Кочуров А.В.

РЕШЕНИЕ НЕЯВНЫХ СЕТОЧНЫХ УРАВНЕНИЙ НА ГРАФИЧЕСКИХ ВЫЧИСЛИТЕЛЬНЫХ УСТРОЙСТВАХ МЕТОДОМ ПИРАМИД

(ФГБОУ ВПО «Самарский государственный аэрокосмический университет им. академика С.П. Королева (национальный исследовательский университет)»)

Вопросам решения сеточных уравнений на графических процессорах (GPU) посвящено множество работ [1,2]. Приведенные в этих работах методы требуют, чтобы сеточная область целиком помещалась в видеопамяти. При этом объем видеопамяти позволяет обрабатывать трехмерные сеточные области размером в сотни узлов [2]. На практике, например, при проектировании оптических элементов с линейными размерами в микроны и нанометровыми неоднородностями, требуемый размер сеточной области составляет тысячи узлов по одному направлению [3]. В связи с этим существует потребность в методах решения сеточных уравнений, не требующих размещения всей сеточной области в видеопамяти. Тривиальный подход позволяет решить эту задачу путем увеличения коммуникаций между видеокартой и центральным процессором, однако это может привести к значительной деградации производительности.

Для явных сеточных уравнений был ранее предложен подход, основанный на модификации метода пирамид [4], были рассмотрены случай двумерной и трехмерной сеточной области. Данный метод позволяет варьировать длительность коммуникаций за счет дублирования вычислений и тем самым минимизировать общее время выполнения.

В настоящей работе представлен метод решения неявных сеточных уравнений методом пирамид.

Постановка задачи

В качестве иллюстрации предлагаемого метода рассмотрим следующее линейное одномерной однородное уравнение теплопроводности [5] $\frac{\partial}{\partial t}U(x,t)=\alpha^2\frac{\partial^2}{\partial x^2}U(x,t)$ с начальным условием $U(x,0)=\phi(x)$ с краевым условием $U(0,t)=U(W,t)=C$.

Простейшая неявная разностная схема для этого уравнения имеет вид:

$$\left(1+2\alpha^2\frac{h_t}{h_x^2}\right)U_i^{(k+1)}-\alpha^2\frac{h_t}{h_x^2}U_{i+1}^{(k+1)}-\alpha^2\frac{h_t}{h_x^2}U_{i-1}^{(k+1)}=U_i^{(k)} \qquad (1)$$

Обозначим $a=-\alpha^2\frac{h_t}{h_x^2}$, $b=1+2\alpha^2\frac{h_t}{h_x^2}$. Перепишем (1) в матричном виде: $AU^{(k+1)}=U^{(k)}$, где $U^{(k)}=\left(U_1^{(k)},U_2^{(k)},\dots U_N^{(k)}\right)^T$ - сеточная функция, $A\in R^{N\times N}$ - трехдиагональная матрица, элементы главной диагонали которой равны b, а элементы под и над главной диагональю — a.

Поставим следующую задачу: наобходимо по заданному начальному распределению температур $U^{(0)}$ вычислить сеточную функцию через *T* шагов разностной схемы по времени. Это можно сделать, последовательно вычисляя значения *U* на каждом слое по времени с применением циклической редукции или иного алгоритма решения СЛАУ (1).

В случае, если видеопамять не позволяет разместить значения *U* на одном слое по времени, на каждом шаге потребуется копировать в видеопамять участки вектора, и пересылать результаты обратно. Будем в дальнейшем этот подход именовать «тривиальным алгоритмом». Однако шина между графическим процессором (GPU) имеет ограниченную пропускную способность, и подобные пересылки приводят к быстрой деградации вычислений.

Метод пирамид

В настоящей работе предлагается следующий подход:

– процесс вычисления значений $U^{(T)}$ будет разделен на несколько этапов, на которых будут вычислены и сохранены в ОЗУ значения $U^{(h)}$, $U^{(2h)}$, $U^{(3h)}$, $\ldots U^{(T)}$, причем единовременно необходимо сохранять в ОЗУ значения только на одном (последнем вычисленном) слое по времени. Значения на промежуточных слоях не будут целиком представленны в ОЗУ; величину *h* будем называть высотой пирамиды.

1. вектор *U* разбивается на части, каждая из которых представляет собой *R*-компонентный вектор, причем в видеопамяти может быть размещено *R + 2 h r* значений сеточной функции;

2. значения этих частей будем сохранять в видеопамяти в следующем виде: $Y_r^{(k)} = \left(\underbrace{0,0,\ldots 0}_{hr\ нулей} U_{rR}^{(k)}, U_{rR+1}^{(k)}, \ldots U_{(r+1)R-1}^{(k)} \underbrace{0,0,\ldots 0}_{hr\ нулей}\right)$

3. для вычисления $U^{((m+1)h)}$ на основе $U^{(mh)}$ на видеоустройстве обрабатываются блоки Yr следующим образом: $Y_r^{(k+1)} = A^{-1}_{rR-hr:(r+1)R+hr,:} Y_r^{(k)}$, оператор применяется последовательно h раз, полученные в результате вектора пересылаются в ОЗУ и суммируются с накопленным ранее результатом $U^{((m+1)h)}$ с учетом сдвига векторов на r R — h r позиций.

4. Алгоритм для вычисления U(K) выглядит следующим образом:

5. исходная сеточная функция U(0) размещается в ОЗУ;

6. U(k) разбивается на блоки, шаги 3-5 выполняются для каждого блока;

7. значения UrR:(r+1)R-1(k) пересылаются в видеопамять и дополняются нулями, таким образом получается Yr(k);

8. к Yr(k) векторам применяется оператор A-1 h раз, в результате получаем Y(k+h);

9. значения Yr(k+h) пересылаются в ОЗУ;

10. значения U(k+h) получаются путем суммирования Yr(k+h);

11. шаги 2-6 повторяются T/h раз, пока не будут получены значения U(T).

12. Как видно, пересылки м/у ОЗУ и GPU требуются только на каждом h-том шаге по времени, при этом требуется пересылка несколько большего объема данных, чем при использовании тривиального подхода, а также увеличивается общий объем вычислений.

На рисунке 1 представлена схема алгоритма (шаги 2-6). Цифрами на рисунке обозначены шаги алгоритма.

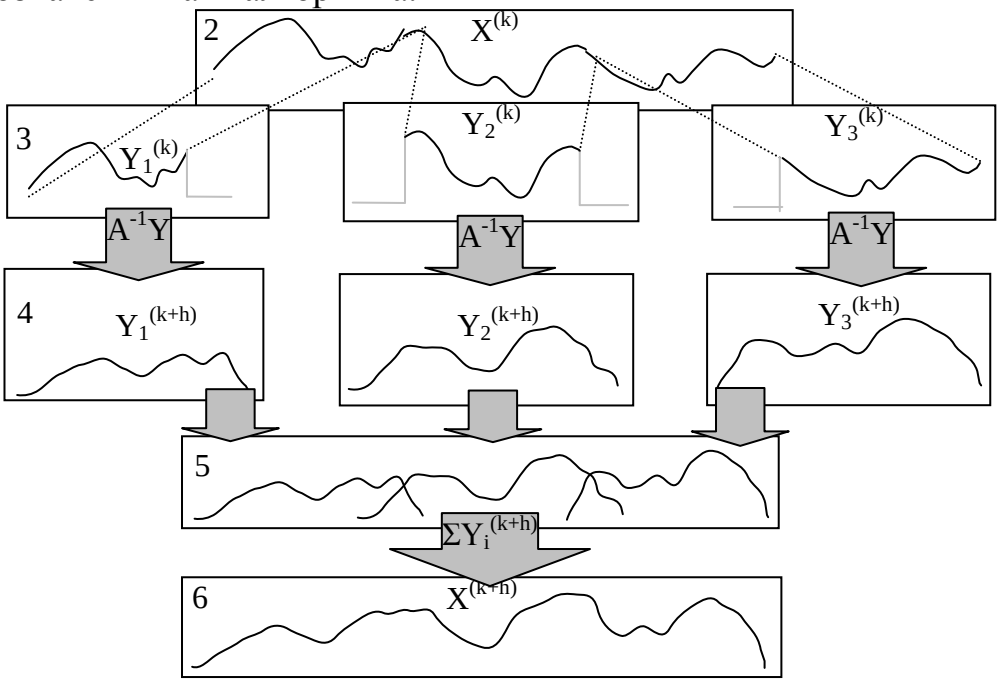

Рис. 1. Общая схема работы алгоритма

Утверждается, что при некоторых ограничениях на значения *a* и *b* выбирая *h* и *r* можно достичь сколь угодно малой погрешности вычислений.

Оценка производительности

Для вычисления *T* слоёв по времени потребуется сделать *T/h* проходов методом пирамид. На каждом проходе будут обработаны все *I*/(*M*-2*r*) блоков, где *M* — число значений, которое можно разместить в видеопамяти. При обработке каждого блока *M*-2*r* значений необходимо переслать в видеопамять и *M* значений — обратно.

Таким образом, для решения поставленной задачи требуется решение $\frac{TI}{M-2r}$ трехдиагональных СЛАУ размерности *M*. Длительность решения одного СЛАУ на GPU можно оценить как $2\left(2\frac{M}{\mu}+\log_2\frac{M}{\mu}\right)\tau_{CR}$, где μ - число ядер на GPU, τ_{CR} - средняя длительность одной редукции на одном ядре.

Если для решения СЛАУ на каждом временном слое каждого блока применять циклическую редукцию, то общая длительность вычислений составит $2\frac{TI}{M-2r}\left(2\frac{M}{\mu}+\log_2\frac{M}{\mu}\right)\tau_{CR}$. Также потребуется осуществить пересылку $2(M-r)\frac{TI}{M-2r}$ значений сеточной функции. Таким образом, общая длительность работы программы может быть оценена как

$\tau = 2\frac{TI}{M-2r}\left(2\frac{M}{\mu}+\log_2\frac{M}{\mu}\right)\tau_{CR}+2(M-r)\frac{TI}{M-2r}\tau_T$, где τ_T - среднее время пересылки одного значения сеточной функции. Если рассматривать r как функцию от высоты пирамид при условии ограниченности погрешности вычислений некоторой наперед заданной величиной, то варьируя высоту можно обеспечить минимальную длительность выполнения при обеспечении допустимой погрешности.

Литература

1. Евстигнеев Н.М. Интегрирование уравнения Пуассона с использованием графического процессора технологии CUDA // Вычислительные методы и программирование, 2009. – Т. 10, № 2. – с. 82 – 88

2. Adams S., Payne J., Boppana R. Finite difference time domain (FDTD simulations using graphics processors) // Proceedings DoD High Performance Computing Modernization Program Users Group Conference, 2007. – pp. 334 – 338

3. Pavelyev V.S., Karpeev S.V., Dyachenko P.N., Miklyaev Y.V. Fabrication of three-dimensional photonics crystals by interference lithography with low light absorption // Journal of Modern Optics, 2009. – Vol. 56, № 9. – pp. 1133 – 1136

4. Головашкин Д.Л., Кочуров А.В. Решение сеточных уравнений на графических вычислительных устройствах. Метод пирами // Вычислительные технологии, 2012. – Т. 17, №3. – с. 55 – 69

5. Самарский А.А. Теория разностных схем. – М.: Наука, 1977. – 656 с.