КЛЕТОЧНЫЙ АСИНХРОННЫЙ МЕТОД РЕШЕНИЯ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ В ЧАСТНЫХ ПРОИЗВОДНЫХ НА ГПУ

А.П. Карпенко, М.П. Погосский

Введение. Численное решение краевых задач для уравнений в частных производных (ДУЧП) является одной из центральных проблем математического моделирования. В силу высокой вычислительной сложности этих задач разработано большое число параллельных методов и алгоритмов их решения, ориентированных на различные классы параллельных вычислительных систем. В последние годы в связи с интенсивным внедрением в вычислительную практику графических процессорных устройств (ГПУ) значительное внимание уделяется разработке параллельных методов и алгоритмов, ориентированных на этот класс систем. Целью работы является исследование эффективности, так называемого клеточного асинхронного метода решения ДУЧП на ГПУ. В работе дана постановка задачи и описание указанного метода ее решения, приведена схема и результаты вычислительного эксперимента по исследованию погрешности метода с помощью имитационного моделирования. На основе результатов исследования обоснована целесообразность модификации метода, заключающаяся в периодической глобальной синхронизации асинхронных итераций. Рассматриваем используемую схему отображения алгоритма, реализующего модифицированный метод асинхронных итераций, на архитектуру ГПУ. Представляем результаты численных экспериментов на двух классах ГПУ. Формулируем основные результаты работы и перспективы ее развития.

Постановка задачи и схема метода. Рассматриваем для простоты записи одномерную модель распространения тепла в стержне в форме уравнения теплопроводности

$$\frac{\partial U(x,t)}{\partial t}=a^2\frac{\partial^2 U(x,t)}{\partial x^2} \tag{1}$$

при заданных граничных и начальных условиях.

Выполним дискретизацию задачи по пространству и времени с шагами h и t_h соответственно. Значения переменных x, t в узлах полученной сетки обозначим x_i, t_i, а значения непрерывной функции $U(x,t)$ обозначим $U(x_i,t_j)$. В результате ДУЧП (1) приобретает форму системы конечно-разностных уравнений

$$\frac{w_{i,j}-w_{i,j-1}}{t_h}-a^2\frac{w_{i+1,j}-2w_{i,j}+w_{i-1,j}}{h^2}=0,\ i=[1:m],\ j=[1:l] \tag{2}$$

с нулевыми начальными и граничными условиями. Здесь m, l – числа шагов дискретизации по переменным x, t соответственно, $w_{i,j}$ - компоненты искомой разностной функции W_h.

Введем обозначение $q_h=\frac{t_h a^2}{h^2}$. Тогда система (2) приобретает форму системы линейных алгебраических уравнений (СЛАУ) вида

$$w_{i,j-1}=(1+2q_h)w_{i,j}-q_h w_{i+1,j}-q_h w_{i-1,j},\ i=[1:m],\ j=[1:l]$$

или в матричной форме - вида

$$L_h W_h=U_h, \tag{3}$$

где L_h - разностный оператор, задающий отображение $L_h: R^n\to R^n$; U_h - сетевая функция.

В общем случае вид оператора L_h определяет используемый конечно-разностный шаблон. С точки зрения распараллеливания вычислений основным параметром шаблона является его степень соседства r - максимальное расстояние между центральным узлом шаблона и его периферийными узлами. Соседей узла (i,j), в смысле используемого шаблона, называем окружением этого узла, а их число обозначаем a_h.

Рассматриваем итерационный клеточный локально-асинхронный метод решения СЛАУ (3). В работе [1], в котором рассматривается теория этого метода, изложение ведется в терминах клеточных нейронных сетей. Поэтому можно сказать, что целью нашей работы является исследование эффективности реализации клеточных нейронных сетей на ГПУ.

Каждому из узлов $w_{i,j}$ рассматриваемой конечно-разностной сетки ставим в соответствие клетку $w_i(j); i=[1:n],\ j=[1:l]$, динамику которой определяет итерационная формула вида

$$w_i(j)=w_i(j-1)-\frac{K}{h^2}(l_i(w_k(s_k(j)),k\in J)-u_i), \tag{4}$$

где $s_k(j), 0 \le s_k(j) \le j$ - задержка между текущим временем t_j и временем $t_{s_k(j)}$, которому соответствует состояние клетки w_k, принадлежащей окружению рассматриваемого узла $w_{i,j}$; J - набор a_h штук номеров клеток этого окружения; $K = t_h a_h^2$. Отметим, что ситуация $s_k(j) = j$ определяет синхронный клеточный метод.

Базовое свойство асинхронного метода состоит в том, что для продолжения итераций в данном узле сеточной области не нужно ожидать завершения итераций в соседних узлах.

Исследование погрешности метода. Для экспериментального исследования погрешности асинхронного метода разработана его C++ имитационная модель. Рассматриваем одномерную задачу теплопроводности для стержня, температура на концах которого равна 60 и 100 градусов. Используем конечно-разностную сетку с числом узлов $n = 100$ и трех точечный шаблон. В каждый момент времени t_j производим по 100 итераций. Моделирование асинхронности вычислений выполняем по следующей схеме.

С каждой из клеток $w_i(j), i = [1:n]$ связываем ее вес $v_{i,j}$, определяемый формулами

$$v_{i,j} = v_{i,j-1} + d_h(1 - V_{i,j-1}),\ v_{i,j} < p;$$
$$v_{i,j} = v_{i,j-1} - d_h V_{i,j-1},\ v_{i,j} \ge p.$$

Здесь d_h - случайная величина равномерно распределенная в интервале $[0{,}01; D_h]$, p – величина порога. Полагаем, что обновление состояния клетки происходит только в том случае, когда ее вес превышает этот порог.

В качестве меры погрешности метода используем величины e_M, s_M - оценки математического ожидания (МО) и среднего квадратичного отклонения (СКО) максимальной ошибки разности решений задачи, полученных на указанном шаблоне методом конечных разностей и асинхронным методом. Некоторые результаты исследования иллюстрирует рисунок 1.

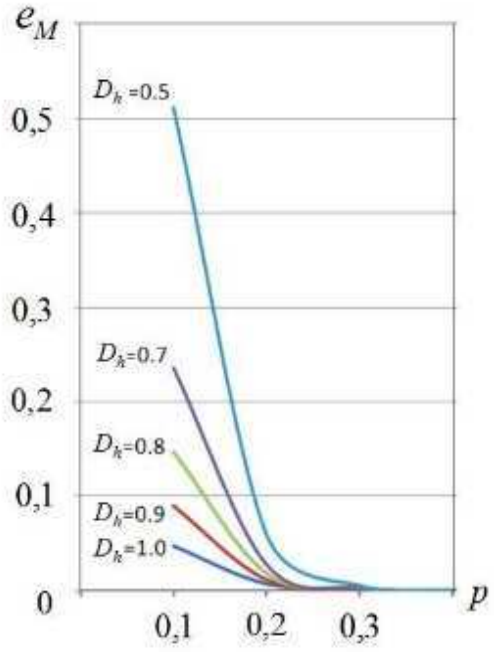


Рисунок 1 — Оценка МО погрешности метода e_{M} в функции величины порога p

Рисунок 1 показывает, что погрешность асинхронного метода может достигать неприемлемых величин. В результате возникает идея ограничить число асинхронных итераций для чего периодически проводить их глобальную синхронизацию. Далее рассматриваем только модифицированный указанным образом метод, называемый условно-асинхронным.

Отображение алгоритма на ГПУ. Условно-асинхронный клеточный метод реализован на архитектуре *NVidia CUDA* для трехмерной модели теплопроводности

$$\frac{\partial U(x,y,z,t)}{\partial t} = a^2\left(\frac{\partial^2 U(x,y,z,t)}{\partial x^2} + \frac{\partial^2 U(x,y,z,t)}{\partial y^2} + \frac{\partial^2 U(x,y,z,t)}{\partial z^2}\right), \quad (5)$$

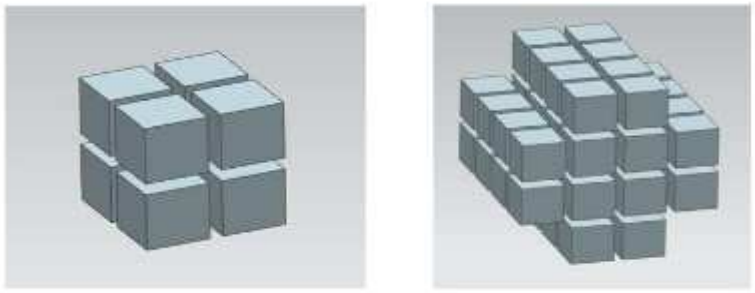
аналогичной модели (1). Расчеты выполнены для кубического тела, грани которого имеют постоянные температуры 50, 60, 70, 100, 120 и 200 градусов соответственно. Начальные условия внутри границ приняты нулевыми.

С точки зрения эффективности реализации соответствующего алгоритма на ГПУ, ключевой проблемой является проблема оптимального разбиения клеток на блоки. В работе выполнен анализ эффективности двух вариантов такого разбиения [2]. Полагаем, что используется семи точечный шаблон и каждой клетке соответствует *CUDA*-нить.

Первый вариант использует разбиение, представленное на рисунке 2а. На рисунке 2б к клеткам, показанным на рисунке 2а, добавлены клетки, информационно связанные с ними. Преимуществами данного разбиения являются возможность использования блоков большого размера и минимальное число смежных клеток из соседних блоков. В то же время, разбиение обладает существенными недостатками: не позволяет в полной мере использовать преимущества блочного чтения из памяти; требует использования достаточно сложной индексации клеток; из-за асинхронности алгоритма клетки блока оказываются сильно подверженными влиянию краевых эффектов [2].

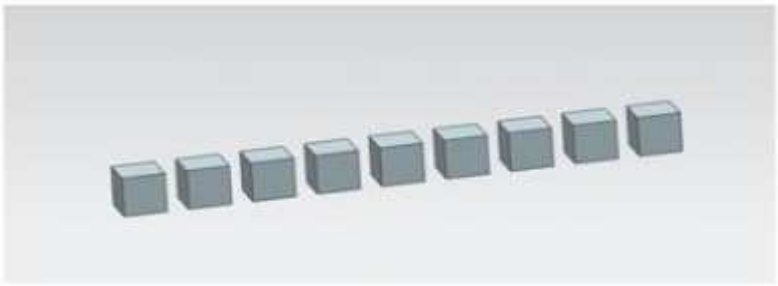
Второй вариант предполагает, что каждый из блоков представляет собой строку клеток по одному из измерений расчетной области (рисунок 3). Такое разбиение лишено всех недостатков первого варианта, но одновременно лишено и его преимуществ [2].

С целью ограничения периода асинхронного выполнения нитей в *CUDA*-программе, реализующей условно-асинхронный алгоритм, приняты следующие решения. Во-первых, число итераций в пределах каждого запуска блока ограничено величиной *N*, что ограничивает степень асинхронности метода. Во-вторых, производится барьерная синхронизация блоков, реализованная с помощью функции завершения выполнения всех нитей на ГПУ.

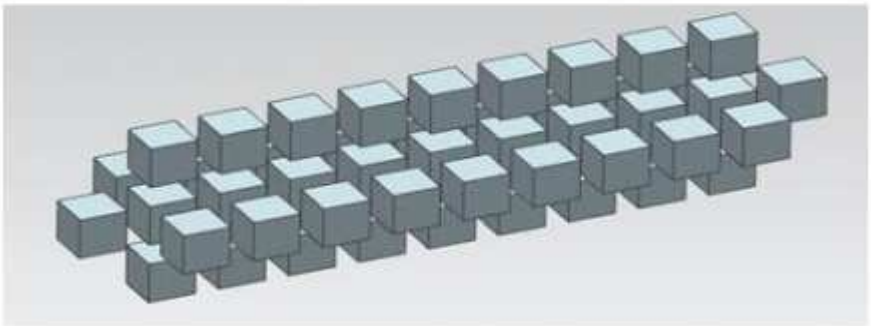


а) б)

Рисунок 2 - К первому варианту разбиения клеток на блоки



а)



б)

Рисунок 3 — Ко второму варианту разбиения на блоки

Программная реализация алгоритма поддерживает два отображения условно-асинхронного алгоритма на ГПУ: 1) с использованием только глобальной памяти; 2) с использованием разделяемой памяти. В зависимости от модели используемого ГПУ и размерности расчетной сетки оптимальной может быть как первая, так и вторая реализации.

Первая реализация использует второй вариант разбиения. Строки блоков ориентированы вдоль оси *z*. Нить имеет одномерный адрес в блоке (*threadIdx.x*), а блок - двухмерный адрес в расчетной сетке (*blockIdx.x*, *blockIdx.y*). Таким образом, клетку в пространстве (*x*, *y*, *z*) и соответствующую нить идентифицирует тройка (*threadIdx.x*, *blockIdx.y*, *blockIdx.z*).

Вторая реализация также использует второй вариант разбиения и указанную выше идентификацию нитей. Отличия реализаций состоит в том, что в данном случае перед пересчетом состояний клеток нити копируют в разделяемую память предыдущие и текущие значения величин, определяющих эти состояния. Кроме того, в разделяемую память копируются текущие состояния клеток-соседей.

Вычислительный эксперимент выполнен на системе, состоящей центрального процессора и двух ГПУ *GeForce GTS*250 и *GeForce GTX*470. Для оценки ускорения *S*, которое обеспечивает алгоритм и его программная реализация, он был реализован также на центральном процессоре системы. Все эксперименты выполнены на расчетной сетке 128x128x128. Традиционно при параллельном решении ДУЧП эффективность алгоритма иллюстрируют зависимостью ускорения, которое он обеспечивает, от числа узлов используемой расчетной сетки. Для асинхронного алгоритма более показательным является ускорение в функции параметра *K* (формула (4)), который определят скорость сходимости алгоритма, т. е. число итераций, необходимых для перехода клеточной сети в устойчивое состояние (которое и соответствует решению задачи). Указанную зависимость иллюстрируют рисунки 4, 5, первый из которых получен с помощью первой программной реализации и ГПУ *GeForce GTS*250, а второй - с помощью второй реализации и того же ГПУ. На указанных и

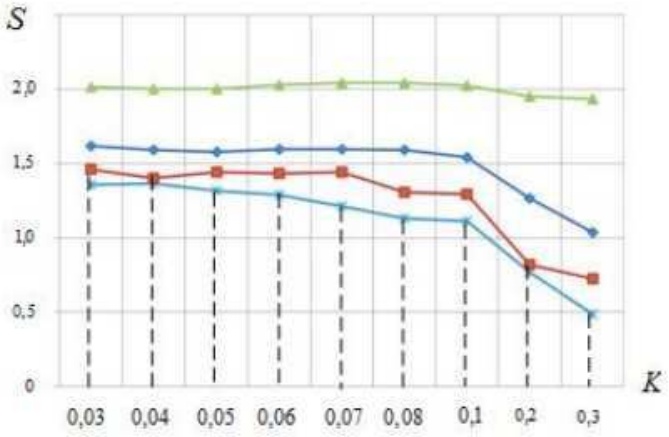
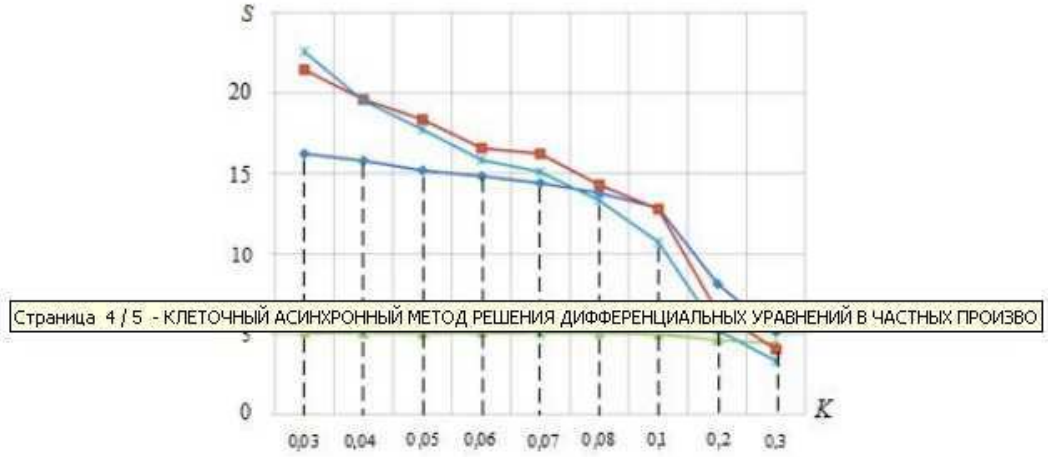


Рисунок 4 - Ускорение вычислений в функции параметра *K*: реализации 1; ГПУ *GeForce GTS*250



Страница 4 / 5 - КЛЕТОЧНЫЙ АСИНХРОННЫЙ МЕТОД РЕШЕНИЯ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ В ЧАСТНЫХ ПРОИЗВО

Рисунок 5 - Ускорение вычислений в функции параметра *K*: реализации 2; ГПУ *GeForce GTS*250

последующих рисунках треугольниками, ромбиками, квадратиками и крестиками помечены кривые для $N=1,5,10,15$ соответственно.

Рисунки показывают, как и следовало ожидать, значительно более высокую эффективность вычислений при использовании разделяемой, а не глобальной памяти. В реализации 1 ускорение слабо зависит от числа итераций *N*. Во второй реализации с увеличением *N* растет доля вычислений, в которых глобальная память не используется. Это обстоятельство обеспечивает более высокое ускорение реализации 2 по сравнению с первой реализацией. Однако с ростом величины *N* возрастает число итераций, необходимых для перехода сети в устойчивое состояние. По этой причине при $N>10$ ощутимый прирост ускорения не наблюдается. Существует достаточно широкий диапазон значений *K*, при которых ускорение меняется не слишком существенно.

Аналогичные результаты, полученные при использовании ГПУ *GeForce GTX*470, иллюстрируют рисунки 6, 7. Рисунки показывают, что первая реализация обеспечивает при $K\in[0,01;0,08]$ и всех рассматриваемых числах итераций *N* ускорение в диапазоне от примерно 20 до 110. Вторая реализация в тех же условиях дает ускорение от 20 до 130.

Заключение. В работе с помощью имитационной модели выполнено исследование эффективности асинхронного клеточного метода решения ДУЧП. На основе результатов исследования предложена модификация этого метода, предполагающая регулярную глобальную синхронизацию вычислений. Выполнено две *CUDA*-реализации соответствующего алгоритма, одна из которых предполагает использование только глобальной памяти ГПУ, а другая - разделяемой памяти. Проведено широкое экспериментальное исследование эффективности предложенной модификации метода, соответствующего алгоритма и указанного программного обеспечения. Результаты исследования показывают перспективность развития асинхронного клеточного метода решения ДУЧП на ГПУ.

В продолжении работы планируется исследование эффективности метода в функции других его свободных параметров и, в частности, в функции размерности расчетной сетки.

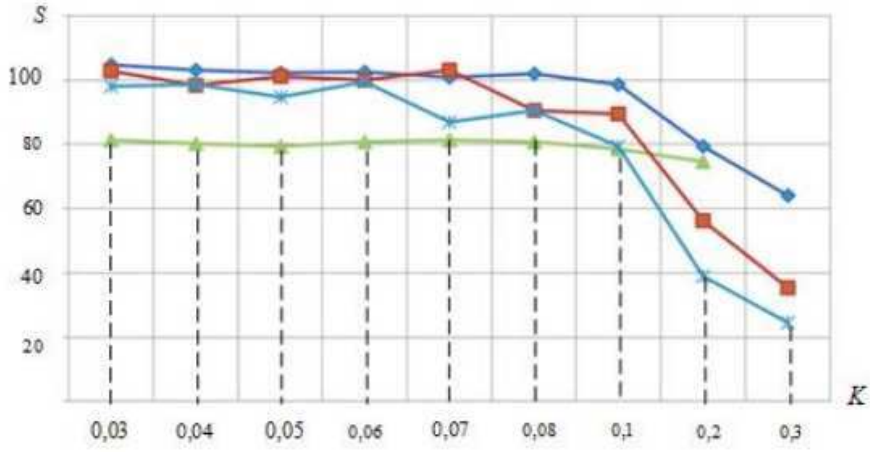


Рисунок 6 Ускорение вычислений в функции параметра *K*: реализации 1; ГПУ *GeForce GTX*470

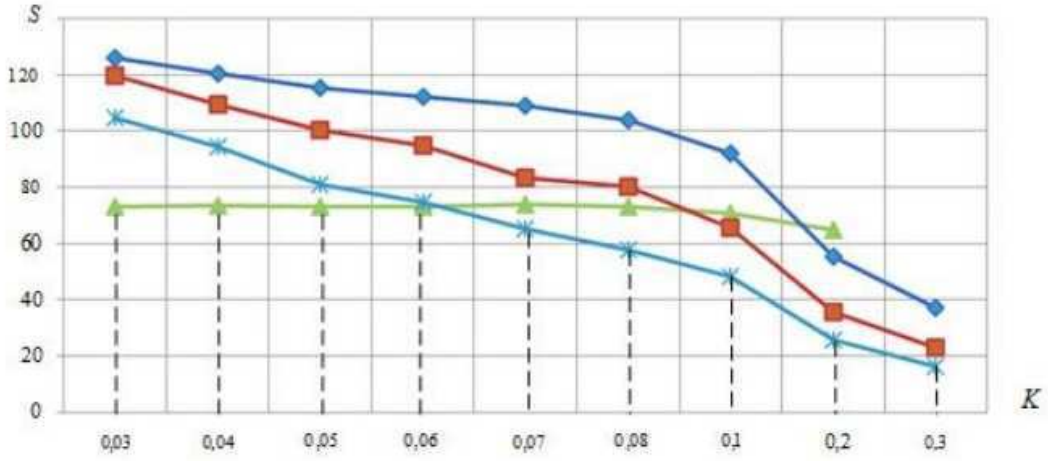


Рисунок 7 Ускорение вычислений в функции параметра *K*: реализации 2; ГПУ *GeForce GTX*470

ЛИТЕРАТУРА:

1. Б.Б. Нестеренко, М.А. Новотарский Локально-асинхронный метод решения уравнений математической физики на клеточных нейронных сетях // «Компьютерное моделирование и интеллектуальные системы», сборник научных трудов, 2007, с.296 – 307.
2. А.Е. Алексеенко, А.М. Казённов Реализация клеточных автоматов «игра Жизнь» с применением технологий CUDA и OpenCL // Компьютерные исследования и моделирование, 2010, Т.2, №3, с. 323–326.