ПАКЕТ ПРОГРАММ ЛОГОС. ИНТЕГРАЦИЯ НЕКОТОРЫХ ФАЙЛОВЫХ ФОРМАТОВ ХРАНЕНИЯ СЕТОЧНЫХ ДАННЫХ

А. И. Лопаткин, И. В. Логинов

ФГУП «РФЯЦ-ВНИИЭФ», г. Саров Нижегородской обл.

В настоящее время в математическом отделении ИТМФ продолжается разработка параллельной системы постобработки ScientificView [1]. Данная система предназначена для фильтрации, отображения и табличного анализа результатов моделирования физических процессов на разностных сетках. Изначально система ScientificView поддерживала формат хранения данных ЕФР1, ЕФР2 [2], который разработан в математическом отделении ИТМФ как универсальное средство чтения/записи расчетных данных для большинства расчетных методик отделения. ЕФР – единый файл-разрез, в котором сосредоточена вся необходимая информация о рассчитываемой задаче на определенный момент времени. На данный момент в математическом отделении ведется работа по созданию открытой версии системы ScientificView для передачи во внешние организации. Для обеспечения конкурентоспособности разрабатываемой системы наряду с такими системами визуализации, как ParaView [3], VizIt, TecPlot и др. необходимо обеспечить поддержку файловых форматов, которые получили широкое распространение во всем мире и поддерживаются большинством популярных систем визуализации и постобработки. Обеспечение обработки данных, представленных в файловых форматах сторонних систем, позволяет применять к этим данным алгоритмы фильтрации, имеющиеся в системе ScientificView, а также обеспечить верификацию работы системы с другими системами визуализации.

Доклад описывает концепцию интеграции сторонних файловых форматов в параллельную систему постобработки ScientificView, подходы и средства, используемые для обеспечения поддержки конкретных файловых форматов, приводятся сравнительные характеристики с другими системами визуализации и постобработки касательно скорости загрузки данных, кратко описываются новые возможности, обусловленные необходимостью поддержки новых форматов, а также дается описание механизма преобразования конечно-элементного представления топологии к ребро-граневому.

Файловые форматы хранения сеточных данных предполагают описание хранения в файле информации о типе сетки (регулярная, нерегулярная), размерности сетки, описание связи между ячейками для нерегулярных сеток.

Ориентируемая на обработку данных формата ЕФР, для обеспечения потребностей ряда математических методик система постобработки ScientificView потребовала предоставления возможности обработки данных, сохраненных в других файловых форматах. Отчасти это было обусловлено недостаточностью возможностей хранения данных, предоставляемых данным форматом. Формат ЕФР2 предоставляет возможность сохранения нерегулярной топологии в ребро-граневом представлении, когда ячейка сетки представляется как совокупность граней, в то время как возможность сохранения в конечно-элементном представлении (ячейка представляется совокупностью узлов) отсутствует.

Процедура внедрения дополнительного формата в ScientificView предполагает реализацию модуля чтения, предоставляющего стандартный набор методов доступа к данным, получения информации по обрабатываемой задаче, инициализацию требуемых параметров, функционал по выделению временных шагов. Данный модуль должен являться наследником базового класса, обеспечивающего чтение данных. Подобная схема наследования позволяет программистам в дальнейшем не заботиться о типе используемого класса и типе файлового формата и позволяет применить весь функционал системы ScientificView к данным, полученным из сторонних форматов.

Одной из первостепенных задач в этом направлении стояла задача интеграции файловых форматов библиотеки VisualizationToolkit и формата системы LS-Dyna D3Plot. Написание модулей для считывания различных файловых форматов собственными силами часто оказывается трудоемкой задачей по причине большого числа форматов и сложности описываемых структур данных. Накопленный в математическом отделении опыт показал, что одной из программ, поддерживающей большое число популярных файловых форматов, является визуализатор ParaView, основой которого является библиотека Visualization Toolkit (VTK) [4, 5]. Библиотека VTK развивается, как Open Source проект, что допускает ее использование в собственных разработках. Примечательно то, что подавляющее большинство модулей для считывания/записи данных из различных файловых форматов, поддерживаемых визуализатором ParaView, реализованы непосредственно в библиотеке VTK и могут использоваться в других программах. Результатом работы модулей считывания данных VTK является набор, состоящий из одного или нескольких объектов данных определенного типа (по классификации VTK). Таким образом, одним из простых способов поддержки внешних файловых форматов в сис-

теме ScientificView будет использование классов библиотеки VTK и дальнейшее написание модулей для преобразования структур данных VTK в собственные структуры программы. Данный подход позволит значительно сократить календарное время разработки, так как перед разработчиком не будет стоять необходимости в детальном изучении файловых форматов и написании базовых модулей для считывания данных непосредственно из файлов. Интеграция форматов хранения регулярных сеточных данных библиотеки VTK полностью вписывается в изложенную концепцию. Выполнив необходимую инициализацию структур хранения, реализовав необходимые методы данных, удалось обеспечить возможность обработки данных, представленных в одном из следующих форматов: vts,vti, vtr, pvts, pvti, pvtr. На рис. 1 представлен результат отображения данных, представленных в формате vts в ParaView (слева) и в ScientificView (справа). Время загрузки и отображения данных составило 9,6 и 10,5 соответственно. Объем требуемой оперативной памяти 385 и 356 Mb для ParaView и ScientificView соответственно. Размерность 200x200x200 ячеек.

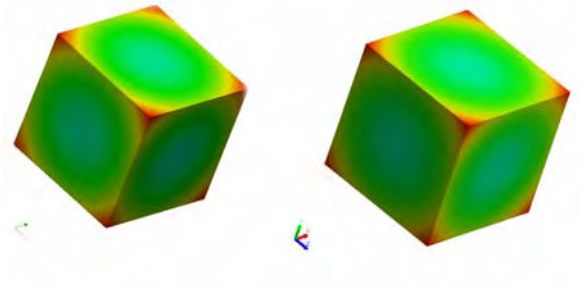

Рис. 1. Результат отображения данных, представленных в формате vts, в ParaView (слева) и в ScientificView (справа)

Внедрение же форматов хранения нерегулярных сеточных данных (vtu, pvtu – представляет собой ссылки на несколько составляющих его фрагментов, представленных в формате vtu) в свою очередь потребовало преобразования конечно-элементного представления топологии к ребро-граневому. Поскольку введение в систему возможности визуализации и обработки сеток, представленных в конечно-элементном виде, потребовало бы значительных трудозатрат, и изменение большинства алгоритмов обработки данных (фильтров) для нерегулярной трехмерной области. Суть данного преобразования заключается в том, чтобы ячейка описывалась не совокупностью узлов, а совокупностью граней, ее составляющих. В свою очередь грани должны быть представлены именно как совокупность узлов. Также необходимо обеспечить, чтобы ячейки с общей гранью ссылались на одну и ту же грань во избежание дублирования и увеличения объема требуемых ресурсов (рис. 2).

Поскольку библиотека VTK ориентирована на конечно-элементное представление топологии, в ней реализованы методы получения конкретной грани для ячейки, а также прочие методы получения информации по ячейке (тип, общее число граней и т. д.).

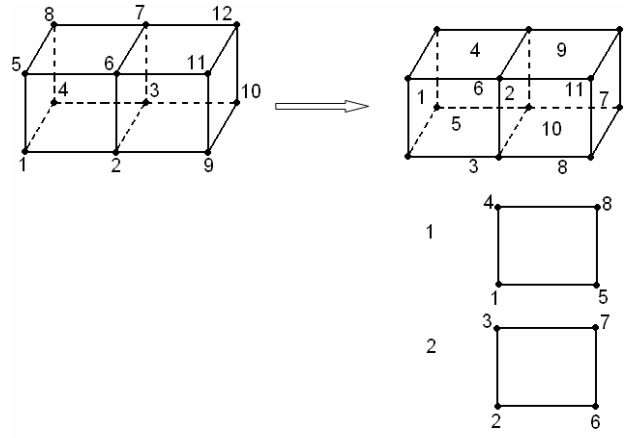

Рис. 2. Конечно-элементное представление топологии (слева) и ребро-граневое (справа)

Также VTK предоставляет функционал по поиску соседей на сетке, что необходимо для предотвращения дублирования граней. В связи с необходимостью конвертирования топологии получен ощутимый проигрыш касательно скорости загрузки данных в сравнении с ParaView. На рис. 3 представлен результат отображения данных, представленных в формате pvtu, в ParaView (слева) и ScientificView (справа). Время загрузки и отображения данных составило 10,629 и 15,1 с для ParaView и ScientificView соответственно. Размерность 1261448 ячеек в 96 файловых фрагментах.

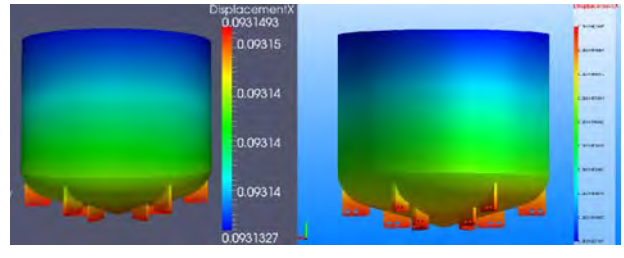

Рис. 3. Результат отображения данных, представленных в формате pvtu, в ParaView (слева) и в ScientificView (справа)

Наряду с интеграцией сторонних форматов в скалярную версию системы ScientificView, обеспечена декомпозиция данных по процессорам при возможности. Так для регулярных данных обеспечена матричная и полистовая декомпозиция. Для нерегулярных форматов библиотеки VTK декомпозицию данных удалось осуществить только для формата pvtu, который представляет из себя несколько распределенных по файлам фрагментов общей сетки, сохраненных в отдельные vtu файлы. Собственно каждый из этих фрагментов по возможности загружается отдельным процессором. На рис. 4 представлен результат декомпозиции нерегулярных данных, представленных в формате pvtu.

В процессе интеграции сторонних файловых форматов стала очевидной необходимость отображения временных шагов. Под временным шагом понимается вся необходимая информация о задаче на определенный момент времени. Совокупность временных

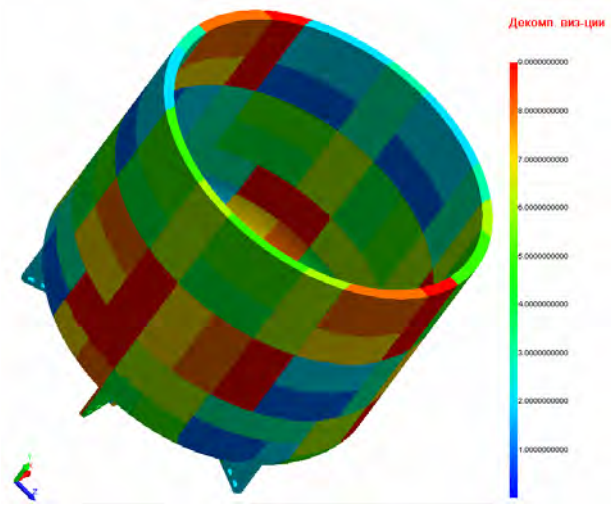

Рис. 4. Результат декомпозиции данных, представленных в формате pvtu, в ScientificView

шагов характеризуется неизменной топологией. С целью отображения нескольких временных шагов, представленных в одном из форматов библиотеки VTK, система визуализации ParaView использует собственный формат pvd (ParaView Data). Частичную поддержку данного формата потребовалось обеспечить в рамках системы ScientificView. В частности, одной из ключевых задач являлось обеспечение отображения временных шагов, представленных в файлах формата pvtu.

В целях экономии трудозатрат по внедрению формата системы LS-Dyna D3Plot также решено было воспользоваться готовым решением библиотеки VTK. Поскольку все сторонние форматы библиотека VTK преобразует к собственным структурам хранения, предполагалось, что, изменив модуль чтения формата vtu по части инициализации и методов доступа, удастся обеспечить обработку данных в формате d3plot системой ScientificView. Однако было выявлено, что часть файлов некорректно обрабатываются ParaView, в том время как LS-PrePost обеспечивает их корректное отображение. Данная проблема была устранена посредством изменения части исходного кода модуля библиотеки VTK, обеспечивающего чтение формата d3plot. Кроме того, потребовалась модификация данного модуля и по обработке ячеек, удаляемых из счета, при переходе от одного временного шага к другому, поскольку в результате работы исходного варианта сеточная топология менялась от шага к шагу, что характеризуется значительными временными затратами. Параллельно с этим была разработана и внедрена схема, обеспечивающая корректную обработку данных ячеек в системе ScientificView. Основная суть данной схемы заключается в том, что при наличии определенного сеточного массива, содержащего признак удаления ячейки, происходит перестроение поверхностных (отображаемых) граней. Кроме того, была обеспечена возможность отображения конкретного временного шага при наличии шапки, содержащей основную информацию по задаче, в то время как исходный модуль чтения библиотеки VTK обеспечивал возможность работы с конкретным шагом только при наличии всех предыдущих. На рис. 5 приведены результаты отображения данных, представленных в формате d3plot, в LS-PrePost и ScientificView. Время загрузки данных в этом случае сравнивать некорректно, поскольку используются разные схемы подгрузки данных. LS-PrePost подгружает данные по мере необходимости, в то время как в ScientificView происходит полное кэширование всех данных. Если же сравниваться с ParaView, то время загрузки и отображения данных для ParaView составило 16,5 с, для ScientificView – около 19 с. Объем требуемой оперативной памяти составил 750 и 691 для ParaView и ScientificView соответственно.

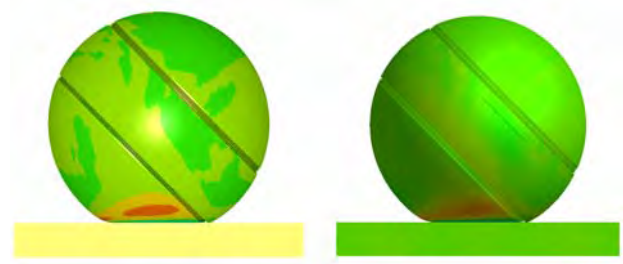

Рис. 5. Результат отображения данных, представленных в формате d3plot, в LS-PrePost (слева) и ScientificView (справа)

Появившийся совсем недавно формат EФР3 вобрал в себя все имеющиеся в EФР2 возможности с учетом недостающих. Так в EФР3 появилась возможность сохранения топологии в конечно-элементном представлении, временные шаги и ряд других возможностей, поддержку которых было необходимо обеспечить в ScientificView. Конечно-элементная топология представляется в EФР3 в виде массива с типами ячеек (рисунок) и списком, содержащим индексы узлов для каждой ячейки. Отсутствие методов получения граней, методов поиска соседства в качестве первоначального лобового решения натолкнуло на мысль конвертирования топологии EФР3 в структуры хранения нерегулярных сеток библиотеки VTK и воспользоваться предоставляемым ею функционалом. Однако подобный вариант оказался неприемлемым по скорости загрузки данных, а также по затратам оперативной памяти, поэтому были реализованы собственные структуры представления ячеек, для каждого типа ячеек необходимые методы получения граней, числа узлов для каждой грани и прочее. Также реализовывался алгоритм поиска соседства по конкретной грани. Суть алгоритма заключается в первоначальном составлении соответствия для каждой точки всех ячеек, которым она принадлежит. При очередном поиске соседа для узлов грани ячейки из составленного соответствия извлекаются ячейки, содержащие эти же узлы, после чего проверяется, содержатся ли все узлы грани исходной ячейки в предполагаемой соседней ячейке. Реализация собственных структур хранения

и методов преобразования позволила снизить временные затраты на конвертирование топологии практически в два раза относительно затрат на конвертирование топологии из структур хранения нерегулярной сетки библиотеки VTK. Результат отображения поддерживаемых в ЕФР3 типов конечных элементов представлен на рис. 6.

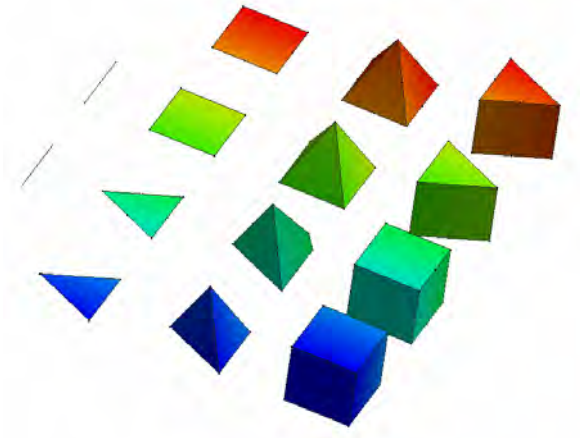

Рис. 6. Результат отображения данных, представленных в формате ЕФР3

Интеграция же файловых форматов, имеющих ребро-граневое представление топологии, не требует каких-либо серьезных преобразований. Представление топологии в текстовом формате ngeom практически полностью совпадает с представлением топологии, на которое ориентируется ScientificView. В результате для интеграции данного формата потребовалось лишь реализовать методы доступа к данным, заполнить внутренние структуры хранения. Наиболее трудоемкой оказалась возможность отображения граничных условий (так называемых патчей), информация о которых также сохраняется в ngeom файле. Файл ngeom хранит информацию о топологии и граничных условиях, данные сеточных массивов сохраняются в специальных usr файлах, методы чтения и предоставления доступа к данным из которых также были реализованы.

Интеграция формата cgns также потребовала конвертирования топологии. Однако, воспользовавшись уже реализованными для ефр3 структурами и методами, это удалось обеспечить в относительно короткие сроки. Для предоставления доступа к файлу для формата cgns используется поставляемая с исходных кодах библиотека Libcgns, обеспечивающая методы получения и чтения необходимой информации. Для формата cgns также было реализовано выделение патчей.

В рамках работы, описанной в докладе, сформировалась схема интеграции сторонних файловых форматов в систему ScientificView. Данная схема позволяет обеспечить возможность обработки данных, представленных в сторонних для ScientificView форматах, посредством реализации требуемых (ставшими стандартными) методов по доступу к данным.

Литература

1. Потехин А. Л., Логинов И. В., Козачек Ю. В. и др. ScientificView – параллельная система постобработки результатов, полученных при численном моделировании физических процессов // Вопросы атомной науки и техники. 2008. Вып. 4. С. 37–45.

2. Волгин А. В., Красов А. В., Кузнецов М. Ю., Тарасов В. И. Библиотека ЕФР для универсального представления расчетных данных // Труды РФЯЦ-ВНИИЭФ. Выпуск 11. Саров, 2007. С. 130–135.

3. Henderson A., Ahren J. etc. The ParaView guide. Published by Kitware, Inc. 2004.

4. Schroeder W. J., Martin K. M., Lorensen W. E. The Visualization Toolkit An Object Oriented Approach to 3D Graphics. 2nd Edition. New Jersey: Prentice Hall, 1998.

5. Schroeder W. J., Martin K. M., Law C. C., Hoffman W. A. The VTK User’s Guide. Kitware, Inc, 2004.