем случае требуется каждый байт линейного адресного пространства процесса сопровождать метаданными, что является практически невозможным. Вместо этого линейное адресное пространство процесса разбивается на фиксированные блоки, каждый из которых сопровождается метаданными о состоянии (подход, подобный прямому отображению физических адресов на кеш-память процессора) [5, 6]. Это приводит к тому, что множеству областей памяти соответствуют одни метаданные, что является источником возникновения ложных конфликтов. *Ложный конфликт* (false conflict)— это ситуация, при которой два или более потока во время выполнения транзакции обращаются к разным участкам линейного адресного пространства, но отображаемым на одни и те же метаданные. Поэтому runtime-система воспринимает такую ситуацию как конфликт (data race), хотя на самом деле таковой отсутствует.

Ложные конфликты существенно снижают эффективность параллельных STM-программ. Поэтому остро стоит задача разработки алгоритмов обнаружения и сокращения числа ложных конфликтов в реализациях STM.

В данной работе предлагается метод оптимизации ложных конфликтов по результатам предварительного профилирования C/C++ STM-программы. Для чего разработан программный инструментарий, который позволяет выполнять профилирование STM-программ и варьировать параметры реализации runtime-библиотеки STM в компиляторе GCC (libitm).

Работа организована следующим образом. Раздел 1 вводит основные понятия связанные с программной транзакционной памятью, содержит примеры использования данного примитива синхронизации, описывает основные аспекты реализации STM в runtime-системах, а также содержит описание предложенного метода сокращения числа ложных конфликтов. Раздел 2 раскрывает детали реализации программного инструментария для оптимизации ложных конфликтов, возникающих при выполнении параллельных программ с транзакционной памятью. В разделе 3 показаны основные результаты экспериментального исследования предложенного метода оптимизации ложных конфликтов.

1. Метод оптимизации обнаружения конфликтов

Международным комитетом ISO по стандартизации языка C++, в рамках рабочей группы WG21, ведутся работы по внедрению транзакционной памяти в стандарт языка. Окончательное внедрение планируется в стандарт C++17. На сегодняшний день предложен черновой вариант спецификации поддержки транзакционной памяти в C++ [7]. Она реализована в компиляторе GCC, начиная с версии 4.8 и предоставляет ключевые слова __transaction_atomic, __transaction_relaxed для создания транзакционных секций, а также __transaction_cancel для принудительной отмены транзакции.

Для выполнения транзакционных секций runtime-системой создаются транзакции. *Транзакция* (transaction) — это конечная последовательность операций транзакционного

чтения/записи памяти. Операция *транзакционного чтения* выполняет копирование содержимого указанного участка общей памяти в соответствующий участок локальной памяти потока. *Транзакционная запись* копирует содержимое указанного участка локальной памяти в соответствующий участок общей памяти, доступной всем потокам.

Инструкции транзакций выполняются потоками параллельно (конкурентно). После завершения выполнения транзакция может быть либо *зафиксирована* (commit), либо *отменена* (cancel). Фиксация транзакции подразумевает, что все сделанные в рамках нее изменения памяти становятся необратимыми. При отмене транзакции ее выполнение прерывается, а состояние всех модифицированных областей памяти восстанавливается в исходное с последующим перезапуском транзакции (*откат транзакции*, rollback).

Отмена транзакции происходит в случае *обнаружения конфликта* — ситуации, при которой два или более потока обращаются к одному и тому же участку памяти и как минимум один из них выполняет операцию записи.

Для разрешения конфликта разработаны различные походы, например, можно приостановить на некоторое время или отменить одну из конфликтующих транзакций.

На рис. 2 представлен пример создания транзакционной секции, в теле которой выполняется добавление элемента в хэш-таблицу множеством потоков. После выполнения тела транзакционной секции каждый поток приступит к выполнению кода, следующего за ней, в случае отсутствия конфликтов. В противном случае поток повторно будет выполнять транзакцию до тех пор, пока его транзакция не будет успешно зафиксирована.

```
/* Совместно используемая хеш-таблица */
hashtable_t *h;
/* Код потоков */
void *thread_start(void *arg) {
    struct data *d = (struct data *)arg;
    prepareData(d);
    /* Транзакционная секция */
    __transaction_atomic {
        /* Добавление элемента в хеш-таблицу */
        struct data *d = (struct data *)arg;
        hashtable_insert(h, d);
    }
    saveData(d);
    return NULL;
}
```

Рис. 2. Добавление пары $(key, value)$ в хеш-таблицу h

Основными аспектами реализации транзакционной памяти в runtime-системах являются:

- политика обновления объектов в памяти;
- стратегия обнаружения конфликтов;
- метод разрешения конфликтов.

Политика обновления объектов в памяти определяет, когда изменения объектов в рамках транзакции будут записаны в память. Распространение получили две основные политики — ленивая и ранняя. *Ленивая* политика обновления объектов в памяти (lazy version management) откладывает все операции с объектами до момента фиксации транзакции.

Все операции записываются в специальном журнале (redo log), который при фиксации используется для отложенного выполнения операций. Очевидно, что это замедляет операцию фиксации, но существенно упрощает процедуры ее отмены и восстановления. Примером реализаций ТП, использующих данную политику, являются RSTM-LLT [8] и RSTM-RingSW [9, 11].

Ранняя политика обновления (eager version management) предполагает, что все изменения объектов сразу записываются в память. В журнале отката (undo log) фиксируются все выполненные операции с памятью. Он используется для восстановления оригинального состояния модифицируемых участков памяти в случае возникновения конфликта. Эта политика характеризуется быстрым выполнением операции фиксации транзакции, но медленным выполнением процедуры ее отмены. Примерами реализаций, использующих раннюю политику обновления данных, являются GCC (libitm), TinySTM [5], LSA-STM [6], Log-TM [10], RSTM [8] и др.

Момент времени, когда инициируется алгоритм обнаружения конфликта, определяется *стратегией обнаружения конфликтов*. При *отложенной стратегии* (lazy conflict detection) алгоритм обнаружения конфликтов запускается на этапе фиксации транзакции [11]. Недостатком этой стратегии является то, что временной интервал между возникновением конфликта и его обнаружением может быть достаточно большим. Эта стратегия используется в RSTM-LLT [8] и RSTM-RingSW [8, 9, 11].

Пессимистичная стратегия обнаружения конфликтов(eager conflict detection) запускает алгоритм их обнаружения при каждой операции обращения к памяти. Такой подход позволяет избежать недостатков отложенной стратегии, но может привести к значительным накладным расходам, а также, в некоторых случаях, может привести к увеличению числа откатов транзакций. Стратегия реализована в TinySTM [5], LSA-STM [6] и TL2 [12]. В компиляторе GCC (libitm) реализован комбинированный подход к обнаружению конфликтов — отложенная стратегия используется совместно с пессимистической.

Для обнаружения конфликтных операций требуется отслеживать изменения состояния используемых областей памяти. Информация о состоянии может соответствовать областям памяти различной степени гранулярности. Выбор гранулярности обнаружения конфликтов — один из ключевых моментов при реализации программной транзакционной памяти.

На сегодняшний день используются два уровня гранулярности: *уровень программных объектов* (object-based STM) и *уровень слов памяти* (word-based STM). Уровень программных объектов подразумевает отображение объектов модели памяти языка (объекты C++, Java, Scala) на метаданные runtime-библиотеки. При использовании уровня слов памяти осуществляется отображение блоков линейного адресного пространства процесса на метаданные. Метаданные хранятся в таблице, каждая строка которой соответствует объекту программы или области линейного адресного пространства процесса. В строке содержатся номер транзакции, выполняющей операцию чтения/записи памяти; номер версии отображаемых данных; их состояние и др. Модификация метаданных выполняется runtime-системой с помощью атомарных операций процессора.

В данной работе рассматривается реализация программной транзакционной памяти в компиляторе GCC, использующая уровень слов памяти (в версиях GCC 4.8+ размер блока – 16 байт).

На рис. 3 представлен пример организации метаданных транзакционной памяти с использованием уровня слов памяти (GCC 4.8+). Линейное адресное пространство процес-

са фиксированными блоками циклически отображается на строки таблицы, подобно кешу прямого отображения. Выполнение операции записи приведет к изменению поля «состояние» соответствующей строки таблицы на «заблокировано». Доступ к области линейного адресного пространства, у которой соответствующая строка таблицы помечена как «заблокировано», приводит к конфликту.

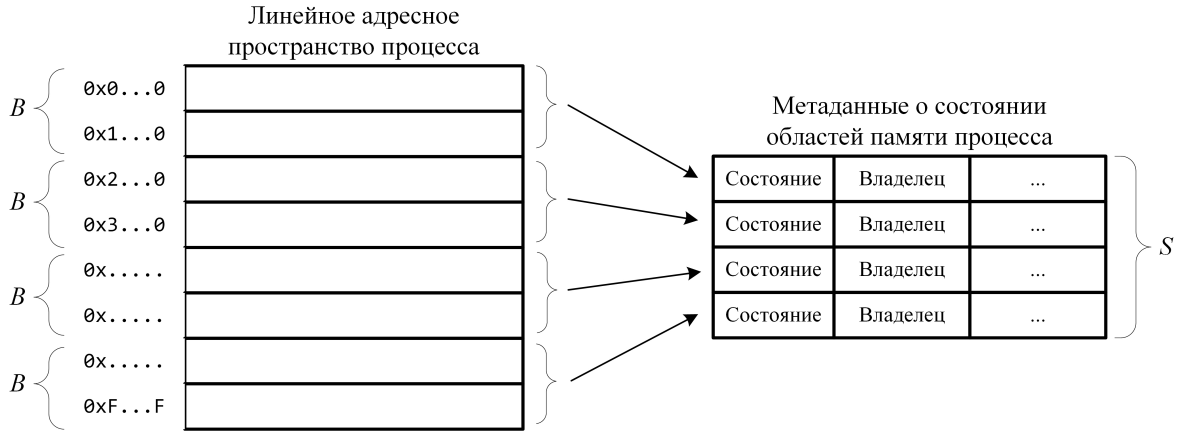


Рис. 3. Таблица с метаданными GCC 4.8+ (word-based STM): $B = 16$, $S = 2^{19}$

Основными параметрами транзакционной памяти с использованием уровня слов памяти являются число S строк таблицы и количество B адресов линейного адресного пространства, отображаемых на одну строку таблицы. От выбора этих параметров зависит число ложных конфликтов — ситуаций аналогичных ситуации ложного разделения данных при работе кеша процессора. В текущей реализации GCC (4.8-5.1) эти параметры фиксированы [13].

При отображении блоков линейного адресного пространства процесса на метаданные runtime-библиотеки возникают коллизии. Это неизбежно, так как размер таблицы метаданных гораздо меньше размера линейного адресного пространства процесса. Коллизии приводят к возникновению ложных конфликтов. *Ложный конфликт* — ситуация, при которой два или более потока во время выполнения транзакции обращаются к разным участкам линейного адресного пространства, но сопровождаемым одними и теми же метаданными о состоянии, и как минимум один поток выполняет операцию записи. Таким образом, ложный конфликт — это конфликт, который происходит не на уровне данных программы, а на уровне метаданных runtime-библиотеки.

Возникновение ложных конфликтов приводит к откату транзакций так же, как и возникновение обычных конфликтов, несмотря на то что состояние гонки за данными не возникает, что влечет за собой увеличение времени выполнения STM-программ. Сократив число ложных конфликтов, можно существенно уменьшить время выполнения программы.

На рис. 4 показан пример возникновения ложного конфликта в результате коллизии отображения линейного адресного пространства на строку таблицы. Поток 1 при выполнении операции записи над областью памяти с адресом $A1$ захватывает соответствующую строку таблицы. Выполнение операции чтения над областью памяти с адресом $A2$ потоком 2 приводит к возникновению конфликта, несмотря на то что операции чтения и записи выполняются над различными адресами. Последнее обусловлено тем, что 1 и 2 отображены на одну строку таблицы метаданных.

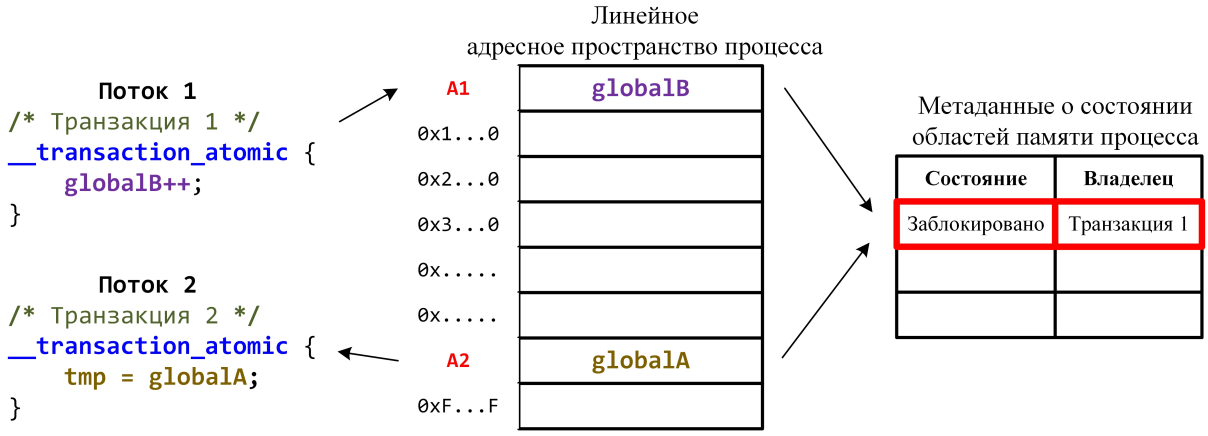


Рис. 4. Пример возникновения ложного конфликта при выполнении двух транзакций (GCC 4.8+)

В работе [14] для минимизации числа ложных конфликтов предлагается использовать вместо таблицы с прямой адресацией (как в GCC 4.8+), в которой индексом является часть линейного адреса, хеш-таблицу, коллизии в которой разрешаются методом цепочек. В случае отображения нескольких адресов на одну запись таблицы каждый адрес добавляется в список и помечается тэгом для идентификации (рис. 5). Такой подход позволяет избежать ложных конфликтов, однако накладные расходы на синхронизацию доступа к метаданным существенно возрастают, так как значительно увеличивается количество атомарных операций «сравнение с обменом» (compare and swap — CAS).

Рис. 5. Хеш-таблица для хранения метаданных

Авторами предложен метод, позволяющий сократить число ложных конфликтов в STM-программах. Предполагается, что метаданные организованы в виде таблицы с прямой адресацией. Суть метода заключается в автоматической настройке параметров S и B таблицы под динамические характеристики конкретной STM-программы. Метод включает три этапа.

Этап 1. Внедрение функций библиотеки профилирования в транзакционные секции. На первом этапе выполняется компиляция C/C++ STM-программы с использованием разработанного модуля анализа транзакционных секций и внедрения вызова функций библиотеки профилирования (модуль расширения GCC). В ходе статического анализа транзакционных секций STM-программ выполняется внедрение кода для регистрации обращений

к функциям Intel TM ABI (_ITM_beginTransaction, _ITM_comitTransaction, _ITM_LU4, _ITM_WU4 и др.). Детали реализации модуля описаны ниже.

Этап 2. Профилирование программы. На данном этапе выполняется запуск STM-программы в режиме профилирования. Профилировщик регистрирует все операции чтения/записи памяти в транзакциях. В результате формируется протокол (trace), содержащий информацию о ходе выполнения транзакционных секций:

- адрес и размер области памяти, над которой выполняется операция;
- временная метка (timestamp) начала выполнения операции.

Этап 3. Настройка параметров таблицы. По протоколу определяются средний размер W читаемой/записываемой области памяти во время выполнения транзакций. По значению W подбираются субоптимальные параметры B и S таблицы, с которыми STM-программа компилируется. Эксперименты с тестовыми STM-программами из пакета STAMP (6 типов STM-программ) позволили сформулировать эвристические правила для подбора параметров B и S по значению W. Значение параметра S целесообразно выбирать из множества $\{2^{18}, 2^{19}, 2^{20}, 2^{21}\}$. Значение параметра B выбирается следующим образом:

- если $W = 1$ байт, то $B = 2^4$ байт;
- если $W = 4$ байт, то $B = 2^6$ байт;
- если $W = 8$ байт, то $B = 2^7$ байт;
- если $W >= 64$ байт, то $B = 2^8$ байт.

2. Программный инструментарий для сокращения ложных конфликтов

Авторами разработан программный инструментарий (STM false conflict optimizer) для оптимизации ложных конфликтов, возникающих при выполнении параллельных программ с транзакционной памятью. Инструментарий позволяет выполнять профилирование STM-программ. Информация, полученная в результате профилирования, предоставляет достаточно сведений о динамических характеристиках транзакционных секций для того, чтобы ответить на вопрос: «Фиксации каких транзакций или операции над какими данными приводят к отмене других транзакций?». Кроме этого, разработанное программное средство позволяет определить значения субоптимальных значений параметров реализации runtime-системы ТП, а именно число строк таблицы метаданных о состоянии областей памяти и количество адресов линейного адресного пространства, отображаемых на одну строку таблицы.

2.1. Функциональная структура пакета

Программный инструментарий состоит из трех основных компонентов (рис. 6):

- модуль внедрения функций библиотеки профилирования в код транзакционных секций ($tm\_prof\_analyzer$);
- библиотека профилирования параллельной программы с транзакционной памятью ($libitm\_prof$);
- модуль анализа протокола выполнения транзакционных секций, установки значений параметров реализации ($tm\_proto\_analyzer$).

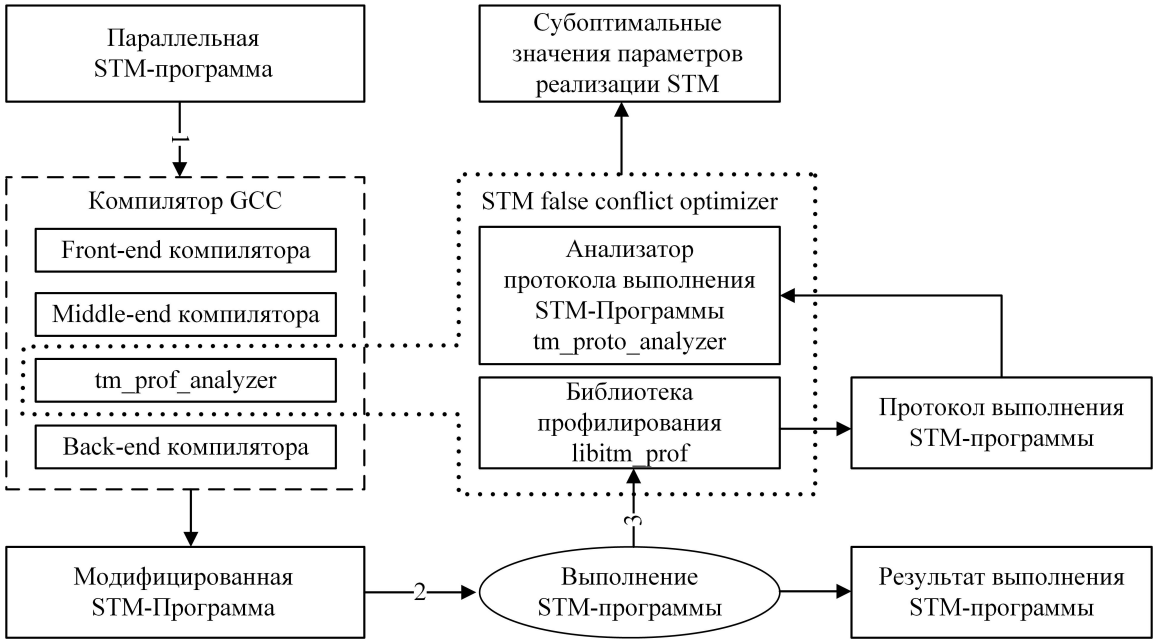


Рис. 6. Функциональная структура разработанного пакета; 1 — компиляция STM-программы; 2 — запуск STM-программы под управление профилировщика; 3 — обращение к функциям профилировщика

2.2. Внедрение функций профилировщика

STM-компилятор осуществляет трансляцию транзакционных секций в последовательность вызовов функций runtime-системы поддержки TM [15]. Компания Intel предложила спецификацию ABI для runtime-систем поддержки транзакционной памяти — Intel TM ABI [16]. Компилятор GCC, библиотека libitm, реализует этот интерфейс начиная с версии 4.8. На рис. 7 представлен пример трансляции компилятором GCC транзакционной секции в обращения к функциям Intel TM ABI.

В общем случае последовательность выполнения транзакции следующая:

1. Создание транзакции (вызов _ITM_beginTransaction) и анализ ее состояния. Если состояние транзакции содержит флаг принудительной отмены, то выполнение продолжается с метки <L3>, т.е. осуществляется выход из транзакции, иначе выполнение тела транзакции начинается с метки <L2>.
2. Выполнение транзакции. Если выполняется принудительная отмена транзакции, то в состоянии устанавливается флаг принудительной отмены (a_abortTransaction) и управление передается метке <L1>.
3. Попытка фиксации транзакции (вызов _ITM_commitTransaction). В случае возникновения конфликта транзакция отменяется, в состояние транзакции записывается причина отмены и выполнение транзакции повторяется начиная с метки <L1>.

Разработанный модуль $tm\_prof\_analyzer$ внедрения функций библиотеки профилирования выполнен в виде встраиваемого модуля компилятора GCC. Программист компилирует STM-программу с ключом $-fplugin = tm\_prof\_analyzer.so$. Модуль внедрения выполняет анализ промежуточного представления $GIMPLE$ транзакционных секций и добавляет функции регистрации обращений к функциям Intel TM ABI: регистрация начала транзакции и ее фиксации, транзакционное чтения/запись областей памяти.

```
int a, b;                                       ...
...                                             state = _ITM_beginTransaction()
__transaction_atomic {                      <L1>:
    if (a == 0)                                 if (state & a_abortTransaction)
        b = 1;                                      goto <L3>;
    else                                        else
        a = 0;                                      goto <L2>;
}                                           <L2>:
...                                             if (_ITM_LU4(&a) == 0)
                                                    _ITM_WU4(&b, 1);
                                                else
                                                    _ITM_WU4(&a, 0);
                                                _ITM_commitTransaction();
                                            <L3>:
                                                ...
/*Исходная Транзакционная секция*/          /*Трансформированная транзакционная секция*/
```

Рис. 7. Трансляция транзакционной секции компилятором GCC; код слева — исходная транзакционная секция; код справа — промежуточное представление трансформированной транзакционной секции

На рис. 8 представлен пример внедрения вызовов функций библиотеки профилирования в транзакционную секцию. Функции с префиксом $tm\_prof\_$ выполняют регистрацию событий.

Во время выполнения STM-программы под управлением профилировщика ($libitm\_prof$), функции регистрации обращений к интерфейсам Intel TM ABI заносят в протокол адреса и размер областей памяти, над которыми выполняются операции, а также время начала выполнения операций. После завершения выполнения STM-программы формируется протокол выполнения программы, на основе которого модуль анализа ($tm\_proto\_analyzer$) осуществляет выбор субоптимальных параметров таблицы метаданных транзакционной памяти.

3. Эксперименты

Экспериментальное исследование проводилось на вычислительной системе, оснащенной двумя четырехъядерными процессорами Intel Xeon E5420. В данных процессорах отсутствует поддержка аппаратной транзакционной памяти (Intel TSX). В качестве тестовых программ использовались многопоточные STM-программы из пакета STAMP [9, 11, 12]. Число потоков варьировалось от 1 до 8. Тесты собирались компилятором GCC 5.1.1. Операционная система GNU/Linux Fedora 21 x86_64.

В рамках экспериментов измерялись значения двух показателей:

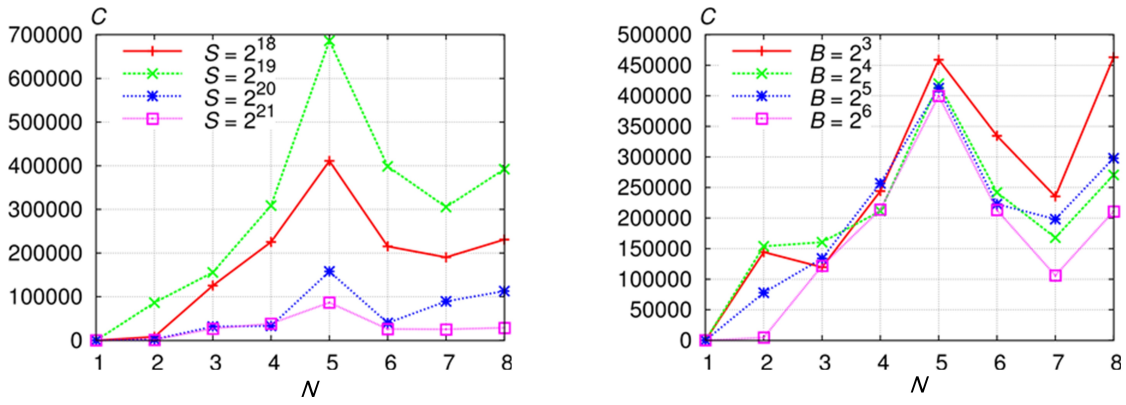
- время t выполнения STM-программы;
- количество C ложных конфликтов в программе.

На рис. 9a, 9b, 10a, 10b показана зависимость количества C ложных конфликтов и времени t выполнения теста от числа потоков при различных значениях параметров B и S. Результаты приведены для программы genome из пакета STAMP. В ней порядка 10 транзакционных секций, реализующих операции над хеш-таблицей и связными списками. Видно, что увеличение значений параметров S и B приводит к уменьшению числа возможных кол-

```
int a, b;                                       ...
...                                             state = _ITM_beginTransaction()
__transaction_atomic {                          tm_prof_begin(state);
    if (a == 0)                             <L1>:
        b = 1;                                  if (state & a_abortTransaction)
    else                                            goto <L3>;
        a = 0;                                  else
}                                                   goto <L2>;
...                                             <L2>:
                                                tm_prof_operation(sizeof(a));
                                                if (_ITM_LU4(&a) == 0) {
                                                    tm_prof_operation(sizeof(b));
                                                    _ITM_WU4(&b, 1);
                                                } else {
                                                    tm_prof_operation(sizeof(a));
                                                    _ITM_WU4(&a, 0);
                                                }
                                                _ITM_commitTransaction();
                                                tm_prof_commit();
                                            <L3>:
                                                ...
/*Исходная Транзакционная секция*/          /*Трансформированная транзакционная секция*/
```

Рис. 8. Встраивание в транзакционную секцию функций библиотеки профилирования; код слева — исходная транзакционная секция; код справа — промежуточное представление трансформированной транзакционной секции

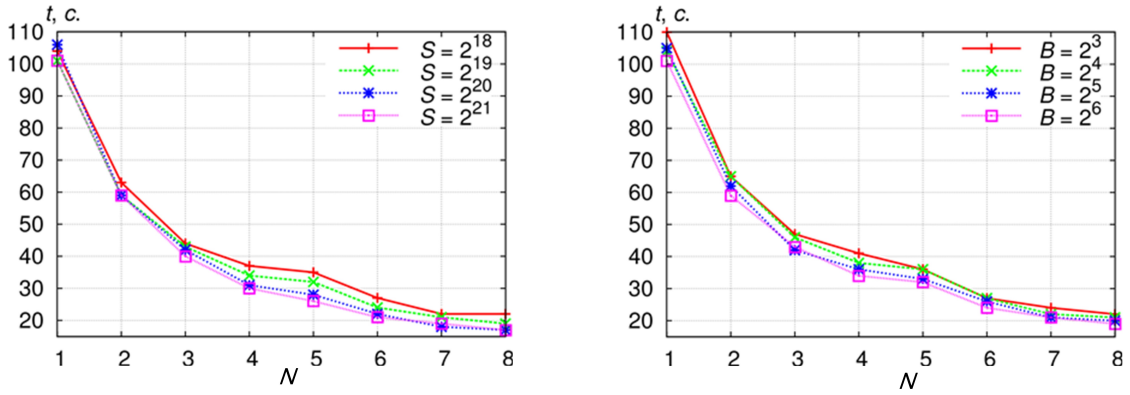
лизий (ложных конфликтов), возникающих при отображении адресов линейного адресного пространства процесса на записи таблицы. При размере таблицы 2^{21} записей, на каждую из которых отображается 2^6 адресов линейного адресного пространства, достигается минимум времени выполнения теста genome, а также минимум числа ложных конфликтов.



(a) $S = 2^{19}$ (b) $B = 2^6$

Рис. 9. Зависимость числа C ложных конфликтов от числа N потоков

Время выполнения теста genome удалось сократить в среднем на 20% за счет минимизации числа ложных конфликтов.



(a) $S = 2^{19}$ (b) $B = 2^6$

Рис. 10. Зависимость времени t выполнения теста от числа N потоков

Заключение

Основным вкладом данной работы является метод оптимизации параметров внутренних структур данных runtime-библиотеки транзакционной памяти компилятора GCC (libitm) под конкретное приложение. Создан программный инструментарий сокращения числа ложных конфликтов в STM-программах. Программный инструментарий позволяет осуществлять инструментацию и профилирование параллельных STM-программ. Результаты профилирования показывают хронологию выполнения транзакций. В данной работе результаты профилирования используются только для выбора значений параметров реализации runtime-библиотеки транзакционной памяти, однако, они могут быть использованы и для других целей, например, для выбора политика обновления объектов в памяти и стратегии обнаружения конфликтов.

Предложенный метод был экспериментально исследован на тестовых программах из пакета STAMP. Эксперименты показали, что применяя разработанный метод, время выполнения теста genome удалось сократить на 20% за счет минимизации числа ложных конфликтов.

В дальнейшем планируется разработать алгоритм реализаций программной транзакционной памяти без централизованного хранения метаданных о состоянии областей памяти процесса.

Работа выполнена в СПбГЭТУ при финансовой поддержке Министерства образования и науки Российской Федерации в рамках договора № 02.G25.31.0149 от 01.12.2015 г.

Литература

1. Herlihy M., Shavit N. The Art of Multiprocessor Programming. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2008. 508 p.
2. Hendler D., Shavit N., Yerushalmi L. A scalable lock-free stack algorithm // Proceedings of the sixteenth annual ACM symposium on Parallelism in algorithms and architectures. 2004. P. 206–215.
3. Кузнецов С.Д. Транзакционная память. URL: www.citforum.ru/programming/digest/transactional_memory/ (дата обращения: 21.06.2016).
4. Shavit N., Touitou D. Software Transactional Memory // Proceedings of the fourteenth annual

ACM symposium on Principles of distributed computing. Aug. 1995. New York, NY, USA, P. 204–213.

5. Felber P., Fetzer C.,Marlier P., Riegel T. Time-based Software Transactional Memory // IEEE Transactions on Parallel and Distributed Systems. December 2010. Vol. 21, Issue 12. P. 1793–1807.
6. Riegel T., Felber P., Fetzer C. A Lazy Snapshot Algorithm with Eager Validation // 20th International Symposium on Distributed Computing. 2006.
7. Luchango V., Maurer J., Moir M. Transactional memory for C++. URL: http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2013/n3718.pdf (дата обращения: 21.06.2016).
8. Scott M.L. Rochester Software Transactional Memory Runtime. URL: www.cs.rochester.edu/research/synchronization/rstm/ (дата обращения: 21.06.2016).
9. Spear M.F., Dalessandro L., Marathe V.J., Scott M.L. A comprehensive strategy for contention management in software transactional memory // Proc. 14th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming. February 2009. P. 141–150.
10. Moore K.E., Bobba J., Moravan M.J., Hill M.D., Wood D.A. LogTM: Log-based transactional memory // Proceedings of the 12th International Symposium on High-Performance Computer Architecture. February 2006. P. 254–265.
11. Spear M.F., Michael M.M., Praun C. RingSTM: scalable transactions with a single atomic instruction // Proceedings of the 20th Annual Symposium on Parallelism in Algorithms and Architectures. June 2008. P. 275–284.
12. Dice D., Shalev O., Shavit N. Transactional locking II // Proceedings of the 20th International Symposium on Distributed Computing. September 2006. Vol. 4167. P. 194–208.
13. Felber P., Fetzer C., Riegel T. Dynamic performance tuning of word-based software transactional memory // Proceedings of the 17th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming. 2008. P. 237–246.
14. Zilles C., Rajwar R. Implications of false conflict rate trends for robust software transactional memory. URL: http://zilles.cs.illinois.edu/papers/tm_false_conflicts.iiswc2007.pdf (дата обращения: 21.06.2016).
15. Olszewski M., Cutler J., Steffan J.G. JudoSTM: A Dynamic Binary-Rewriting Approach to Software Transactional Memory // Proceedings of the 16th International Conference on Parallel Architecture and Compilation Techniques. 2007. P. 365–375.
16. Intel Corporation. Intel Transactional Memory Compiler and Runtime Application Binary Interface. URL: https://software.intel.com/sites/default/files/m/5/a/2/a/f/8097-Intel_TM_ABI_1_0_1.pdf (дата обращения: 21.06.2016)

DOI: 10.14529/cmse160404

OPTIMIZATION OF CONFLICT DETECTION IN PARALLEL PROGRAMS WITH TRANSACTIONAL MEMORY

© 2016 I.I. Kulagin[1], M.G. Kurnosov[2]

[1]*Siberian State University of Telecommunications and Information Science (Kirova 86, Novosibirsk, 630102 Russia),*

[2]*Saint-Petersburg Electrotechnical University "LETI"(Professora Popova 5, St. Petersburg, 197376 Russia)*

E-mail: ivan.i.kulagin@gmail.com, mkurnosov@gmail.com

Received: 10.03.2016

Transactional memory is a perspective abstraction for the creating a scalable parallel programs for multi-core systems. It will be included in C++17. In this work, are proposed optimization method of conflicts detection, that accur in parallel programs with the software transactional memory during execution. The autors have implemented a module for GCC compiler for profiling parallel programs with software transactional memory and a tool for adaptive tuning runtime-library. The efficiency of method is investigated on the STAMP benchmarks.

Keywords: software transactional memory, parallel programming, profiling, compilers.

FOR CITATION

Kulagin I.I., Kurnosov M.G. Optimization of Conflict Detection in Parallel Programs with Transactional Memory. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. 2016. vol. 5, no. 4. pp. 46–60. (in Russian) DOI: 10.14529/cmse160404.

References

1. Herlihy M., Shavit N. *The Art of Multiprocessor Programming.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2008. 508 p.

2. Hendler D., Shavit N., Yerushalmi L. A Scalable Lock-free Stack Algorithm. *Proceedings of the Sixteenth Annual ACM Symposium on Parallelism in Algorithms and Architectures.* 2004. pp. 206–215. DOI: 10.1145/1007912.1007944.

3. Kuznetsov S.D. *Transaktsionnaya pamat* [Transactional Memory]. Available at: http://citforum.ru/programming/digest/transactional_memory/ (accessed: 21.06.2016).

4. Shavit N., Touitou D. Software Transactional Memory. *Proceedings of the Fourteenth Annual ACM Symposium on Principles of Distributed Computing.* Aug. 1995. New York, NY, USA. pp. 204–213. DOI: 10.1145/224964.224987.

5. Felber P., Fetzer C.,Marlier P., Riegel T. Time-based Software Transactional Memory. *IEEE Transactions on Parallel and Distributed Systems.* December 2010. vol. 21, issue 12. pp. 1793–1807. DOI: 10.1109/TPDS.2010.49.

6. Riegel T., Felber P., Fetzer C. A Lazy Snapshot Algorithm with Eager Validation. *20th International Symposium on Distributed Computing.* 2006. DOI: 10.1007/11864219_20.

7. Luchango V., Maurer J., Moir M. *Transactional Memory for C++*. Available at: http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2013/n3718.pdf (accessed: 21.06.2016).

8. Scott M.L. *Rochester Software Transactional Memory Runtime*. Available at: www.cs.rochester.edu/research/synchronization/rstm/ (accessed: 21.06.2016).

9. Spear M.F., Dalessandro L., Marathe V.J., Scott M.L. A Comprehensive Strategy for Contention Management in Software Transactional Memory. *Proc. 14th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*. February 2009. pp. 141–150. DOI: 10.1145/1594835.1504199.

10. Moore K.E., Bobba J., Moravan M.J., Hill M.D., Wood D.A. LogTM: Log-based Transactional Memory. *Proceedings of the 12th International Symposium on High-Performance Computer Architecture*. February 2006. pp. 254–265. DOI: 10.1109/HPCA.2006.1598134.

11. Spear M.F., Michael M.M., Praun C. RingSTM: Scalable Transactions with a Single Atomic Instruction. *Proceedings of the 20th Annual Symposium on Parallelism in Algorithms and Architectures*. June 2008. pp. 275–284. DOI: 10.1145/1378533.1378583.

12. Dice D., Shalev O., Shavit N. Transactional Locking II. *Proceedings of the 20th International Symposium on Distributed Computing*. September 2006. vol. 4167. pp. 194–208. DOI: 10.1007/11864219_14.

13. Felber P., Fetzer C., Riegel T. Dynamic Performance Tuning of Word-based Software Transactional Memory. *Proceedings of the 17th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*. 2008. pp. 237–246. DOI: 10.1145/1345206.1345241.

14. Zilles C., Rajwar R. *Implications of False Conflict Rate Trends for Robust Software Transactional Memory*. Available at: http://zilles.cs.illinois.edu/papers/tm_false_conflicts.iiswc2007.pdf (accessed: 21.06.2016).

15. Olszewski M., Cutler J., Steffan J.G. JudoSTM: A Dynamic Binary-Rewriting Approach to Software Transactional Memory. *Proceedings of the 16th International Conference on Parallel Architecture and Compilation Techniques*. 2007. pp. 365–375. DOI: 10.1109/PACT.2007.4336226.

16. *Intel Corporation. Intel Transactional Memory Compiler and Runtime Application Binary Interface*. Available at: https://software.intel.com/sites/default/files/m/5/a/2/a/f/8097-Intel_TM_ABI_1_0_1.pdf (accessed: 21.06.2016)

УДК 004.4, 378.147 DOI: 10.14529/cmse160405

ОБЗОР АДАПТИВНЫХ МОДЕЛЕЙ ЭЛЕКТРОННОГО ОБУЧЕНИЯ

© 2016 г. Н.С. Силкина, Л.Б. Соколинский
Южно-Уральский государственный университет
(454080 Челябинск, пр. им. В.И. Ленина, д. 76),
E-mail: silkinans@susu.ru, Leonid.Sokolinsky@susu.ru
Поступила в редакцию: 07.10.2016

В статье дается обзор адаптивных моделей электронного обучения. В каждой модели рассматривается структура и способы предоставления образовательного контента. Проводится анализ сильных и слабых сторон рассмотренных моделей электронного обучения. Общее слабое место — отсутствие предопределенной дидактической структуры образовательного объекта. Данный недостаток существенно ограничивает возможность автоматической проверки электронного учебного курса на дидактическую полноту. В заключении статьи определяются черты новой модели электронного обучения, включающей в себя средства для описания дидактической структуры образовательных объектов.

Ключевые слова: электронное обучение, электронный учебный курс, модель электронного обучения, образовательный объект, дидактическая структура.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Силкина Н.С., Соколинский Л.Б. Обзор адаптивных моделей электронного обучения // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2016. Т. 5, № 4. С. 61–76. DOI: 10.14529/cmse160405.

Введение

Данная статья является продолжением обзора моделей и стандартов электронного обучения, представленного авторами в работе [1]. В настоящем обзоре мы сконцентрировали внимание на высокоуровневых моделях отечественных и зарубежных ученых, в той или иной мере отражающих концепцию адаптивного электронного обучения. *Адаптивное электронное обучение* (*Adaptive e-Learning*) [2, 3] является сегодня активно развивающимся направлением в сфере образования, под которым понимают совокупность психологических, дидактических и педагогических методов, учитывающих поведение и состояние человека в процессе обучения, опирающуюся на методы инженерии знаний. Основы адаптивного электронного обучения были заложены в середине XX века. В 1960 году Н. Краудером был предложен алгоритм *разветвленного программированного обучения* [4]. Основным отличием данного подхода является введение индивидуальных путей прохождения учебного материала. Путь для каждого учащегося определяет сама программа в процессе обучения, основываясь на ответах учащихся. Следующий этап развития адаптивного электронного обучения связан с использованием гипертекстовых и мультимедийных технологий [5–7], которые, по существу, стали базовой технологической платформой образовательного контента. Это, во-первых, дало возможность использовать в качестве образовательных объектов не только текстовую информацию, но также графику, аудио и видеоинформацию. Во-вторых, дало толчок для развития большого количества различных подходов к разветвленному программированному обучению.

На сегодняшний день существует большое количество стандартов и моделей электронного обучения. Основным стандартом в области электронного обучения является стандарт SCORM [1], объединяющий в себе целый ряд моделей, содержащих требования к программной структуре среды электронного обучения, спецификации прикладных программных интерфейсов и описания структур данных. Стандарт SCORM позволяет обеспечить совместимость различных LMS и возможность переноса и многократного использования образовательных объектов. Наряду с этим существуют высокоуровневые модели электронного обучения, которые описывают структуру образовательного контента и правила предоставления образовательных объектов учащемуся. Среди них выделяется ряд моделей, в которых наибольшее внимание уделяется вопросам кастомизации обучения (адаптивные модели). В настоящей статье дается обзор четырех наиболее интересных с нашей точки зрения моделей этого класса.

Статья организована следующим образом. В разделе 1 приводится описание модели KFS (Knowledge Flow Structure), основанной на понятии потока знаний. Раздел 2 посвящен рассмотрению модели DCM (Dynamic content model), использующей для организации и представления знаний карту понятий. В разделе 3 мы обсуждаем модель, предложенную А.В. Сололовым, основными понятиями которой являются учебный элемент, граф содержания и спецификация учебных элементов. В разделе 4 рассматривается модель CDCGM (Competency-driven content generation model), предполагающая, что весь учебный материал, доступный разработчику электронного учебного курса, хранится в виде образовательных объектов, связанных с банком компетенций. В заключении суммируются выводы, сделанные нами по результатам обзора, и намечаются черты новой модели электронного обучения, предусматривающей средства для описания дидактической структуры образовательных объектов.

1. Модель KFS

Модель KFS (Knowledge Flow Structure) [8, 9] основана на понятии потока знаний. В KFS-модели базовым элементов является учебный ℓ-блок, при изучении которого у студента должно сформироваться некое целевое знание B. Для изучения ℓ-блока могут понадобиться входные знания A_i (см. рис. 1). Это могут быть результаты изучения других ℓ-блоков этого курса, знания из других курсов или вообще из другой области (*базовые знания курса*).

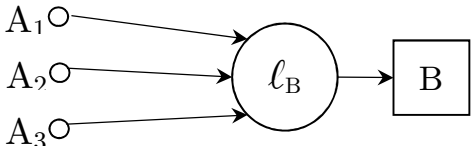


Рис. 1. Учебный ℓ-блок учебного курса KFS-модели

Между ℓ-блоками могут существовать связи по знаниям, которые указывают, что информация, изложенная в одном ℓ-блоке курса, используется в другом ℓ-блоке. Такие связи называются *потоком знаний* (*Knowledge Flow*) в электронном учебном курсе. Содержательная интерпретация блоков учебного материала полностью определяется разработчиком курса, явно указывающим из каких блоков он состоит.

Схема курса представляет собой ориентированный граф, узлами которого являются учебные ℓ-блоки, а дугами связи по передаче знаний из блока в блок. Очевидно, что в курс могут вести несколько входов, а результат последнего ℓ-блока является и результатом всего курса. Полученный граф должен быть связным и не иметь циклов. То есть прямо или косвенно весь учебный материал связан и работает на общий результат. Возможные пути от начального до конечного блока представляют возможные варианты изучения курса, называемыми *блок-схемой изучения курса.*

Каждый блок может завершаться проверкой выходного знания. Более того, разработчик может устанавливать проверку знаний и при входе в блок. Для оценивания знаний могут использоваться разные методики, одна из которых приведена в [10]. Таким образом учебный курс наделяется точками контроля, в которых происходит ветвление обучения. Вводятся два типа условий ветвления блок-схемы изучения курса. *Условия выходного контроля* R — условия, которые реализуют процесс повторного обучения (restudy). *Условия входного контроля* U — условия, которые не позволяют ученику изучать новый ℓ-блок, если он не обладает необходимыми знаниями.

Проектирование курса начинается с одного блока, для которого необходимо прописать его входы и выход — знания и умения, вырабатываемые в этом курсе. Далее проводится несколько этапов детализации. Каждый этап детализации порождает новый слой в представлении учебного материала. На любом этапе граф курса может быть автоматически разложен в ярусно-параллельную, на основе которой могут задаваться возможные пути изучения курса. Пример детализации блок-схемы курса представлен на рис. 2.

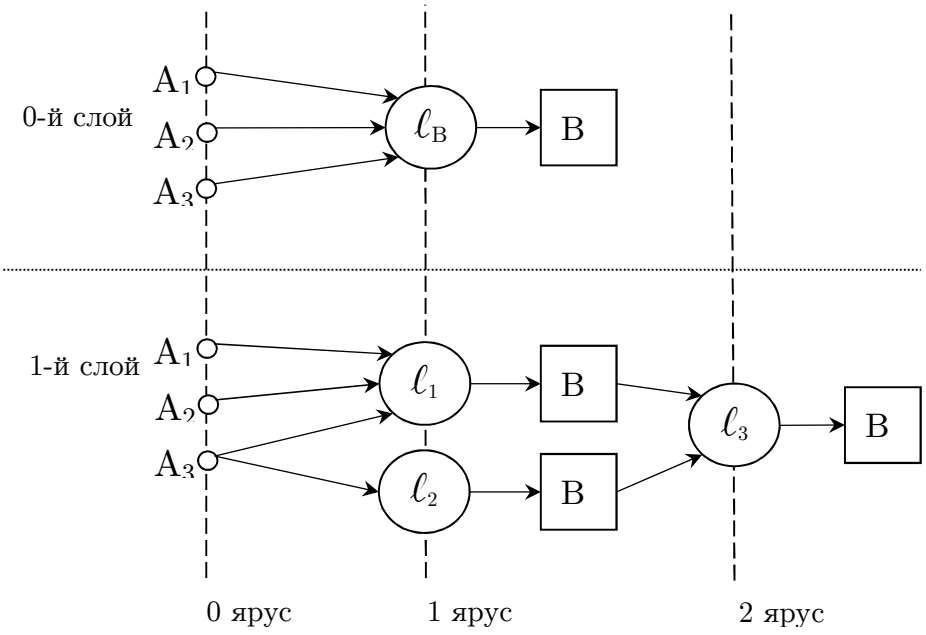


Рис. 2. Пример детализации блок-схемы курса в KFS-модели

KFS-модель позволяет разрабатывать учебный курс любого уровня сложности, делая это поэтапно, постепенно усложняя курс. Использование KFS-модели позволяет ре-

шать задачи формирования и изучения учебного материала, а также задачи мониторинга и анализа учебного процесса в целом и отдельных студентов в частности.

При такой модели курса студент может самостоятельно изучать курс, переходя от раздела к разделу, в соответствии с путями, проложенными преподавателем. Студент всегда знает, где он находится, как он туда пришел, может просмотреть свои результаты и выбрать дальнейший путь. При этом в курсе преподаватель может предусмотреть повторное изучение тех или иных разделов, как в случае неудовлетворительной оценки с точки зрения преподавателя, так и в случае, если студент сам недоволен своим результатом. Поскольку в модель заложено время изучения, этот процесс может контролироваться автоматически.

Для решения задачи мониторинга и анализа достаточно иметь экземпляр структуры курса для каждого студента, на котором фиксируется процесс его изучения. Обрабатывая соответствующим образом экземпляры курса каждого студента, всегда можно провести исчерпывающий анализ изучения курса в разных аспектах.

Электронный учебный курс в KFS-модели имеет достаточно сложную структуру ввиду наличия большого количества узлов, связей между ними, а также условий входа и выхода, что существенно ограничивает возможность переноса части учебного материала из одного курса в другой.

2. Модель DCM

Модель DCM (Dynamic Content Model) [11–13] основана на широко используемом инструменте для организации и представления знаний — карте понятий (Concept Map) [14]. *Карта понятий* представляет собой циклический граф, узлами которого являются понятия, а дугами — связи между понятиями. Каждый узел графа помечается словом или словосочетанием. Пример карты понятий приведен на рис. 3. Карта понятий является удобным инструментом для представления структуры знаний, однако она не позволяет отражать процесс обучения в динамике. Модель DCM восполняет этот недостаток.

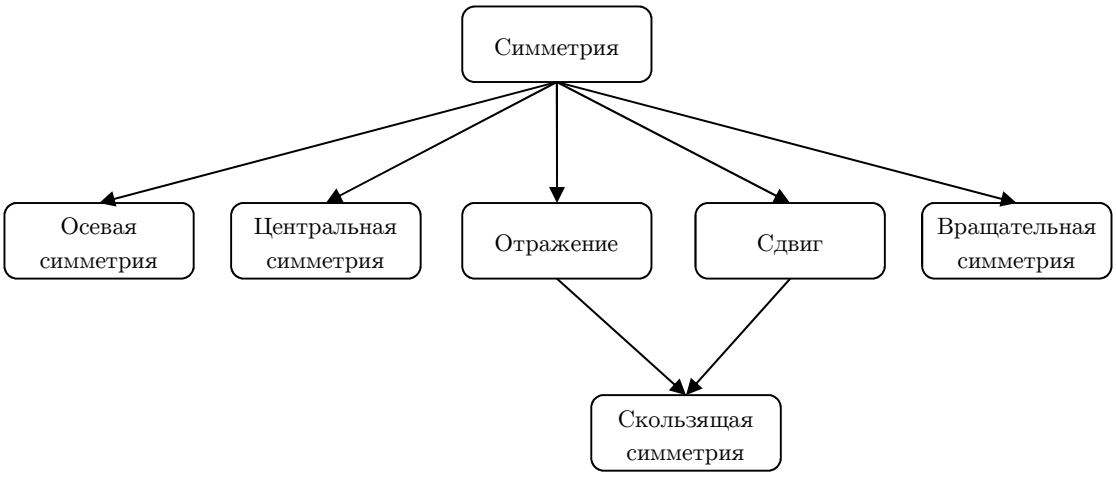


Рис. 3. Карта понятий «Симметрия в геометрии»

В модели DCM знания представляются в виде *учебных модулей* (*Content Units*), которые в свою очередь состоят из компонент двух типов: *образовательные ресурсы* (*Learning Resources*) и *контрольные мероприятия* (*Evaluations*). В качестве образовательных ресурсов могут фигурировать учебники, интернет-ресурсы, ссылки на статьи в Википедии и др. В качестве контрольных мероприятий могут использоваться тесты, задания и др. Идеологически учебные модули соответствуют отдельным понятиям в карте понятий. Из учебных модулей формируются три типа карт: карта знаний, карта обучения и карта студента. Все указанные карты имеют структуру ориентированного графа.

Карта знаний (*knowledge map*) содержит все доступные учебные модули из некоторой предметной области. Дуги в карте знаний отображают зависимости с семантикой «должен знать», определяющие порядок изучения учебного материала. На рис. 4 показан упрощенный пример карты знаний из области геометрии. Метками T_i обозначаются темы, соответствующие тому или иному понятию, R_i обозначают образовательные ресурсы, E_i — контрольные мероприятия.

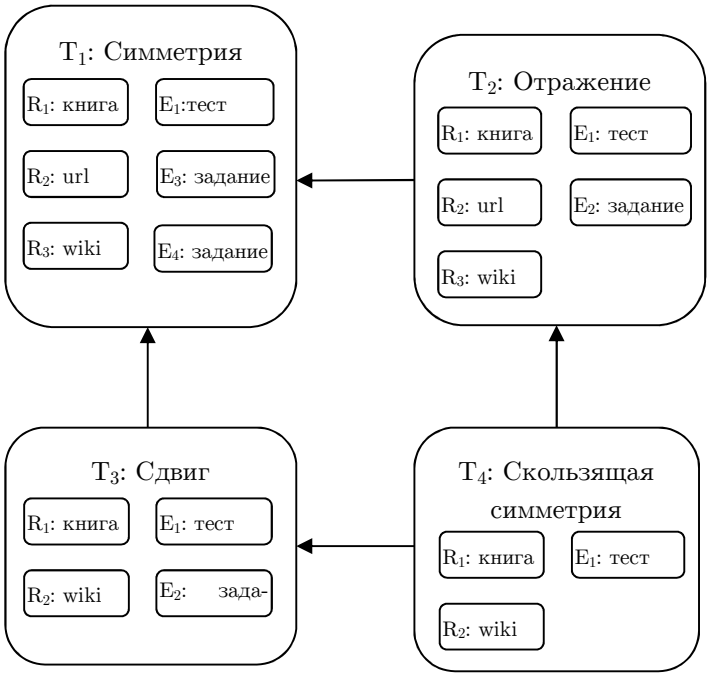


Рис. 4. Карта знаний

Карта обучения (*Learning Map*) строится на основе карты знаний и служит основой для формирования конкретного электронного учебного курса. При трансформации карты знаний в карту обучения преподаватель может выполнять следующие действия:

- удалять отдельные компоненты учебных модулей или целые модули;
- вводить зависимости «должен знать» как между отдельными компонентами внутри учебного модуля, так и между компонентами различных учебных модулей.

На рис. 5 представлен пример карты обучения полученной на основе карты знаний, изображенной на рис. 4. В данном примере в карте обучения опущен учебный модуль T4, в модуле T3 удалены контрольное мероприятие E2 и образовательный ресурс R1, добавлена связь между модулями T3 и T2, а также введены зависимости между компонентами модуля T1. В модели DCM предполагается, что, если внутри учебного модуля

между образовательными ресурсами не установлено явных зависимостей «должен знать», то их можно изучать в любом порядке. Если же отсутствуют явные зависимости между контрольным мероприятием и образовательными ресурсами модуля, то это означает, что для выполнения соответствующего контрольного мероприятия необходимо изучить все имеющиеся в модуле образовательные ресурсы.

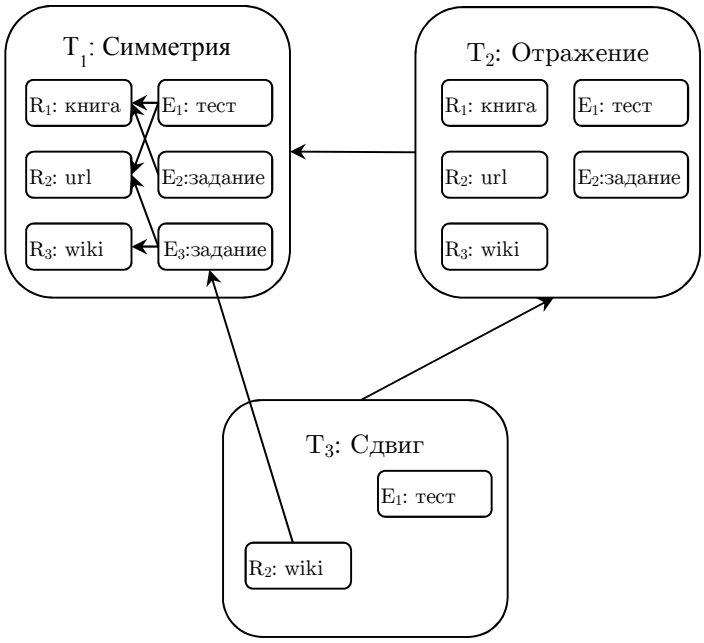


Рис. 5. Карта обучения

Карта студента (*Student Map*) служит для мониторинга процесса обучения конкретного студента, предоставляя механизмы для оценки и обратной связи с обучаемым. Карта студента содержит те же самые элементы, что и карта обучения, однако, вместо связей «должен знать» в ней задаются связи между компонентами всех учебных модулей, определяющие порядок освоения учебного материала, не противоречащий связям «должен знать» в соответствующей карте обучения. При этом должен получиться связный ациклический граф. Отметим, что контрольные мероприятия в карте студента содержат не вопросы и задания, а фактические ответы студента. На рис. 6 представлен пример карты студента, полученной на основе карты обучения, изображенной на рис. 5.

Карта знаний содержит весь существующий учебный материал по определенной предметной области, при формировании карты обучения преподаватель должен выбрать из карты знаний только те учебные модули, которые предусмотрены программой курса. При этом он должен обеспечить выполнение следующих двух условий: при удалении избыточного учебного модуля должны быть удалены все влияющие на него модули, за исключением тех, которые предусмотрены программой курса и всех на них влияющих; должны быть удалены все учебные модули, изученные в предыдущих курсах. Сетевая структура знаний, лежащая в основе модели DCM, делает этот процесс сложным для преподавателя, так как при удалении модулей может быть нарушена связность графа, представляющего карту обучения.

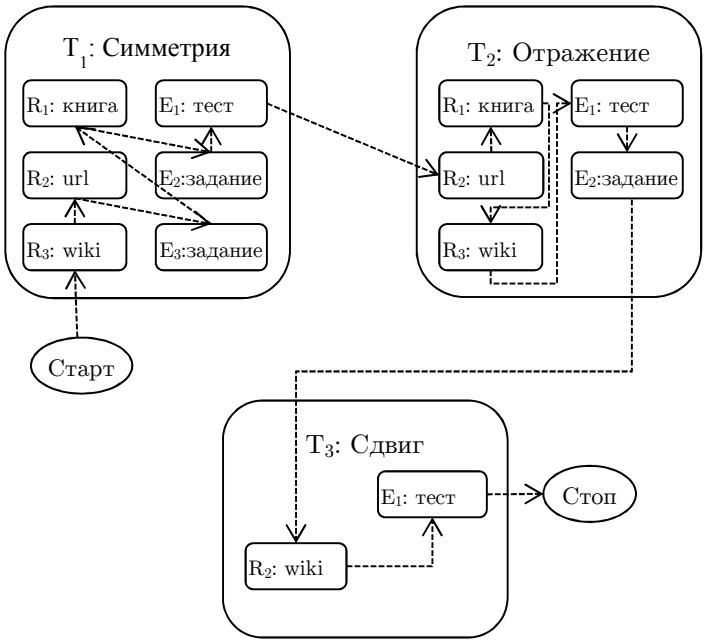


Рис. 6. Карта студента

3. Модель Соловова

Основными понятиями модели Соловова [15, 16] являются учебный элемент, граф содержания и спецификация учебных элементов. При проектировании курса весь учебный материал разбивают на отдельные блоки, называемые *учебные элементы* (*УЭ*). Под УЭ понимают объекты, явления, понятия, методы деятельности, отобранные из соответствующей науки и внесенные в программу учебной дисциплины.

Совокупность УЭ представляют в виде ориентированного дерева, называемого *графом содержания* (*ГС*). Узлами графа являются УЭ, ребрами — связи между ними. УЭ, расположенный на уровень ниже некоторого УЭ, не является простой декомпозицией содержания вышестоящего УЭ (содержание нижестоящих УЭ может детализировать, раскрывать отдельные компоненты содержания связанного с ними вышестоящего УЭ). И, наоборот, содержание вышестоящего УЭ, хотя и интегрирует содержание связанных с ним нижестоящих УЭ, но не является их простым объединением. Математической моделью ГС является его матрица смежности. Пример ГС и соответствующей матрицы смежности изображен на рис. 7.

После структурирования и отбора содержания учебного материала для каждого УЭ формулируют требования по уровню усвоения α, уровню представления β и уровню осознанности γ. Полученные требования представляют в виде таблицы и называют *спецификацией УЭ*. На рис. 8 представлен пример спецификации УЭ, соответствующей ГС, изображенному на рис. 7. Для каждого показателя спецификации УЭ иногда заполняют две колонки таблицы, в первой из которых указывают «стартовый» показатель (уровень до обучения), во второй — «финишный» (требуемый уровень после обучения).

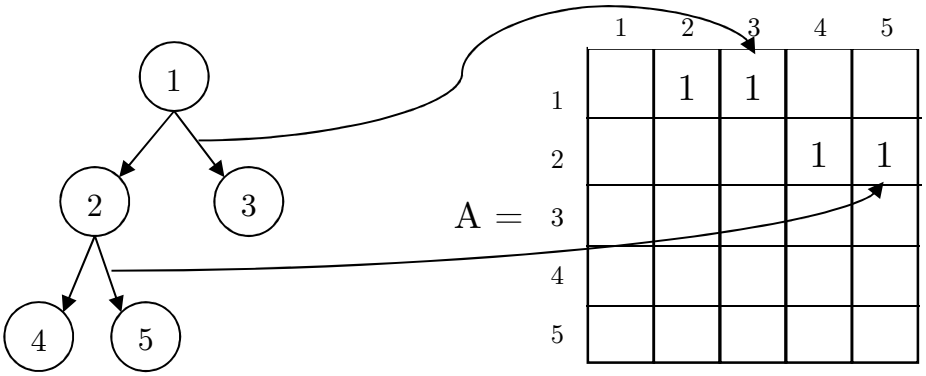


Рис. 7. ГС и его матрица смежности

Совокупность ГС и спецификации УЭ называется моделью содержания учебного материала.

№	Наименование УЭ	α	β	γ
1.	Орграфы	2	3	2
2.	Орграфы и матрицы	2	3	2
3.	Связность	1	3	2
4.	Матрица смежности	2	3	2
5.	Матрица расстояний	1	3	2

Рис. 8. Спецификация УЭ

На основе теории ориентированных графов автор вывел ряд свойств ГС и интегральных характеристик модели содержания учебного материала. Например, число учебных элементов ГС, число уровней структуризации, вектор структуризации, степень разветвленности модели содержания учебного материала, средний уровень представления, средний уровень усвоения и средний уровень осознанности учебного материала. Используя интегральные характеристики можно анализировать и сравнивать различные учебные материалы между собой, оценивать трудоемкость подготовки курса.

После построения модели содержания учебного материала определяется последовательность изучения УЭ, для чего устанавливаются логические связи между УЭ курса (по принципу «изучается прежде») с помощью матриц отношений очередности. Количество строк и столбцов матрицы отношений очередности равно количеству УЭ курса. При заполнении ячеек матрицы отношений очередности анализируют отношение очередности между двумя учебными элементами. Единицу ставят в ячейку, если учебный элемент, указанный в номере строки, должен изучаться после учебного элемента, указанного в номере столбца. Противоположное отношение очередности обозначают нулем или оставляют соответствующую ячейку матрицы пустой. Все ячейки главной диагонали матрицы отношений очередности заполняют единицами. Последовательность изучения учебных элементов при обучении определяется в процессе обработки матрицы отношений очередности, суммируя коэффициенты каждой строки матрицы. Чем больше сумма, тем позже должен изучаться соответствующий учебный элемент. На рис. 9 пред-

ставлен пример матрицы отношений очередности (и соответствующий орграф), соответствующей ГС, изображенному на рис. 7.

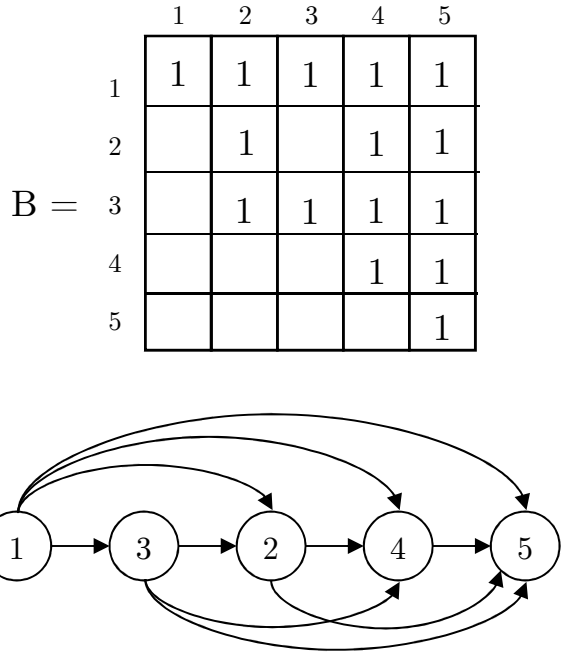


Рис. 9. Матрица отношений очередности и соответствующий орграф

4. Модель CDCGM

Модель CDCGM (Competency-Driven Content Generation Model) [17, 18] предполагает, что весь учебный материал, доступный разработчику электронного учебного курса, хранится в виде образовательных объектов LO (Learning Objects), связанных с банком компетенций. Образовательный объект может включать в себя учебный материал различного рода и предназначен для описания некоторого понятия. Компетенция —это спецификация знаний, умений и навыков (ЗУН) [19], которые должен приобрести обучаемый в результате изучения образовательных объектов. В модели CDCGM различаются входные и выходные компетенции. Входные компетенции хранятся в профиле обучаемого и представляют собой ЗУНы, которыми обучаемый владеет до начала изучения электронного учебного курса. Выходные компетенции представляют собой цели обучения и включают в себя ЗУНы, которые обучаемый приобретет в результате изучения данного курса. Модель CDCGM позволяет по заданным входному и выходному наборам компетенций осуществить автоматический отбор образовательных объектов и организовать их в виде древовидной структуры, адаптированной под конкретного обучаемого. На рис. 10 изображена общая концепция модели CDCGM.

Отбор образовательных объектов осуществляется на основе анализа профиля обучаемого, целей обучения и метаданных LO. Компетенции в модели CDCGM описываются согласно требованиям RCD-модели (Reusable Competency Definition), описанной в работе [20]. С каждым образовательным объектом связываются компетенции, определяющие предусловия и результаты изучения LO. Таким образом, компетенция, которая является результатом изучения LO, может быть предусловием изучения другого LO. За каждым

LO закрепляется блок метаданных LO (Metadata) или кратко MD, содержащий ссылки на компетенции (см. рис. 11).

Метаданные LO описываются согласно требованиям LOM-модели (Learning Object Metadata), описанной в работах [21, 22]. Для описания ссылок на компетенции используется элемент «Classification» и его атрибуты «Purpose» и «Taxon Path» для определения предусловия и результатов изучения соответственно. С каждым LO может быть связано несколько описаний «Classification» для определения более одного предусловия и более одного результата изучения.

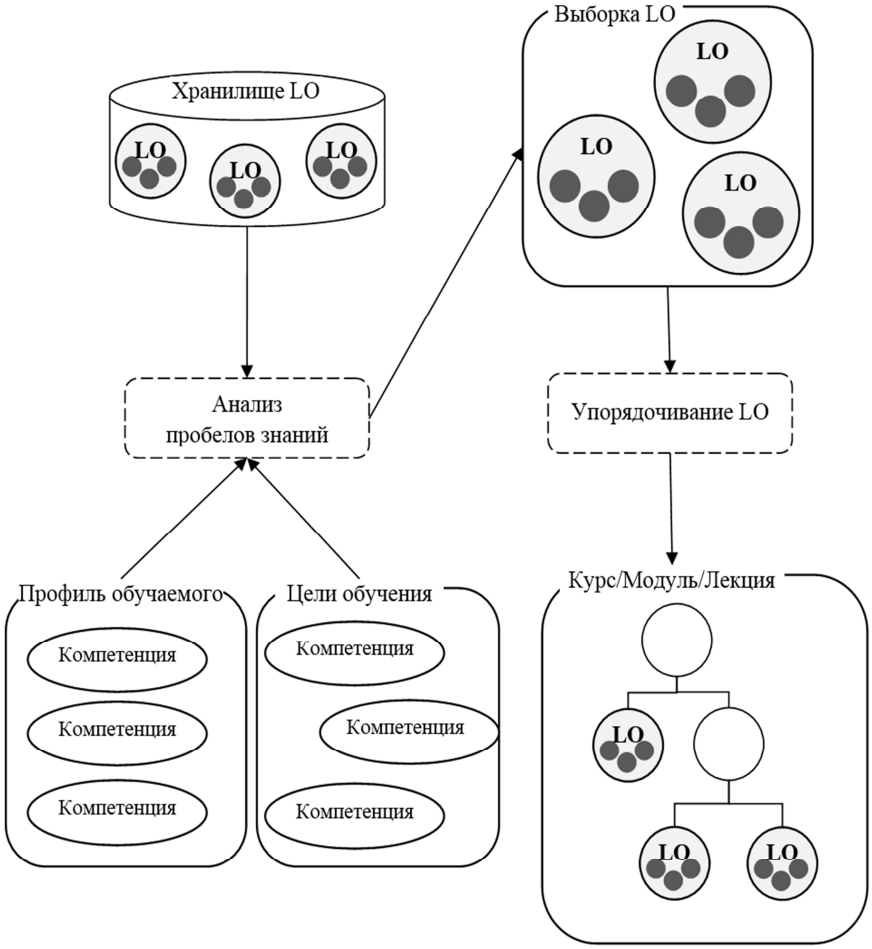


Рис. 10. Концепция модели CDCGM

Профиль обучаемого и цели обучения являются входными параметрами для процесса анализа пробелов в знаниях. На этом этапе выполняется поиск в хранилище образовательных объектов, которые заполняют пробелы между текущим профилем обучаемого и целями обучения. Таким образом, результатом работы данного этапа является неупорядоченное множество образовательных объектов, заполняющих пробелы между текущими знаниями обучаемого и ожидаемыми результатами обучения.

На следующем этапе полученное множество упорядочивается таким образом, чтобы базовые LO были представлены учащемуся в первую очередь. Проблему упорядочивания LO авторы рассматривают с точки зрения классической задачи удовлетворения ограничений (constraint satisfaction problem) [23, 24]. В данном случае пространство решений включает в себя все возможные способы упорядочивания LO и имеет размер $n!$, где n — количество LO, отобранных на предыдущем этапе. Нас интересуют только те

упорядочивания, которые удовлетворяют всем ограничениям, накладываемым метаданными LO, содержащими указатели на связанные компетенции. Для нахождения такого решения авторы используют метод роя частиц (Particle Swarm Optimization, PSO) [25, 26].

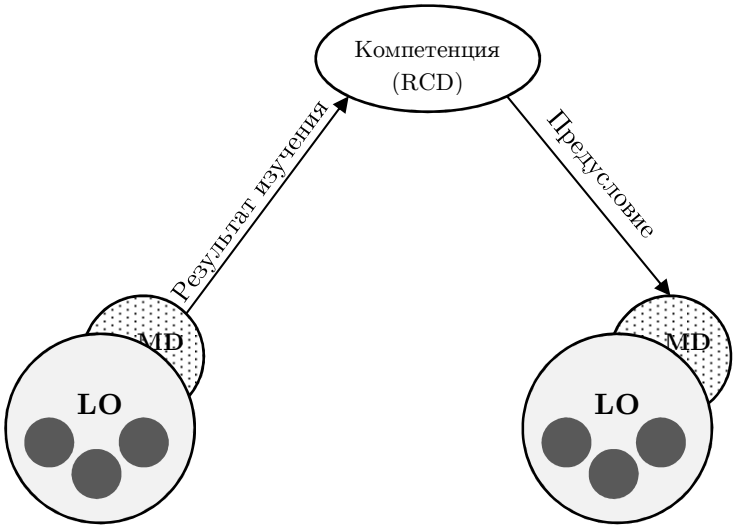


Рис. 11. Связь модулей через компетенцию

Таким образом, CDCGM-модель позволяет автоматически персонифицировать электронный учебный курс, в зависимости от компетенций учащегося, которыми он владеет. В качестве недостатка данного подхода можно указать то, что задача упорядочения контента в некоторых случаях может не иметь решения. Другой проблемой является то, что при увеличении количества образовательных объектов значительно возрастает количество возможных перестановок, что приводит к существенному усложнению поиска решения.

Заключение

На сегодняшний день разработано достаточно большое количество высокоуровневых моделей электронного обучения. Среди них выделяется ряд моделей, в которых наибольшее внимание уделяется вопросам кастомизации обучения. К этому классу относятся модели, представленные в настоящем обзоре. Все рассмотренные модели обладают одним общим недостатком: они не предусматривают деление образовательного объекта на дидактические единицы предопределенных типов. Это существенным образом ограничивает возможность автоматической проверки электронного учебного курса на дидактическую полноту. Кроме этого, отсутствие предопределенных типов дидактических единиц препятствует автоматическому выделению и переносу в другой курс части учебного материала, относящиеся к одному дидактическому типу. В работах [27, 28] авторами настоящего обзора предложен прототип новой модели электронного обучения, в которой каждый образовательный объект должен включать в себя шесть дидактических компонент следующих типов: теоретическое описание понятия; пример, иллюстрирующий те или иные отличительные черты понятия; упражнение для самостоятельного выполнения; тестовое задание; слайд презентации; библиографическая ссылка. Каждый

компонент обладает своими дидактическими возможностями. Модель допускает возможность расширения набора стандартных дидактических способов представления учебного материала с учетом специфики каждой конкретной специальности или направления подготовки. Разработка и внедрение в практику электронного обучения подобных моделей является актуальной задачей, решение которой должно привести к созданию соответствующего стандарта.

Литература

1. Силкина Н.С., Соколинский Л.Б. Модели и стандарты электронного обучения // Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика. 2014. Т. 3, № 4. С. 5–35.
2. Розенберг И.Н. Обучение по гибкой траектории // Современное дополнительное профессиональное педагогическое образование. 2015. № 1 (1). С. 64–72.
3. Shute V., Towle B. Adaptive E-Learning // Educational psychologist. 2003. № 38(2). P. 105–114.
4. Краудер Н.А. О различиях между линейным и разветвленным программированием // Программированное обучение за рубежом: Сб. статей / Под ред. И.И. Тихонова. М.: Высшая школа. 1968. С. 58–67.
5. Брусиловский П.Л. Адаптивные интеллектуальные технологии в сетевом обучении // Новости искусственного интеллекта. 2002. № 5. С. 25–31.
6. Брусиловский П.Л. Адаптивные обучающие системы в World Wide Web: обзор имеющихся в распоряжении технологий // International Forum of Educational Technology & Society. URL: http://ifets.ieee.org/russian/depository/WWWITS.html (дата обращения: 13.09.2016).
7. Engelbart D.C. Toward Augmenting the Human Intellect and Boosting our Collective IQ // Communications of the ACM. Vol. 38, No. 8. 1995. P. 30–33.
8. Курганская Г.С. Модель представления знаний и система дифференцированного обучения через Интернет на его основе // Известия Челябинского Научного Центра. 2000. Вып. 2. С. 84–88.
9. Курганская Г.С. Облачные технологии интернет-образования на основе KFS модели представления знаний // Вестник Бурятского государственного университета. 2013. № 9. С. 69–75.
10. Курганская Г.С. Математическое и программное обеспечение системы сопоставительной оценки // Доклады Всесоюзного семинара. Томск. 1990. С. 76–83.
11. Eide S., Kristensen T., Lamo Y. A Model for Dynamic Content Based E-learning systems // Proceedings of the 2008 Euro American Conference on Telematics and Information Systems (EATIS '08). ACM, New York, NY, USA, 2008. Article 2. 8 p.
12. Kristensen T., Lamo Y., Mughal K., Tekle K. M., Bottu A. K. Towards a Dynamic, Content Based E-Learning Platform // Proceedings of the 10th IASTED International Conference on Computers and Advanced Technology in Education (CATE '07). ACTA Press, Anaheim, CA, USA, 2007. P. 107–114.
13. Kristensen T., Lamo Y., Hinna K.R., Hole G.O. Dynamic Content Manager — A New Conceptual Model for E-Learning // Proceedings of the International Conference on Web Information Systems and Mining (WISM '09). Springer-Verlag, Berlin, Heidelberg, 2009. P. 499–507.

14. Novak J.D., Canas A.J. The Theory Underlying Concept Maps and How to Construct Them. Technical Report IHMC CmapTools 2006-01 Rev 2008-01. Florida Institute for Human and Machine Cognition, USA. 2008. URL: http://cmap.ihmc.us/docs/theory-of-concept-maps (дата обращения: 10.07.2016).
15. Соловов А.В. Математические модели содержания и процессов электронного обучения // Телекоммуникации и информатизация образования, 2006. № 4. С. 20–37.
16. Соловов А.В. Математическое моделирование содержания, навигации и процессов электронного обучения в контексте международных стандартов и спецификаций. Лекция-доклад // Труды Всероссийской научно-практической конференции с международным участием «Информационные технологии в обеспечении нового качества высшего образования (14–15 апреля 2010 г., Москва, НИТУ МИСиС)». М.: Исследовательский центр проблем качества подготовки специалистов, 2010. 52 с.
17. De-Marcos L., Martinez J.J., Gutierrez J.A. Swarm Intelligence in e-Learning: A Learning Object Sequencing Agent based on Competencies // Proceedings of the 10th annual conference on Genetic and evolutionary computation GECCO'08 (July 12–16, 2008, Atlanta, Georgia, USA). 2008. P. 17–24.
18. De-Marcos L., Pages C., Martinez J.J., Gutierrez J.A. Competency-Based Learning Object Sequencing Using Particle Swarms // Proceedings of 19th IEEE International Conference on Tools with Artificial Intelligence ICTAI 2007 (29–31 October 2007). Vol. 2. P. 111–116.
19. Wilkinson J. A Matter of Life or Death: Re-engineering Competency-based Education through the Use of a Multimedia CD-ROM // Proceedings of IEEE International Conference on Advanced Learning Technologies, 2001. P. 205–208.
20. IEEE 1484.20.1. Standard for Learning Technology – Data Model for Reusable Competency Definitions. 2007.
21. IEEE 1484.12.1. Draft Standard for Learning Object Metadata. 2002. URL: http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf (дата обращения: 20.03.2016).
22. IEEE 1484.12.3. Standard for Learning Technology — Extensible Markup Language (XML) Schema Definition Language Binding for Learning Object Metadata. 2005.
23. Schoofs L., Naudts B. Ant Colonies are Good at Solving Constraint Satisfaction Problems // Proceedings of the 2000 Congress on Evolutionary Computation (16–19 July 2000, La Jolla, CA). Vol. 2. P. 1190–1195.
24. Tsang, E. Foundations of Constraint Satisfaction. Academic Press Limited, 1993. 421 p.
25. Hinchey M.G., Sterritt R., Rouff C. Swarms and Swarm Intelligence // Computer. 2007. Vol. 40 (4). P. 111–113.
26. Kennedy J., Eberhart R. Particle swarm optimization // Proceedings of IEEE International Conference on Neural Networks (Perth, WA, Australia, 1995). Vol. 1944. P. 1942–1948
27. Силкина Н.С., Соколинский Л.Б. Система UniCST — универсальная среда электронного обучения // Системы управления и информационные технологии. 2010. № 2. С. 81–86.
28. Жигальская (Силкина) Н.С. Моделирование дидактической структуры электронных учебных комплексов // Вестник Южно-Уральского государственного университета.

Серия: Математическое моделирование и программирование. 2008. № 27(127). Вып. 2. С. 4–9.

DOI: 10.14529/cmse160405

SURVEY OF ADAPTIVE E-LEARNING MODELS

© 2016 N.S Silkina, L.B. Sokolinsky

South Ural State University (pr. Lenina 76, Chelyabinsk, 454080 Russia),
E-mail: silkinans@susu.ru, Leonid.Sokolinsky@susu.ru

Received: 07.10.2016

The article provides an overview of adaptive e-learning models. For each model, the structure and the representation methods of educational content are described. An analysis of the strong and week features of the presented e-learning models is discussed. The main lack is the absence of predefined didactic structure of the learning objects. This essentially restricts automatic checking the didactic completeness of the e-learning course. In the conclusion of the paper, we presented an outline of a new e-learning model, which includes the facilities of describing the didactic structure of learning objects.

Keywords: e-learning, e-learning course, the model of e-learning, learning objects, didactic structure.

FOR CITATION

Silkina N.S., Sokolinsky L.B. Survey of Adaptive E-learning Models. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. 2016. vol. 5, no. 4. pp. 61–76. (in Russian) DOI: 10.14529/cmse160405.

References

1. Silkina N.S., Sokolinsky L.B. E-learning Models and Standards. *Vestnik Yuzhno-Uralskogo gosudarstvennogo universiteta. Serija: Vychislitelnaja matematika i informatika* [Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering]. 2014. vol. 3, no. 4. pp. 5–35. DOI: 10.14529/cmse140401. (in Russian)
2. Rosenberg I.N. Training for Flexible Trajectory. *Sovremennoe dopolnitelnoe professionalnoe pedagogicheskoe obrazovanie* [Modern Additional Pedagogical Education]. 2015. no. 1 (1). pp. 64–72. (in Russian)
3. Shute V., Towle B. Adaptive E-Learning. *Educational psychologist.* 2003. no. 38(2). pp. 105–114. DOI: 10.1207/S15326985EP3802_5.
4. Krauder N.A. About the Differences between Linear and Branched Programming. *Programmirovannoe obuchenie za rubexhom: Sbornik statej* [Programmed Training Abroad: Digest of Articles] / Edited by I.I. Tihonov. M.: Vysshaya shkola. 1968. pp. 58–67. (in Russian)
5. Brusilovsky P.L. Adaptive Intellect Technologies in Network Training. *Novosti iskusstvennogo intellekta* [News of Artificial Intelligence]. 2002. no. 5. pp. 25–31. (in Russian)
6. Brusilovsky P.L. Adaptive Learning Systems in the World Wide Web: an Overview of Available Technologies. *International Forum of Educational Technology & Society.* Available at: http://ifets.ieee.org/russian/depository/WWWITS.html (accessed: 13.09.2016)

7. Engelbart D.C. Toward Augmenting the Human Intellect and Boosting our Collective IQ. *Communications of the ACM.* vol. 38, no. 8. 1995. pp. 30–33. DOI: 10.1145/208344.208352.
8. Kurganskaya G.S. The Model of Knowledge Representation and the System of Differentiated Instruction Through the Internet on the Basis Thereof. *Izvestiya Chelyabinskogo Nauchnogo Centra* [Proceedings of the Chelyabinsk Scientific Center]. 2000. vol. 2. pp. 84–88. (in Russian)
9. Kurganskaya G.S. Cloud Technologies of the Internet Education Based on KFS Knowledge Representation Model. *Vestnik burjatskogo gosudarstvennogo universiteta* [Bulletin of the Buryat State University]. 2013. no. 9. pp. 69–75. (in Russian)
10. Kurganskaya G.S. Mathematical and Software of a Comparative Evaluation System. *Doklady Vsesoyuznogo seminara* [Reports of the All-Union Seminar]. Tomsk. 1990. pp. 76–83. (in Russian)
11. Eide S., Kristensen T., Lamo Y. A Model for Dynamic Content Based E-learning Systems. *Proceedings of the 2008 Euro American Conference on Telematics and Information Systems (EATIS '08).* ACM, New York, NY, USA, 2008. Article 2. 8 p. DOI: 10.1145/1621087.1621089.
12. Kristensen T., Lamo Y., Mughal K., Tekle K. M., Bottu A. K. Towards a Dynamic, Content Based E-Learning Platform. *Proceedings of the 10th IASTED International Conference on Computers and Advanced Technology in Education (CATE '07).* ACTA Press, Anaheim, CA, USA, 2007. pp. 107–114.
13. Kristensen T., Lamo Y., Hinna K.R., Hole G.O. Dynamic Content Manager — A New Conceptual Model for E-Learning. *Proceedings of the International Conference on Web Information Systems and Mining (WISM '09).* Springer-Verlag, Berlin, Heidelberg, 2009. pp. 499–507. DOI: 10.1007/978-3-642-05250-7_52.
14. Novak J.D., Canas A.J. *The Theory Underlying Concept Maps and How to Construct Them. Technical Report IHMC CmapTools 2006-01 Rev 2008-01.* Florida Institute for Human and Machine Cognition, USA. 2008. Available at: http://cmap.ihmc.us/docs/theory-of-concept-maps (accessed: 10.07.2016)
15. Solovov A.V. Mathematical Models of Content and Processes of E-learning. *Telekommunikacii i informatizaciya obrazovaniya* [Telecommunications and Informatization of Education]. 2006. no. 4. pp. 20–37. (in Russian)
16. Solovov A.V. Mathematical Modeling of Content, Navigation, and Processes Electronic-learning in the Context of International Standards and Specifications. Lecture-report. *Informacionnye tehnologii v obespechenii novogo kachestva vysshego obrazovaniya: Trudy Vserossijskoj nauchno-prakticheskoj konferencii s mezhdunarodnym uchastiem (Moskva, 14–15 aprelya 2010)* [Information Technology to Provide a New Quality of Higher Education: Proceedings of the All-Russian Scientific-Practical Conference with International Participation (Moscow, Russia, April 14–15, 2010)]. M.: Research Center of training quality problems. 2010. 52 p. (in Russian)
17. De-Marcos L., Martinez J.J., Gutierrez J.A. Swarm Intelligence in e-Learning: A Learning Object Sequencing Agent based on Competencies. *Proceedings of the 10th annual conference on Genetic and evolutionary computation GECCO'08 (July 12–16, 2008, Atlanta, Georgia, USA).* 2008. pp. 17–24. DOI: 10.1145/1389095.1389099.

18. De-Marcos L., Pages C., Martinez J.J., Gutierrez J.A. Competency-Based Learning Object Sequencing Using Particle Swarms. *Proceedings of 19th IEEE International Conference on Tools with Artificial Intelligence ICTAI 2007 (29–31 Oct. 2007)*. vol. 2. pp. 111–116. DOI: 10.1109/ICTAI.2007.14.
19. Wilkinson J. A Matter of Life or Death: Re-engineering Competency-based Education through the Use of a Multimedia CD-ROM. *Proceedings of IEEE International Conference on Advanced Learning Technologies*, 2001. pp. 205–208. DOI: 10.1109/ICALT.2001.943901.
20. IEEE 1484.20.1. *Standard for Learning Technology – Data Model for Reusable Competency Definitions*. 2007. DOI: 10.1109/IEEESTD.2008.4445693.
21. IEEE 1484.12.1. *Draft Standard for Learning Object Metadata*. 2002. Available at: http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf (дата обращения: 20.03.2016).
22. IEEE 1484.12.3. *Standard for Learning Technology — Extensible Markup Language (XML) Schema Definition Language Binding for Learning Object Metadata*. 2005. DOI: 10.1109/IEEESTD.2006.215176.
23. Schoofs L., Naudts B. Ant Colonies are Good at Solving Constraint Satisfaction Problems. *Proceedings of the 2000 Congress on Evolutionary Computation (16–19 July 2000, La Jolla, CA)*. vol. 2. pp. 1190–1195. DOI: 10.1109/CEC.2000.870784.
24. Tsang, E. *Foundations of Constraint Satisfaction*. Academic Press Limited, 1993. 421 p. DOI: 10.1016/B978-0-12-701610-8.50021-5.
25. Hinchey M.G., Sterritt R., Rouff C. Swarms and Swarm Intelligence. *Computer*. 2007. vol. 40 (4). pp. 111–113. DOI: 10.1109/MC.2007.144.
26. Kennedy J., Eberhart R. Particle Swarm Optimization. *Proceedings of IEEE International Conference on Neural Networks (Perth, WA, Australia, 1995)*. vol. 1944. pp. 1942–1948.
27. Silkina N.S., Sokolinsky L.B. UniCST System — a Universal E-learning Environment. *Sistemy upravleniya i informacionnye tehnologii* [Control Systems and Information Technology]. 2010. no. 2. pp. 81–86. (in Russian)
28. Zhigalskaya (Silkina) N.S. Modeling of the E-learning Packages Didactic Structure. *Vestnik Yuzhno-Uralskogo gosudarstvennogo universiteta. Serija: Matematicheskoe modelirovaniei programmirovanie* [Bulletin of the South Ural State University. Series: Mathematical Modelling, Programming & Computer Software]. 2008. no. 27(127). vol. 2. pp. 4–9. (in Russian)

УДК 519.63, 524.3 DOI: 10.14529/cmse160406

ЧИСЛЕННОЕ ГИДРОДИНАМИЧЕСКОЕ МОДЕЛИРОВАНИЕ АСТРОФИЗИЧЕСКИХ ТЕЧЕНИЙ НА ГИБРИДНЫХ СУПЕРЭВМ, ОСНАЩЕННЫХ УСКОРИТЕЛЯМИ INTEL XEON PHI*

© 2016 г. И.М. Куликов[1], И.Г. Черных[2], Э.И. Воробьев[3], А.В. Снытников[1], Д.В. Винс[2], А.А. Московский[4], А.Б. Шмелёв[4], В.А. Протасов[5], А.А. Серенко[5], В.Е. Ненашев[5], В.А. Вшивков[1], А.С. Родионов[6], Б.М. Глинский[2], А.В. Тутуков[7]

[1]*Лаборатория параллельных алгоритмов решения больших задач, Институт вычислительной математики и математической геофизики Сибирского отделения РАН (630090 Новосибирск, пр. Академика Лаврентьева, д. 6),*

[2]*Сибирский суперкомпьютерный центр, Институт вычислительной математики и математической геофизики Сибирского отделения РАН (630090 Новосибирск, пр. Академика Лаврентьева, д. 6),*

[3]*Лаборатория космических исследований, Южный федеральный университет (344090 Ростов-на-Дону, пр. Стачки, д. 194),*

[4]*ЗАО «РСК Технологии» (121170 Москва, Кутузовский пр., д. 36, стр. 23),*

[5]*Новосибирский государственный технический университет (630073 Новосибирск, пр. К. Маркса, д. 20),*

[6]*Лаборатория моделирования динамических процессов в информационных сетях, Институт вычислительной математики и математической геофизики Сибирского отделения РАН (630090 Новосибирск, пр. Академика Лаврентьева, д. 6),*

[7]*Отдел физики и эволюции звезд, Институт Астрономии РАН (119017 Москва, ул. Пятницкая, д. 48)*

E-mail: kulikov@ssd.sscc.ru, chernykh@parbz.sscc.ru, eduard.vorobiev@univie.ac.at, snytav@gmail.com, wns.dmitry@gmail.com, moskov@rsc-tech.ru, alexeysh@rsc-tech.ru, inc_13@mail.ru, fafnur@yandex.ru, arni.12@mail.ru, vsh@ssd.sscc.ru, alrod@sscc.ru, gbm@opg.sscc.ru, atutukov@inasan.ru

Поступила в редакцию: 13.04.2016

*Статья рекомендована к публикации программным комитетом Международной научной конференции «Параллельные вычислительные технологии — 2016»

В работе представлены исследования кода AstroPhi для численного моделирования астрофизических течений на гибридных суперЭВМ, оснащенных ускорителями Intel Xeon Phi. Описан со-дизайн вычислительной модели для описания астрофизических объектов. Детально описаны особенности параллельной реализации и исследования производительности кода AstroPhi. Представлены результаты моделирования взаимодействия межгалактического ветра и дисковой галактики. Для кода AstroPhi было достигнуто 134-кратное ускорение в рамках одного ускорителя Intel Xeon Phi, 75-процентная масштабируемость при использовании 224 ускорителей Intel Xeon Phi. На расчетной сетке $7168 \times 1024 \times 1024$ было достигнуто 47 процентов от пиковой скалярной производительности ускорителя Intel Xeon Phi при использовании 53760 нитей.

Ключевые слова: Высокопроизводительные вычисления, вычислительная астрофизика, ускорители Intel Xeon Phi.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Куликов И.М., Черных И.Г., Воробьев Э.И., Снытников А.В., Винс Д.В., Московский А.А., Шмелёв А.Б., Протасов В.А., Серенко А.А., Ненашев В.Е., Вшивков В.А., Родионов А.С., Глинский Б.М., Тутуков А.В. Численное гидродинамическое моделирование астрофизических течений на гибридных суперЭВМ, оснащенных ускорителями Intel Xeon Phi // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2016. Т. 5, № 4. С. 77–97. DOI: 10.14529/cmse160406.

Введение

Математическое моделирование играет ключевую роль в современной астрофизике. Оно является универсальным инструментом для исследования нелинейных эволюционных процессов во Вселенной. Одними из важнейших задач, решаемых вычислительной астрофизикой, являются задачи столкновения [1] и эволюции галактик [2], процессы коллапса звезд [3], химокинетические процессы в галактиках [4]. При конструировании математической модели следует учитывать достижения современной астрономии. Так актуальным является учет магнитного поля в галактиках, так как его наличие обнаружено в рукавах галактики M51 [5] и влияет на процесс звездообразования. Таким образом, изучение астрофизических процессов усложняется необходимостью учета большого числа подсеточных физических процессов. Кроме того, состав астрофизических объектов состоит из нескольких ингридиентов, для описания которых используются различные математические модели. Данное обстоятельство усложняет разработку эффективных кодов для исследования астрофизических проблем на суперкомпьютерах.

Для моделирования сложных астрофизических процессов в высоком разрешении необходимо использовать наиболее мощные суперкомпьютеры. Два из Top-3 (четыре из Top-10) суперкомпьютера в ноябрьской версии 2015 года списка Top-500 оснащены графическими ускорителями и ускорителями Intel Xeon Phi. Ожидается, что первый суперкомпьютер экзафлопсной производительности будет построен на основе гибридного подхода. Разработка кодов для гибридных суперкомпьютеров не сугубо техническая задача, а отдельная сложная научная задача, требующая со-дизайна алгоритмов на всех стадиях решения задачи — от физической постановки до инструментов разработки.

Несмотря на большое число разработанных кодов для решения астрофизических задач [6] остается большое число нерешенных проблем в области математических моделей, численных методов и программных реализаций для изучения астрофизических течений. Авторским коллективом уже на протяжении нескольких лет развивается гибридный эйлерово-лагранжевый подход для решения астрофизических задач. В настоящей статье будет при-

ведено краткое описание и подробное исследование оригинального кода AstroPhi [7] для моделирования динамики астрофизических объектов.

В первом разделе будут описаны основные компоненты со-дизайна вычислительной схемы, второй раздел посвящен описанию новой версии кода AstroPhi, которая была основана на работе 2013 года [8] и представляет собой расширение кода на большее число математических моделей при использовании native режима ускорителя Intel Xeon Phi. В третьем разделе приведены результаты исследования производительности кода, четвертый раздел посвящен вычислительным экспериментам по изучению астрофизических течений на различных масштабах. В заключении приведены основные результаты работы и перспективы дальнейшего развития вычислительной модели.

1. Концепция со-дизайна вычислительной схемы

Главный фокус наших исследований направлен на моделирование динамики галактик. Поэтому численная модель астрофизических течений ориентирована в основном на описание компонент галактик и подсеточных процессов. В работе [9] были исследованы вопросы со-дизайна численных моделей астрофизики и физики плазмы. Расссмотрим основные этапы со-дизайна численных моделей для решения астрофизических проблем.

1. *Этап формулировки физической задачи.* Главными ингридиентами галактик является газовая компонента, которая описывает межзвездный газ и равномерно распределенную пыль, и бесстолкновительная компоненты, которая используется для описания звездной компоненты и темной материи. Основными подсеточными физическими процессами являются процессы звездообразования, эффект от взрыва сверхновых, функции охлаждения и нагревания, а также химические реакции [10].
2. *Этап математической формализации.* Для описания газовой компоненты используются уравнения гравитационной газовой динамики, которые расширяются на уравнения односкоростной многокмопнентной гравитационной газовой динамики с эффективным показателем адиабаты в случае учета химических реакций. Для описания бесстолкновительной компоненты используются уравнения для первых моментов бесстолкновительного уравнения Больцмана. Такой подход был исследован и успешно использован для решения задач эволюции [2, 10] и столкновения галактик [6, 11]. Такой способ описания бесстолкновительной компоненты позволяет позволяет сформулировать термодинамически согласованную модель звездообразования и эффекта от взрыва сверхновых.
3. *Этап построения численного метода решения.* Особенностью математической формализации является описание газовой и бесстолкновительной компонент галактик с помощью системы гиперболических уравнений. Таким образом, мы можем сформулировать единый численный метод решения гиперболических уравнений. В следующем разделе численный метод будет описан более подробно. Использование единого численного метода позволяет записать единый параллельный алгоритм. В основе такого алгоритма лежит локальность вычислений, что достаточно эффективно проецируется на современные архитектуры суперкомпьютеров.
4. *Этап выбора структур данных.* Используемые структуры данных в случае решения гиперболических уравнений полностью согласуются с выбором расчетных сеток. В оригинальном подходе используются регулярные сетки, что позволяет сформулировать простой подход к организации параллельных вычислений [12]. Основным из трендов современного численного решения гиперболических уравнений является технология по-

движных сеток. В случае использования регулярных структур данных может быть описана технология комфорных подвижных сеток, которая позволяет эффективно моделировать большое число задач механики сплошной среды. При этом сохраняются параллельные алгоритмы, используемые для вычислений на регулярных сетках. В настоящее время, такая технология подвижных сеток не реализована в коде AstroPhi, но в перспективе такая реализация планируется.

5. *Этап учета архитектуры суперкомпьютера.* В наших исследованиях используются гибридные суперкомпьютеры с ускорителями Intel Xeon Phi. Логическая архитектуры такого суперкомпьютера представляется в виде линейки ускорителей, взаимодействующих напрямую (в случае использования native режима) или через CPU (в случае использования offload режима). В рамках одного ускорителя вычисления разбиваются на большое (несколько сотен) нитей. Организация вычислений в оригинальном методе позволяет исключить взаимодействие между нитями в рамках одного ускорителя на основных этапах метода, либо сводить такие вычисления к минимуму. Такое взаимодействие возникает в случае вычисления шага по времени из условия Куранта.
6. *Этап использования средств разработки.* Организация вычислений оригинального численного метода и архитектуры используемых суперкомпьютеров позволяют нам ограничиться библиотекой MPI для организации межпроцессных взаимодействий и технологией OpenMP для организации многопоточных вычислений.

В следующем разделе будет подробнее описана реализация каждого этапа.

2. Код AstroPhi

Для описания газовой компоненты будем использовать систему уравнений односкоростной многокомпонентной гравитационной газовой динамики, записанную в эйлеровых координатах:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{u}) = \mathcal{S} - \mathcal{D},$$

$$\frac{\partial \rho_i}{\partial t} + \nabla \cdot (\rho_i \vec{u}) = -s_{ij} + \mathcal{S}\frac{\rho_i}{\rho} - \mathcal{D}\frac{\rho_i}{\rho},$$

$$\frac{\partial \rho \vec{u}}{\partial t} + \nabla \cdot (\rho \vec{u}\vec{u}) = -\nabla p - \rho \nabla(\Phi) + \vec{v}\mathcal{S} - \vec{u}\mathcal{D},$$

$$\frac{\partial \rho E}{\partial t} + \nabla \cdot (\rho E \vec{u}) = -\nabla \cdot (p\vec{v}) - (\rho \nabla(\Phi), \vec{u}) - \Lambda + \Gamma + \varepsilon\frac{\mathcal{S}}{\rho} - \varepsilon\frac{\mathcal{D}}{\rho},$$

$$\frac{\partial \rho \varepsilon}{\partial t} + \nabla \cdot (\rho \epsilon \vec{u}) = -(\gamma - 1)\rho\epsilon \nabla \cdot \vec{u} - \Lambda + \Gamma + \varepsilon\frac{\mathcal{S}}{\rho} - \varepsilon\frac{\mathcal{D}}{\rho},$$

$$\rho E = \frac{1}{2}\rho \vec{u}^2 + \rho\varepsilon,$$

$$p = (\gamma - 1)\rho\varepsilon,$$

Для описания бесстолкновительной компоненты будем использовать систему уравнений для первых моментов бесстолкновительного уравнения Больцмана, записанную также в эйлеровых координатах:

$$\frac{\partial n}{\partial t} + \nabla \cdot (n\vec{v}) = \mathcal{D} - \mathcal{S},$$

$$\frac{\partial n\vec{v}}{\partial t} + \nabla \cdot (n\vec{v}\vec{v}) = -\nabla \Pi - n\nabla(\Phi) + \vec{u}\mathcal{D} - \vec{v}\mathcal{S},$$

$$\frac{\partial \rho W}{\partial t} + \nabla \cdot (\rho W \vec{v}) = -\nabla \cdot (\Pi \vec{v}) - (n\nabla(\Phi), \vec{v}) + \varepsilon \frac{\mathcal{D}}{\rho} - \varepsilon \frac{\mathcal{S}}{\rho},$$

$$\frac{\partial \Pi_{\xi\xi}}{\partial t} + \nabla \cdot (\Pi_{\xi\xi} \vec{v}) = -2\Pi \nabla \cdot \vec{u} + \varepsilon \frac{\mathcal{D}}{3\rho} - \varepsilon \frac{\mathcal{S}}{3\rho},$$

$$\rho W = \frac{1}{2}\rho \vec{v}^2 + \frac{\Pi_{xx} + \Pi_{yy} + \Pi_{zz}}{2},$$

Уравнение Пуассона для обеих компонент записывается в виде:

$$\Delta \Phi = 4\pi G(\rho + n),$$

где p — давление газа, ρ_i — плотность i компоненты смеси газа, s_{ij} — скорость прохождения химических реакций, $\rho = \sum_i \rho_i$ — плотность смеси газа, n — плотность бесстолкновительной компоненты, $\vec{u}$ — скорость газовой компоненты, $\vec{v}$ — скорость бесстолкновительной компоненты, ρE — плотность полной механической энергии газа, ρW — плотность полной механической энергии бесстолкновительной компоненты, Φ — гравитационный потенциал, ε — плотность внутренней энергии газа, γ — эффективный показатель адиабаты, $\Pi_{\xi\xi} = (\Pi_{xx}, \Pi_{yy}, \Pi_{zz})$ — диагональный тензор дисперсии скоростей бесстолкновительной компоненты, $\mathcal{S}$ — скорость образования сверхновых звезд, $\mathcal{D}$ — скорость звездообразования, Λ — функция охлаждения, Γ — функция нагревания от взрыва сверхновых звезд. Мы не будем вводить подробности описания каждого терма для описания подсеточной физики, так как подробности их применения могут быть найдены в работах [4, 10, 13].

2.1. Описание численного метода

Для численного решения уравнений гравитационной газовой динамики был использован оригинальный численный метод, основанный на комбинации метода Годунова, метода разделения операторов и кусочно-параболического метода на локальном шаблоне для обеспечения высокого порядка точности [14, 15].

Система уравнений решается в два этапа: эйлеров, на котором решаются уравнения без адвективных членов, и лагранжев, на котором происходит адвективный перенос гидродинамических величин. На эйлеровом этапе гидродинамические уравнения для обеих компонент записываются в неконсервативной форме и исключаются адвективные члены. В результате такая система на интерфейсе двух ячеек имеет аналитическое решение, которое используется для записи потоков через интерфейс двух ячеек [16]. Для повышения порядка точности используется кусочно-параболический метод на локальном шаблоне (PPML), который состоит в построении локальных парабол внутри ячеек для каждой гидродинамической величины. Главное отличие PPML от классического PPM метода состоит в использовании локального шаблона для вычислений. Это позволяет на этапе параллельной реализации, в основе которой геометрическая декомпозиция расчетной области, использовать только один слой перекрытия подобластей, что упрощает реализацию граничных условий и уменьшает количество пересылок, следовательно способствует росту эффективности параллельной реализации. На лагранжевом этапе используется аналогичный численный подход.

На данный момент решение уравнения Пуассона основано на Fast Fourier Transform методе. Это связано с тем, что решение уравнения Пуассона занимает несколько процентов от времени счета, но в дальнейшем мы планируем перейти к итерационным методам решения таким как SOR и CGM. После решения уравнения Пуассона и гидродинамических уравнений происходит корректировка решения переопределенной системы уравнений, для этого

используется оригинальная процедура для сохранения полной энергии системы и гарантии неубывания энтропии [17, 18].

В результате разработанный численный метод решения обладает следующими свойствами: высокий порядок точности на гладких решениях и малая диссипация в случае разрывных решений; отсутствие необходимости введения члена искусственной вязкости или ограничителей; инвариантность получаемого численного решения относительно поворота и отсутствие карбункул-эффектов; гарантированное неубывание энтропии; возможность расширения на более сложные гидродинамические модели; простота программной реализации; потенциально бесконечная масштабируемость. Последний пункт нам наиболее важен и основан на том факте, что все вычисления в ячейках происходят независимо, регулярно и на локальном шаблоне.

Численный метод был протестирован на следующих задачах:

1. Одномерные тесты Годунова о распаде разрыва.
2. Одномерный тест Аксенова с непрерывным периодическим решением.
3. Задача Седова о точечном взрыве.
4. Двумерная неустойчивость Релея–Тейлора.
5. Двумерная неустойчивость Кельвина–Гельмгольца.
6. Задача коллапса Эврарда.

Подробное описание численного метода и его верификация приведена в работе [19]. Также разработано расширение численного метода на решение МГД уравнений [20].

2.2. Декомпозиция расчетной области

Со-дизайн [9] физико-математической модели, численного метода и структур данных позволяет использовать геометрическую декомпозицию расчетной области с одним слоем перекрытия подобластей. Такую возможность мы имеем за счет построения парабол на предыдущем шаге, что требует только локального взаимодействия между ячейками. На рис. 1 приведены процентные соотношения между этапами.

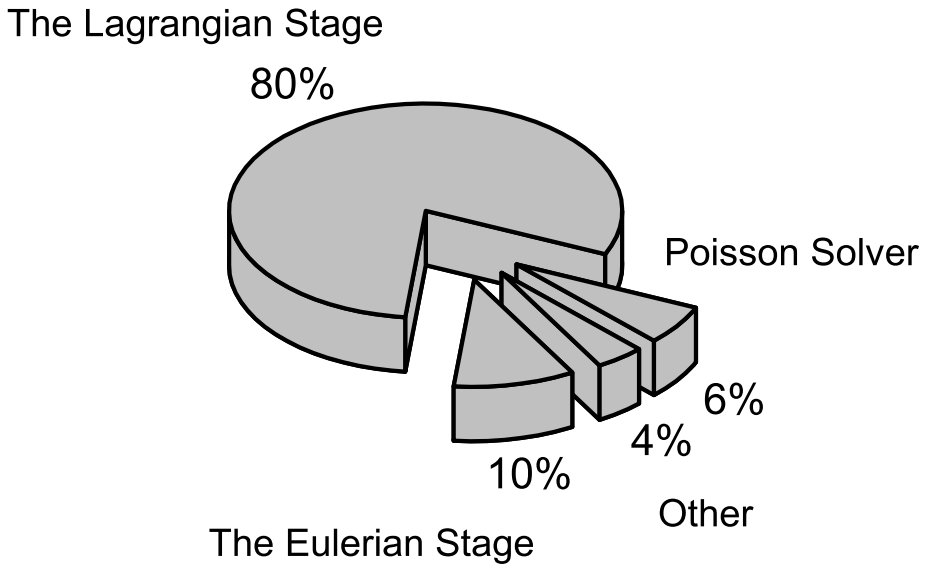


Рис. 1. Процентное соотношение между этапами в коде AstroPhi

Для решения уравнения Пуассона, в основе которого быстрое преобразование Фурье для суперЭВМ с распределенной памятью была использована библиотека FFTW [21]. В основе этой библиотеки лежит процедура ALLTOALL, которая «транспонирует» трехмерный массив, перераспределяя значительные объемы памяти между всеми процессами. Безусловно, это дорогая сетевая операция, которая требует отказа от всего алгоритма в случае использовании сколь либо значительного количества вычислителей. Однако, эта процедура в случае использования сетевой инфраструктуры InfiniBand не занимает критическое время и, по всей видимости, оптимизирована на низком сетевом уровне [22].

Основными этапами вычислительной схемы являются эйлеров и лагранжев этапы. Мы сосредоточимся именно на этих этапах, как на наиболее затратных. Также вне нашего рассмотрения в плане ускорения останутся процедуры, в которых «деление» импульса на функцию плотности и перезапись массивов. В этих процедурах фактически происходит копирование памяти из одной области в другую, в дальнейшем мы также рассмотрим эти операции отдельно с точки зрения обобщенной функции MEMCPY.

Отдельно остановимся на процедуре вычисления шага по времени, исходя из условия Куранта. В случае использования графических ускорителей данная процедура была реализована только на CPU [6] (также было сделано и в коде GAMER [23]). Причина этого — отсутствие эффективной реализации редуцирующей операцией min в технологии CUDA. В то время как в OpenMP такая операция эффективно реализована. Стоимость этой процедуры составляет порядка одного процента от общего времени вычислений и практически не влияет на эффективность параллельной реализации. Однако, при увеличении количества графических ядер до нескольких тысяч и стократного ускорения в рамках одного графического процессора суммарно всех остальных процедур, может возникнуть курьезная ситуация, когда процедура вычисления шага по времени будет выполняться дольше всех остальных. При том, что авторами уже было достигнуто 55-кратное ускорение в рамках одного GPU [6] и количество графических ядер в одном ускорителе увеличивается, то такая ситуация может быть достигнута в ближайшие пару лет. Стоит отметить, что такая проблема в принципе невозможна на ускорителях Intel Xeon Phi.

Использование равномерной сетки в декартовых координатах для решения уравнений гидродинамики позволяет использовать произвольную декартову топологию для декомпозиции расчетной области. Такая организация вычислений имеет потенциально бесконечную масштабируемость. В коде AstroPhi используется многоуровневая одномерная декомпозиция расчетной области. По одной координате внешнее одномерное разрезание происходит средствами технологии MPI, внутри каждой подобласти разрезание происходит средствами OpenMP, адаптированного для MIC-архитектур (рис. 2).

Такой подход использовался также и в первой версии программного кода AstroPhi [7] с учетом использования offload режима. Такая декомпозиция связана с топологией и архитектурой гибридного суперЭВМ RSC PetaStream, который был использован для вычислительных экспериментов.

2.3. Шаблоны программирования для Intel Xeon Phi

Для использования Intel Xeon Phi использован регулярный шаблон вычислений, который следует из схемы декомпозиции расчетной области и состоит в распределении работ по нитям (см. рис. 3).

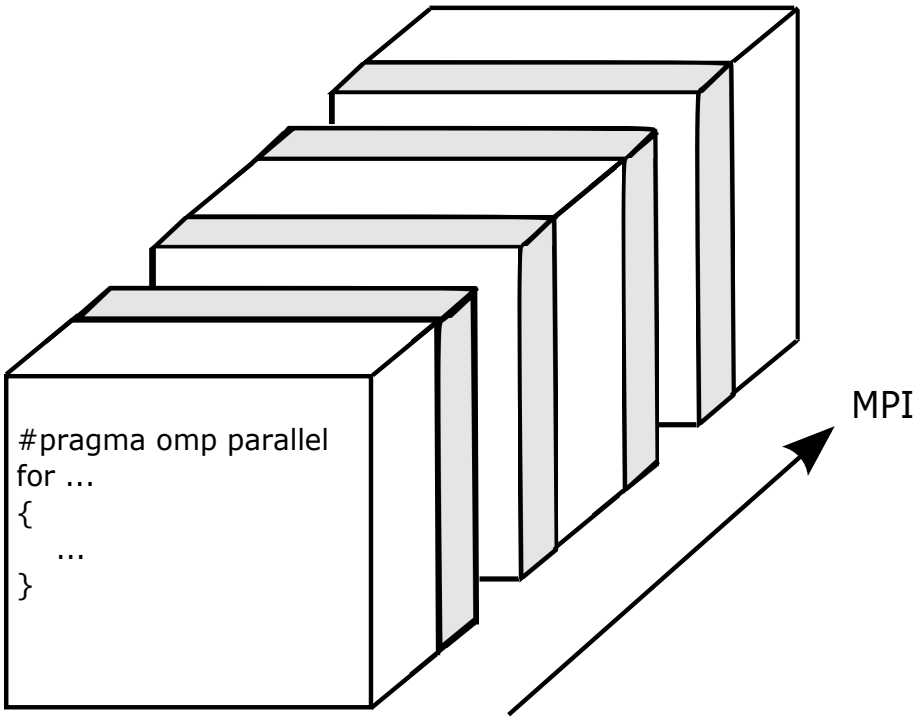


Рис. 2. Схема геометрической декомпозиции в коде AstroPhi

В листинге приведена заготовка для использования offload режима использования ускорителя Intel Xeon Phi, аналогичный подходу используемому в работе [7].

2.4. Шаблоны сетевых взаимодействий

Межпроцессное взаимодействие средствами MPI осуществляется с помощью шаблона передачи по двунаправленному списку (см. рис. 4) крайних элементов одномерного массива размером N элементов.

Указанный шаблон является очень простым, однако именно на нем построены более сложные межпроцессные взаимодействия обмена срезами трехмерных массивов.

3. Исследование производительности

Для экспериментов были использованы два гибридных суперкомпьютера на основе архитектуры RSC PetaStream: МВС-10П МСЦ РАН (64 ускорителя Intel Xeon Phi 7120 D) и Политехник RSC PetaStream СПбПУ (256 ускорителя Intel Xeon Phi 5120 D). Далее приведем исследования производительности различных подсистем кода: исследование ускорения, масштабируемости, имитационного моделирования масштабируемости, пропускной способности памяти и скорость сетевых коммуникаций. В наших исследованиях вопросы масштабируемости и ускорения были исследованы на обеих архитектурах, вопросы связанные с организацией вычислений были проведены на суперкомпьютере МВС-10П МСЦ РАН.

В силу разного объема памяти на ускорителях Intel Xeon Phi 7120 D исследование ускорения проводилось на сетке 512^3, на ускорителях Intel Xeon Phi 5120 D была использована сетка 512×256^2. Это максимальные размеры сеток, которые могут поместиться в один ускоритель. Для измерения ускорения замерялось время каждого этапа численного метода, в секундах, а затем вычислялась их сумма при различном числе используемых логических

```
...
// Offload/Native mode
#define NATIVE /* OFFLOAD */
// Number of MIC-threads
#define MIC_NUM_THREADS 240
...
#ifdef OFFLOAD
#pragma offload_attribute (push,target(mic))
#endif
double foo(double *a, double x, int index)
{
 return a[index] * x ;
}
#ifdef OFFLOAD
#pragma offload_attribute (pop)
#endif
...
#ifdef OFFLOAD
#pragma offload target (mic) in (a:length(N)) \
        out(c:length(N))
#endif
{
 #pragma omp parallel for default(none) shared(a,x,c) \
         num_threads(MIC_NUM_THREADS)
 for(i=0;i<N;i++)
  c[i] = foo(a,x,i);
}
...
```

Рис. 3. Шаблон работы с процедурами на Intel Xeon Phi

ядер (Threads). Ускорение P *(SpeedUp)* вычислялось по формуле 1:

$$P = \frac{Total_1}{Total_K}, \tag{1}$$

где $Total_1$ — время вычислений на одном логическом ядре, $Total_K$ — время вычислений при использовании K логических ядер. Результаты исследований ускорения для суперкомпьютера МВС-10П МСЦ РАН (JSCC) и Политехник RSC PetaStream СПбПУ (SPb) приведены на рис. 5.

Таким образом, было получено 134-кратное ускорение (масштабируемость в сильном смысле) в рамках одного ускорителя Intel Xeon Phi 7120 D и 84-кратное ускорение в рамках одного ускорителя Intel Xeon Phi 5120 D. Такие значения ускорения по всей видимости напрямую связаны с производительностью каждого ускорителя. Так исследования ускорения (и дальнейшей масштабируемости) на суперкомпьютере МВС-10П МСЦ РАН было сделано в апреле 2015 года, а исследования на суперкомпьютере Политехник RSC PetaStream СПбПУ было сделано в ноябре 2015 года.

```
#define COMM   MPI_COMM_WORLD
#define STATUS MPI_STATUS_IGNORE
#define TR 1 // "to right" communications
#define TL 2 // "to left" communications
...
if(rank == 0)
{ buffer[0] = a[N-2];
  MPI_Send(buffer,1,MPI_DOUBLE,rank+1,TR,COMM);
  MPI_Recv(buffer,1,MPI_DOUBLE,rank+1,TL,COMM,STATUS);
  a[N-1] = buffer[0]; }
if(rank == size-1)
{ MPI_Recv(buffer,1,MPI_DOUBLE,rank-1,TR,COMM,STATUS);
  a[0] = buffer[0];
  buffer[0] = a[1];
  MPI_Send(buffer,1,MPI_DOUBLE,rank-1,TL,COMM); }
if(rank!=0 && rank!=size-1)
{ MPI_Recv(buffer,1,MPI_DOUBLE,rank-1,TR,COMM,STATUS);
  a[0] = buffer[0];
  buffer[0] = a[N-2];
  MPI_Send(buffer,1,MPI_DOUBLE,rank+1,TR,COMM);
  MPI_Recv(buffer,1,MPI_DOUBLE,rank+1,TL,COMM,STATUS);
  a[N-1] = buffer[0];
  buffer[0] = a[1];
  MPI_Send(buffer,1,MPI_DOUBLE,rank-1,TL,COMM); }
...
```

Рис. 4. Шаблон сетевых взаимоедйствий средствами MPI

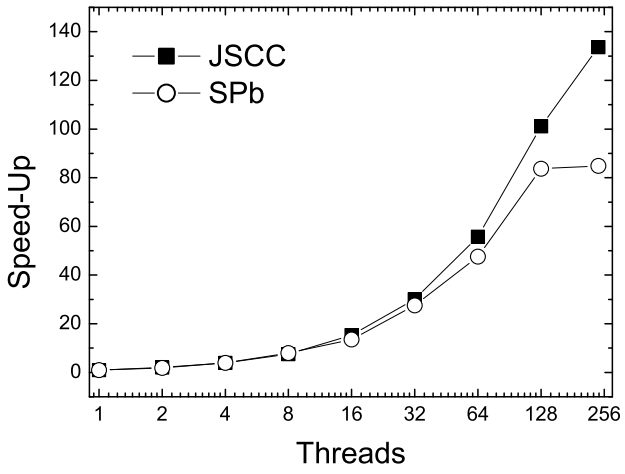


Рис. 5. Исследование ускорения кода AstroPhi

Проводилось исследование масштабируемости кода AstroPhi на ускорителях Intel Xeon Phi 7120 D на сетке $512p \times 512 \times 512$, на ускорителях Intel Xeon Phi 5120 D была использована сетка $512p \times 256 \times 256$ в обеих случаях использовались четыре логических ядра на каждый ускоритель, где p — число используемых ускорителей. Таким образом, на каждый ускоритель приходится одинаковый размер подобласти при любом числе исследуемых уско-

рителей. Для исследования масштабируемости замерялось время каждого этапа численного метода, в секундах, а затем вычислялась их сумма (Total) при различном числе используемых ускорителей Intel Xeon Phi (MIC). Масштабируемость T *(Scalability)* вычислялось по формуле

$$T = \frac{Total_1}{Total_p}, \tag{2}$$

где $Total_1$ — время вычислений на одном ускорителе при использовании одного ускорителя, $Total_p$ — время вычислений на одном ускорителей при использовании p ускорителей. Результаты исследований масштабируемости приведены на рис. 6.

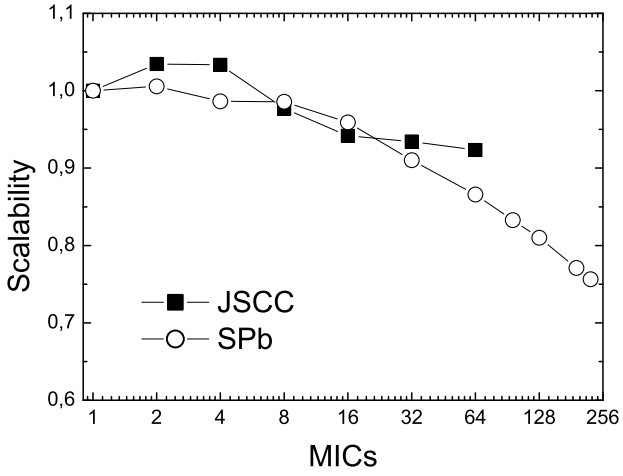


Рис. 6. Исследование масштабируемости кода AstroPhi

Таким образом, была получена 92-процентная эффективность (масштабируемость в слабом смысле) на 64 ускорителях Intel Xeon Phi 7120 D и 75-процентная эффективность на 224 ускорителях Intel Xeon Phi 5120 D. Заметим, что эффективность быстрее просаживается на суперкомпьютере СПбПУ, что вероятно связано со сложностью сетевой инфраструктуры и дополнительными сетевыми расходами на организацию обменов.

Особенностью оригинального подхода является возможность простой геометрической декомпозиции расчетной области и последующим обменом граничных значений между только соседними вычислительными узлами. Имитационная модель организации вычислений строится из следующих предположений:

1) для нахождения общего времени выполнения вычислений на каждом этапе будем предполагать, что нам известно среднее время вычислений на одну ячейку, таким образом, предполагая однородность вычислений по всей расчетной области;
2) в качестве вычислительного узла выбирается ускоритель Intel Xeon Phi полностью, тем самым не моделируется масштабируемость вычислений внутри одного устройства;
3) время выполнения коммуникаций будем считать линейной функцией от числа передаваемых элементов с учетом латентности;
4) количество передаваемых элементов, а, следовательно, и время передачи, после каждого из этапов численного метода одинаково;
5) используется сетевая инфраструктура ССКЦ ИВМиМГ СО РАН;
6) используется одномерная декомпозиция расчетной области;

7) рассматривается только экстенсивность вычислительной системы при сохранении производительности отдельного устройства.

Построенная на таких допущениях имитационная модель кода AstroPhi была смоделирована с помощью комплекса AGNES [24] на различном числе модельных ускорителей. Вычислительные эксперимены показали, что программный комплекс AstroPhi может быть с 70-процентной эффективностью масштабируем до одного миллиона вычислительных устройств. Такое число ускорителей соответствует экзафлопсному уровню вычислений.

4. Вычислительные эксперименты

4.1. Моделирование образования крупномасштабных космологических структур

С момента времени, соответствующему $z = 99$ будем рассматривать расширяющуюся кубическую область с длиной куба $L = 100/h$ Mpc $= 3 \times 10^{23}/h$ m, и периодическими граничными условиями по каждому измерению. В качестве характерного значения плотности взято значение $\rho = 1,88 \times 10^{-26} h^2$ kg/m^3. Доля темной энергии $\Omega_\Lambda = 0,73$, темной материи $\Omega_D = 0,226$, видимой барионной материи $\Omega_B = 0,044$ (в начальный момент времени предполагается отсутствие звезд). Температура газовой компоненты $T = 10^4$ K. Постоянная Хаббла $H = 67,8$ км/сек/мпс. Для задания начальных данных задаются малые флуктуации равномерно распределенной плотности. Для задания случайных возмущений формируется нормальное распределение с амплитудой, соответствующей энергетическому космологическому спектру. Затем выполняется обратное преобразование Фурье. В рамках двухфазной многокомпонентной гидродинамической модели с учетом космологического расширения и подсеточных процессов было смоделировано (см. рис. 7) образование крупномасштабных космологических структур — волос (филаменты в зарубежной литературе), стен (блинчики Я.Б. Зельдовича в российской литературе), скоплений (кластеры в зарубежной литературе) галактик, пустот (войды в зарубежной литературе).

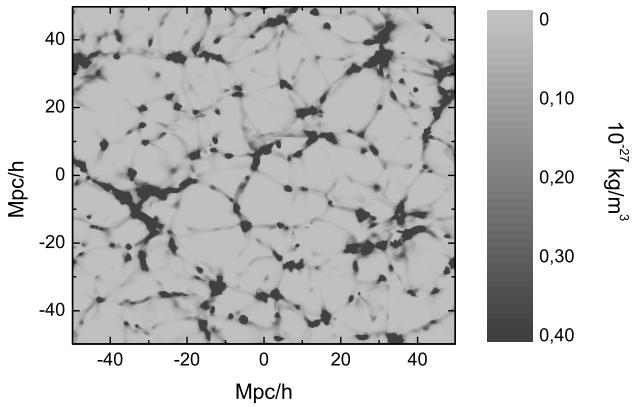


Рис. 7. Плотность темной материи в момент $z = 0$

В результате вычислительного эксперимента было показано качественное соответствие структуры смоделированного и наблюдаемых скоплений, количественное соответствие масс смоделированных галактик и расстояний между ними с наблюдаемыми значениями.

4.2. Моделирование образования спиральных рукавов галактик

Объяснен механизм образования спиральных неустойчивостей в галактическом диске в модели изотермической гидродинамики, приводящий к образованию многорукавных галактик (двух-, четырех- и семирукавная структура) в ходе развития гравитационной неустойчивости. Определены параметры для образования каждого вида галактик (см. рис. 8).

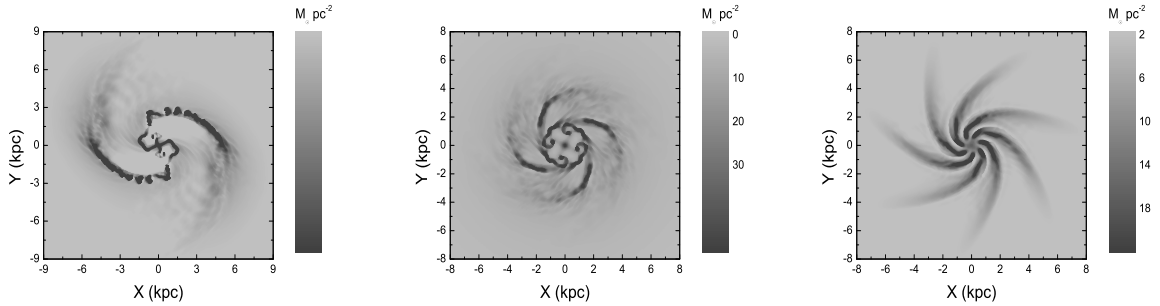


Рис. 8. Столбцевая плотность (в $M_{\odot}pc^{-2}$) двухрукавной (слева), четырехрукавной (посередине) и семирукавной (справа) галактик

4.3. Моделирование образования спиральных рукавов галактик

Исследована задача образования молекулярных облаков в ходе развития МГД турбулентности (см. рис. 9).

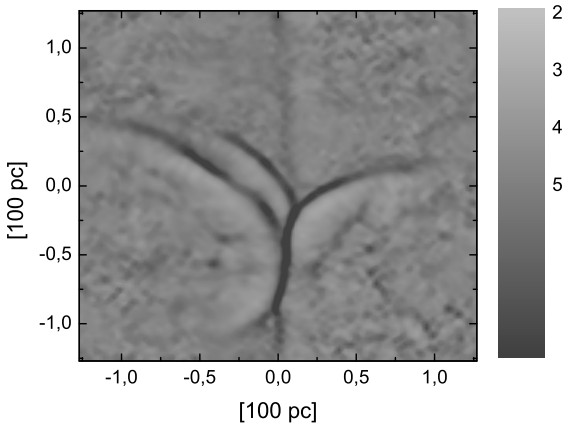


Рис. 9. Задача развития МГД турбулентности межзвездной среды. Концентрация газа в $см^{-3}$ в момент времени $t = 15$ млн. лет

После процесса ионизации водорода происходит процесс образования облачных структур. Для вычислительного эксперимента использовалась сетка 512^3 ячеек, для которой также была проанализирована зависимость альфвеновской скорости от плотности газа (см. рис. 10 слева) и косинуса угла колинеарности между векторами скорости и магнитного поля от плотности газа (см. рис 10 справа).

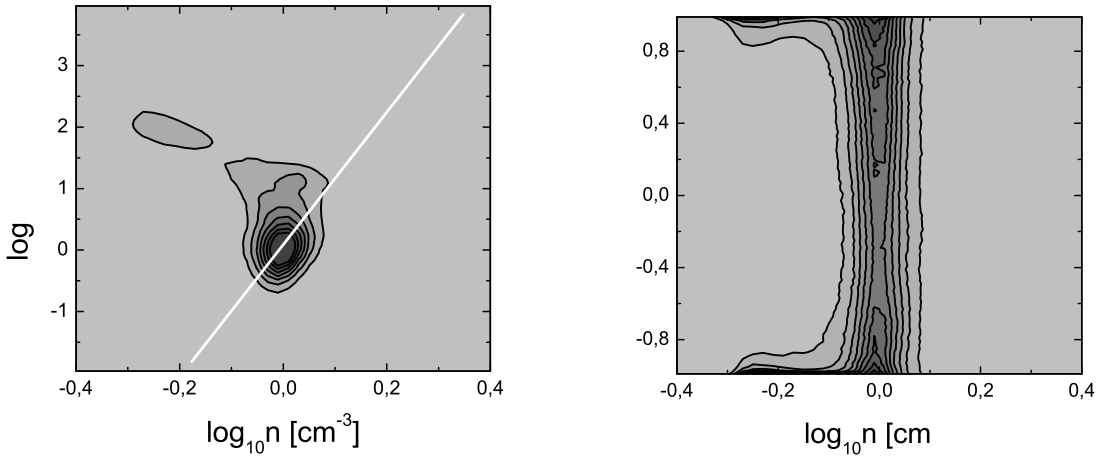


Рис. 10. Задача развития МГД турбулентности межзвездной среды. Зависимость альфвеновской скорости от плотности газа (слева) и косинуса угла колинеарности между векторами скорости и магнитного поля от плотности газа (справа)

Из рисунков видно, что для альфвеновского числа Маха прослеживается корреляция $\mathcal{M} \sim n^2$, показанная белой линией, и большая часть облака $n > 10$ см$^{-3}$ попадают в сверхальфвеновскую область (см. рис. 10 слева). Причина возникновения такого режима связано с самоорганизацией в замагниченной турбулентной межзвездной среде в трансальфеновском режиме $\mathcal{M} \sim 1$ при $n \sim 1$. При таких плотностях (см. рис 10 справа) контуры косинуса угла колинеарности между векторами скорости и магнитного поля образуют седловидную структуру, что говорит о том, что сжатие происходит вдоль силовых линий магнитного поля. Затем за счет влияния самогравитации происходит дальнейшее увеличение массы и плотности облаков. В свою очередь, в полученных плотных облаках турбулентность является только сверхальфвеновской с числом Маха $\mathcal{M} > 100$.

4.4. Задача высоко-скоростного столкновения дисковой галактики с межгалактическим ветром

В качестве одного из вычислительных экспериментов выбрана задача высокоскоростного столкновения дисковой галактики с межгалактическим ветром в гидродинамической модели. В результате такого взаимодействия образуется механизм набегающего потока и происходит обтекание с образованием неустойчивостью за галактикой. Образование подобных неустойчивостей и хвостов важны для изучения механизма образования пекулярных галактик и процесса звездообразования [25, 26]. Дисковая галактика задается равновесной конфигурацией сферического гало с NFW-профилем плотности и равновесным экспонециальным профилем плотности с дифференциальным вращением. Общая масса галактики составляет $M = 10^{13} M_{\odot}$. Скорость набегающего потока составляет $v = 600$ км/с. Постановка задачи изображена на рис. 11.

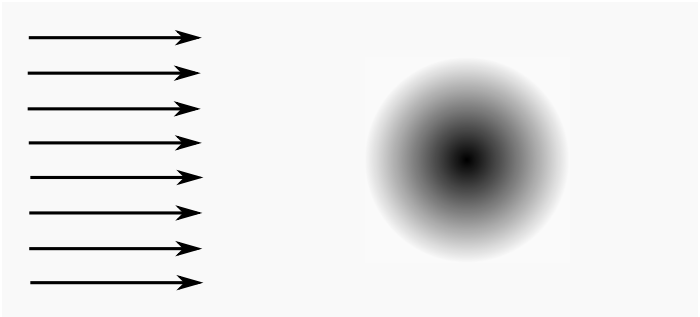


Рис. 11. Постановка задачи набегания газа на галактику

Результаты моделирования представлены на рис. 12, которые согласуются с результатами аналогичного моделирования [26] и наблюдениями [27]. Расчеты были проведены на последовательности сеток от $896 \times 128 \times 128$ до $7168 \times 1024 \times 1024$. На последней сетке было достигнуто 47 процентов от пиковой скалярной производительности ускорителя Intel Xeon Phi при использовании 53760 нитей.

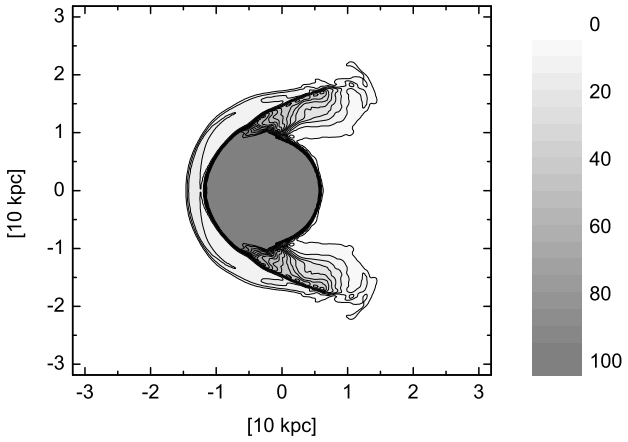


Рис. 12. Результаты моделирования. Столбцевая плотность в $M_{\odot}pc^{-2}$

На рис. 12 видно образование хвоста за фронтом галактики, который образуется вследствие набегания газа на галактику.

Заключение

В работе были представлены исследования кода AstroPhi для численного моделирования астрофизических течений на гибридных суперЭВМ, оснащенных ускорителями Intel Xeon Phi. Подробно описан со-дизайн вычислительной модели для описания астрофизических объектов. Детально описаны особенности параллельной реализации и исследования производительности кода AstroPhi. Для кода AstroPhi было достигнуто 134-кратное ускорение в рамках одного ускорителя Intel Xeon Phi, 75-процентная масштабируемость при использовании 224 ускорителей Intel Xeon Phi. Представлены результаты моделирования взаимодействия межгалактического ветра и дисковой галактики. На расчетной сетке $7168 \times 1024 \times 1024$ было достигнуто 47 процентов от пиковой скалярной производительности ускорителя Intel Xeon Phi при использовании 53760 нитей. В будущем планируется разработка векторизованного варианта кода AstroPhi, что позволит получить сверхвысокую производительность вплоть до 1 терафлопса на один ускоритель Intel Xeon Phi.

Работа поддержана грантом Российского фонда фундаментальных исследований 15-31-20150 мол-а-вед, 15-01-00508 и 16-07-00434, грантом Президента РФ МК – 6648.2015.9. Работа выполнена при частичной поддержке проектной части госзадания № 3.961.2014/К Министерства образования и науки Российской Федерации (Э.И. Воробьев).

Литература

1. Tutukov A., Lazareva G., Kulikov I. Gas Dynamics of a Central Collision of Two Galaxies: Merger, Disruption, Passage, and the Formation of a New Galaxy // Astronomy Reports. 2011. Vol. 55, No. 9. P. 770–783.
2. Mitchell N., Vorobyov E., Hensler G. Collisionless Stellar Hydrodynamics as an Efficient Alternative to N-body Methods // Monthly Notices of the Royal Astronomical Society. 2013. Vol. 428, No. 3. P. 2674–2687.
3. Ardeljan N.V., Bisnovatyi-Kogan G.S., Kosmachevskii G.S., Moiseenko S.G. An implicit Lagrangian code for the treatment of nonstationary problems in rotating astrophysical bodies // Astronomy and Astrophysics Supplement Series. 1996. Vol. 115. P. 573–594.
4. Khoperskov S.A, Vasiliev E.O., Sobolev A.M., Khoperskov A.V. The simulation of molecular clouds formation in the Milky Way // Monthly Notices of the Royal Astronomical Society. 2013. Vol. 428. P. 2311–2320.
5. Fletcher A., Beck R., Shukurov A., Berkhuijsen E., Horellou C. Magnetic fields and spiral arms in the galaxy M51 // Monthly Notices of the Royal Astronomical Society. 2014. Vol. 412. P. 2396–2416.
6. Kulikov I. GPUPEGAS: A New GPU-accelerated Hydrodynamic Code for Numerical Simulations of Interacting Galaxies // The Astrophysical Journal Supplement Series. 2014. Vol. 214, Id. 12.
7. Kulikov I.M., Chernykh I.G., Snytnikov A.V., Glinskiy B.M., Tutukov A.V. AstroPhi: A code for complex simulation of dynamics of astrophysical objects using hybrid supercomputers // Computer Physics Communications. 2015. Vol. 186. P. 71–80.
8. Куликов И.М., Черных И.Г., Глинский Б.М. AstroPhi: программный комплекс для моделирования динамики астрофизических объектов на гибридных суперэвм, оснащенных ускорителями Intel Xeon Phi // Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика. 2013. Т. 2, № 4. С. 57–79.
9. Glinskiy B., Kulikov I., Snytnikov A., Romanenko A., Chernykh I., Vshivkov V. Co-design of Parallel Numerical Methods for Plasma Physics and Astrophysics // Supercomputing frontiers and innovations. 2015. Vol. 1, No. 3. P. 88–98.
10. Vorobyov E., Recchi S., Hensler G. Stellar hydrodynamical modeling of dwarf galaxies: simulation methodology, tests, and first results // Astronomy & Astrophysics. 2015. Vol. 579, Id. A9.
11. Kulikov I., Chernykh I. Glinskiy B., Weins D., Shmelev A. Astrophysics simulation on RSC massively parallel architecture // Proceedings – 2015 IEEE/ACM 15th International Symposium on Cluster, Cloud, and Grid Computing, CCGrid 2015. 2015. P. 1131–1134.
12. Vshivkov V., Lazareva G., Snytnikov A., Kulikov I. Supercomputer Simulation of an Astrophysical Object Collapse by the Fluids-in-Cell Method // Lecture Notes of Computer Science. 2009. Vol. 5698. P. 414–422.

13. Kulikov I., Chernykh I., Katysheva E., Protasov A., Serenko A. The numerical simulation of interacting galaxies by means of hybrid supercomputers // Bulletin NCC: Numerical analysis. 2015. Vol. 17. P. 17–33.

14. Popov M., Ustyugov S. Piecewise parabolic method on local stencil for gasdynamic simulations // Computational Mathematics and Mathematical Physics. 2007. Vol. 47, No. 12. P. 1970–1989.

15. Popov M., Ustyugov S. Piecewise parabolic method on a local stencil for ideal magnetohydrodynamics // Computational Mathematics and Mathematical Physics. 2008. Vol. 48, No. 3. P. 477–499.

16. Vshivkov V., Lazareva G., Snytnikov A., Kulikov I., Tutukov A. Hydrodynamical code for numerical simulation of the gas components of colliding galaxies // The Astrophysical Journal Supplement Series. 2011. Vol. 194, Id. 47.

17. Godunov S., Kulikov I. Computation of Discontinuous Solutions of Fluid Dynamics Equations with Entropy Nondecrease Guarantee // Computational Mathematics and Mathematical Physics. 2014. Vol. 54. P. 1012–1024.

18. Vshivkov V., Lazareva G., Snytnikov A., Kulikov I., Tutukov A. Computational methods for ill-posed problems of gravitational gasodynamics // Journal of Inverse and Ill-posed Problems. Vol. 19, No. 1. P. 151–166.

19. Kulikov I., Vorobyov E. Using the PPML approach for constructing a low-dissipation, operator-splitting scheme for numerical simulations of hydrodynamic flows // The Journal of Computational Physics. 2016. Vol. 317. P. 318–346.

20. Kulikov I., Chernykh I., Snytnikov A., Protasov V., Tutukov A., Glinsky B. Numerical Modelling of Astrophysical Flow on Hybrid Architecture Supercomputers // In Parallel Programming: Practical Aspects, Models and Current Limitations (ed. M. Tarkov). 2015. P. 71–116.

21. Frigo M., Johnson S. The Design and Implementation of FFTW3 // Proceedings of the IEEE. 2005. Vol. 93, No. 2. P. 216–231.

22. Kalinkin A., Laevsky Y., Gololobov S. 2D Fast Poisson Solver for High-Performance Computing // Lecture Notes in Computer Science. 2009. Vol. 5698. P. 112–120.

23. Schive H., Tsai Y., Chiueh T. GAMER: a GPU-accelerated Adaptive-Mesh-Refinement Code for Astrophysics // The Astrophysical Journal. 2010. Vol. 186. P. 457–484.

24. Podkorytov D., Rodionov A., Sokolova O., Yurgenson A. Using Agent-Oriented Simulation System AGNES for Evaluation of Sensor Networks // Lecture Notes of Computer Science. 2010. Vol. 6235. P. 247–250.

25. Jaffe Y.L., Smith R., Candlish G., Poggianti B.M., Sheen Y.-K., Verheijen M.A.W. BUDHIES II: A phase-space view of HI gas stripping and star-formation quenching in cluster galaxies // Monthly Notices of the Royal Astronomical Society. 2015. Vol. 448. P. 1715–1728.

26. Vollmer B., Cayatte V., Balkowski C., Duschl W.J. Ram pressure stripping and galaxy orbits: The case of the Virgo cluster // The Astrophysical Journal. 2001. Vol. 561. P. 708–726.

27. Cayatte V., Kotanyi C., Balkowski C., van Gorkom J.H. A very large array survey of neutral hydrogen in Virgo Cluster spirals. 3: Surface density profiles of the gas // The Astronomical Journal. 1994. Vol. 107, No. 3. P. 1003–1017.

DOI: 10.14529/cmse160406

NUMERICAL HYDRODYNAMICS SIMULATION OF ASTROPHYSICAL FLOWS AT INTEL XEON PHI SUPERCOMPUTERS

© 2016 I.M. Kulikov[1], I.G. Chernykh[2], E.I. Vorobyov[3], A.V. Snytnikov[1], D.V. Weins[2], A.A. Moskovsky[4], A.B. Shmelev[4], V.A. Protasov[5], A.A. Serenko[5], V.E. Nenashev[5], V.A. Vshivkov[1], A.S. Rodionov[6], B.M. Glinsky[2], A.V. Tutukov[7]

[1]*Laboratory of Parallel Algorithms for Solving Large Problems, Institute of Computational Mathematics and Mathematical Geophysics SB RAS (pr. Ac. Lavryenteva 6, Novosibirsk, 630090 Russia),*

[2]*Siberian Supercomputer Center, Institute of Computational Mathematics and Mathematical Geophysics SB RAS (pr. Ac. Lavryenteva 6, Novosibirsk, 630090 Russia),*

[3]*Laboratory of Space Research, Southern Federal University (pr. Stachki 194, Rostov-on-Don, 344090 Russia),*

[4]*ZAO RSC Technologies (Kutuzovskiy pr. 36, building 23, Moscow, 121170 Russia),*

[5]*Novosibirsk State Technical University (pr. K. Marksa 20, Novosibirsk, 630073 Russia),*

[6]*Laboratory of Dynamic Processes Simulation in Information Networks, Institute of Computational Mathematics and Mathematical Geophysics SB RAS (pr. Ac. Lavryenteva 6, Novosibirsk, 630090 Russia),*

[7]*Department of Stellar Physics and Evolution, Institute of Astronomy RAS (Pyatnitskaya St. 48, Moscow, 119017 Russia)*

E-mail: kulikov@ssd.sscc.ru, chernykh@parbz.sscc.ru, eduard.vorobiev@univie.ac.at, snytav@gmail.com, wns.dmitry@gmail.com, moskov@rsc-tech.ru, alexeysh@rsc-tech.ru, inc_13@mail.ru, fafnur@yandex.ru, arni.12@mail.ru, vsh@ssd.sscc.ru, alrod@sscc.ru, gbm@opg.sscc.ru, atutukov@inasan.ru

Received: 13.04.2016

In this paper we propose a research of AstroPhi code for numerical simulation of astrophysical flows at Intel Xeon Phi supercomputers. The co-design of a computational astrophysics model are described. The parallel implementation and scalability tests of the AstroPhi code are presented. The results of simulation of interaction between intergalactic wind and a disk galaxy are provided. For AstroPhi code a 134x speed-up with one Intel Xeon Phi accelerator and 75% weak scaling efficiency on 224x Intel Xeon Phi accelerators was obtained. We got peak of performance on a $7168 \times 1024 \times 1024$ mesh size by means 53760 RSC PetaStream threads.

Keywords: high performance computing, numerical astrophysics, Intel Xeon Phi accelerators.

FOR CITATION

Kulikov I.M., Chernykh I.G., Vorobyov E.I., Snytnikov A.V., Weins D.V., Moskovsky A.A., Shmelev A.B., Protasov V.A., Serenko A.A., Nenashev V.E., Vshivkov V.A., Rodionov A.S., Glinsky B.M., Tutukov A.V. Numerical Hydrodynamics Simulation of Astrophysical Flows at Intel Xeon Phi Supercomputers. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. 2016. vol. 5, no. 4. pp. 77–97. (in Russian) DOI: 10.14529/cmse160406.

References

1. Tutukov A., Lazareva G., Kulikov I. Gas Dynamics of a Central Collision of Two Galaxies: Merger, Disruption, Passage, and the Formation of a New Galaxy. *Astronomy Reports*. 2011. vol. 55, no. 9. pp. 770–783. DOI: 10.1134/S1063772911090083.

2. Mitchell N., Vorobyov E., Hensler G. Collisionless Stellar Hydrodynamics as an Efficient Alternative to N-body Methods. *Monthly Notices of the Royal Astronomical Society*. 2013. vol. 428, no. 3. pp. 2674–2687. DOI: 10.1093/mnras/sts228.

3. Ardeljan N.V., Bisnovatyi-Kogan G.S., Kosmachevskii G.S., Moiseenko S.G. An Implicit Lagrangian Code for the Treatment of Nonstationary Problems in Rotating Astrophysical Bodies. *Astronomy and Astrophysics Supplement Series*. 1996. vol. 115. pp. 573–594.

4. Khoperskov S.A, Vasiliev E.O., Sobolev A.M., Khoperskov A.V. The Simulation of Molecular Clouds Formation in the Milky Way. *Monthly Notices of the Royal Astronomical Society*. 2013. vol. 428. pp. 2311–2320. DOI: 10.1093/mnras/sts195.

5. Fletcher A., Beck R., Shukurov A., Berkhuijsen E., Horellou C. Magnetic Fields and Spiral Arms in the Galaxy M51. *Monthly Notices of the Royal Astronomical Society*. 2014. vol. 412. pp. 2396–2416. DOI: 10.1111/j.1365-2966.2010.18065.x.

6. Kulikov I. GPUPEGAS: A New GPU-accelerated Hydrodynamic Code for Numerical Simulations of Interacting Galaxies. *The Astrophysical Journal Supplement Series*. 2014. vol. 214, id. 12. DOI: 10.1088/0067-0049/214/1/12.

7. Kulikov I.M., Chernykh I.G., Snytnikov A.V., Glinskiy B.M., Tutukov A.V. AstroPhi: A Code for Complex Simulation of Dynamics of Astrophysical Objects Using Hybrid Supercomputers. *Computer Physics Communications*. 2015. vol. 186. pp. 71–80. DOI: 10.1016/j.cpc.2014.09.004.

8. Kulikov I.M., Chernykh I.G., Glinskiy B.M. AstroPhi: a Hydrodynamical Code for Complex Modelling of Astrophysical Objects Dynamics by Means of Hybrid Architecture Supercomputers on Intel Xeon Phi Base. *Vestnik Yuzhno-Uralskogo gosudarstvennogo universiteta. Serija: Vychislitelnaja matematika i informatika* [Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering]. 2013. vol. 2, i. 4. pp. 57–79. DOI: 10.14529/cmse130405. (in Russian)

9. Glinskiy B., Kulikov I., Snytnikov A., Romanenko A., Chernykh I., Vshivkov V. Co-design of Parallel Numerical Methods for Plasma Physics and Astrophysics. *Supercomputing Frontiers and Innovations*. 2015. vol. 1, no. 3. pp. 88–98.

10. Vorobyov E., Recchi S., Hensler G. Stellar Hydrodynamical Modeling of Dwarf Galaxies: Simulation Methodology, Tests, and First Results. *Astronomy & Astrophysics*. 2015. vol. 579, id. A9. DOI: 10.1051/0004-6361/201425587.

11. Kulikov I., Chernykh I. Glinskiy B., Weins D., Shmelev A. Astrophysics Simulation on RSC Massively Parallel Architecture. *Proceedings – 2015 IEEE/ACM 15th International Symposium on Cluster, Cloud, and Grid Computing, CCGrid 2015*. 2015. pp. 1131–1134. DOI: 10.1109/ccgrid.2015.102.

12. Vshivkov V., Lazareva G., Snytnikov A., Kulikov I. Supercomputer Simulation of an Astrophysical Object Collapse by the Fluids-in-Cell Method. *Lecture Notes of Computer Science*. 2009. vol. 5698. pp. 414–422. DOI: 10.1007/978-3-642-03275-2_41.

13. Kulikov I., Chernykh I., Katysheva E., Protasov A., Serenko A. The Numerical Simulation of Interacting Galaxies by Means of Hybrid Supercomputers. *Bulletin NCC: Numerical Analysis*. 2015. vol. 17. pp. 17–33.

14. Popov M., Ustyugov S. Piecewise Parabolic Method on Local Stencil for Gasdynamic Simulations. *Computational Mathematics and Mathematical Physics*. 2007. vol. 47, no. 12. pp. 1970–1989. DOI: 10.1134/s0965542507120081.

15. Popov M., Ustyugov S. Piecewise Parabolic Method on a Local Stencil for Ideal Magnetohydrodynamics. *Computational Mathematics and Mathematical Physics*. 2008. vol. 48, no. 3. pp. 477–499. DOI: 10.1134/s0965542508030111.

16. Vshivkov V., Lazareva G., Snytnikov A., Kulikov I., Tutukov A. Hydrodynamical Code for Numerical Simulation of the Gas Components of Colliding Galaxies. *The Astrophysical Journal Supplement Series*. 2011. vol. 194, id. 47. DOI: 10.1088/0067-0049/194/2/47.

17. Godunov S., Kulikov I. Computation of Discontinuous Solutions of Fluid Dynamics Equations with Entropy Nondecrease Guarantee. *Computational Mathematics and Mathematical Physics*. 2014. vol. 54. pp. 1012–1024. DOI: 10.1134/s0965542514060086.

18. Vshivkov V., Lazareva G., Snytnikov A., Kulikov I., Tutukov A. Computational Methods for Ill-posed Problems of Gravitational Gasodynamics. *Journal of Inverse and Ill-posed Problems*. vol. 19, no. 1. pp. 151–166. DOI: 10.1515/jiip.2011.027.

19. Kulikov I., Vorobyov E. Using the PPML Approach for Constructing a Low-Dissipation, Operator-Splitting Scheme for Numerical Simulations of Hydrodynamic Flows. *The Journal of Computational Physics*. 2016. vol. 317. pp. 318–346. DOI: 10.1016/j.jcp.2016.04.057.

20. Kulikov I., Chernykh I., Snytnikov A., Protasov V., Tutukov A., Glinsky B. Numerical Modelling of Astrophysical Flow on Hybrid Architecture Supercomputers. *In Parallel Programming: Practical Aspects, Models and Current Limitations (ed. M. Tarkov)*. 2015. pp. 71–116.

21. Frigo M., Johnson S. The Design and Implementation of FFTW3. *Proceedings of the IEEE*. 2005. vol. 93, no. 2. pp. 216–231. DOI: 10.1109/jproc.2004.840301.

22. Kalinkin A., Laevsky Y., Gololobov S. 2D Fast Poisson Solver for High-Performance Computing. *Lecture Notes in Computer Science*. 2009. vol. 5698. pp. 112–120. DOI: 10.1007/978-3-642-03275-2_11.

23. Schive H., Tsai Y., Chiueh T. GAMER: a GPU-accelerated Adaptive-Mesh-Refinement Code for Astrophysics. *The Astrophysical Journal*. 2010. vol. 186. pp. 457–484. DOI: 10.1088/0067-0049/186/2/457.

24. Podkorytov D., Rodionov A., Sokolova O., Yurgenson A. Using Agent-Oriented Simulation System AGNES for Evaluation of Sensor Networks. *Lecture Notes of Computer Science*. 2010. vol. 6235. pp. 247–250. DOI: 10.1007/978-3-642-15428-7_24.

25. Jaffe Y.L., Smith R., Candlish G., Poggianti B.M., Sheen Y.-K., Verheijen M.A.W. BUDHIES II: A Phase-Space View of HI Gas Stripping and Star-Formation Quenching in Cluster Galaxies. *Monthly Notices of the Royal Astronomical Society*. 2015. vol. 448. pp. 1715–1728. DOI: 10.1093/mnras/stv100.

26. Vollmer B., Cayatte V., Balkowski C., Duschl W.J. Ram Pressure Stripping and Galaxy Orbits: The Case of the Virgo Cluster. *The Astrophysical Journal*. 2001. vol. 561. pp. 708–726. DOI: 10.1086/323368.

27. Cayatte V., Kotanyi C., Balkowski C., van Gorkom J.H. A Very Large Array Survey of Neutral Hydrogen in Virgo Cluster Spirals. 3: Surface Density Profiles of the Gas. *The Astronomical Journal*. 1994. vol. 107, no. 3. pp. 1003–1017. DOI: 10.1086/116913.

Анатолий Васильевич Панюков

(к 65-летию со дня рождения)

28 ноября 2016 г. исполняется 65 лет со дня рождения члена редколлегии журнала, заслуженного работника высшей школы, доктора физико-математических наук, профессора Панюкова Анатолия Васильевича.

Анатолий Васильевич Панюков родился в г. Копейске, в семье рабочих. До него в семье высшего образования не имел никто. В 1971 г. с отличием окончил дневное отделение Челябинского радиотехнического техникума по специальности «Радиоаппаратостроение» и сразу же был призван на службу в Советскую Армию. Служил в войсках дальней космической связи, демобилизован в звании сержанта. Трудовой стаж начался в июне 1970 г. с производственных практик на Челябинском радиозаводе. После службы в армии работал в должности радиоинженера в НИИ измерительной техники ПО «Полет» и без отрыва от производства обучался на специальности «Автоматика и телемеханика» вечернего отделения Челябинского политехнического института. В 1976 г., по настойчивой рекомендации доцентов кафедры прикладной математики В.А. Штрауса, М.М. Гольденберга и М.Р. Решетова был переведен на дневное отделение приборостроительного факультета, который в 1980 г. закончил с отличием по специальности «Прикладная математика» и был распределен на должность младшего научного сотрудника кафедры прикладной математики ЧПИ. К моменту окончания вуза имел 17 авторских свидетельств на изобретения, научные публикации, медаль Минвуза СССР за лучшую научную работу.

В 1980 г. начал работу под руководством Б.В. Пельцвергера над проектом САПР обустройства нефтяных и газовых месторождений Западной Сибири. В ходе этих работ были получены значимые результаты в области дискретной оптимизации и математического моделирования. В 1982 г. поступил в очную аспирантуру кафедры прикладной математики, которую успешно окончил в ноябре 1985 г. с завершением работы над диссертацией, вернувшись на должность младшего научного сотрудника кафедры. В мае 1986 г. защитил диссертацию на соискание ученой степени кандидата физико-математических наук в Институте кибернетики им. В.М. Глушкова АН УССР (Киев). С октября 1986 г. переведен на должность старшего преподавателя кафедры Высшей математики № 2, с ноября 1988 г. — доцента кафедры прикладной математики, с ноября 1990 г. переведен на должность доцента кафедры информатики факультета экономики и управления, в связи с ее организацией. Ученое звание доцента по кафедре прикладной математики присуждено в июле 1991 г. В период с 1994 г. по 2000 г. активно занимался профориентационной работой среди школьников в рамках программы «Шаг в будущее». Ученики М.И. Германенко, В.В. Горбик, Д.В. Будуев и Д.Н. Малов были лауреатами программы, а впоследствии успешно защити-

ли кандидатские диссертации. В октябре 1999 г. без отрыва от преподавательской работы защитил диссертацию на соискание ученой степени доктора физико-математических наук в ВЦ РАН (Москва). В марте 2000 г. года был избран на должность профессора кафедры информатики.

С 2000 г. активно занимался становлением и развитием фундаментального математического образования в Южно-Уральском государственном университете. В мае 2001 г. ему было предложено организовать и возглавить кафедру экономико-математических методов и статистики, для подготовки выпускников по специальности «Математические методы в экономике» и новой для вуза и региона специальности «Статистика», обучение опиралось на мощные математический и экономический блоки. На должном уровне обеспечивалось базовое образование по статистике. Как следствие, выпускники обеих специальностей — неоднократные победители всероссийских олимпиад по названным специальностям. С участием Анатолия Васильевича были организованы механико-математический факультет, а затем и факультет вычислительной математики и информатики, в составе которого кафедра находилась до реструктуризации университета. На кафедре была открыта и успешно функционировала аспирантура по специальностям 05.13.18 «Математическое моделирование, численные методы и комплексы программ», 05.13.17 «Теоретические основы информатики», 08.00.13 «Математические и инструментальные методы экономики». А.В. Панюков являлся членом диссертационных советов Д 122.298.02 при ЮУрГУ (с 2001 по 2007 гг.), Д 212.298.14 при ЮУрГУ (с 2008 г. по настоящее время), ДМ 212.298.18 при ЮУрГУ (с 2012 г. по настоящее время), ДМ 212.189.07 при ПермГУ (с 2009 по 2012 гг.). К 2016 г. кафедра имела в своем составе оборудованную силами кафедры многофункциональную лабораторию, методический кабинет, кабинет заведующего. Кафедра приобрела опыт организации учебного процесса по семи образовательным программам. За пятнадцать лет руководства кафедрой профессор А.В. Панюков смог создать ядро из высококвалифицированных и абсолютно надежных опытных преподавателей (А.А. Кощеев, И.В. Парасич, В.И. Дударева, Н.С. Колотова , Т.А. Макаровских (Панюкова), С.У. Турлакова, А.Д. Липенков и др.), привлечь на кафедру более 30 преподавателей, создать среди коллег атмосферу доброжелательности, желания заниматься наукой и педагогическим трудом. Двери его кабинета были всегда открыты как для преподавателей, так и для студентов. Многие выпускники стремились сохранить связь с кафедрой работая почасовиками или поступив в аспирантуру (А.А. Макаева (Остренок), Т.С. Лыкова, Е.С. Исакова, Е.А. Савицкий и др.).

В июле 2016 г. кафедра экономико-математических методов и статистики включена во вновь образовавшуюся в результате реструктуризации университета кафедру математического и компьютерного моделирования. Многие ученики Анатолия Васильевича - молодые амбициозные преподаватели кафедры получили заманчивые предложения работать во вновь организованных структурах вуза.

В настоящее время А.В. Панюков работает в должности профессора кафедры математического и компьютерного моделирования.

Энциклопедичность и фундаментальность знаний Анатолия Васильевича, умение войти практически в любую задачу позволили ему после защиты докторской диссертации [1] подготовить двух кандидатов технических наук (Д.В. Будуев, Д.Н. Малов), двух кандидатов экономических наук [4, 5], четырех кандидатов физико-математических наук (А.Т. Латипова, М.И. Германенко, В.А. Голодов, Р.Э. Шангин) осуществить научное консультирование двух докторов наук [2, 3].

Направления диссертационных работ учеников представляют развитие основных направлений его докторской диссертации.

1. Развитие неперебoрных методов решения комбинаторных задач.
2. Разработка и анализ решения обратных задач и неустойчивых проблем.
3. Приложения к решению задач естествознания, техники, экономики.
4. Исследование возможностей высокопроизводительных гетерогенных вычислительных систем для эффективного осуществления точной дробно-рациональной арифметики.

В результате выполнении работ по развитию теории и техники безошибочных дробно-рациональных вычислений получены следующие результаты.

1. Показана полиномиальная битовая сложность безошибочного решения задачи линейного программирования [11, 18].
2. Введена избыточная позиционная система счисления и доказана масштабируемость алгоритмов выполнения операции алгебраического сложения в этой системе [24].
3. Разработаны масштабируемые алгоритмы реализации безошибочных вычислений с применением гетерогенных вычислительных систем [24]. Данные алгоритмы реализованы в виде библиотеки классов. Библиотека дает потенциальную возможность для широкого применения разработанных методов решения обратных задач:
   (a) спектрально-статистического метода [16];
   (b) метода анализа моделей регрессии и авторегрессии на основе установленной связи взвешенного и обобщенного методов наименьших модулей [19];
   (c) метода интервального погружения [25].

Разработаны и исследуются три группы методов решения обратных задач.

1. Спектрально-статистический метод [16].
2. Метод анализа моделей регрессии и авторегрессии на основе установленной связи взвешенного и обобщенного методов наименьших модулей [19].
3. Метод интервального погружения [25].

Данные методы нашли применение в технологиях неразрушающего контроля [2], в технической диагностике [3], в геофизике [6], в экономических и социально-поведенческих исследованиях [13, 20].

Разработаны алгоритмы решения задачи Штейнера на сверхбольших разреженных графах [17]. Программная реализация алгоритмов включена в САПР «Нефть» и использовалась при разработке нефтяных и газовых месторождений Западной Сибири. В дальнейших исследованиях данные результаты были обобщены на случай задачи Вебера для древовидной сети [14] на случай задачи Вебера для сетей в виде k-дерева и сетей с ограниченной древовидной шириной [14]. Разработаны эффективные алгоритмы для задачи коммивояжера, проблемы маршрутизации для CAD/CAM систем технологической подготовки процессов раскроя [21, 22].

Анатолий Васильевич — профессор, активно работающий со студентами и аспирантами. В настоящее время он проводит все виды учебных занятий по дисциплинам «Методы оптимизации», «Теория игр и исследование операций» для студентов бакалавриата по направлению «Прикладная математика и информатика»; «Современные проблемы прикладной математики и информатики», «Непрерывные математические модели», «Теория принятия решений» и семинара по математическому и информационному обеспечению экономической деятельности для студентов магистратуры по направлению прикладная математика и информатика.

Литература

1. Панюков А.В. Модели решения задач построения и идентификации геометрического размещения (исследование, алгоритмы, применения): диссертация ... доктора физико-математических наук: 05.13.16 / Вычислительный центр имени А.А. Дородницына Российской академии наук. Москва, 1999. 252 л.

2. Валиев М.М. Математическое моделирование электромагнитных систем контроля качества ферромагнитных изделий: диссертация ... доктора технических наук: 05.13.18 / Южно-Уральский государственный университет. Челябинск, 2003. 259 л.

3. Тырсин А.Н. Робастная параметрическая идентификация моделей диагностики на основе обобщенного метода наименьших модулей: Диссертация ... доктора технических наук: 05.13.18 / Южно-Уральский государственный университет. Челябинск, 2007. 327 л.

4. Будина Е.С. Математические и инструментальные методы оценки рисков в розничном кредитовании на основе композиции статистического и экспертного подходов: диссертация ... кандидата экономических наук: 08.00.13 / Пермский государственный университет. Пермь, 2010. 185 л.

5. Тетин И.А. Модель конкурентного взаимодействия участников рынка страховых услуг в условиях цикла андеррайтинга: диссертация ... кандидата экономических наук: 08.00.13 / Пермский национальный исследовательский политехнический университет. Пермь, 2016. 128 л.

6. Панюков А.В., Будуев Д.В., Малов Д.Н. Системы пассивного мониторинга грозовой деятельности // Вестник Южно-Уральского государственного университета. Серия: Математика. Механика. Физика. 2003. Т. 8, № 4. С. 11–24.

7. Панюков А.В., Будуев Д.В. Алгоритм определения расстояния до местоположения молниевого разряда // Электричество. 2001. № 4. С. 10–21.

8. Latipova A.T., Panyukov A.V. Numerical Techniques for Finding Equilibrium In Von Neumann's Model // Computational Mathematics and Mathematical Physics. November 2008. Vol. 48, Iss. 11. P. 1999–2006. DOI: 10.1134/S0965542508110080.

9. Панюков А.В., Латипова А.Т. Оценка положения равновесия в модели Неймана при интервальной неопределенности исходных данных // Вестник Уфимского государственного авиационного технического университета. 2008. Т. 10, № 2. С. 150–153.

10. Панюков А.В., Германенко М.И. Безошибочное решение систем линейных уравнений // Вестник Южно-Уральского государственного университета. Серия: Математика. Физика. Химия. 2009. № 10. С. 33–45.

11. Panyukov A.V., Gorbik V.V. Using Massively Parallel Computations for Absolutely Precise Solution of the Linear Programming Problems // Automation and Remote Control. July 2016. Vol. 77, Iss. 7. P. 1208–1215. DOI: 10.1134/S0005117912020063.

12. Панюков А.В., Богушов А.К. Применение гетерогенных вычислительных систем для решения задачи идентификации параметров положения дипольного источника излучения // Вестник Пермского университета. Серия: Математика. Механика. Информатика. 2012. № 3(11). С. 17–22.

13. Панюков А.В., Коновалова Е.Д. Анализ эффективности адаптивности государственного регулирования к изменениям ситуаций на рынках с высокой степенью монополизации // Вестник Пермского университета. Серия: Экономика. 2012. № 3(1). С. 58–68.

14. Панюков А.В., Шангин Р.Э. Точный алгоритм решения дискретной задачи Вебера для k-дерева // Дискретный анализ и исследование операций. 2014. Т. 21, № 3. С. 78–94.

15. Panyukov A.V., Shangin R.E. Algorithm for the discrete Weber's problem with an accuracy estimate // Automation and Remote Control 2016. Vol. 77, Iss. 7. P. 1208–1215. DOI: 10.1134/S0005117916070079.

16. Panyukov A.V., Bogushov A.K. The Spectral Statistical Method for Determining the Location Parameters of a Dipole Source of Electromagnetic Radiation // Radiophysics and Quantum Electronics. 2016. Vol. 59, Iss. 4. P. 278–288. DOI: 10.1007/s11141-016-9696-4

17. Panyukov A.V. The Steiner Problem in Graphs: Topological Methods of Solution // Automation and Remote Control. March 2004. Vol65, Iss. 3. P. 439–448 DOI: 10.1023/B:AURC.0000019376.31168.20.

18. Панюков А.В. Представление суммы Минковского для двух полиэдров системой линейных неравенств // Вестник Южно-Уральского государственного университета. Серия: Математическое моделирование и программирование. 2012. № 40(299). С. 108–119.

19. Панюков А.В. Об устойчивом оценивании параметров авторегрессионных моделей на основе обобщенного метода наименьших модулей // Вестник НГУЭУ. 2015. № 4. С. 339–346.

20. Konovalova E.D., Panyukov A.V. Government Control of Stackelberg Equilibrium at Natural Monopoly // Vestnik of Ufa State Aviation Technical University. 2014. Vol. 18, No. 5. P. 73–78.

21. Makarovskikh T.A., Panyukov A.V. *AOE*-trails Constructing for a Plane Connected 4-regular Graph // Supplementary Proceedings of the 9th International Conference on Discrete Optimization and Operations Research and Scientific School (DOOR 2016). Vladivostok, Russia, September 19 - 23, 2016. CEUR Workshop Proceedings. 2016. Vol. 1623. P. 821–826.

22. Makarovskikh T.A., Panyukov A.V., Savitskiy E.A. Mathematical Models and Routing Algorithms for CAM of Technological Support of Cutting Processes // Proceedings of 8th IFAC Conference on Manufacturing Modelling, Management and Control(MIM 2016) Troyes, France, 28—30 June 2016. IFAC-PapersOnLine 49-12. 2016. Vol. 49, Iss. 12. P. 821–826. DOI: 10.1016/j.ifacol.2016.07.876.

23. Panyukov A.V. Estimation of the Location of an Arbitrarily Oriented Dipole Under Single-point Direction Finding // Journal of Geophysical Research: Atmospheres. 1996. Vol. 101, Iss. 10. P. 14977–14982. DOI: 10.1023/B:AURC.0000019376.31168.20

24. Panyukov A.V. Scalability of Algorithms for Arithmetic Operations in Radix Notation // Reliable Computing. 2015. Vol. 19, Iss. 4. P. 417–434.

25. Panyukov A.V. Computing Best Possible Pseudo-Solutions to Interval Linear Systems of Equations // Reliable Computing. 2013. Vol. 19, Iss. 2. P. 215–228.

В.И. Дударева

СВЕДЕНИЯ ОБ ИЗДАНИИ

Научный журнал «Вестник ЮУрГУ. Серия «Вычислительная математика и информатика» основан в 2012 году.

Свидетельство о регистрации ПИ ФС77-57377 выдано 24 марта 2014 г. Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций.

Журнал включен в Реферативный журнал и Базы данных ВИНИТИ; индексируется в библиографической базе данных РИНЦ. Журнал размещен в открытом доступе на Всероссийском математическом портале MathNet. Сведения о журнале ежегодно публикуются в международной справочной системе по периодическим и продолжающимся изданиям «Ulrich's Periodicals Directory».

Решением Президиума Высшей аттестационной комиссии Министерства образования и науки Российской Федерации журнал включен в «Перечень рецензируемых научных изданий, в которых должны быть опубликованы основные научные результаты на соискание ученой степени кандидата наук, на соискание ученой степени доктора наук» по следующим отраслям и группам специальностей: 05.13.00 – информатика, вычислительная техника и управление; 01.01.00 – математика; 25.00.00 – науки о Земле (№421).

Подписной индекс научного журнала «Вестник ЮУрГУ», серия «Вычислительная математика и информатика»: 10244, каталог «Пресса России». Периодичность выхода — 4 выпуска в год (февраль, май, август и ноябрь).

ПРАВИЛА ДЛЯ АВТОРОВ

1. Правила подготовки рукописей и пример оформления статей можно загрузить с сайта серии `http://vestnikvmi.susu.ru`. **Статьи, оформленные без соблюдения правил, к рассмотрению не принимаются.**
2. Адрес редакции научного журнала «Вестник ЮУрГУ», серия «Вычислительная математика и информатика»:

   Россия 454080, г. Челябинск, пр. им. В.И. Ленина, 76, ЮУрГУ, кафедра СП,

   ответственному секретарю Цымблеру М.Л.
3. Адрес электронной почты редакции: `vestnikvmi@susu.ru`
4. **Плата с авторов за публикацию рукописей не взимается, и гонорары авторам не выплачиваются.**