УДК 004.42

ИСПОЛЬЗОВАНИЕ ГИБРИДНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ К РЕШЕНИЮ КРАЕВОЙ ЗАДАЧИ ДЛЯ УРАВНЕНИЯ НЕРАЗРЫВНОСТИ МОДИФИЦИРОВАННЫМ МЕТОДОМ ТРАЕКТОРИЙ

Ефремов А.А.
Научный руководитель – к.ф.-м.н., доцент Карепова Е.Д.
Сибирский федеральный университет

Уравнение неразрывности входит в состав многих математических моделей сплошной среды. Несмотря на то, что данная проблема находится под пристальным вниманием мирового математического сообщества уже очень долго, существующие сегодня явные численные методы его решения накладывают сильные ограничения на шаг по времени.

Одним из возможных подходов к численному интегрированию уравнения неразрывности является метод траекторий. Использование этого метода дает явную по времени разностную схему, однако данная модификация метода налагает более мягкие ограничения на шаг по времени по сравнению с явными разностными схемами.

Данный метод позволяет вычислять значение ρ в уравнениях следующего вида:

$$\frac{\partial \mathrm{p}}{\partial \mathrm{t}} + \frac{\partial(\mathrm{up})}{\partial \mathrm{x}} + \frac{\partial(\mathrm{vp})}{\partial \mathrm{y}} = \mathrm{f(t,\ x,\ y)}$$

Следующая разностная схема позволяет вычислить ρ:

$$\frac{1}{\tau}\left(\rho_{ij}^{k} - \frac{1}{h_x h_y}\int\limits_{d\Omega_{ij}} \rho_h^{k-1} d\Omega_{ij}\right) = f_{ij}^{k}$$

Также предполагаем, что известно точное решение, а также функции скорости u и v:

$$\rho(t,x,y) = 1.1 + \sin(t * x * y),\ N = 10 \cdot 2^n,\ n = 0, \dots, 7,\ \tau = \frac{h}{5},\ \mathrm{t} \in [0,1]$$

$$u(t,y,x) = 100 \cdot y(1-y)arctg(x)$$

$$v(t,x,y) = arctg(x(1-x) \cdot y(1-y)\frac{1+t}{10}$$

Для того, чтобы проверить модификацию метода траекторий на практике, была написана последовательная программа-решатель. Также, поскольку в явных разных схемах отсутствует зависимость по данным между временными слоями, данный метод также заранее предполагает эффективную параллельную реализацию. С целью исследовать различные подходы к распараллеливанию с использованием гибридных вычислительных систем, были дополнительно написаны две параллельные реализации данного метода с использованием технологий OpenMP и NVIDIA CUDA. Также целью работы является оценка производительности при использовании различных технология параллельного программирования.

Результаты сравнения производительности представлены на следующих диаграммах:

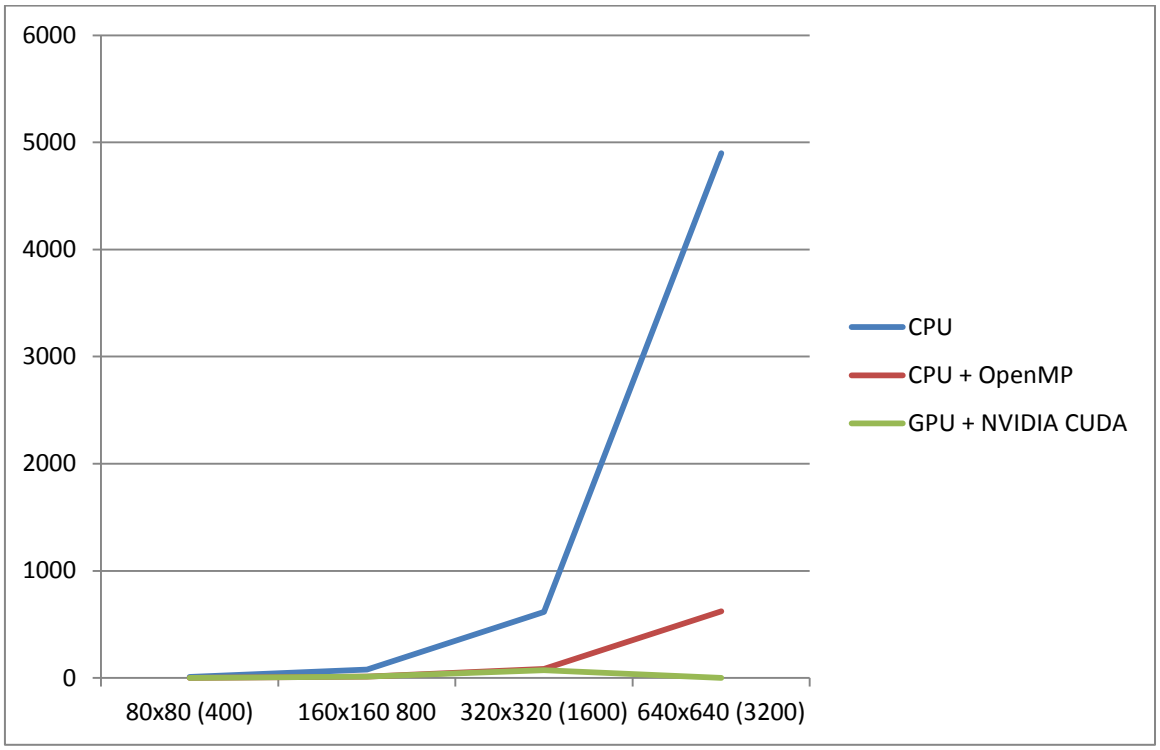

Диаграмма 1. Сравнение времени расчетов CPU, CPU + OpenMP, GPU в секундах. Оптимизированный вариант (-O3)

	CPU	CPU + OpenMP	GPU + NVIDIA CUDA
80x80 (400)	9.8	1,783	2,55
160x160 800	77,6	12,88	13,06
320x320 (1600)	616.9	85,54	74,02
640x640 (3200)	4900.0	622,47	*

Таблица 1. Время расчетов в секундах

	CPU/OpenMP	OpenMP / GPU	CPU / GPU
80x80 (400)	5.5	0,698327031	3,847261
160x160 800	6.03	0,986316344	5,9476751
320x320 (1600)	7.2	1,155657914	8,3344895
640x640 (3200)	7.87	*	*

Таблица 2. Ускорение расчетов с использованием различных техник распараллеливания

Символ ‘*’ в таблицах означает, что расчеты провести не удалось.

При создании параллельной версии программы на GPGPU возникли трудности с масштабируемостью. При запуске расчетных сеток размером более, чем 640*640 элементов возникли проблемы с ограничениями GPGPU Nvidia Telsa C2050. В

дальнейшем планируется доработать подход к распараллеливанию на GPGPGU. В частности изменить функции создания сетки для CUDA ядра, а также вынести часть расчетов в дополнительные ядра.

Численные эксперименты проводились на высокопроизводительном вычислительном сервере Flagman RX240T8.2. Данный сервер расположен в ИВМ СО РАН. Ниже приведены основные характеристики сервера:

- процессоры: 2 шт. 2.93-3.33GHz Intel® Xeon® X5670 Westmere-EP SixCore w/HyperThreading 6.4GT/s QPI, 12MB Smart cache;
- установленная память – 40GB DDR-III PC3-10600 ECC Registered;
- GPGPU: 8 шт. nVidia® Tesla® C2050 PCI-Express ×16 3072 MB GDDR5;

Графические ускорители Tesla C2050 обладают следующими характеристиками:

- Количество CUDA ядер, шт: 448;
- Пропускная способность, Гб/с: 144;
- Объём видеопамяти, Мб: 3072;
- Шина видеопамяти, бит: 512;
- Теоретическая производительность (двойная точность), гигафлопс: 515,2;

Для реализации алгоритма использовалось следующее ПО:

- ОС: Ubuntu Linux 11.04;
- Языки программирования: C/C++, CUDA C/C++;
- Компиляторы: GNU GCC 4.5.2, NVCC release 5.0, ICPC 13.1.0
- Дополнительное ПО: CUDA GDB