УДК 519.6

ScientificView — ПАРАЛЛЕЛЬНАЯ СИСТЕМА ПОСТОБРАБОТКИ РЕЗУЛЬТАТОВ, ПОЛУЧЕННЫХ ПРИ ЧИСЛЕННОМ МОДЕЛИРОВАНИИ ФИЗИЧЕСКИХ ПРОЦЕССОВ

А. Л. Потехин, В. И. Тарасов, С. А. Фирсов, И. В. Логинов,
В. А. Никитин, М. Г. Кузнецов, Н. В. Попова, А. К. Деманова, Ю. В. Козачек
(РФЯЦ-ВНИИЭФ)

Описывается параллельная система постобработки ScientificView, реализованная в математическом отделении РФЯЦ-ВНИИЭФ. Система предназначена для фильтрации, отображения и табличного анализа результатов моделирования физических процессов на разностных сетках регулярного типа.

Приведены графики изменения времени работы некоторых алгоритмов обработки в зависимости от числа используемых процессоров.

Введение

Мощные многопроцессорные ЭВМ позволяют более детально описывать физические процессы за счет применения более точных моделей, описываемых уравнениями математической физики, при решении которых разностными методами требуются существенные вычислительные ресурсы и большое количество счетных точек. Для оценки качества проведенных расчетов необходимы развитые средства анализа и постобработки, в частности системы визуализации, способные быстро обрабатывать большой объем данных. Добиться этого с помощью программ, выполняемых в однопроцессорном режиме, затруднительно по причине ограниченности ресурсов ПЭВМ.

В математическом отделении РФЯЦ-ВНИИЭФ накоплен опыт использования параллельных систем визуализации, в частности, проводится адаптация системы ParaView [1] для обработки данных, рассчитанных с помощью применяемых в отделении математических методик. Работы по адаптации показали, что, несмотря на имеющийся большой инструментарий ParaView, реализация в нем алгоритмов, учитывающих особенности методик, связана с большими трудозатратами (например, поддержка адаптивно встраиваемых сеток), либо просто невозможна (например, поддержка обработки неструктурированных сеток с ячейками произвольной формы без их дробления на примитивы ParaView). Внедрение в ParaView для полноценной обработки расчетных данных неграфических средств (табличного просмотра, расчета интегральных характеристик и т. д.), также связано с большими трудозатратами. Поэтому было решено создать параллельную систему постобработки ScientificView на основе собственных программных продуктов, а именно системы визуализации программы расчета начальных данных 3D-РНД [2], программы табличной обработки данных EFR-Viewer [3], библиотеки ЕФР [4] для доступа к данным на диске.

Выбор схемы построения системы

Наиболее широко используются три схемы построения параллельных систем постобработки, в которых в параллельном режиме функционируют модули чтения и процедуры фильтрации данных. Отличие схем заключается в выполнении процедур подготовки изображения: они могут быть распараллелены (схема с максимальным распараллеливанием), выполняться в однопроцессорном режиме на ПЭВМ (схема клиент-сервер) или на одном процессе, принадлежащем параллельному полю (схема удаленного клиента).

При реализации системы на основе программ, разработанных в математическом отделении

РФЯЦ-ВНИИЭФ, было решено использовать схему клиент-сервер с клиентской частью, функционирующей под операционной системой (ОС) Windows, и серверной частью, запущенной под ОС Linux в параллельном режиме на многопроцессорной ЭВМ с распределенной памятью (рис. 1).

Эта схема оказалась проще в реализации, поскольку для создания клиентской части (интерфейса пользователя, системы отображения) достаточно было выделить соответствующие процедуры из системы визуализации комплекса 3D-РНД. Серверная часть была создана за счет выделения в отдельное кросс-платформенное приложение и распараллеливания процедур чтения и обработки данных. Несмотря на простоту, реализация системы по данной схеме позволила увеличить объемы и скорость обработки расчетных данных по сравнению с программами, выполняемыми на ПЭВМ в однопроцессорном режиме.

Связь клиента и серверной части осуществляется посредством сокетов по протоколам TCP/IP. Для этого среди процессов серверной части выделяется специальный *управляющий* процесс. Управляющий процесс осуществляет прием запросов от клиента, распределяет нагрузку по данному запросу на остальные (*подчиненные*) процессы сервера, собирает от них результаты обработки.

Управляющий процесс не выполняет чтения и обработки сеточных данных, поскольку это приводит к замедлению работы всей серверной части. В будущем, при реализации процедур формирования изображения в параллельном режиме, каждый подчиненный процесс будет формировать часть изображения по имеющимся у него данным, а итоговое изображение предполагается формировать на управляющем процессе.

Поскольку связь клиента на ПЭВМ осуществляется лишь с одним процессом серверной части, многопроцессорная ЭВМ может быть неоднородной и не для всех ее узлов обязательно существование прямого соединения с ПЭВМ пользователя.

Структуры и потоки данных

Необходимо было разработать структуры и потоки данных, позволяющие с минимальными трудозатратами провести первичную реализацию системы и упростить ее развитие. Поскольку речь идет о графической системе, под развитием в первую очередь понимается создание новых возможностей по обработке данных (различных алгоритмов фильтрации и анализа), а также возможностей по обработке данных новых типов (хранящихся в файлах новых форматов или полученных в результате моделирования ранее не рассматриваемых физических процессов). Разрабатываемая архитектура должна была также обеспечить кросс-платформенность программной реализации.

При проектировании были выделены несколько информационных уровней, каждый из которых представляется некоторой программной реализацией (рис. 2):

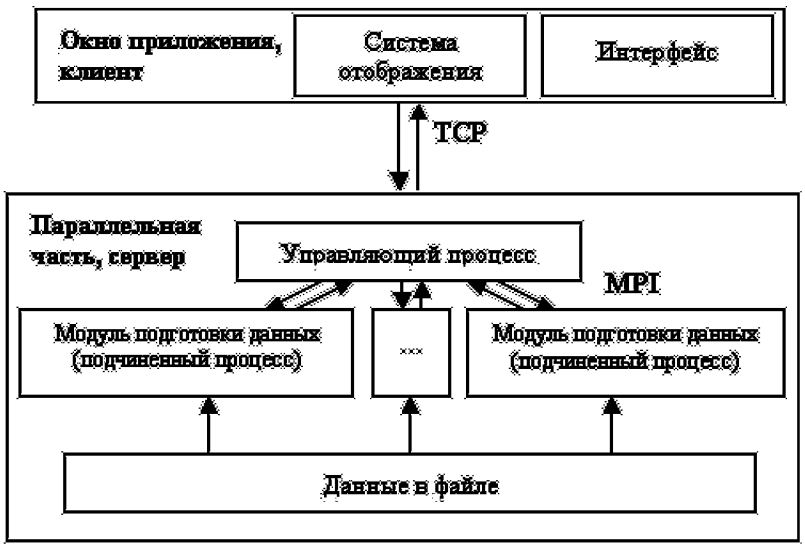

Рис. 1. Схема построения ScientificView

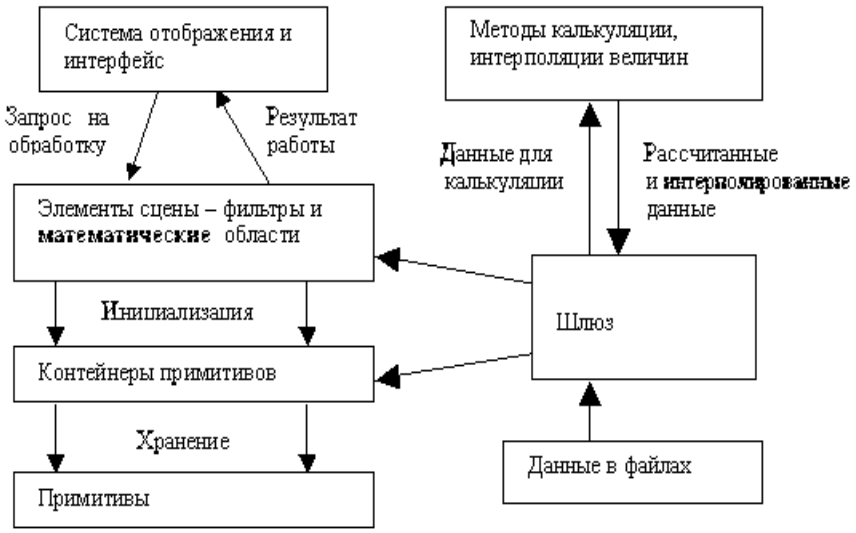

Рис. 2. Основные структуры и потоки данных

1. *Геометрические примитивы*: точки, векторы, отрезки, многоугольники. Каждый примитив хранит часть информации, принадлежащей либо исходным сеточным данным (математическим областям), либо результатам обработки (в первую очередь результатам фильтрации). Примитивы, как правило, самодостаточны и не нуждаются в информации о своем реальном окружении (в частности, одна точка ничего "не знает" о соседней).
2. *Контейнер примитивов.* Контейнер предназначен для хранения однотипных примитивов (например многоугольников), а также для их обработки: добавления, удаления, сортировки, поиска дубликатов и т. д. В каждом контейнере хранится информация о взаимосвязях находящихся в нем примитивов.
3. *Элементы сцены.* Под элементами сцены понимаются объекты, хранящие результаты работы фильтрующих алгоритмов или расчетные данные. Элементы сцены используют контейнеры для хранения своих геометрических примитивов.

Доступ к данным в файлах или оперативной памяти осуществляется через шлюзовую систему. Она же обеспечивает доступ к данным, полученным интерполяцией или калькуляцией по специальным формулам.

Взаимодействие пользователя с ScientificView организовано через интерфейсную часть и систему отображения, осуществляющую вывод графической информации.

Потоки данных, связывающие модули и информационные слои, обрабатываются ядром системы визуализации. Это позволяет при разработке новых алгоритмов воспользоваться ранее реализованными базовыми конструкциями, например процедурами инициализации объектов, получения данных из файла и отображения результатов обработки.

Процедуры обработки запросов

Взаимодействие клиентской и серверной частей приложения осуществляется на основе запросов, которые группируются в пакеты четырех типов.

Первый тип пакета — *информационный.* Он указывает принимающей стороне (как правило, серверу), какую операцию необходимо выполнить, а также размер данных, которые будут переданы в качестве параметров операции следующим типом пакета — *пакетом с данными.*

Содержимое пакета с данными зависит от проводимой операции, правила формирования пакета согласованы между отправляющей и принимающей сторонами.

Если после выполнения запрашиваемой операции результат обработки данных с серверной стороны будет передаваться на клиентскую сторону несколькими пакетами, то сначала сервер посылает клиенту *процедурный* пакет. Этот пакет содержит информацию, необходимую для организации приема обработанных данных. Сами данные передаются пакетом с данными.

На информационный и процедурный типы пакетов принимающая сторона отвечает *пакетом отклика*, что является подтверждением факта приема.

Пакеты обрабатываются процедурами собственной разработки. Простейший сценарий обработки запроса приведен на рис. 3.

Поскольку взаимодействие с клиентом осуществляет только управляющий процесс сервера, он должен разослать полученную от клиента информацию всем остальным процессам сервера, а также собрать результаты обработки данных для передачи клиенту. Это осуществляется за счет скрытого взаимодействия процессов серверной части при обработке запроса. Например, функция приема информационного пакета (см. рис. 3, п. 2 справа) на управляющем процессе выполняет передачу содержимого пакета на подчиненные процессы сервера с помощью интерфейса передачи сообщений MPI. Функция передачи процедурного пакета (см. рис. 3, п. 8 справа), вызванная на подчиненном процессе, передаст данные с помощью MPI управляющему процессу сервера, а тот, в свою очередь, — клиенту.

Реализация скрытого взаимодействия позволила использовать одинаковые алгоритмы обработки на управляющем и подчиненных процессах. На основе системы запросов выполнено распараллеливание основных алгоритмов обработки данных, реализованных в версии системы визуализации программы 3D-РНД для ПЭВМ.

Режимы отображения

Интерпретация значений величин цветом при отображении исходных сеточных данных и результатов их обработки возможна несколькими способами: с помощью заливки полигонов, цветной сетки или цветных узлов сетки, отображения внешних границ сетки (рис. 4, см. также цветную вкладку).

В качестве величины для интерпретации выступает любая величина, определенная в ячейках или узлах сетки (рис. 5, см. также цветную вкладку).

При отображении объектов возможно применение эффекта прозрачности, упрощающего анализ сеточных данных, находящихся "внутри" объекта (рис. 6, см. также цветную вкладку).

Некоторые алгоритмы графической обработки данных

В параллельном режиме реализованы классические алгоритмы построения векторных полей, сечений плоскостью и изоповерхностей (рис. 7, см. также цветную вкладку) [1, 5], а также алгоритмы, предназначенные для обработки данных на структурированных сетках: выделение слоя граней с постоянным значением одного из индексов (рис. 8,*а*, см. также цветную вкладку); выделение подобласти, заданной интервалами из-

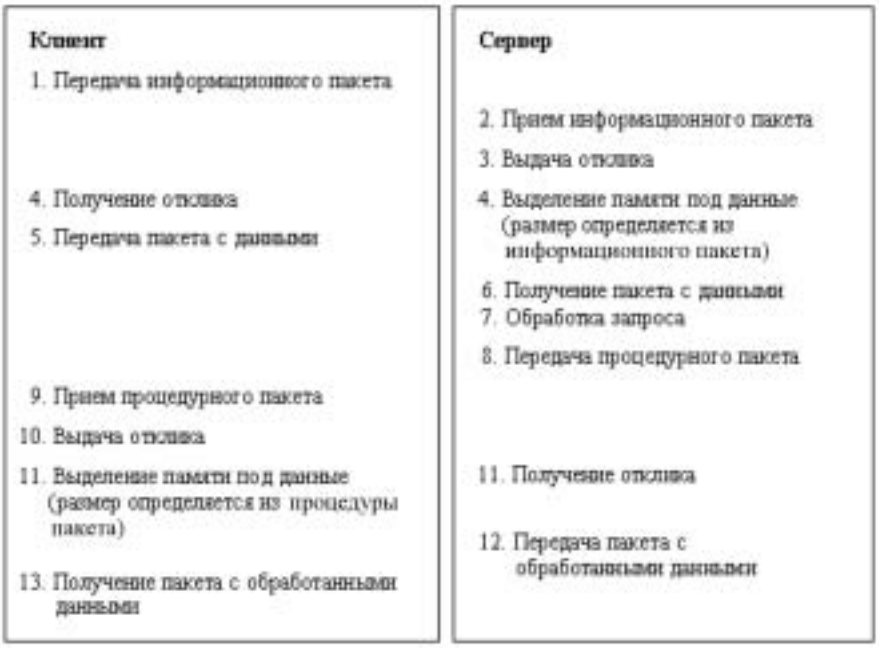

Рис. 3. Простейший сценарий обработки запроса

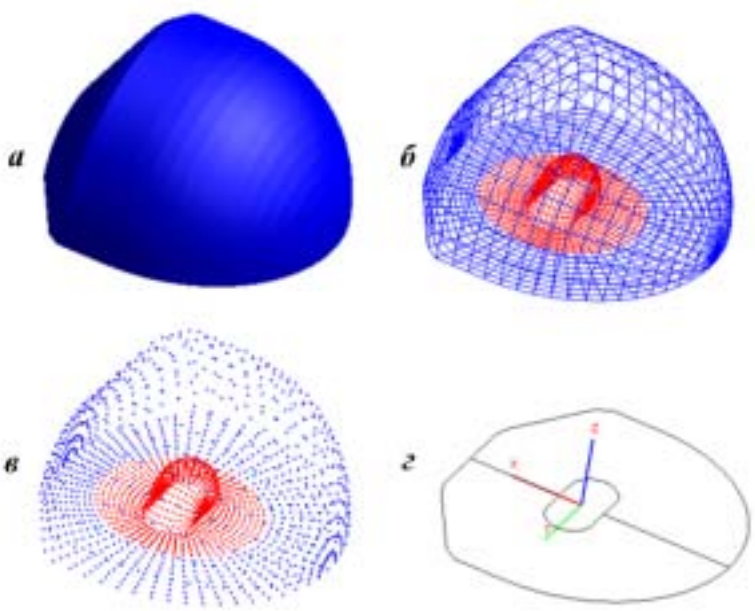

Рис. 4. Способы интерпретации величин цветом: *а* — заливка полигонов; *б* — цветная сетка; *в* — цветные узлы сетки; *г* — отображение границ

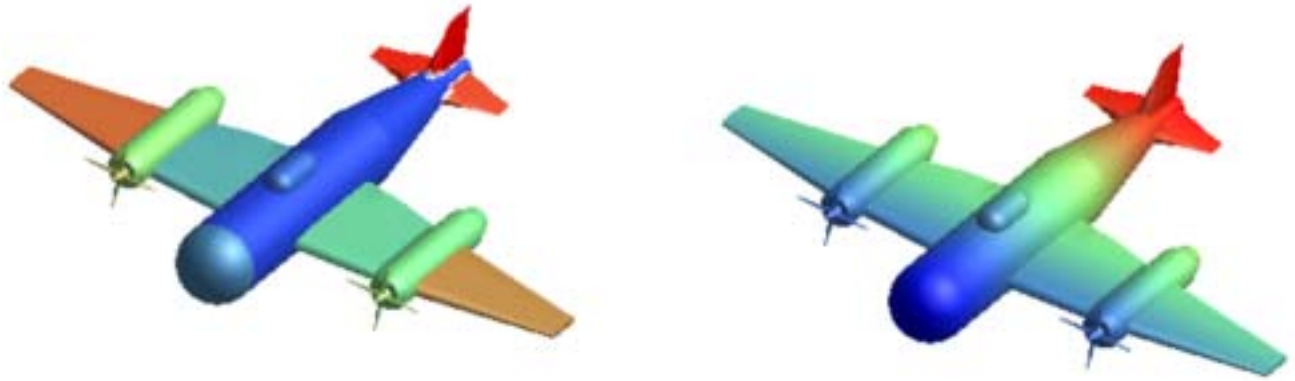

Рис. 5. Заливка многоугольников: слева — по величинам из ячеек (вещество); справа — по величинам из узлов сетки (координата X)

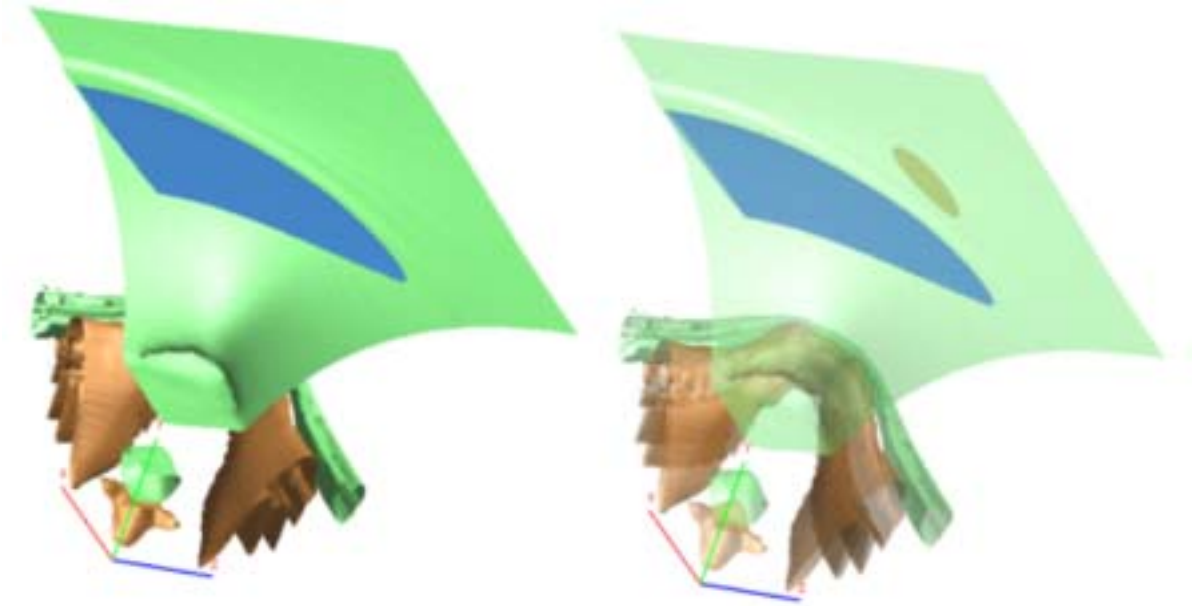

Рис. 6. Отображение изоповерхности: слева — обычное; справа — с применением прозрачности

менения индексов (рис. 8,*б*, см. также цветную вкладку).

Для анализа топологии сетки реализован алгоритм, наиболее часто применяемый при обработке данных на неструктурированных сетках (фильтр "Проба"). Он позволяет просмотреть окрестность выбранного узла или ячейки — номера граней и узлов примыкающих ячеек и т. д. (рис. 9, см. также цветную вкладку).

Для выделения ячеек или узлов, значения величин в которых находятся в заданном интервале, реализован фильтр "Интервал". Он, в частности, позволяет выделить ячейки с определенным веществом (рис. 10, см. также цветную вкладку).

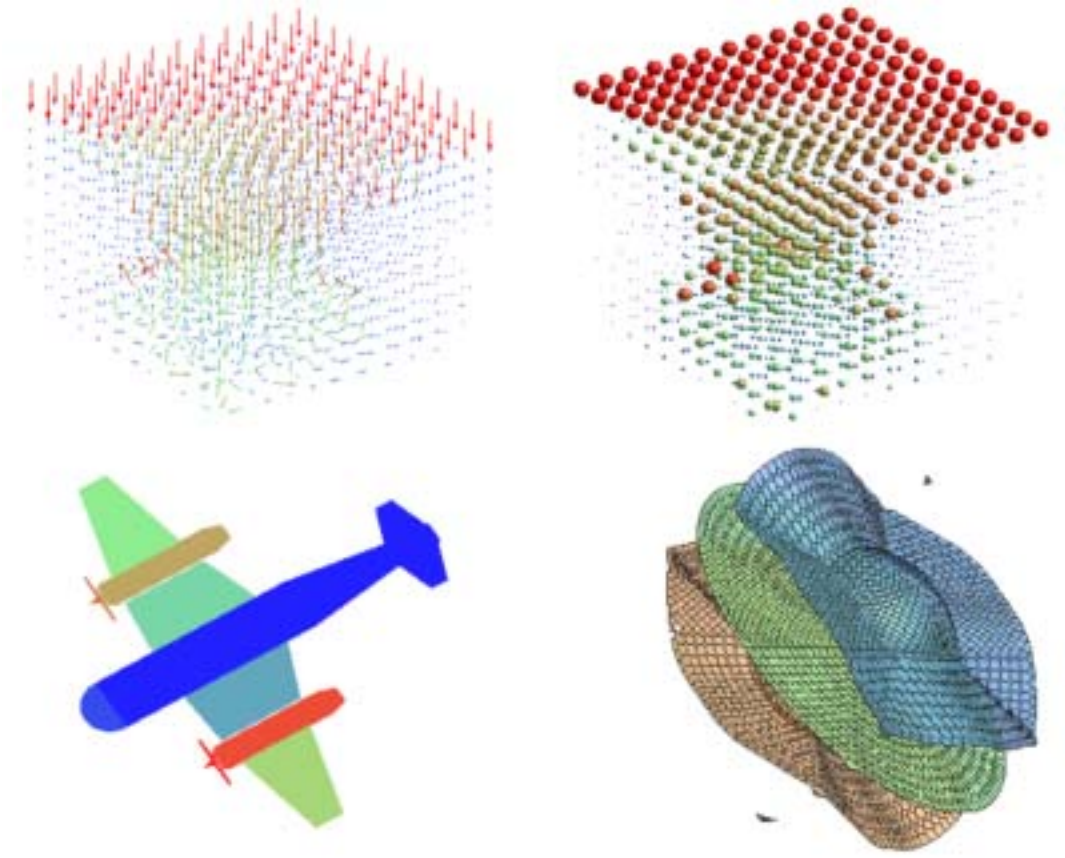

Рис. 7. Результат работы фильтров: сверху — "Векторное поле" (отображение в двух режимах); снизу — "Сечение" (слева), "Изоповерхность" (справа)

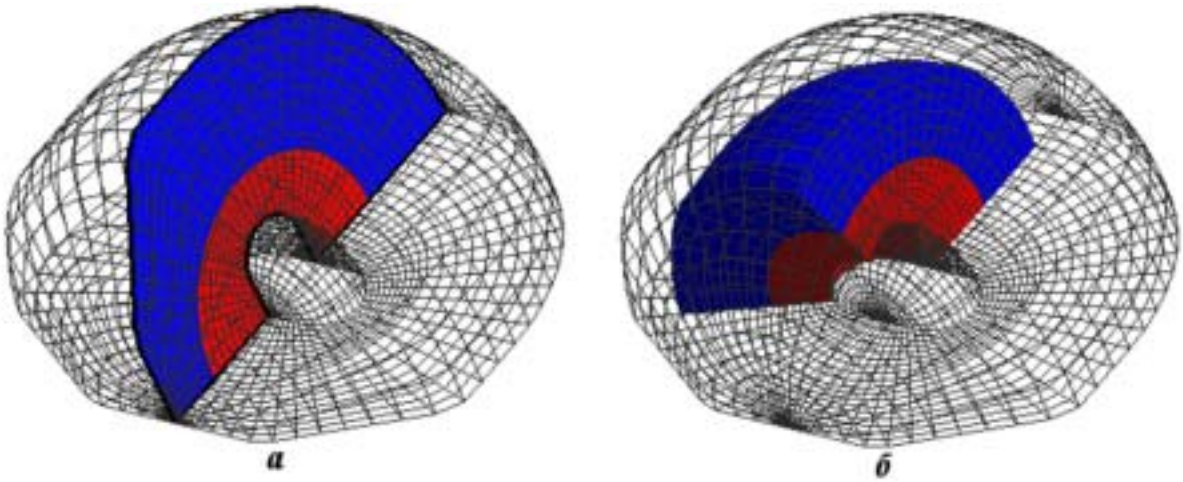

Рис. 8. Построение слоя граней (*а*) и подобласти со структурированной сеткой (*б*)

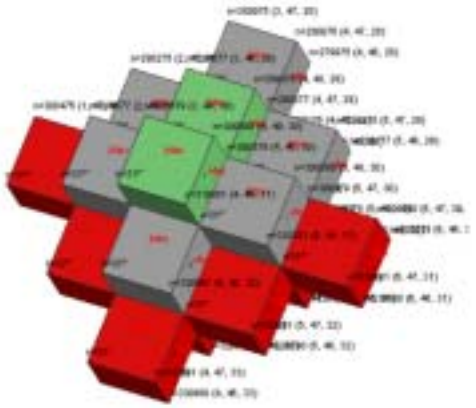

Рис. 9. Анализ топологии с помощью фильтра "Проба"

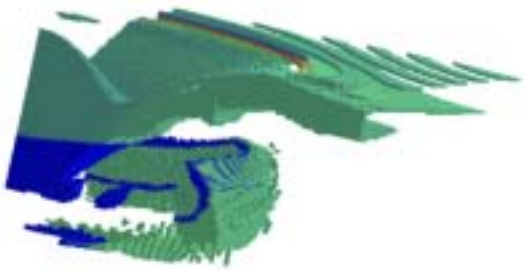

Рис. 10. Результат работы фильтра "Интервал"

В последних версиях системы появилась возможность *иерархической фильтрации*: применения одного алгоритма фильтрации к результатам работы другого. Это позволяет, например, выделить часть счетного пространства, построить там сечение и на нем построить векторное поле (рис. 11, см. также цветную вкладку).

Алгоритмы табличной обработки данных

Большинство алгоритмов табличной обработки изначально было реализовано в программе EFR-Viewer. Посредством гипертекстовых интерфейсов (рис. 12) имеется возможность:

- просмотра общих характеристик задачи, числа, типа и параметров математических областей;
- просмотра набора массивов, списков, структур, определенных для каждой области;
- расчета интегральных характеристик, сравнения балансов сеточных величин из двух файлов-разрезов;
- внесения изменений в любую составляющую файла-разреза.

Большинство из этих алгоритмов внедрено в ScientificView.

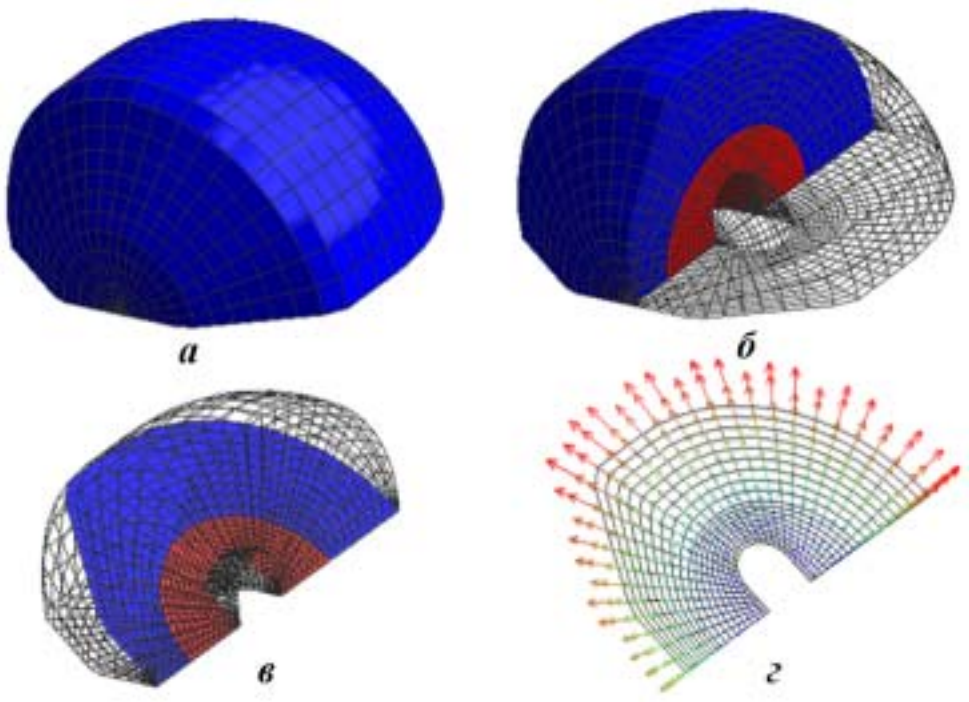

Рис. 11. Применение иерархической фильтрации center: *а* — исходное счетное пространство; *б* — выделение части счетного пространства; *в* — построение сечения; *г* — построение векторного поля

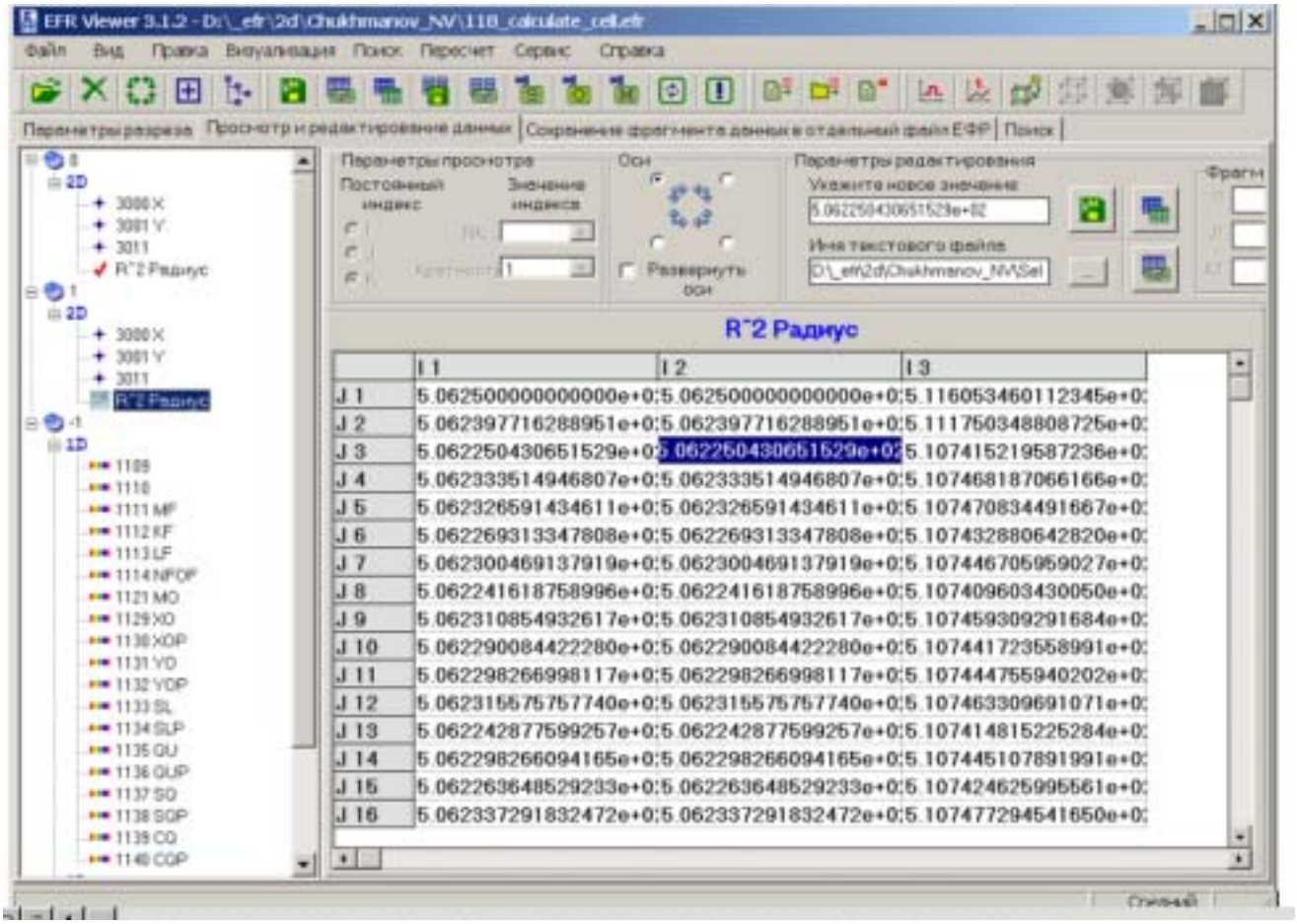

Рис. 12. Диалоговое окно просмотра значения массивов выбранной области

Результаты тестирования

При численном моделировании физических процессов по программам математического отделения РФЯЦ-ВНИИЭФ в трехмерном случае, как правило, используются разностные сетки с числом ячеек около 100 млн. В дальнейшем число ячеек должно достигнуть 500—1 000 млн. Поэтому для проверки применимости ScientificView при обработке таких объемов данных был создан набор тестовых задач с числом ячеек 500 млн.

Тестирование проводилось на 8—48 процессорах серверной части. При использовании 48 процессоров время открытия файлов и построения основных фильтров составило ~ 40 секунд. Система постобработки обеспечила при этом отображение 3—5 кадров в секунду, потребовав менее 1 Гбайт оперативной памяти на ПЭВМ и менее 400 Мбайт на каждом процессоре сервера. ScientificView, как показало тестирование, обеспечивает приемлемые времена обработки.

Наиболее слабым местом системы является ее клиентская часть. Это, как отмечалось выше, связано с ограниченными ресурсами ПЭВМ, необходимыми для функционирования системы отображения. Решение этой задачи видится в двух направлениях — увеличения мощности и памяти ПЭВМ и распараллеливания процедур отображения.

На рис. 13 приведены графики изменения времени считывания данных из файла и применения фильтров "Сечение плоскостью" и "Векторное поле" в зависимости от числа процессоров в серверной части.

Перспективы развития

В ближайшее время планируется провести распараллеливание алгоритмов обработки данных на неструктурированных сетках и ряда дополнительных фильтров для обработки данных на структурированных сетках: выделения гра-

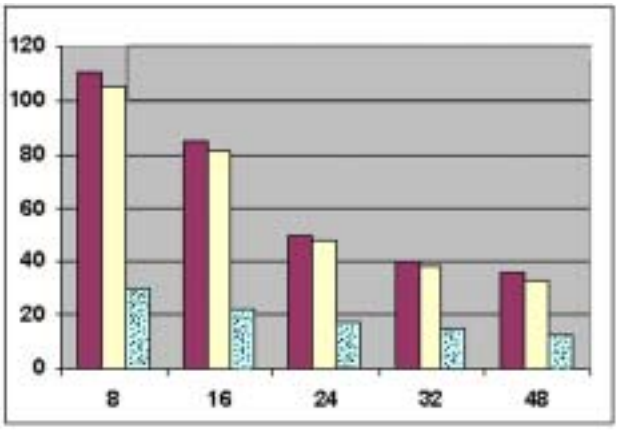

Рис. 13. Графики изменения времени работы (в секундах) алгоритмов в зависимости от числа используемых процессоров в серверной части: столбцы слева — чтение данных из файла; столбцы справа — применение фильтра "Векторное поле"; столбцы в центре — применение фильтра "Сечение плоскостью"

ницы раздела веществ, построения линии профиля. Необходимо также провести дополнительную оптимизацию процедур чтения и обработки данных с целью их ускорения. В дальнейшем планируется реализовать возможности подготовки изображения в параллельном режиме (параллельный рендеринг), альтернативного представления данных (воксельная и стереографика), управления камерой посредством специальных манипуляторов (*виртуальных перчаток*).

Список литературы

1. *Henderson A., Ahren J., etc.* The ParaView guide. Kitware Inc., 2004.
2. *Потехин А. Л., Никитин В. А., Кузнецов М. Г.* Система визуализации в программе расчета трехмерных начальных данных. Обработка данных в трехмерном контексте // VI науч.-тех. конф. "Молодежь в науке". Саров, 30 октября—1 ноября 2007 г.
3. *Попова Н. В., Деманова А. К.* Программа EFR VIEWER как средство быстрого анализа результатов численного моделирования физических процессов // VI науч.-тех. конф. "Молодежь в науке". Саров, 30 октября—1 ноября 2007 г.
4. *Волгин А. В., Красов А. В., Кузнецов М. Ю., Тарасов В. И.* Библиотека ЕФР для универсального представления расчетных данных // Труды РФЯЦ-ВНИИЭФ. 2007. Вып. 11. С. 130—135.
5. *Зибаров А. В.* Cutting-edge CFD solutions for science and industry (http:\\www.cfd.ru 10.02.2006).

Статья поступила в редакцию 17.06.08.