ПРИМЕНЕНИЕ ВЫСОКОПРОИЗВОДИТЕЛЬНЫХ ВЫЧИСЛЕНИЙ В ЗАДАЧАХ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ

УДК 004.272.2

М. В. Якобовский

ВЫЧИСЛИТЕЛЬНЫЙ ЭКСПЕРИМЕНТ НА МНОГОПРОЦЕССОРНЫХ СИСТЕМАХ: АЛГОРИТМЫ И ИНСТРУМЕНТЫ

Обсуждаются проблемы, возникающие при проведении вычислительных экспериментов на многопроцессорных системах. Рассматриваются вопросы рациональной декомпозиции, огрубления триангулированных поверхностей, иерархической обработки и распределенной визуализации больших объемов сеточных данных.

Ключевые слова: *многопроцессорные системы, распределенная визуализация, рациональная декомпозиция, математическое моделирование, вычислительный эксперимент, параллельные алгоритмы, триангуляция, огрубление данных.*

Введение. Современные многопроцессорные системы обладают производительностью более 10^{14} операций с плавающей точкой в секунду, что потенциально позволяет за короткое время выполнять большие объемы вычислений. С помощью суперкомпьютеров возможно проведение вычислительных экспериментов, направленных на решение фундаментальных и прикладных задач газовой динамики, горения, микро- и наноэлектроники, экологии и многих других. Несмотря на то что потребность в проведении подобных экспериментов велика, большая часть современных суперкомпьютеров востребована далеко не в полной мере. Тому есть весомые причины.

Фундаментальной проблемой создания методов, позволяющих эффективно использовать совокупную мощность множества процессоров, является необходимость разработки алгоритмов, обладающих существенным запасом внутреннего параллелизма на всех этапах расчета. Каждый шаг решения задачи должен содержать достаточное количество взаимно независимых операций, выполнение которых возможно одновременно на всех выделенных для расчета процессорах. В идеале *все* множество операций, необходимых для решения задачи, следует равномерно распределить между *всеми* процессорами на протяжении *всего* времени выполнения расчета. Указанная проблема, безусловно, имеет определяющий характер, но она далеко не единственная.

Рост вычислительной мощности суперкомпьютеров обеспечивается сегодня за счет экстенсивного увеличения числа обрабатывающих элементов — процессоров, процессорных ядер, мультитредовых устройств и им подобных (далее — процессоров). Увеличение вычислительной мощности за счет роста числа поддерживаемых потоков команд, а не за счет скорости обработки одного потока, обусловливает несоответствие между возможностями тради-

ционных средств подготовки и анализа данных и способностью многопроцессорных систем к генерации больших массивов результатов.

Одним из наиболее активно используемых методов изучения процессов, протекающих в сложных многомерных объектах, является метод математического моделирования. Часто этот метод является единственной возможностью изучения сложных нелинейных явлений. В силу естественных временных ограничений невозможен натурный эксперимент при изучении глобальных изменений климата. В соответствии с действующими международными соглашениями невозможен натурный эксперимент в области изучения поведения вещества в экстремальных условиях ядерного взрыва. Дорог и практически невоспроизводим натурный эксперимент, имеющий своей целью определение оптимальных режимов добычи углеводородов на нефтяных месторождениях. Список можно продолжить, но принцип ясен — там, где натурный эксперимент невозможен, необходим эксперимент вычислительный. В его рамках создается математическая модель изучаемого явления. Использование методов математической физики приводит к описанию объекта исследования системой нелинейных многомерных уравнений в частных производных, решение которых определяется с помощью численных методов. Непрерывная среда заменяется дискретным аналогом — конечной сеткой по времени и пространству. Дифференциальные уравнения, действующие в непрерывном пространстве, заменяются алгебраическими, действующими в дискретном пространстве разностной сетки или конечных элементов. Решение алгебраических уравнений возлагается на суперкомпьютер.

Накоплен значительный опыт применения многопроцессорных систем для моделирования физических и технологических процессов, но он ориентирован в первую очередь на параллельные системы средней производительности, содержащие относительно небольшое число процессоров. При переходе к вычислительным системам, количество процессоров в которых исчисляется сотнями и более, требуется создание адекватных средств обработки больших объемов данных.

Увеличение числа процессоров, используемых для решения задачи, приводит к уменьшению объема вычислений, выполняемых на каждом из процессоров. В свою очередь, это ведет к относительному росту доли накладных расходов, обусловленных необходимостью обеспечения взаимодействия процессов передачи данных между процессорами и наличием других операций, обеспечивающих согласованное решение задачи на многопроцессорной системе. С ростом числа процессоров наступает насыщение — момент, после которого увеличение числа процессоров уже не приводит к сокращению времени решения задачи. Таким образом, за счет многопроцессорности сложно сокращать время решения задачи, но с помощью суперкомпьютеров можно решать более сложные задачи, увеличивая степень детализации изучаемых объектов, включая в рассмотрение дополнительные факторы, что влечет за собой увеличение объемов данных, описывающих эти объекты.

Оперирование большими объемами данных предъявляет новые требования и к используемым алгоритмам, и к самим принципам организации работы на суперкомпьютере. Работа становится невозможной без привлечения параллельных алгоритмов и средств, поддерживающих всю цепочку действий, требуемых для численного моделирования на подробных сетках, в том числе методов формирования геометрических моделей высокого качества, генераторов поверхностных и пространственных сеток, средств декомпозиции сеток, библиотек распределенного ввода—вывода, алгоритмов и библиотек балансировки загрузки процессоров, средств визуализации результатов крупномасштабных экспериментов и многих других.

Будем считать объем данных „большим“, если выполняются некоторые из следующих условий:

— вычислительной мощности любого отдельно взятого процессорного узла недостаточно для обработки всего объема данных за приемлемое время;

— оперативной памяти любого отдельно взятого процессорного узла недостаточно для хранения всего объема обрабатываемых данных;

— пропускной способности сетей передачи данных недостаточно для передачи за разумное время всего объема полученной информации от сервера хранения данных до рабочего места пользователя.

Рассмотрим в рамках настоящей статьи проблемы рациональной декомпозиции сеток и визуализации сеточных данных.

Декомпозиция данных. Будем руководствоваться широко использующимся на практике методом геометрического параллелизма, в соответствии с которым элементы расчетной сетки и ассоциированные с ними сеточные данные равномерно распределяются по процессорам компактными группами. За счет того что каждый процессор обрабатывает размещенные на нем данные, вычислительная нагрузка так же равномерно распределяется по процессорам.

Можно сформулировать первый критерий разбиения сеток: данные следует распределить так, чтобы объемы вычислительной работы, выполняемой на каждом процессоре, были одинаковыми.

Второй критерий связан с необходимостью минимизации сопровождающих расчет накладных расходов. Распределим сеточные данные так, чтобы сократить объемы передаваемых между процессорами в ходе вычислений данных. В контексте рассматриваемых задач на суперкомпьютер возлагается задача решения систем алгебраических уравнений. Есть два основных метода построения таких систем — на основе явных и неявных схем. Использование явных схем выглядит наиболее привлекательными, поскольку не требует привлечения дорогостоящих в вычислительном плане алгоритмов обращения матриц большого размера. С точки зрения описания нестационарных физических процессов явные схемы также имеют преимущество, поскольку они естественным образом отражают связь между значениями физических переменных, соответствующими последовательно моделируемым моментам времени. Системы алгебраических уравнений, соответствующие явным разностным схемам, обладают важным свойством локальности, позволяющим эффективно использовать многопроцессорные системы с распределенной памятью. Оно заключается в том, что для вычисления новых значений физических величин в некотором узле сетки необходимо знать о значениях сеточных функций только в тех узлах, что находятся на небольшом (по числу ребер) расстоянии от обрабатываемого узла. В первом приближении это означает, что объем данных, передаваемых между процессорами, будет определяться числом узлов, соседствующих с узлами, хранящимися на других процессорах.

Теперь можно сформулировать второй критерий: данные следует распределить так, чтобы минимизировать число ребер, соединяющих узлы, хранящиеся на разных процессорах.

Таким образом, задача рациональной декомпозиции сетки сводится к разбиению вершин графа сетки на заданное число классов эквивалентности — доменов. Требуется найти такой вариант разбиения сетки на домены, содержащие одинаковое количество вершин, при котором число ребер, соединяющих вершины из разных доменов (число „разрезанных“ ребер), минимально. Задача рационального разбиения графов принадлежит классу NP-полных, что заставляет использовать для ее решения эвристические алгоритмы.

Существует ряд пакетов декомпозиции графов, среди которых выделяются ParMETIS, CHACO, PARTY, JOSTLE и SCOTCH. Наиболее эффективны иерархические методы рационального разбиения графов (B. Hendrickson, R. Leland, G. Karypis, V. Kumar и др.) [1—3]. В их основе лежит следующая последовательность действий: огрубление графа (построение последовательности уменьшающихся в размере вложенных графов), начальная декомпозиция огрубленного графа на заданное число доменов, восстановление графа и локальное уточнение границ доменов.

Огрубление графа выполняется путем многократного объединения пар соседних вершин в одну, в результате чего формируется граф с небольшим числом агрегированных вершин, каждой из которых соответствует подмножество вершин исходного графа.

Начальная декомпозиция может быть выполнена с помощью практически любого алгоритма, вплоть до алгоритма полного перебора, поскольку число вершин огрубленного графа может быть сокращено до достаточно малого значения. Но, как правило, качество такой декомпозиции будет низким, поскольку оно непосредственно зависит от равномерности процесса огрубления графа. При разбиении огрубленного графа практически неизбежно формируются домены несовпадающих весов, так как веса агрегированных вершин могут иметь значительный разброс. В связи с этим необходимо выполнять локальное уточнение границ доменов, что позволяет уравновесить веса доменов и уменьшить число ребер, пересекающих их границы, например, с помощью алгоритмов Kernighan-Lin (KL) и Fiduccia-Mattheyses (FM) [4], обладающих относительно низкой вычислительной сложностью.

Для ряда задач актуальны два дополнительных критерия декомпозиции:

— минимизация максимальной степени домена (число соседних доменов) с целью снижения числа актов обмена данными на каждом шаге по времени и уменьшения сложности вычислительного алгоритма;

— обеспечение связности каждого из описывающих домен подграфов.

Перечисленные ранее алгоритмы не позволяют контролировать связность и максимальную степень доменов. Одним из методов, обеспечивающих формирование начального приближения высокого качества, является алгоритм спектральной бисекции. Применяя его рекурсивно, можно разбивать граф на произвольное число частей. Декомпозиция графа на две части может быть выполнена с помощью упорядочения вершин графа по значениям компонент вектора Фидлера — собственного вектора, соответствующего наибольшему ненулевому собственному значению спектральной матрицы графа (матрицы Лапласа) [3, 5, 6]. Разбиение графа на большее число частей, согласно компонентам вектора Фидлера, может быть использовано для формирования доменов, имеющих малое число соседей. Существенным недостатком метода является сложность определения компонент вектора Фидлера вырожденной спектральной матрицы графа, что ограничивает число вершин разбиваемого графа.

Для обеспечения связности подграфов каждого из доменов может быть использован инкрементный метод [7]. Определив глубину произвольной вершины как кратчайшее расстояние от нее до множества граничных вершин домена, можно ввести понятие ядра заданного уровня. Ядро уровня k определим как подграф, образованный вершинами глубиной не менее k и ребрами, инцидентными только этим вершинам. Алгоритм инкрементного роста ориентирован на формирование доменов, в каждом из которых ядра заданного уровня связны, что является более существенным условием, чем требование связности каждого из доменов. Тестирование показывает, что использование инкрементного алгоритма обеспечивает и малое число разрезанных ребер, и высокий уровень связности ядер доменов.

Иерархическая обработка больших сеток. Моделирование на многопроцессорных вычислительных системах центров коллективного пользования сопряжено с выполнением длительных вычислений с помощью большого количества сравнительно коротких сеансов. Число используемых процессоров может изменяться от одного сеанса к другому, что вынуждает многократно решать задачу балансировки загрузки. Несмотря на высокую эффективность иерархических алгоритмов, разбиение нерегулярных расчетных сеток большого размера занимает значительное время. Для снижения потерь целесообразно использовать иерархический метод хранения и обработки больших сеток. В соответствии с ним расчетная сетка предварительно разбивается на множество блоков небольшого размера — микродоменов. В дальнейшем сетка хранится в виде набора микродоменов и графа, определяющего их взаимосвязи, — макрографа. Каждая из вершин макрографа соответствует микродомену. Перед

очередным сеансом расчета производится разбиение макрографа, и каждому процессору назначается список микродоменов. Поскольку в макрографе значительно меньше вершин, чем в исходной сетке, его разбиение требует меньшего времени и может быть выполнено последовательными алгоритмами. Дополнительный выигрыш по времени достигается за счет возможности распределенного хранения больших графов как совокупности микродоменов, что значительно уменьшает накладные расходы на чтение и запись сеток большим числом процессоров.

Распределенная визуализация. С ростом числа процессоров, используемых для проведения вычислительных экспериментов, сложность задачи преобразования больших объемов сеточных данных к виду, пригодному для наглядного отображения, существенно возрастает. Для визуализации уже недостаточно ресурсов персональных компьютеров (ПК) — рабочих мест пользователей [8—10]. Наиболее естественным путем решения проблемы является использование технологии клиент—сервер, в соответствии с которой (рис. 1):

— серверная часть системы визуализации, как правило, выполняемая на многопроцессорной системе, обеспечивает обработку большого объема данных;

— клиентская часть системы визуализации, выполняясь на ПК, обеспечивает интерфейс взаимодействия с пользователем и непосредственное отображение данных, подготовленных сервером, используя при этом аппаратные возможности персонального компьютера как для построения наглядных визуальных образов (с помощью графических ускорителей, стереоустройств), так и для управления ими (с помощью многомерных манипуляторов).

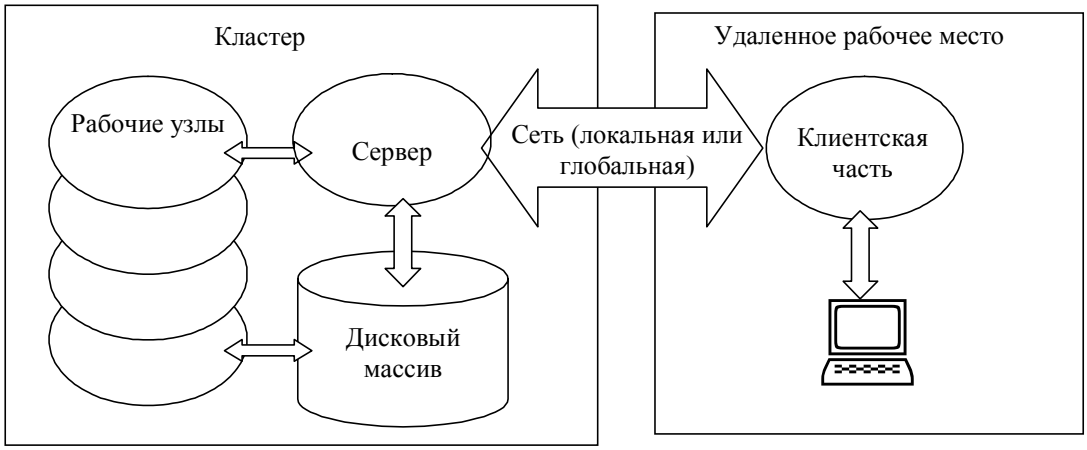

Рис. 1

На сервере целесообразно подготавливать данные, обеспечивающие формирование клиентом именно трехмерного образа объекта, манипулирование которым с целью его изучения с различных направлений возможно уже без дополнительных обращений к серверу. В этом заключается одно из существенных отличий обсуждаемого подхода от методов, используемых в большинстве доступных систем визуализации, выполняющих на сервере „рендеринг“ — формирование двумерного растрового образа изучаемого объекта.

Одним из наиболее мощных и наглядных методов визуализации трехмерных скалярных данных является визуализация изоповерхностей, описываемых множеством треугольников. Современные видеоускорители аппаратно поддерживают отображение массивов треугольников, что значительно повышает наглядность визуализации как за счет высокой скорости вывода данных на экран, так и за счет возможности автоматического формирования стереоизображений.

Ядро системы визуализации представлено рядом алгоритмов фильтрации и огрубления первичных данных, позволяющих аппроксимировать результаты вычислений ограниченным объемом данных, достаточным, однако, для восстановления изучаемого образа с заданным уровнем качества. Фактически к алгоритму огрубления предъявляются два требования, вызывающие потребность обеспечить интерактивный режим работы, а следовательно — высокую

скорость предоставления пользователю визуального образа. За короткое время необходимо огрубить данные и передать результат на компьютер пользователя. Следовательно, во-первых, алгоритм огрубления должен работать быстро. Во-вторых, необходимо выполнить огрубление данных *до объема*, заданного временем, отведенным на передачу данных на компьютер пользователя. Число описывающих изоповерхность узлов велико и по порядку величины может совпадать с числом узлов исходной трехмерной сетки, поэтому и необходим этап ее огрубления до размеров, допускающих их передачу через медленные каналы связи за короткое время. Необходимые коэффициенты „сжатия“ — отношения объемов данных, описывающих исходную изоповерхность и ее образ, достигают сотен тысяч и более, поэтому следует использовать методы сжатия с потерей точности. В первую очередь визуально воспринимаются основные контуры и формы трехмерного объекта, спроецированного на двумерный экран, следовательно, при изучении объекта „в целом“ деталями можно пожертвовать. При необходимости фрагменты объекта можно изучить с большим увеличением и меньшей потерей точности.

Простейшие алгоритмы сжатия, основанные на снижении точности представления вещественных чисел, описывающих сетку, и последующей компрессии с помощью стандартных алгоритмов, подобных групповому кодированию (*RLE*) или кодированию строк (*LZW*), значительного выигрыша не дают. Большую часть объема данных о триангуляции составляет целочисленная информация, описывающая ее топологию — связи между узлами. Стандартными алгоритмами сжатия без потерь эта информация практически не обрабатывается. Таким образом, основной интерес представляют алгоритмы, формирующие некоторую новую триангулированную поверхность, аппроксимирующую исходную изоповерхность, но содержащую значительно меньшее количество узлов.

Огрубление триангулированных поверхностей. Эффективны алгоритмы огрубления триангулированных поверхностей на основе методов редукции [11, 12]. Их основная идея заключается в итерационном удалении из исходной поверхности некоторого количества узлов таким образом, чтобы триангуляция, определенная на оставшихся точках, аппроксимировала исходную поверхность с требуемой точностью. В качестве иллюстрации на рис. 2, *а* представлена триангулированная сфера (точек 98 880, треугольников 195 884), а на рис 2, *б* — результат огрубления сферы с помощью редукции (точек 215, треугольников 375). Методы редукции допускают огрубление изоповерхностей, обладающих достаточно сложной структурой, например, имеющих самопересечения.

а) *б*)

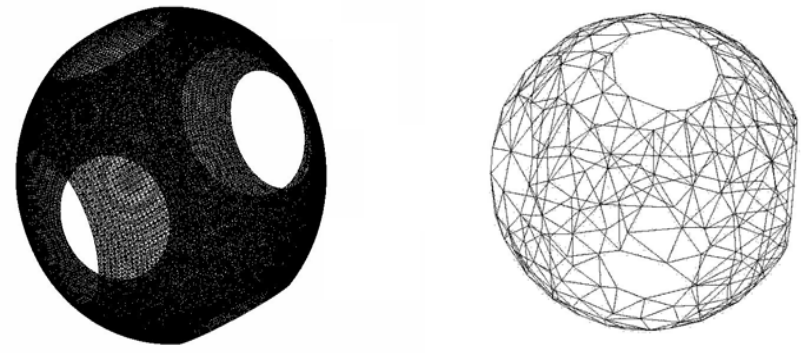

Рис. 2

Параллельный алгоритм построения и огрубления изоповерхностей основан на методе геометрического параллелизма. Каждый процессор формирует часть изоповерхности, проходящую через обрабатываемый процессором фрагмент сетки — домен. Результаты, полученные при огрублении фрагментов изоповерхности, передаются на один процессор. Однако с помощью такого алгоритма невозможно обработать произвольно большой объем данных. Суммарный объем данных, описывающих огрубленные на процессорах фрагменты, не может

превышать объема оперативной памяти того единственного процессорного узла, на котором выполняется окончательная обработка изоповерхности. Проблема решается разбиением процессоров на группы и введением дополнительных промежуточных этапов. После первого этапа огрубления результаты, полученные всеми процессорами одной группы, передаются на один из процессоров этой же группы. На каждом из таких процессоров выполняется второй этап огрубления, полученные результаты передаются на один процессор для окончательной обработки. При необходимости можно использовать каскадную схему, увеличив число этапов, что в принципе снимает ограничения на исходный объем обрабатываемых данных.

Каскадная схема позволяет получить дополнительный выигрыш — существенно улучшить качество аппроксимации поверхности. При частичном огрублении на каждом из процессоров присутствуют узлы двух типов: внутренние и граничные. Узел считается граничным, если множество опирающихся на него треугольников распределено по нескольким процессорам. Разрешено удаление только внутренних узлов, граничные не могут быть удалены, поскольку удаление разных узлов одной и той же границы разными процессорами может привести к возникновению разрывов триангулированной поверхности. При сильном огрублении внутренней части поверхности исключается большое число внутренних узлов. Поскольку все граничные узлы сохраняются, на огрубленной таким образом изначально гладкой поверхности возникают артефакты — изломы, описываемые большим числом точек, сглаживание которых на заключительном этапе затруднительно. Каскадная многоэтапная схема позволяет применять на каждом шаге меньшее сжатие внутренней части, что положительно сказывается на картине в целом. Результаты триангуляции передаются на персональный компьютер пользователя, где выполняется отображение поверхности.

На рис. 3, *а* приведена изоповерхность распределения плотности среды, полученная при моделировании обтекания летательного аппарата с использованием тетраэдральной сетки, содержащей более 2 млн узлов и 14 млн тетраэдров (визуализация разработанной системой RemoteViewer).

На рис. 3, *б* приведен результат визуализации изоповерхности плотности среды, полученный с помощью системы Tecplot, широко используемой для визуализации научных данных на персональных компьютерах. Сравнение рис. 3, *а* и *б* показывает хорошее совпадение формы изоповерхностей, полученных с помощью разных инструментов, что подтверждает высокое качество образов, формируемых описанными алгоритмами визуализации.

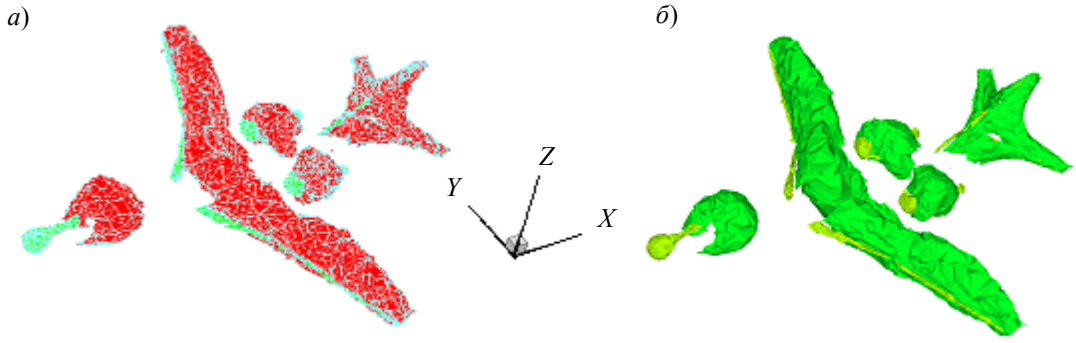

Рис. 3

Заключение. В заключение следует подчеркнуть, что рассмотренные в статье методы обработки результатов проводимых на суперкомпьютерах вычислительных экспериментов актуальны, если другие методы оказываются неприменимыми. Они актуальны, если используются объемы данных, для обработки которых недостаточно ресурсов ПК и недостаточно возможностей хорошо развитых и обладающих богатой функциональностью последовательных пакетов прикладных программ.

Работа выполнена при поддержке проекта РФФИ № 08-07-00458-а, 09-01-00292-а.

СПИСОК ЛИТЕРАТУРЫ

1. *Hendrickson B. and Leland R. A.* Multi-Level Algorithm for Partitioning Graphs // Tech. Rep. SAND93-1301, Sandia National Laboratories, Albuquerce. October 1993.

2. *Hendrickson B. and Leland R.* An Improved Spectral Graph Partitioning Algorithm for Mapping Parallel Computations // SIAM J. Sci. Comput. 1995. Vol. 16, N 2.

3. *Karypis G., Kumar V.* Multilevel Graph Partitioning Schemes // ICPP (3). 1995. P.113-122.

4. *Fiduccia C. and Mattheyses R.* A linear time heuristic for improving network partitions // Proc. 19th IEEE Design Automation Conf. 1982. P. 175—181.

5. *Fiedler M.* Eigenvectors of aciyclic matrices // Czechoslovak Mathematical J. 1975. Vol. 25(100). P. 607—618.

6. *Fiedler M.* A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory // Czechoslovak Mathematical J. 1975. Vol. 25(100). P. 619—633.

7. *Якобовский М. В.* Инкрементный алгоритм декомпозиции графов // Вестн. Нижегородского Университета им. Н. И. Лобачевского. Сер. Математическое моделирование и оптимальное управление. Вып. 1(28). Н. Новгород: Изд-во ННГУ, 2005. С. 243—250.

8. *Iakobovski M. V., Karasev D. E., Krinov P. S., Polyakov S. V.* Visualisation of grand challenge data on distributed systems // Proc. Symp. Mathematical Models of Non-Linear Excitations, Transfer, Dynamics, and Control in Condensed Systems and Other Media. Moscow—London: Plenum Publishers, 2001. P. 71—78.

9. *Якобовский М. В.* Обработка сеточных данных на распределенных вычислительных системах // Вопр. атомной науки и техники. Сер. Математическое моделирование физических процессов. 2004. Вып. 2. С. 40—53.

10. *Iakobovski M., Nesterov I., Krinov P.* Large distributed datasets visualization software, progress and opportunities // Computer Graphics & Geometry. 2007. Vol. 9, N 2, P. 1—19.

11. *Krinov P.S., Iakobovski M.V., Muravyov S.V.* Large Data Volume Visualization on Distributed Multiprocessor Systems // Parallel Computational Fluid Dynamics: Advanced numerical methods software and applications. Proc. of the Parallel CFD 2003 Conf. Moscow, Russia. Amsterdam: Elsevier, 2004. P. 433—438.

12. *Кринов П.С., Якобовский М.В., Муравьёв С.В.* Визуализация данных большого объема в распределенных многопроцессорных системах // Высокопроизводительные параллельные вычисления на кластерных системах. Мат. 3-го Междунар. науч.-практич. семинара. Н. Новгород: Изд-во ННГУ, 2004. С. 81—88.

Сведения об авторе

Михаил Владимирович Якобовский — д-р физ.-мат. наук, профессор; Институт математического моделирования РАН, сектор программного обеспечения вычислительных систем и сетей, Москва; зав. сектором; E-mail: lira@imamod.ru

Рекомендована институтом

Поступила в редакцию
10.03.09 г.