УДК 004.4

ОБ ОДНОМ СПОСОБЕ ОРГАНИЗАЦИИ СТРУКТУР ДАННЫХ В СИСТЕМАХ НАУЧНОЙ ВИЗУАЛИЗАЦИИ

А. Л. Потехин
(ФГУП "РФЯЦ-ВНИИЭФ", г. Саров Нижегородской области)

Использование систем визуализации является одним из наиболее простых и удобных способов анализа результатов математического моделирования. Один из важных вопросов, который приходится решать разработчикам при создании систем визуализации, является вопрос выбора структур данных. Необходимо соблюсти компромисс между расходом оперативной памяти и скоростью работы, удобством применения для конкретных алгоритмов и универсальностью и т. д.

Описываются структуры данных, разработанные в рамках параллельной системы постобработки ScientificView, причины выбора таких структур. Приводится сравнение их скоростных и расходных характеристик с характеристиками структур в системе со схожими возможностями.

Ключевые слова: системы научной визуализации, графическая обработка данных, постобработка результатов математического моделирования, параллельные алгоритмы.

Введение

В РФЯЦ-ВНИИЭФ разработано несколько десятков комплексов программ, предназначенных для численного моделирования различных физических процессов: гидродинамики, теплопроводности, нейтронной физики, прочностного анализа, детонации и т. д. В основном при моделировании используются разностные двумерные и трехмерные сетки различных типов, хотя имеются комплексы, в которых моделирование выполняется с помощью методов молекулярной и кластерной динамики.

Исторически во многих комплексах создавались собственные программы для графического анализа результатов моделирования. Особенно бурный рост таких программ пришелся на 1990—2000 гг., когда появились развитые библиотеки для работы с компьютерной графикой, такие как DirectX [1] и OpenGL [2]. Однако появление супер-ЭВМ с тысячами процессорных ядер и обеспечение работы на них моделирующих комплексов остро поставили вопрос о постобработке данных сверхбольшого объема. Стало понятно, что их быстрая и качественная постобработка возможна только в параллельном режиме. Поскольку реализация параллельной постобработки — задача существенно более сложная, чем создание последовательной программы, в целях экономии трудозатрат разработчиков было принято решение о создании единой параллельной системы постобработки ScientificView [3].

Вопросы хранения и обработки исходных данных

Как уже отмечалось, ScientificView разрабатывалась как система для постобработки результатов моделирования на различных типах сеток. На практике возможны двумерные и трехмерные данные, заданные на структурированных и неструктурированных сетках, а также на наборах материальных точек (кластеры, молекулы, частицы) (рис. 1).

В целях оптимального расхода оперативной памяти для хранения всех этих типов данных разработаны собственные структуры. Что касается структурированных сеток, то ввиду их простой матричной структуры нет необходимости хранить топологию данных. При обработке же неструктурированных сеток топология может быть достаточно сложной. В ScientificView, как и в большинстве других систем визуализа-

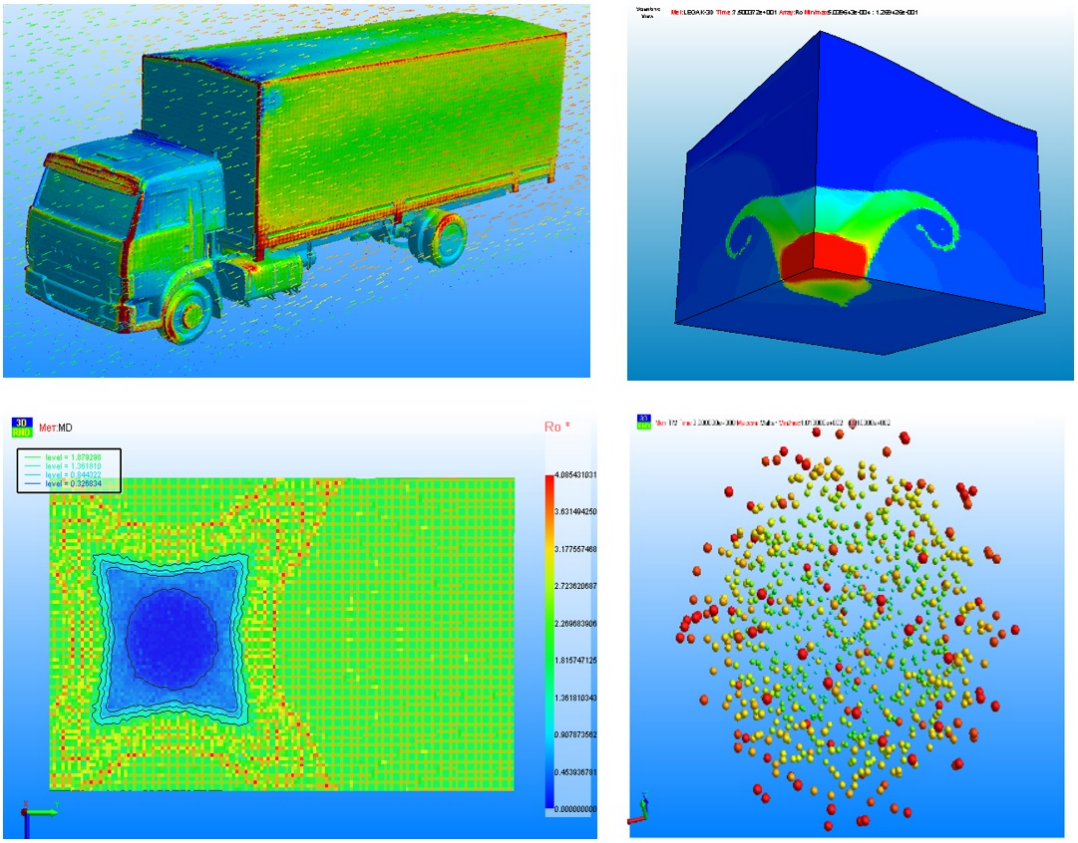


Рис. 1. Примеры отображения в ScientificView различных типов обрабатываемых данных

ции, она считывается разово при первичном открытии файла.

В практике применения ScientificView из всех расчетных данных, использующих неструктурированные сетки, примерно 70 % получены при моделировании конечно-объемными методами [4]. В этом случае каждая ячейка представляет собой список граней, грань — список узлов сетки, узлы описываются координатами. Число граней в ячейке может быть произвольным, число узлов в грани — тоже. В таком виде, как правило, представляются результаты моделирования задач аэрогидродинамики, тепломассопереноса, фильтрации, некоторых других. Число ячеек в практических задачах — до 1 млрд.

Оставшиеся 30 % результатов счета задач получаются при моделировании в конечно-элементном представлении [5], в котором сетка является набором ячеек нескольких видов, например шестигранников, призм, тетраэдров. Для каждой ячейки задан ее тип и набор узлов. Восстановление формы ячеек (и, в частности, набора граней) возможно за счет строгой типизации ячеек. В таком виде, как правило, представляются результаты моделирования задач прочности. Число ячеек в таких задачах редко достигает 30 млн.

Несмотря на то, что для хранения одинаковой топологии (например, из шестигранников или тетраэдров) конечно-объемный вариант представления требует больше оперативной памяти, нежели конечно-элементный, в целях универсальности и исходя из доминирующего количества задач конечно-объемного типа, именно к конечно-объемному представлению приводятся все обрабатываемые в ScientificView неструктурированные данные.

Еще одним важным вопросом является вопрос об обработке сеточных величин, заданных в центрах ячеек, узлах сетки либо в центрах граней или ребер ячеек. В обрабатываемых задачах, как правило, встречается от 20 до 70 массивов. Понятно, что в рамках одной сессии графического анализа практически не возникает ситуаций, когда необходимо использование всех, без исключения, массивов.

Рассмотрим пример данных, определенных на структурированной сетке из 1 млрд ячеек. Пусть

используется 30 сеточных массивов с двойной точностью (8 байт на значение), тогда суммарный объем данных составляет примерно 240 Гб. В то же время, если пользователь реально будет использовать в сессии постобработки 5 сеточных массивов и 3 массива координат, системе визуализации потребуется 64 Гб оперативной памяти, т. е. в 4 раза меньше по сравнению с исходными данными.

ScientificView, как правило, работает с файловыми форматами, позволяющими получать данные порциями. Используются как собственные файловые форматы (в частности, ЕФР [6]), так и внешние, широко распространенные (форматы семейства VTK [7], d3plot [8], cgns [9] и др.). Это означает, что для получения подмножества данных нет необходимости считывать весь файл, достаточно прочитать только необходимую часть. При этом скорость чтения практически линейно зависит от объема данных, которые реально необходимы. Значит, на начальном этапе можно не считывать лишние данные вовсе: ограничиться чтением в память координат (как самых востребованных величин), а все остальные данные считывать по необходимости, например при смене пользователем отображаемой величины. Для описанного примера с 1 млрд ячеек расход оперативной памяти в ScientificView составит только 24 Гб, т. е. в 10 раз меньше, чем требовалось бы в случае чтения всех данных.

Как будет показано на результатах тестирования, такой подход позволяет не только экономить оперативную память, но и обеспечивает время постобработки, сравнимое с вариантом предварительного считывания всех данных.

Графическая обработка данных с помощью алгоритмов фильтрации

Для анализа результатов моделирования физических процессов на разностных сетках широко используется вычисление различных интегральных характеристик, построение графиков зависимостей величин от времени и вдоль пространственных кривых и другие алгоритмы, которые можно отнести к методам числовой обработки информации. В ScientificView реализовано около 20 алгоритмов числовой обработки, но все же основное время пользователь тратит на выполнение более 30 доступных алгоритмов графической обработки (или фильтрации) данных: построение различных сечений, полей, изоповерхностей, скрытие части данных по различным критериям и т. д. Во время сессии постобработки число применяемых фильтров нередко достигает 10—15, поэтому вопрос о правильном выборе структур данных для алгоритмов фильтрации не менее важен, чем для исходных данных.

Практически все фильтры в результате своей работы порождают два множества: множество точек пространства и множество многоугольников, опирающихся на эти точки. Этих двух множеств вполне достаточно для отображения результатов фильтрации любых входных данных, обрабатываемых в ScientificView.

В целях снижения трудозатрат на реализацию новых алгоритмов фильтрации была реализована *базовая часть* системы, обеспечивающая:

- запуск и обработку диалогов для задания параметров фильтрации;
- хранение результатов фильтрации;
- отображение результатов фильтрации в различных режимах, построение упрощенных сеточных моделей для этих результатов в целях ускорения их отображения [10];
- вычисление координат точек, нормалей, определение общих габаритов;
- получение новых значений для цветовой интерпретации сеточной величины в многоугольниках и/или точках при смене пользователем отображаемой величины;
- вычисление интегральной информации.

Таким образом, для реализации новых алгоритмов фильтрации необходимо реализовать лишь саму функцию фильтрации (например, построение фрагмента сечения для конкретной ячейки), большинство остальных операций будет поддержано базовой частью системы.

Так же, как и в случае работы с исходными данными, при формировании результатов фильтрации ScientificView явно не хранит значения всех доступных сеточных массивов. При необходимости происходит чтение требуемого подмножества данных из файла. Более того, даже в случае фильтрации данных, заданных на неструктурированных сетках, не хранится информация о топологии получившегося при фильтрации объекта. Вместо этого используется битовая карта с признаком попадания ячейки сетки в результаты фильтрации. По этой информации также может быть быстро установлена принадлежность результатам фильтрации узла сетки.

Рассмотрим работу системы на примере применения к исходным трехмерным данным фильтра "Интервал". С помощью этого фильтра будем выделять множество ячеек, у которых в заданном пользователем интервале находится значение объема ячейки. Укрупненно работа фильтра состоит из следующих этапов:

1. Получение из файла значений объемов для всех ячеек.
2. Цикл по всем ячейкам исходных данных. Если объем ячейки попадает в заданный интервал, а объем соседней с ней ячейки — нет, то разделяющая эти ячейки грань добавляется в результат фильтрации (в контейнер для хранения многоугольников). Принадлежащие грани узлы сетки также автоматически попадают в результат фильтрации (но уже в контейнер узлов). Для попавшей в заданный интервал ячейки устанавливается признак в битовой карте.
3. Выполнение базовой частью системы общих операций, не зависящих от алгоритма фильтрации: получение координат для всех узлов сетки, попавших в результат фильтрации, построение нормалей и т. п.
4. Удаление данных об объемах, считанных из файла.

Результат для описанного примера представлен на рис. 2. Слева — реальное изображение: синим цветом закрашены ячейки, выделенные фильтром "Интервал", прозрачным серым — прочие исходные данные. Справа — схематичное представление плоского сечения исходных данных и результатов фильтрации, приведенное для упрощения восприятия. Грани, разделяющие ячейки (изображены на правой части рисунка тонкими линиями), отображаются только для лучшего понимания реализации и в ScientificView реально не хранятся. Сечение граничных многоугольников, полученных при фильтрации и используемых при визуализации объекта слева на рис. 2, изображено жирной линией синего цвета на этом рисунке справа. Там же ячейки с выполняющимся критерием отображены серым цветом, с невыполняющимся — белым.

Если теперь на наборе отфильтрованных "Интервалом" ячеек необходимо построить изоповерхность (например, модуля скорости), будут выполняться следующие действия:

1. Получение из файла значений модулей скорости для всех узлов.
2. Цикл по всем ячейкам исходных данных. Если ячейка попала в результат ранее проведенной фильтрации (в данном случае результат работы фильтра "Интервал"), то она обрабатывается, иначе пропускается.
3. Для каждой обрабатываемой ячейки построение фрагмента изоповерхности на основе координат и значений модулей скорости. Полученные многоугольники вместе со своими точками сохраняются в качестве результата фильтрации.
4. Автоматическое построение нормалей, получение габаритов и другие операции, выполняемые базовой частью системы.
5. Удаление данных о модуле скорости, считанных из файла.

Результат построения изоповерхности показан на рис. 3. Слева — реальное изображение, справа — схематичное представление плоского сечения исходных данных (ячейки белого цвета) и результатов фильтрации "Интервалом" (ячейки

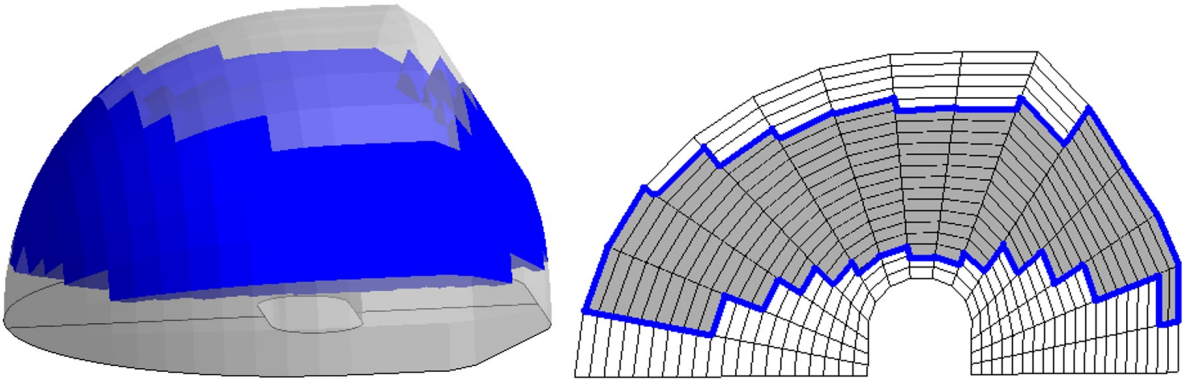


Рис. 2. Изображение результата применения фильтра "Интервал"

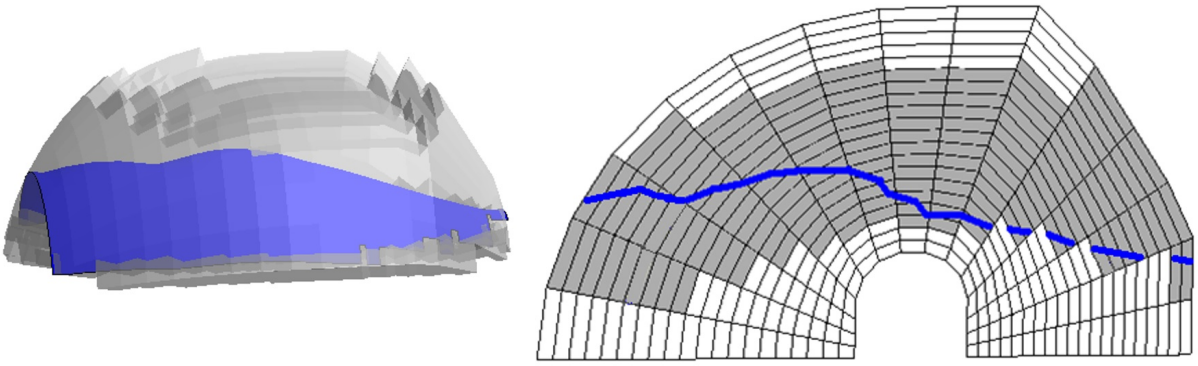


Рис. 3. Изображение результата применения фильтра "Изоповерхность" к множеству ячеек, ранее выделенных фильтром "Интервал"

серого цвета), приведенное для упрощения восприятия. Сечение изоповерхности отображено жирными линиями синего цвета.

Схема работы в параллельном режиме

Принято рассматривать три способа распараллеливания [11] (остальные способы в той или иной степени можно отнести к одному из указанных): распараллеливание на уровне задач (task parallelism), конвейерная параллельность (pipeline parallelism) и распараллеливание на уровне данных (data parallelism).

При реализации ScientificView использовалось распараллеливание на уровне данных [12]. Вся система работает по технологии клиент-сервер, где клиент запускается на ПЭВМ пользователя и предоставляет графический интерфейс для управления системой. Серверная часть запускается с применением MPI на многопроцессорной кластерной ЭВМ под OS Linux, где происходит вся существенная обработка: чтение данных с диска, фильтрация, формирование изображения (рендеринг). При этом в серверной части приложения выделяется один специальный "головной" процесс. Он обеспечивает взаимодействие клиентской и серверной частей, а также проводит итоговое формирование изображения (рис. 4).

При обработке структурированных данных распределение исходных данных по рабочим ядрам проводится следующим образом:

- всем ядрам приписывается для обработки примерно одинаковое количество ячеек;
- топология данных, обрабатываемых каждым ядром, остается регулярной;
- размеры регулярных блоков обработки для всех ядер по возможности совпадают.

Рис. 4. Общая схема работы ScientificView в параллельном режиме

Пример распределения структурированных данных показан на рис. 5.

При обработке неструктурированных данных используется декомпозиция, полученная моделирующей программой при счете. Если при моделировании использовалось N процессорных ядер, при постобработке через ScientificView возможно использование любого числа ядер от 1 до N, хотя на практике оптимальное соотношение *количество ядер/производительность* достигается на числе ядер, составляющем 10—20 % от N. Пример распределения неструктурированных данных между обрабатывающими их ядрами показан на рис. 6.

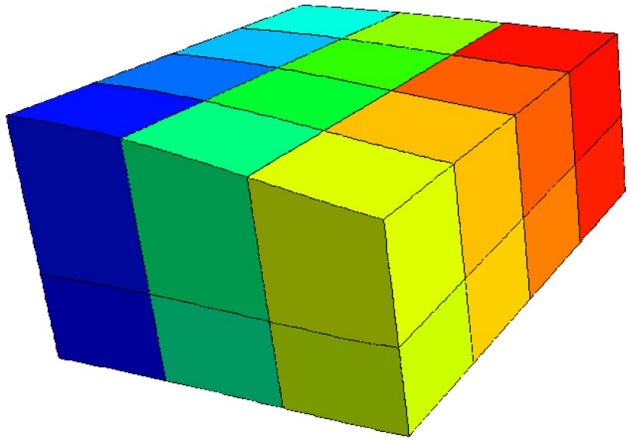


Рис. 5. Распределение структурированных данных по процессорным ядрам

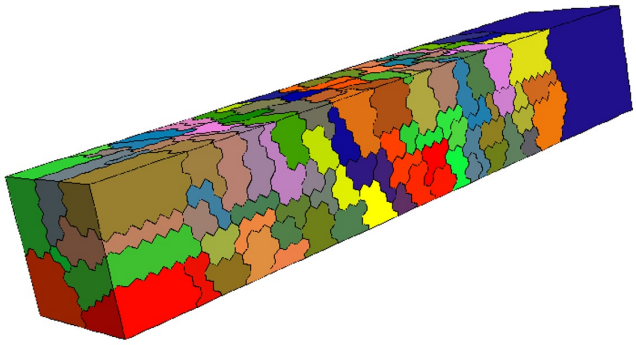


Рис. 6. Распределение неструктурированных данных по процессорным ядрам

Результаты сравнения скоростных и расходных характеристик с характеристиками ParaView

Для оценки правильности решений, принятых при разработке основных структур данных ScientificView, проводилось сравнительное тестирование ScientificView и системы научной визуализации ParaView [13]. ParaView является одной из наиболее популярных в мире систем подобного типа, обладает широкими функциональными возможностями по обработке данных, способна обрабатывать данные в параллельном режиме. Необходимо отметить, что система визуализации ParaView, основанная на библиотеке VTK [7], использует существенно отличающийся подход к организации структур данных. В VTK и ParaView созданы разные объекты для хранения разных типов обрабатываемых данных (структурированные и неструктурированные, двумерные и трехмерные и т. д.). Данные, как правило, считываются полностью при первичном открытии файла, при фильтрации происходит сохранение полученных топологии и копии части сеточных данных.

При проведении тестирования оценивались как скорость работы систем, так и суммарный расход оперативной памяти. Использовался параллельный режим работы, засечки снимались стандартными средствами программ и операционной системы. Входными данными послужил тест с сеткой внутри куба с единичной стороной. Файловое представление — стандартное для рассмотренных систем: формат ЕФР для ScientificView и vts/vtk для ParaView. Ячейки сетки также состоят из кубов, число ячеек для случая структурированной сетки $1\,000 \times 1\,000 \times \times 1\,000 = 1$ млрд, для неструктурированной — $460 \times 460 \times 460 \sim 100$ млн. В узлах сетки с координатами X, Y, Z заданы следующие величины:

- Val1 $= Z^2 + X^2 + Y^2$;
- Val2 $= \sin(15X)\sin(15Y)\sin(15Z)$.

При обработке структурированной сетки использовалось 20 MPI-потоков серверной части приложений, неструктурированной — 10 потоков.

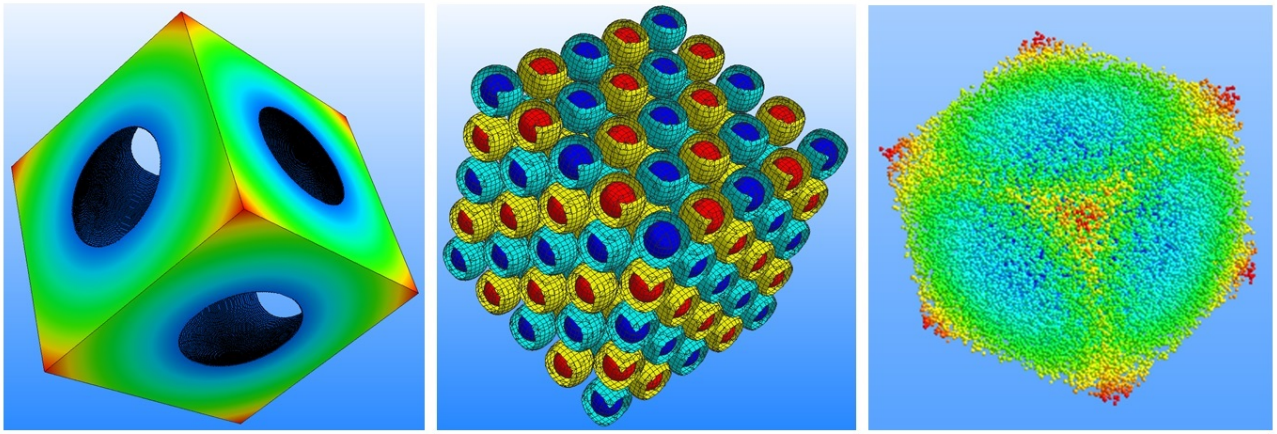
После загрузки исходных данных применялись три алгоритма фильтрации, формирующих соответственно три типа данных в виде множества трехмерных ячеек, многоугольников и наборов векторов:

- "Интервал" (в ParaView "Threshold") для значений величины Val1 от 0,3 до 0,75;
- "Изоповерхность" (в ParaView "Contour") для величины Val2 с четырьмя изоуровнями: −0,65, −0,32, 0,32, 0,65;
- "Векторное поле" (в ParaView "Glyph") — 50 000 векторов, случайно распределенных внутри куба.

Результаты такой фильтрации в ScientificView представлены на рис. 7.

Времена выполнения алгоритмов и суммарный расход оперативной памяти приведены в таблице.

Как видно из приведенных данных, были получены в целом сопоставимые времена выполнения алгоритмов, в то время как расход оперативной памяти в ScientificView оказался существенно меньше, особенно в алгоритмах фильтрации, формирующих полноценные трехмерные ячейки или распределение векторов/точек.



а *б* *в*

Рис. 7. Результаты применения алгоритмов фильтрации в ScientificView: *а* — "Интервал"; *б* — "Изоповерхность"; *в* — "Векторное поле"

Сравнение времени (в секундах) и суммарного расхода памяти (в Гб) при выполнении алгоритмов в системах ScientificView и ParaView

Система, тип данных	"Интервал"		"Изоповерхность"		"Векторное поле"	
	Время	Память	Время	Память	Время	Память
ScientificView, структурированные данные	8	8,2	20	17,3	4,3	0,075
ParaView, структурированные данные	14	47	16	41	4	7,7
ScientificView, неструктурированные данные	2,5	1,2	5,5	3,1	0,5	0,075
ParaView, неструктурированные данные	3,5	6,9	4	5,5	3,5	7,7

Заключение

В результате проведенных работ разработаны структуры данных, обеспечивающие в ScientificView возможность постобработки результатов математического моделирования, полученных с использованием различных типов сеток: двумерных и трехмерных, структурированных и неструктурированных.

Сравнение с одной из лучших в мире систем визуализации ParaView показало, что предложенный подход к реализации структур данных и методов их обработки вполне конкурентоспособен. Получена сопоставимая скорость обработки при существенно более низком расходе оперативной памяти.

Список литературы

1. *Миллер Т.* DirectX 9 с управляемым кодом. Программирование игр и графика. М.: КомБук, 2005.
Miller T. DirectX 9 s upravlyaemym kodom. Programmirovanie igr i graphika. M.: KomBuk, 2005.
2. *Segal M., Akeley K.* The OpenGL Graphics System: a Specifications (Version 3.2 (Core Profile)). The Cronos Group Inc., 2009. www.opengl.org.
3. *Потехин А. Л., Логинов И. В., Козачек Ю. В., Никитин В. А., Кузнецов М. Ю., Деманова А. К., Попова Н. В., Фирсов С. А.* Система постобработки ScientificView. Принципы построения. Основные возмож-

ности // Тр. конф. "Научная конференция им. А. А. Бунатяна". Снежинск: ФГУП "РФЯЦ-ВНИИТФ", 2007. С. 57—62.
Potekhin A. L., Loginov I. V., Kozachek Yu. V., Nikitin V. A., Kuznetsov M. Yu., Demanova A. K., Popova N. V., Firsov S. A. Sistema postobrabotki ScientificView. Printsipy postroeniya. Osnovnye vozmozhnosti // Tr. konf. "Nauchnaya konferentsiya im. Bunatyana". Snezhinsk: FSUE "RFYaTs-VNIITF", 2007. S. 57—62.

4. *Смирнов Е. М., Зайцев Д. К.* Метод конечных объемов. Научно-технические ведомости. СПбГПУ, 2004.
Smirnov E. M., Zaytsev D. K. Metod konechnykh obyemov. Nauchno-tekhnicheskiye vedomosti. SPbGPU, 2004.

5. *Галлагер Р.* Метод конечных элементов. Основы: Пер. с англ. М.: Мир, 1984.
Gallager R. Metod konechnykh elementov. Osnovy: Per. s angl. M.: Mir, 1984.

6. *Олесницкая К. К., Антипин И. А., Петрова М. А.* Коллективный доступ к файловым данным на вычислительных системах с различной архитектурой в библиотеке ЕФР // Тр. XVII Межд. конф. "Супервычисления и математическое моделирование". Саров: ФГУП "РФЯЦ-ВНИИЭФ", 2018. С. 392—399.
Olesnitskaya K. K., Antipin I. A., Petrova M. A. Kollektivnyy dostup k faylovym dannym na vychislitelnykh sistemakh s razlichnoy arkhitekturoy v biblioteke EFR // Tr. XVII Mezd. konf. "Supervychisleniya i matematicheskoe modelirovanie". Sarov: FGUP "RFYaTs-VNIIEF", 2018. S. 392—399.

7. VTK File Formats. https://kitware.github.io/vtk-examples/site/VTKFileFormats/.

8. LS-DYNA Database Binary Output Files. https://ftp.lstc.com/anonymous/outgoing/trent001/manuals/ls-dyna_database.pdf.

9. CFD General Notation System. https://cgns.github.io/.

10. *Потехин А. Л.* Методы быстрого формирования изображения в параллельной системе постобработки ScientificView // Тр. межд. конф. "Компьютерное моделирование 2009". С.-Пб.: Изд-во Политех. ун-та, 2009. С. 159—164.
Potekhin A. L. Metody bystrogo formirovaniya izobrazheniya v parallelnoy sisteme postobrabotki ScientificView // Tr. mezhd. konf. "Kompyuternoe modlirovanie 2009". S.-Pb.: Izd-vo Politekh. un-ta, 2009. S. 159—164.

11. *Ahrens J., Law Ch.* A parallel approach for efficiently visualizing extremely large, time-varying datasets. Report Los Alamos National Laboratory. LAUR-00-1620, 2000.

12. *Потехин А. Л., Логинов И. В., Козачек Ю. В., Никитин В. А., Кузнецов М. Ю., Деманова А. К., Попова Н. В., Фирсов С. А.* ScientificView — параллельная система постобработки результатов, полученных при численном моделировании физических процессов // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 2007. Вып. 4. С. 37—45.
Potekhin A. L., Loginov I. V., Kozachek Yu. V., Nikitin V. A., Kuznetsov M. Yu., Demanova A. K., Popova N. V., Firsov S. A. ScientificView — parallelnaya Sistema postobrabotki resultatov, poluchennykh pri chislennom modelirovanii fizicheskikh protsessov // Voprosy atomnoy nauki i tekhniki. Ser. Matematicheskoe modelirovanie fizicheskikh protsessov. 2007. Vyp. 4. S. 37—45.

13. ParaView is an Open-Source Application for Visualizing Two- and Three-Dimensional Data Sets. https://www.paraview.org/about/.

Статья поступила в редакцию 31.03.22.