

Система Интерактивной Визуализации Параллельных Вычислений

В.Л. Авербух, П.А. Васёв, Д.Ю. Горбашевский,
А.Ю. Казанцев, Д.В. Манаков
ИММ УрО РАН, Екатеринбург
averbukh@imm.uran.ru

При создании специализированных систем визуализации для задач, связанных с «большим» счетом (время счета - от нескольких часов до нескольких суток и более), возникает целый ряд проблем, решение которых служат темой этой работы. Серьезная проблема связана непосредственно с организацией on-line визуализации параллельных вычислений.

1. ВВЕДЕНИЕ

Создаваемый нами проект системы визуализации параллельных вычислений предполагает ее функционирование по следующей схеме:

- Задача считается на параллельном вычислителе и оставляет (окончательный или промежуточный) результат в файлах или в памяти процессоров вычислителя.
- На параллельном же вычислителе проводится некоторая предобработка и фильтрация данных, связанная с будущей визуализацией.
- По заданию пользователя строятся модули описания параллельных фильтров и видов отображения, на вход которых подаются данные после предобработки (на первых этапах предлагается использовать предопределенные в системе виды отображения).
- Сама визуализация может идти как на графической рабочей станции, так и на параллельной машине (в последнем случае на ПЭВМ может подаваться собственно растр).

Предполагается рассмотреть два варианта использования системы – в off-line (после окончания счета) и on-line (по ходу счета параллельной программы) режимах. Также рассматривается два варианта размещения средств предобработки и визуализации на процессоры параллельного вычислителя – на те же процессоры, где происходил счет или на дополнительно выделенные процессоры вычислителя.

Таким образом, в проектируемой системе параллельной визуализации предполагается три независимых класса модулей, отвечающих за параллельную предобработку и фильтрацию, за сбор, хранение и передачу данных и, наконец, за визуализацию.

В качестве конкретных примеров, на которых отрабатывается эта схема, рассматриваются задачи математической физики, в частности, связанные с сеточными методами вычислений.

В работе рассмотрены подходы к разработке транспортной подсистемы, а также виды отображения, метафоры и методики визуализации, предложенные в ходе реализации системы.

2. СУЩЕСТВУЮЩИЕ ПОДХОДЫ И РЕШЕНИЯ

Подробное изложение подходов к реализации систем интерактивной визуализации можно найти, например, в [1]. Мы же ограничимся сейчас рассмотрением одной, на наш взгляд наиболее интересной системы.

Oak Ridge CUMULVS. Система CUMULVS[2] предлагает on-line визуализацию, управление ходом вычислений и поддержку контрольных точек. В ее основе системы лежит

весьма лаконичная и изящная парадигма взаимодействия. Так, для внедрения механизма on-line визуализации программист должен для визуализируемого распределенного массива данных выполнить вызов в параллельной программе, в котором указать его логическое имя, тип данных, размерности, декомпозицию по процессорам и указатель на данные текущего процессора. После этого пишется (или берется) скрипт для системы визуализации AVS, в котором говорится, на основе каких массивов как строить визуальное представление.

На наш взгляд единственный и ключевой недостаток данной системы заключается ее жесткой привязке к PVM. PVM на данный момент считается неэффективной библиотекой и практически не используется отечественными параллельными программистами. При этом совместное использование коммуникационных библиотек, например MPI и PVM, современными системами запуска задач не приветствуется.

3. ТРАНСПОРТНАЯ ПОДСИСТЕМА

Одной из составляющих системы визуализации является подсистема взаимодействия процессов.

Создана «транспортная» подсистема, которая позволяет связать несколько программ (выполняющихся на вычислителе и пользовательских станциях) в т.н. домен обмена сообщениями.

Использована классическая парадигма очереди сообщений с некоторыми дополнениями. Подсистема является сохранной асинхронной системой обмена сообщениями [3]. В ее основе находится понятие сообщения - неделимой единицы обмена. Данные в сообщении произвольные. Процессы-участники посылают друг другу сообщения через очереди сообщений. Очередей сообщений может быть несколько, посылать сообщения в очередь может любой участник, которому это разрешено; принимают сообщения из очереди только те участники, которые оформили подписку на данную очередь. Подписка осуществляется на основе логических псевдонимов. Сообщения очереди гарантированно доставляются всем подписавшимся участникам в порядке отправки с учетом приоритетов сообщений и приоритетов отправителей. Разрешается отзываться сообщения - так, участник может отозвать записанные им, но еще не доставленные сообщения, соответствующие некоторой маске атрибутов. Существует класс "приветственных" сообщений: такое сообщение, будучи помещено в очередь, рассылается и вновь подписывающимся участникам, вплоть до отзыва этого сообщения.

Текущая реализация использует возможности сетевых файловых систем и требует наличие общего каталога, доступного с вычислителя и пользовательских станций. Этот каталог должен быть доступен для файловых операций mkdir/open/read/... всем участникам обменов. Обязательна поддержка жестких ссылок.

Разработка подсистемы и работа над тестовыми задачами позволила сделать следующие наблюдения:

- Имеет место высокая латентность обменов. В нашем случае она не критична, поскольку частота обменов "визуальный клиент"- "параллельная программа"

относительно невелика (10-50 сообщений в секунду) (это интуитивное предположение)

- При неудачной комбинации «устаревшая версия сервера samba»—«аномальное поведение клиента nfs на узле»—«высокая частота обменов (>50 сообщений в секунду)»—«узкий канал связи с машиной пользователя» система ведет себя нестабильно и неэффективно.
- Организация vpn-доступа для работы через Internet оказалась нетривиальной задачей.

На основе выказанных наблюдений формулируются следующие выводы:

- Без промежуточных звеньев (менеджеров сообщений) не удастся организовать эффективный асинхронный механизм работы с системой обмена сообщениями. Если продолжать развитие этой системы, придется внедрять такие звенья в инфраструктуру вычислителя.
- Более правильным видится написание полноценной системы промежуточного уровня (возможно и использование каких-либо opensource-проектов).
- Самое важное направление сейчас, после разработки транспортной подсистемы – это создание парадигмы системы визуализации. Даже при написании тестовых программ много времени пришлось потратить на разработку протоколов обмена. Конечному пользователю надо предоставить не обмен сообщениями, а высокоуровневые и простые конструкции.

4. МЕТОДИКИ ВИЗУАЛИЗАЦИИ РЕЗУЛЬТАТОВ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ

Основные данные, получаемые в результате расчетов для аэро- и гидродинамических задач – это массивы значений газодинамических параметров в узлах расчетной трехмерной сетки. Поля газодинамических величин могут быть как скалярными (например, давление, плотность, число Маха, энтропия), так и векторными (поле скоростей). Кроме того, исследователей интересует не только пространственная картина течений, но и распределение величин (плотности, давления) на поверхности объекта, а также плоскостные срезы и одномерные профили полей. Адекватное визуальное представление исследуемого объекта играет важную роль при изучении физических процессов методами численного моделирования.

Сетка состоит из набора стандартных модулей, например, «прямоугольник», «квадрат-цилиндр», «полусфера», «квадрат» и «задонная область».

Каждый модуль состоит из нескольких блоков регулярных сеток, ограниченных границами. С целью удобного и понятного графического представления результатов моделирования предполагается, прежде всего, наглядно отображать сеточные данные больших объемов. Необходима также возможность поиска аномалий в структуре сетки, а также построение в трехмерном пространстве векторных и скалярных полей, изоповерхностей, линий тока и других объектов, могущих быть полезными в исследовании сеточных данных.

Нерегулярность и весьма сложная структура сетки, а также ее большие размеры делают использование одного вида отображения явно недостаточным. Интерфейс основного вида отображения будет перегружен функциями манипуляции. Ввод информации может оказаться сложным, а вывод – тяжёлым для восприятия. Кроме того, при использовании единственного вида отображения, визуальной информации, может оказаться недостаточно, а при масштабировании необходимый уровень детализации, возможно, не будет достигнут. Для решения данных

проблем был разработан комплексный вид отображения с элементами интерактивного взаимодействия. При этом были использованы идеи MiniMap, применяемого в основном, в пакетах обработки растровых изображений (Adobe Photoshop, Microsoft Paint и т.д.) и в компьютерных играх. Потенциал нового вида отображения будет на порядок выше вследствие его гибкости. Заложим в один вид отображения способность к изменению множества параметров, существенно влияющих на адекватность восприятия его содержимого в конкретной ситуации. Размерность, метод рендеринга, набор доступных функций пользовательского интерфейса, все это будет меняться (возможно, автоматически, в зависимости от некоторого набора внешних факторов).

Так, например, при построении сечений сетки плоскостью, удобно спроецировать вид отображения на плоскость перпендикулярную секущей.

Использование нескольких методов рендеринга в одном виде отображения заметно повышает его универсальность. Возможности полигональной графики позволяют очень детально описать блочную структуру сетки, иерархию составных частей и особенности ее внутреннего устройства. Воксельная графика подчеркивает значение сеточных задач, отображая не дискретное множество точек, а трехмерную модель в виде непрерывного массива информации.

Визуальные элементы пользовательского интерфейса, отображающие введенную информацию, также могут считаться самостоятельными видами отображения, входящими в систему. Однако, их влияние на адекватность восприятия информации в целом невелико.

Функции манипуляции данными, наделяя отдельные виды отображения интерактивностью и управляемостью, формируют законченный образ системы комплексных видов отображения. Правильно спроектированная система манипуляции данными, равномерно распределяя свои функции среди визуальных компонент системы, обеспечивает чистоту и ясность человеко-машинного интерфейса. Важнейшая характеристика такой системы, *адекватность*, трактуется как сведение к минимуму различий между тем, что пользователь хотел сделать с данными визуализации и тем, что с этими данными в действительности произошло в результате его манипуляций.

4.1 Применение полигональной графики

Для отображения сеток была разработана среда Transparent Grid Explorer (TGE), которая представляет собой специализированную систему визуализации по выводу сеточных данных с управляемой камерой и системой фильтрации данных

Навигация в системе представляет собой совокупность функций управления камерой. При этом используется проекционная система манипуляции данными. Функции манипуляции данными разделены на группы, такие как «перемещение», «вращение», «построение сечений» и т.д.. Во избежание путаницы и перегруженности вида отображения, эти группы разделены взаимоисключающими режимами манипуляции, так, что при выбранном режиме, доступны только его функции.

Система оперирует данными больших объемов. Без эффективных механизмов фильтрации данных возможно снижение производительности. Также неизбежно падение общей эффективности системы за счет перегруженности видов отображения ненужной информацией.

Работа системы фильтрации данных состоит из двух частей: «пространственной» и «качественной».

Пространственная фильтрация заключается в выделении необходимого объема сеточных данных на основе их пространственного расположения. Очевидно, что одновременный вывод на экран всего слишком большого объема данных влечет за собой снижение уровня детализации. Кроме того, из-за большой некомпактности визуализируемых данных, более далекие от наблюдателя участки могут оказаться вне поля зрения будучи скрытыми близкими участками.

Пространственная фильтрация отчасти решает эту проблему, выделяя область интересующей нас информации из общего ее объема. Функции пространственной фильтрации реализованы с помощью метафоры «альфа-сферы». Суть метафоры состоит в том, что из общего объема данных выделяется сферическая область. При этом визуальная прозрачность объектов, находящихся внутри области, прямо пропорциональна удаленности объектов от центра области. Таким образом, информация, представляющая для нас наибольший интерес, видна лучше всего, а остальная – тем меньше, чем более удалена от центра области рассмотрения. Фильтрация «по значению» необходима для отсеечения ненужных участков визуализируемых данных на основании значения каких-либо характеристик, заданных на этих участках. Для целей фильтрации «по значению» в TGE введен дополнительный компонент пользовательского интерфейса – поле диапазонов (range bar). Внешне поле диапазонов представляет собой узкую полосу, закрашенную градиентом от красного к синему – каждому значению характеристики соответствует цвет на полосе.

«Блочная фильтрация» является наименее универсальным методом фильтрации данных, но тем не менее эффективно применяется в нашей системе.

TGE написана с применением интерфейсов библиотеки DirectX. При запуске программа создает окно графического вывода, размеры которого совпадают с разрешением экрана. Далее создаются DirectX объекты: вершинные буферы, текстуры. Затем инициализируются параметры освещения, матриц трансформации, и общие установки рендеринга.

Следующий этап – чтение сеточных данных из файлов.

Механизмы блочной фильтрации отбирают необходимые блоки из сетки, после чего система обращается к каждому узлу конкретного блока. Подключив механизмы пространственной фильтрации и фильтрации «по значению», система определяет уровень визуальной прозрачности узлов. Затем начинается этап рендеринга. При выводе к сеточным данным также применяются операции со стороны системы манипуляции данными. Отметим, что наша реализация обладает большим потенциалом развития, высокой производительностью и низкими аппаратными и системными требованиями.

В качестве перспективных направлений развития среды можно выделить задачи:

- Доработка модуля построения сечений и изоповерхностей.
- Разработка модуля представления векторных данных в виде полей и линий тока.
- Использование альтернативных видов рендеринга, в добавление к имеющимся.
- Адаптация системы для работы с многопроцессорным вычислителем.

4.2 Применение воксельной графики

При визуализации структуры сеток и данных на них возникают задачи как непосредственно отображения сеток и данных в целом, так и задачи построения сечений и

изоповерхностей на сетках. Было проведено исследование, которое показало, что возможности воксельной графики позволяют реализовать эти задачи, и что виды отображения на основе ее могут служить мощным дополнением к видам отображений на основе полигональной графики. Возможности воксельной графики позволяют отображать внутреннюю структуру всей сетки и ее блоков (используя метафору прозрачности), а уровень современных сечений и изоповерхностей как всей сетки, так и отдельных ее участков

Возможна аппаратная реализация Ray Casting'a. Причем двухпроходный алгоритм реализуется с использованием DirectX9 Pixel Shaders 3.0, где можно использовать циклы и которая поддерживает в настоящее время незначительное число видеокарт (NVIDIA GeForce 6800). Если же использовать DirectX9 Pixel Shaders 2.0, которые поддерживает значительное число видеокарт (начиная с NVIDIA GeForce 5200 и ATI Radeon 9200), придется выполнить несколько проходов, выбирая каждый раз только те воксели, которые расположены на заданной дистанции от точки пересечения луча с передней гранью куба (дистанция при этом меняется в зависимости от прохода).

Данный метод позволяет получать качественные изображения в реальном времени.

Следует отметить, что выбор уровня прозрачности (transfer function) для каждого вокселя оказывает существенное влияние на конечное изображение. То, как это будет сделано, зависит от конкретной задачи визуализации объемных данных.

Использование Pixel Shader'ов позволяет строить изоповерхности с использованием воксельной графики без предварительного просмотра исходных данных, чего не удастся сделать при использовании полигональной графики. Эта технология позволяет получать различные изоповерхности без уменьшения общей производительности.

При отображении сеток с целью анализа структуры ее блоков и данных важную роль играет интерактивность полученного вида отображения. Поэтому, в качестве методик отрисовки вокселей были выбраны методики, основанные на 3D-текстурах. Для отображения вокселей был использован метод Splatting'a и проведено пробное использование аппаратного Ray Casting'a.

Прежде чем отображать сетку с применением 3D-текстуры, необходимо конвертировать данные сетки в соответствующий формат, поддерживаемый видеокартой. Хотя современные видеокарты и поддерживают текстуры в формате с плавающей точкой, такие текстуры занимают в 4 раза больше памяти, чем обычные текстуры в формате R8G8B8A8 (по 8 бит по красный, зеленый, синий каналы и 8 бит на уровень прозрачности), что позволяет использовать текстуры до 256*256*256 на 128Mb видео памяти.

Сетки, используемые для расчетов задач математической физики, являются многоблочными регулярными прямоугольными, а 3D-текстура является регулярной кубической сеткой. Поэтому, прежде всего, необходимо было конвертировать исходную сетку в регулярную кубическую.

После конвертации исходной сетки в регулярную кубическую, можно приступать к отрисовке 3D-текстуры, предварительно выбрав цвет и уровень прозрачности для каждого элемента текстуры.

Большую роль при анализе структуры сетки и данных на ней играет возможность построения сечений. При использовании 3D-текстур сечения сетки легко строятся в реальном времени, причем плоскость сечения может быть

произвольной. Для этого, находятся пересечения плоскости сечения с ребрами воксельного куба, по найденным точкам строим набор треугольников с заданными в вершинах текстурными координатами (u, v, w), полученные методом линейной интерполяции (текстурные координаты в вершинах воксельного куба известны), и затем отрисовываем полученные треугольники. Закраску треугольников, используя трилинейную интерполяцию, обеспечивает видеокарта. Производя манипуляции с текстурными координатами в вершинах воксельного куба можно достичь некоторых полезных эффектов, напр. увеличение или уменьшение просматриваемой области сетки в воксельном кубе.

Был разработан вид отображения, позволяющий в режиме реального времени просматривать динамику сечений отображаемой сетки – “режим радара”. В нем производится перемещение секущей плоскости от одной стороны воксельного куба к другой, что и позволяет просматривать динамику сечений параллельными плоскостями. Присутствует возможность отображение содержимого воксельного куба от текущего сечения до края куба.

Проведенные исследования показывают, что использование воксельной графики является перспективным направлением в области визуализации сеток и данных на них. Наряду с использованием полигональной графики для отображения сеток планируется активное использование воксельной графики в качестве альтернативного вида отображения, что позволит предоставить пользователю дополнительные инструменты для анализа структуры сетки и данных на ней.

Планируется продолжать работы по исследованию методик отрисовки воксельной графики, в частности, большой интерес представляет возможность использования программируемых вершинных и пиксельных конвейеров современных видеокарт, что позволит строить изоповерхности в реальном времени и повысить качество получаемых изображений

Также планируется предоставить возможность пользователю просматривать в воксельном виде отдельные участки сетки с целью улучшения детализации, посредством пересоздания 3D-текстуры для заданного участка, а также производить дальнейшие разработки пользовательского интерфейса с последующей интеграцией с системой полигональных видов отображений сеток и данных на них.

Использование параллельных вычислителей для воксельного отображения больших сеток также представляет интерес.

Программа, осуществляющая конвертацию сеток в формат 3D-текстуры с последующей ее отрисовкой с использованием воксельной графики (Splatting), написана на Visual C++ с использованием интерфейсов DirectX.

Перспективы данной системы следующие:

- Построение изоповерхностей в реальном времени, используя программируемые вершинные и пиксельные конвейеры современных видеокарт.
- Дальнейшее улучшение качества получаемого изображения за счет реализации аппаратного Ray Casting'a.
- Доработка пользовательского интерфейса в целом и системы навигации для создания возможности выделения участка сетки с целью более подробного рассмотрения – путем пересчета 3D-текстуры для выделенного участка.
- Интеграция с системой полигональных видов отображений сеток и данных на них.
- Использование параллельного вычислителя для воксельного рендеринга больших сеток.

5. ЗАКЛЮЧЕНИЕ

В настоящее время разработана транспортная подсистема, которая позволяет нескольким программам (выполняющимся на вычислителе или пользовательских станциях) обмениваться сообщениями.

Разработаны виды отображения для сеточных задач, базирующихся на трехмерных образах, генерация которых идет как с использованием методик триангуляции, так и с использованием воксельной графики. В настоящее время реализованы анимационные виды отображения, дающие представление об общем виде сеток, их сечениях и некоторых особенностях.

Разработаны соответствующие демонстрационные программы.

Работа выполнена при поддержке гранта РФФИ № 04-07-90120.

6. ЛИТЕРАТУРА

- [1] Авербух В.Л., Манаков Д.В, Васёв П.А., Комаровский И.А., Мухачёв А.А., Шинкевич А.Н. Подходы к реализации средств on-line визуализации параллельных вычислений // Супервычисления и математическое моделирование: Тезисы международного семинара, г. Саров, ВНИИЭФ-РФЯЦ, 2003, С. 14-16.
- [2] Geist G.A., Kohl J. A., Papadopoulos P.M. CUMULVS: Providing Fault-Tolerance, Visualization and Steering of Parallel Applications // International Journal of High Performance Computing Applications, Volume 11, Number 3, August 1997, pp. 224-236.
- [3] Таненбаум Э. Распределенные системы. Принципы и парадигмы. – СПб.: Питер, 2003. – 877 с.