Grid-gen: инструмент для генерации численных кодов

Кайгородов П.В.*

Аннотация

Рассматривается архитектура и основные принципы работы **grid-gen** – программы для генерации численных кодов. Цель разработки **grid-gen** – создание удобного инструмента для автоматической генерации высокооптимизированных параллельных численных кодов, предназначенных, прежде всего для решения астрофизических задач. **Grid-gen** позволяет независимо описывать геометрию сетки, определять моделируемые физические процессы, численную схему, начальные и граничные условия. Кроме того, независимо может быть выбрана схема распараллеливания, что позволяет с минимальными изменениями запускать полученный код как на многоядерных рабочих станциях, так и на больших кластерах, использующих MPI и GPU-ускорители. Результатом работы **grid-gen** является оптимизированный код на языке Fortran, включающий (при необходимости) OpenCL-ядра, директивы OpenMP и вызовы MPI. В дальнейшем планируется добавление поддержки C/C++, CUDA и Xeon Phi. В настоящий момент **grid-gen** находится в состоянии активной разработки.

1 Введение

Решение астрофизических задач невозможно без проведения масштабных численных расчетов. Современные задачи требуют применения весьма продвинутых средств моделирования, включающих учет множества физических явлений, исследование многих процессов требует проведения расчетов с высоким пространственным и временны**ы**м разрешением. Кроме того, повышение точности расчетов зачастую невозможно без применения неравномерных, в общем случае – движущихся сеток. Для решения систем физических уравнений в современных кодах применяются весьма сложные численные методы, требующие большого количества операций для вычисления значений на каждом временном шаге. Наконец, высокая вычислительная сложность современных численных моделей требует применения для расчетов высокопроизводительных вычислительных средств, соответственно, коды должны быть к ним адаптированы.

Все эти требования приводят к необходимости создания весьма сложных и громоздких численных кодов, разработка и отладка которых может занимать месяцы и годы. При этом коды должны обеспечивать определенный уровень гибкости, позволяя решать широкий спектр задач, для чего необходима поддержка различных типов граничных условий, возможность модификации физической модели и т.п., что зачастую сказывается на их производительности. Наконец, сложность современных кодов крайне осложняет их отладку и верификацию, практика показывает, что серьезные ошибки могут оставаться необнаруженными в течение лет даже в активно используемых кодах.

С точки зрения исследователя, идеальный численный код должен обладать следующими свойствами:

- Возможность независимой модификации различных его частей – физической модели, начальных и граничных условий, геометрии сетки, численной схемы, методики распараллеливания. При этом изменение одной его части не должно никак затрагивать остальные, например, изменение системы уравнений не должно отражаться на численной схеме, а переход от распараллеливания посредством MPI к OpenMP и обратно не должен влиять на часть кода, отвечающего, например, за сетку.

*email: pasha@inasan.ru, Институт астрономии РАН

- Код должен быть оптимизирован для всего набора современных вычислительных архитектур, включая, в том числе, поддержку кластеров, состоящих из многопроцессорных узлов, имеющих несколько GPU-ускорителей на каждом. Оптимизация должна обеспечивать достижение наивысшей возможной производительности на всех типах архитектур.

- Код должен поддерживать широкий спектр физических и математических моделей, давая, тем не менее возможность глубокого изменения любой своей части для реализации особых требований задачи.

У сожалению, эти требования трудносовместимы в одном коде. Тем не менее, возможно создание инструментов, облегчающих построение численных кодов, попыткой разработки такого инструмента является **grid-gen**. Система **grid-gen** проектировалась таким образом, чтобы, по возможности, удовлетворить всем вышеперечисленным условиям, оставаясь при этом надежным и простым в использовании практическим инструментом. Для реализации **grid-gen** был выбран LISP-подобный язык Scheme, имеющий широкие возможности для мета-программирования. На данный момент общий размер написанного кода составляет порядка 13 000 строк. **Grid-gen** включает в себя следующие модули:

1. Базовый модуль, служащий для связывания и поддержки всех остальных модулей

2. Модули генерации кода, непосредственно переводящие внутреннее представление программной логики в код на языках Fortran и OpenCL (написание модулей для других языков планируется)

3. Модуль оптимизации, сокращающий размер генерируемого кода, а также повышающий его эффективность путем очистки от ненужных/повторных вычислений

4. Модуль газодинамики, содержащий систему уравнений газовой динамики, матрицу гиперболичности, а также наборы ее собственных векторов и собственных значений (планируется добавление модуля МГД и обобщение для многожидкостных моделей)

5. Модули численных схем адвекции (на данный момент поддерживаются схемы Роу и Лакса-Фридрихса)

6. Модуль TVD (поддерживается метод Ошера)

7. Модули энтропийных поправок (Эйнфельдта, Хартена и пр.)

8. Модуль (топологически)-декартовых сеток, поддерживающий генерацию сеток произвольной геометрии (поддержка движущихся сеток планируется)

9. Модуль составных сеток, позволяющий проводить вычисления на соединенных между собой сетках, например, на т.н. «кубизированной сфере»

10. Модули символьной математики, используемые вышеперечисленными модулями для символьных преобразований: дифференцирования, операций над матрицами и векторами, а также для элементарного упрощения выражений

11. Модули распараллеливания (поддерживается MPI, OpenMP, OpenCL)

12. Вспомогательные модули, например, модуль генерации файлов в формате OpenDX, модуль физических констант и т.п.

Создавая код при помощи **grid-gen** пользователь подключает необходимые модули (например, схему Роу, поправку Ошера, MPI и OpenMP), при этом некоторые модули принимают параметры для настройки. Так, например, модуль MPI требует указания максимального и минимального числа параллельных процессов (эти числа могут совпадать) для эффективного разделения сетки на подобласти. Указывается разрешение сетки (при этом сетка может быть одно, двух или трехмерной) и закон преобразования координат. Описываются начальные и граничные условия, при этом для граничных условий

возможен как выбор из готового набора (константные условия, «свободное втекание» или «твердая стенка») так и задание специфичных для задачи условий. Описывается основной цикл работы кода, включающий создание точек сохранения, вывод отладочной информации и т.п. На данный момент для описания кода используется нотация Scheme, в дальнейшем планируется разработка упрощенного внутреннего языка. Типичный размер описания кода на данный момент составляет 100-200 строк.

После запуска, **grid-gen** составляет полную систему уравнений, вычисляемую на каждом шаге. Будучи расписанной «в лоб», подобная система может быть весьма громоздкой, например, схема Роу-Ошера-Эйнфельдта для полной трехмерной системы уравнений газовой динамики порождает выражения с более чем 100 000 операций. Модуль оптимизации позволяет сократить число операций до нескольких сотен, вынося «общие части» выражений и обеспечивая повторное использование величин, вычисленных при обработке предыдущей ячейки сетки. Далее, модуль MPI добавляет код синхронизации, после чего генератор кода создает сам текст программы на языке Fortran, добавляя директивы OpenMP в местах, помеченных соответствующим модулем. В случае, если подключен модуль OpenCL, генерируются также OpenCL-ядра и в соответствующие места кода вставляются вызовы подпрограмм, производящих копирование данных и запуск ядер на счет. Полученный таким образом код сохраняется в виде отдельного файла (нескольких файлов при использовании OpenCL), который в дальнейшем может быть скомпилирован и запущен. Общий размер сгенерированного кода может составлять 10-15 тысяч строк.

В докладе будет представлено описание основных алгоритмов и принципов работы **grid-gen**, примеры тестовых задач, а также обозначено направление будущего развития системы.