УДК 532.5.013

ЧИСЛЕННОЕ РЕШЕНИЕ СИСТЕМЫ УРАВНЕНИЙ НАВЬЕ - СТОКСА С ИСПОЛЬЗОВАНИЕМ ПЛАТФОРМЫ CUDA

Лепп Э. И.
Научный руководитель - Спирин Е. А.
Сибирский федеральный университет

Численное моделирование течений жидкостей и газов получает в последние десятилетия все более широкое распространение благодаря интенсивному росту производительности вычислительной техники. Вычислительная гидродинамика — подраздел механики сплошных сред, включающий совокупность физических, математических и численных методов, предназначенных для вычисления характеристик потоковых процессов. Зарубежное название — (англ. *Computational fluid dynamics, CFD*),

Скорость решения задачи *CFD* зависит от математической формулировки задачи, способа дискретизации уравнений и как следствие от метода решения системы уравнений, от реализации численных методов решения, но в итоге основное влияние оказывает производительность аппаратной платформы, на которой производятся вычисления.

Долго время производители процессорных устройств вели гонку за увеличение тактовой частоты центральных процессоров (CPU). Всего за несколько лет в результате этой гонки тактовые частоты существенно выросли, особенно после выхода процессора Intel Pentium 4. Но гонка быстро приближалась к пределу. После волны огромного прироста тактовых частот (между 2001 и 2003 годами тактовая частота Pentium 4 удвоилась с 1,5 до 3 ГГц), пользователям пришлось довольствоваться десятыми долями гигагерц, которые смогли выжать производители (с 2003 до 2005 тактовые частоты увеличились всего с 3 до 3,8 ГГц).

Для широкого круга задач существенное увеличение производительности дают параллельные вычислительные системы – физические компьютерные, а также программные системы, реализующие тем или иным способом параллельную обработку данных на многих вычислительных узлах.

Сегодня, когда масштаб вычислений требует существенных вычислительных мощностей, для исследовательских, инженерных и других задач, необходим рациональный подход к использованию аппаратного обеспечения. Таким подходом стало использование в вычислениях графических процессоров. Развитие этого направления было обусловлено высокой стоимостью аппаратного обеспечения, способного обрабатывать огромные объемы информации. Так появились технологии, которые позволяют реализовывать на обычных персональных компьютерах, с соответствующим программным и аппаратным обеспечением, миниатюрные вычислительные центры.

Так в 2007 году появилась программно - аппаратная архитектура, позволяющая производить вычисления с использованием графических процессоров NVIDIA, поддерживающих технологию GPGPU (произвольных вычислений на видеокартах). Впервые появились на рынке с выходом чипа NVIDIA восьмого поколения — G80 и присутствует во всех последующих сериях графических чипов, которые используются в семействах ускорителей GeForce, Quadro и NVidia Tesla.

CUDA SDK позволяет программистам реализовывать на специальном упрощённом диалекте языка программирования Си алгоритмы, выполнимые на

графических процессорах NVIDIA, и включать специальные функции в текст программы на Си. CUDA даёт разработчику возможность по своему усмотрению организовывать доступ к набору инструкций графического ускорителя и управлять его памятью, организовывать на нём сложные параллельные вычисления

CUDA (англ. *Compute Unified Device Architecture*) — программно-аппаратная архитектура (рис. 1), позволяющая производить вычисления с использованием графических процессоров NVIDIA, поддерживающих технологию GPGPU (произвольных вычислений на видеокартах).

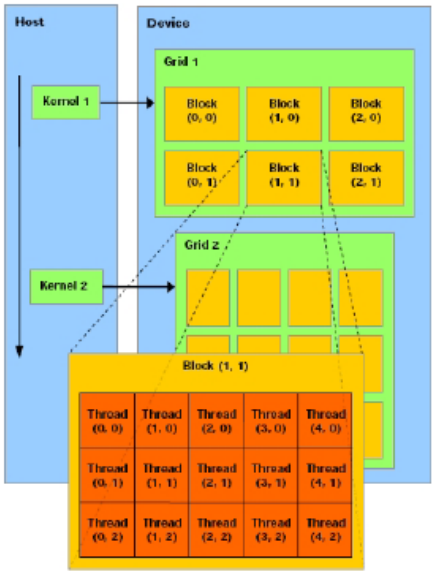

Рисунок 1 - Модель платформы CUDA

В основе *CUDA API* лежит язык Си с некоторыми ограничениями. Для успешной трансляции кода на этом языке, в состав *CUDA SDK* входит собственный Си-компилятор командной строки *nvcc* компании Nvidia. Компилятор *nvcc* создан на основе открытого компилятора *Open64* и предназначен для трансляции *host*-кода (главного, управляющего кода) и device-кода (аппаратного кода) (файлов с расширением .cu) в объектные файлы, пригодные в процессе сборки конечной программы или библиотеки в любой среде программирования.

Преимущества:

По сравнению с традиционным подходом к организации вычислений общего назначения посредством возможностей графических *API*, у архитектуры *CUDA* отмечают следующие преимущества в этой области:

1. Интерфейс программирования приложений *CUDA* (CUDA API) основан на стандартном языке программирования Си с некоторыми ограничениями. По мнению разработчиков, это должно упростить и сгладить процесс изучения архитектуры *CUDA;*
2. Разделяемая между потоками память (shared memory) размером в 16 Кб может быть использована под организованный пользователем кэш с более широкой полосой пропускания, чем при выборке из обычных текстур;
3. Более эффективные транзакции между памятью центрального процессора и видеопамятью;
4. Полная аппаратная поддержка целочисленных и побитовых операций.

Для оценки производительности было решено применить вычислительные ресурсы графических процессоров для решения задачи внешнего обтекания,

формализованной с использованием уравнений Навье - Стокса. Задача решается сеточным методом в двумерной постановке для несжимаемой жидкости.

Математическая формулировка решаемой задачи описывается уравнениями Навье-Стокса в проекциях на оси координат, и дополняется уравнением неразрывности :

$$\frac{\partial U_x}{\partial t}+U_x\frac{\partial U_x}{\partial x}+U_y\frac{\partial U_x}{\partial y}+U_z\frac{\partial U_x}{\partial z}=\nu\left(\frac{\partial^2 U_x}{\partial x^2}+\frac{\partial^2 U_x}{\partial y^2}+\frac{\partial^2 U_x}{\partial z^2}\right)-\frac{1}{\rho}\frac{\partial P}{\partial x}+F_x, \quad (1)$$

$$\frac{\partial U_y}{\partial t}+U_x\frac{\partial U_y}{\partial x}+U_y\frac{\partial U_y}{\partial y}+U_z\frac{\partial U_y}{\partial z}=\nu\left(\frac{\partial^2 U_y}{\partial x^2}+\frac{\partial^2 U_y}{\partial y^2}+\frac{\partial^2 U_y}{\partial z^2}\right)-\frac{1}{\rho}\frac{\partial P}{\partial y}+F_y,$$

$$\operatorname{div}\vec{U}=\frac{\partial U_x}{\partial x}+\frac{\partial U_y}{\partial y}+\frac{\partial U_z}{\partial z}=0. \quad (2)$$

Для определения распределения давления в пространстве решается уравнение Пуассона:

$$\Delta P=\frac{\partial^2 P}{\partial x^2}+\frac{\partial^2 P}{\partial y^2}=-\rho\left(\left(\frac{\partial U_x}{\partial x}\right)^2+2\frac{\partial U_x}{\partial y}\frac{\partial U_y}{\partial x}+\left(\frac{\partial U_y}{\partial y}\right)^2+\frac{\partial D}{\partial t}+U_x\frac{\partial D}{\partial x}+U_y\frac{\partial D}{\partial y}-\nu\Delta D-\operatorname{div}\vec{F}\right). \quad (3)$$

где, $D=\operatorname{div}\vec{U}=\frac{\partial U_x}{\partial x}+\frac{\partial U_y}{\partial y}$.

Решение уравнения Пуассона выполнено методом Либмана:

$$\Delta P=\frac{\partial^2 P}{\partial x^2}+\frac{\partial^2 P}{\partial y^2}=A. \quad (4)$$

$$\frac{P_{i+1,j}-2P_{i,j}+P_{i-1,j}}{h^2}+\frac{P_{i,j+1}-2P_{i,j}+P_{i,j-1}}{h^2}=A_{i,j}, \quad (5)$$

$$P_{i,j}=\tfrac{1}{4}\left(P_{i+1,j}+P_{i-1,j}+P_{i,j+1}+P_{i,j-1}-h^2A_{i,j}\right) \quad (6)$$

Дискретизация производных выполнена с применением явной разностной схемы ВВЦП (вперед по времени, центральная по пространству)

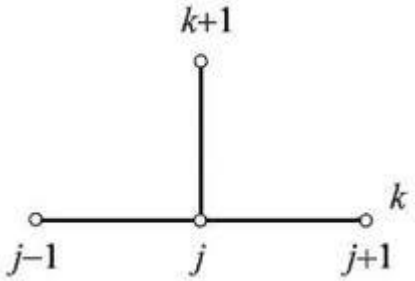

Рисунок X – Явная разностная схема ВВЦП

Для повышения устойчивости при вычислении слагаемых $U_x\frac{\partial D}{\partial x}$ и $U_y\frac{\partial D}{\partial y}$

используют противоточные производные:

$$U_x \frac{\partial A}{\partial x} \Rightarrow \begin{cases} U_{xi,j} \dfrac{A_{i,j} - A_{i-1,j}}{h}, U_{xi,j} \geq 0; \\ U_{xi,j} \dfrac{A_{i+1,j} - A_{i,j}}{h}, U_{xi,j} < 0. \end{cases} \quad (7)$$

Для обеспечения устойчивости сформулированного явного численного решения использован критерий Куранта–Фридрихса–Леви [4].

$$\frac{u_x \Delta t}{\Delta x} + \frac{u_y \Delta t}{\Delta y} < C. \quad (8)$$

Сравнительные тесты на производительность фирмы NVIDIA показали, четырнадцати кратный прирост производительности в большинстве задач, без проведения оптимизации алгоритма работы вычислительных ядер. Тогда как после мероприятий по оптимизации кода и алгоритмов работы вычислительных ядер, производительность достигает трехсот тридцати кратного повышения. В частности, рассматриваемая выше задача, с использованием технологии CUDA, по предварительным расчетам, может быть реализована в режиме реального времени с возможностью варьирования входных параметров.