Параллельная реализация модифицированного метода траекторий для уравнения неразрывности*

А.В. Вяткин [1], А.А. Ефремов [1], Е.Д. Карепова [1,2], В.В. Шайдуров [1]

[1] Институт вычислительного моделирования СО РАН,
[2] Институт математики и фундаментальной информатики СФУ,
Красноярск

Обсуждается параллельная реализация численного решения двумерного уравнения переноса полулагранжевым методом для высокопроизводительных вычислительных систем гибридной архитектуры. Проведено исследование производительности последовательной и нескольких параллельных реализаций алгоритма, выполненных с помощью технологий OpenMP и CUDA для языка Си.

Введение

В течение двух последних десятилетий активно развивались численные алгоритмы [1–4] решения гиперболических уравнений, основанные на смешанном полулагранжевом (semi-Lagrangian) подходе. Преимуществом подхода является выполнение условия Куранта–Фридрихса–Леви без ограничения на шаг по времени [5,6], что делает его удобным инструментом для решения задач с большими значениями скоростей. Полулагранжевый подход также используется при построении численных методов в случаях, когда на соседних слоях по времени необходимо использовать разные пространственные дискретизации вычислительной области [4]. Еще одним достоинством подхода является выполнение закона сохранения в консервативных версиях метода. Консервативность метода основана на выполнении локального закона сохранения, который справедлив для внутренних узлов сетки [1–4].

В работах [7–9] в рамках полулагранжевого подхода обоснован метод численного решения начально-краевой задачи для гиперболического уравнения, записанного в дивергентной форме. Метод основан на точном тождестве для пространственных интегралов,

*Работа выполнялась в рамках гранта РФФИ № 11-01-00224-а, интеграционного проекта СО РАН № 130, проекта Президиума РАН № 18.2.

области интегрирования которых лежат на соседних слоях по времени. Порядок сходимости метода зависит от асимптотической точности вычисления интегралов. В [7–9] представлены теоремы, позволяющие точно учитывать краевые условия и обосновывающие сходимость метода с первым порядком точности.

Основным недостатком полулагранжевого подхода следует считать большую вычислительную емкость и сложность реализации получаемых на его основе численных методов. В связи с этим актуальным является вопрос параллельной реализации этих методов.

В настоящей работе обсуждаются основные проблемы, связанные с распараллеливанием метода, описанного в [9], для высокопроизводительных систем гибридной архитектуры с помощью технологий OpenMP и CUDA.

Постановка дифференциальной задачи и последовательный алгоритм

Пусть $\overline{D} = [0,1] \times [0,1]$ – единичный квадрат. В области $[0, T] \times \overline{D}$ рассмотрим уравнение переноса:

$$\frac{\partial \rho}{\partial t} + \frac{\partial (u\rho)}{\partial x} + \frac{\partial (v\rho)}{\partial y} = 0, \tag{1}$$

где $u(t,x,y)$ и $v(t,x,y)$ – достаточно гладкие, известные в $[0, T] \times \overline{D}$ функции. Пусть $\forall\ t \in [0,T]$ на верхней и нижней границах $\overline{D}$ выполнены условия прилипания:

$$\begin{aligned} u(t,x,y)\big|_{y=0} &= u(t,x,y)\big|_{y=1} = 0, \\ v(t,x,y)\big|_{y=0} &= v(t,x,y)\big|_{y=0} = 0. \end{aligned} \tag{2}$$

Пусть также при $x = 1$ верно:

$$u(t,x,y)\big|_{x=1} \geq 0 \ \forall\ t \in [0, T]. \tag{3}$$

Зададимся также начальными и граничными условиями:

$$\forall\ (x,y) \in \overline{D} \quad \rho(0,x,y) = \rho_{\text{init}}(x,y), \tag{4}$$

$$\forall\ t \in [0, T] \quad \rho(t,0,y) = \rho_{\text{lb}}(t,y). \tag{5}$$

Полулагранжевый подход к построению численного метода решения задачи (1)–(5) в [9] основан на точном интегральном равенстве

$$\int\limits_{\Omega_{i,j}} \rho\left(t_k, x, y\right) d\Omega = \int\limits_{Q_{i,j}^{k-1}} \rho\left(t_{k-1}, x, y\right) dQ + \int\limits_{I_{i,j}^{k-1}} \left(\rho_{lb} u\right)\left(t, 0, y\right) dI, \tag{6}$$

где слева интеграл берется по ячейке сетки Ω_{ij} на текущем слое по времени, а справа – по соответствующему Ω_{ij} криволинейному многоугольнику Q_{ij} на предыдущем слое с учетом возможного дополнительного слагаемого для левой границы расчетной области (рис. 1, слева). В численном методе [9] криволинейный многоугольник Q_{ij} и интегралы в (6) вычисляются с необходимой для достижения нужной сходимости метода точностью.

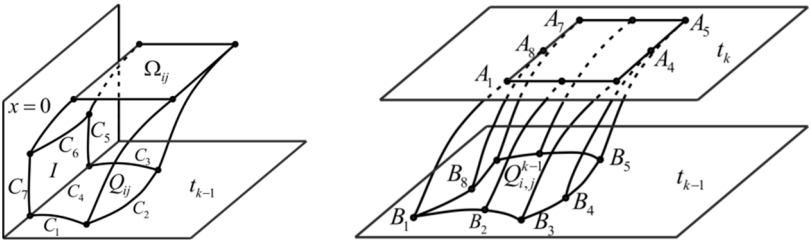


Рис. 1. Основная идея используемого полулагранжевого подхода

Опишем последовательный алгоритм [9].

1. Положим $\rho^h\left(t_0, x_i, y_j\right) = \rho_{\text{init}}\left(x_i, y_j\right)$, $i, j = 0, ..., N$, используя начальные данные (4).
2. *Цикл по времени*: для каждого шага по времени $k = 1, ..., K$ выполнить:
   Цикл по пространству: для каждой ячейки сетки Ω_{ij}^k, $i, j = 0, ..., N$ выполнить:
   2.1. Для каждой точки $A_n\left(x_n, y_n\right)$, $n = 1, .., 8$ (рис. 1, справа) решить систему обыкновенных дифференциальных уравнений для определения соответствующих траекторий $A_n B_n^h$ и координат точек $B_n^h\left(\tilde{x}_n\left(t_{k-1}\right), \tilde{y}_n\left(t_{k-1}\right)\right)$ криволинейного восьмиугольника

P_{ij}^{k-1}.

2.2. Для тех траекторий $A_n B_n^h$, которые пересекают граничную плоскость $P_0 = \{(0, y, t)\}$, определить координаты пересечения траекторий $A_n B_n^h$ с этой плоскостью.

2.3. Вычислить $J_{ij}^{k-1} = \int\limits_{P_{ij}^{k-1}} \rho_h^l \left(t_{k-1}, x, y\right) dP + \int\limits_{L_{i,j}^{k-1}} \left(\rho_{lb} u\right)^l \left(t, 0, y\right) dL$, где в качестве подынтегральных функций используются соответствующие билинейные интерполяционные многочлены, а интегралы берутся по каждому непустому пересечению $P_{i,j}^{k-1} \cap \left\{\left[x_p, x_{p+1}\right) \times \left[y_q, y_{q+1}\right)\right\}$ и $L_{i,j}^{k-1} \cap P_0$ в отдельности.

2.4. Вычислить $\rho^h \left(t_k, x_i, y_j\right) = \dfrac{1}{mes\left(\Omega_{ij}\right)} J_{ij}^{k-1}$, где через $mes\left(\Omega_{ij}\right)$ обозначена площадь ячейки Ω_{ij}.

Конец цикла по пространству.

2.5. При необходимости вычислить нормы решения и ошибки, другую статистическую информацию для текущего шага по времени.

Конец цикла по времени.

Отметим, что наиболее вычислительно трудоемким в алгоритме является анализ расположения криволинейного многоугольника P_{ij}^{k-1} относительно узлов сетки $(k-1)$-го слоя по времени. А именно, для точного, до погрешности компьютерной арифметики, вычисления интегралов на шаге 2.3 применяется процедура разбиения P_{ij}^{k-1} на составные части таким образом, что каждая часть лежит только в одной ячейке Ω_{pq} сетки $(k-1)$-го слоя по времени.

Параллельные реализации алгоритма

Для OpenMP-версии алгоритма, основанной на распараллеливании плотно вложенных циклов, исследовано влияние на производительность технологии HyperThreading (HT), а также проанализирована зависимость ускорения от размерности задачи и количества используемых потоков. Анализ результатов показал следующее.

1. Накладные расходы, связанные с синхронизацией в OpenMP-версии, невелики.

2. Для получения максимального ускорения OpenMP-версии программы выгодно использовать технологию HT с загрузкой максимально возможного количества логических ядер. При имеющихся в нашем распоряжении 12 физических ядрах программе было доступно максимально 24 логических ядра при включенной HT и 12 – при выключенной. Эксперименты показали, что время выполнения программы на 24 нитях при включенной HT в среднем на 14 % меньше времени выполнения этой же программы на 12 нитях без технологии HT (рис. 2).

3. Использование оптимизации компилятора существенно (более чем в 2 раза) уменьшает время работы последовательной программы (табл. 1). Поскольку компилятор никак не оптимизирует код CUDA, то время выполнения CUDA-версий программы не зависело от опций компиляции (табл. 1).

В табл. 1 и на рис. 3 приведены данные по ускорению нескольких параллельных OpenMP- и CUDA-версий программ.

Таблица 1. Время выполнения различных версий программы

Версия	Размерность сетки (количество шагов по времени)				
	80*80 (400)	160*160 (800)	320*320 (1600)	640*640 (3200)	1280*1280 (6400)
Последовательная, без опт.	20,16	159,40	1268,46	10107,80	*
Последовательная, опт. O2	9,99	78,97	626,72	4980,61	39598,90
Последовательная, опт. O3	9,87	78,08	619,80	4936,25	39202,91
OpenMP(12)**, без опт., HT-	1,98	12,52	103,45	819,06	6519,87
OpenMP(12)**, опт. O2, HT-	0,92	7,13	56,10	444,15	3535,53
OpenMP(24)**, опт. O2, HT+	0,91	6,17	49,93	388,20	3080,91
CUDA версия 1	2,55	13,06	74,02	***	***
CUDA версия 2	3,20	8,27	42,60	308,82	**
* – результат за разумное время не получен.					
** – приведены результаты на 12 нитях при отключенной технологии HT и на 24 – при включенной HT.					
** – не удалось запустить ядро из-за нехватки количества регистров.					

Отметим, что имеется ряд существенных ограничений алгоритма для его успешной адаптации для систем гибридной архитектуры с использованием технологии CUDA.

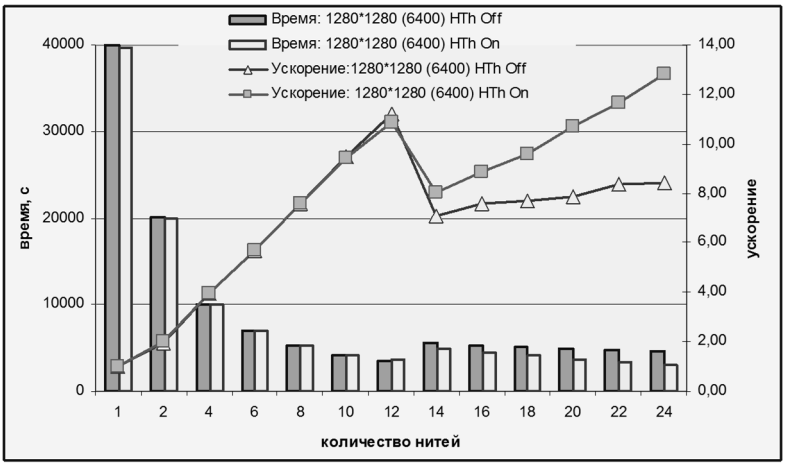


Рис. 2. Время выполнения OpenMP-версии программы (основная ось) и достигнутое ускорение относительно лучшей последовательной версии (дополнительная ось). Сравнение расчетов при включенной и выключенной поддержке технологии HyperThreading

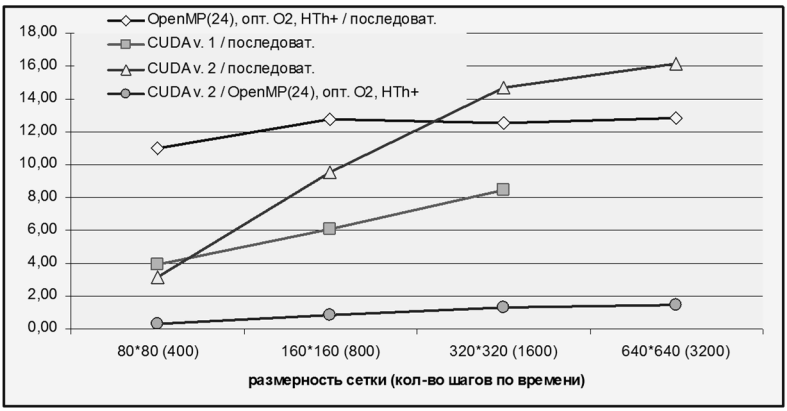


Рис. 3. Ускорение параллельных версий программы

Во-первых, существует традиционное для технологии CUDA ограничение по памяти. В нашем случае оно выражается в ограничении на размерность сетки по пространству, если использовать один поток CUDA для выполнения шагов 2.1–2.4 последовательного алгоритма (ядро CUDA – тело вложенного цикла). Для снятия этого ограничения полезным оказывается подход, основанный на «переиспользовании потоков» для обработки нескольких итераций цикла по пространству.

Другая возможность – использование нескольких графических устройств одновременно. Однако поскольку в полулагранжевом методе разностный шаблон подстраивается под решение и не является равномерным, то реализация обоих подходов имеет ряд дополнительных сложностей.

Во-вторых, анализ расположения ячейки интегрирования относительно разностной сетки на предыдущем шаге по времени содержит много ветвлений и вложенных вызовов функций, что делает актуальным второе традиционно узкое место графических процессоров общего назначения – аппаратное ограничение количества регистров на массиве потоковых мультипроцессоров (Streaming Multiprocessor, SM). В результате для повышения эффективности распараллеливания необходимо существенно менять последовательный алгоритм.

В численных экспериментах на вычислительной системе с видеокартой NVIDIA TESLA C2050 (CC 2.0) достигалось 16-кратное ускорение параллельной версии программы (рис. 3).

Обсуждение результатов

В работе метод, основанный на полулагранжевом подходе, реализован для двумерного гиперболического уравнения в дивергентной форме и распараллелен для вычислительных систем с общей памятью и гибридной архитектурой. Проведены исследования эффективности и производительности параллельных версий.

На первый взгляд сравнение технологий OpenMP и CUDA применительно к нашему алгоритму демонстрирует преимущества OpenMP. Однако хочется заметить, что некорректно сравнивать количество ядер GPU (в нашем случае 448 ядер на видеокарту) и количество ядер CPU (у нас 12 физических или 24 логических). В архитектуре ядра GPU можно найти только один аналог составляющих ядра CPU – арифметико-логическое устройство. Правильнее сравнить ядро CPU с потоковым мультипроцессором (SM), который, кроме ядер GPU, как и ядра CPU, содержит разнообразные устройства управления. Следует также учитывать, что архитектура SM на GPU заточена под работу с графикой, тогда как архитектура CPU оптимизирована для работы с большим числом приложений. Кроме того, потенциал оптимизации CUDA-версий программы пока не исчерпан.

Таким образом, в нашем исследовании проводилось сравнение технологии OpenMP на 12 ядрах CPU и технологии NVIDIA CUDA на 14 потоковых мультипроцессорах. В таком ракурсе наши результаты с

учетом дальнейшего развития CUDA-версий следует интерпретировать как перспективные для графических процессоров.

Литература

1. *Priestley A.* A quasi-conservative version of the semi-Lagrangian advection scheme // Mon. Weather Rev. 1993. Vol. 121. P.621–629.
2. *Scroggs J.S.*, *Semazzi F.H.M.* A conservative semi-Lagrangian method for multidimensional fluid dynamics applications // Numer. Meth. Part. Diff. Eq. 1995. Vol. 11. P.445–452.
3. *Phillips T.N., Williams A. J.* Conservative semi-Lagrangian finite volume schemes // Numer. Meth. Part. Diff. Eq. 2001. Vol. 17. P.403–425.
4. *Iske A.* Conservative semi-Lagrangian advection on adaptive unstructured meshes // Numer. Meth. Part. Diff. Eq. 2004. Vol. 20. P.388–411.
5. *Годунов С.К.* Уравнения математической физики. М.: Наука, 1979. 392 с.
6. *Численное* решение многомерных задач газовой динамики / под ред. С.К. Годунова. М.: Наука, 1976. 400 с.
7. *Xin Wen, Vyatkin A.V., Shaidurov V.V.* Characteristics-like approach for solving hyperbolic equation of first order // Молодой учёный. 2013. № 3(50). С. 5–12.
8. *Xin Wen, Vyatkin A.V., Shaidurov V.V.* Semi-Lagrangian Scheme for soling hyperbolic equation of first order // Молодой учёный. 2013. № 9(56). С. 6–13.
9. *Вяткин А.В., Ефремов А.А., Карепова Е.Д., Шайдуров В.В.* Использование гибридных вычислительных систем для решения уравнения переноса модифицированным методом траекторий //Пятая Международная конференция «Системный анализ и информационные технологии»: Труды конференции. В 2 т. Красноярск: ИВМ СО РАН, 2013. Т. 1. С. 45–55.