УДК 681.3.

© **В.В. Гостюшкин**[1], канд. техн. наук, старш. науч. сотрудник;
С.К. Полумиенко[2], д-р физ.-мат. наук, старш. науч. сотрудник;
С.З. Савин[1], канд. техн. наук, зав. лабораторией

[1]Вычислительный центр ДВО РАН;
[2]Институт телекоммуникаций и глобального информационного пространства, г. Киев

ГРАФИЧЕСКИЕ ПРОЦЕССОРЫ В ЗАДАЧАХ БИОИНФОРМАТИКИ

Разрабатываются принципы использования графических процессоров для решения задач компьютерного автоматизированного анализа совмещенных медицинских изображений по данным однофотонной эмиссионной компьютерной томографии и рентгеновской компьютерной томографии. Приведен пример программного кода для вычисления быстрого преобразования Фурье, значительно ускоряющего расчеты при построении совмещенных изображений трехмерной графики в задачах радионуклидной диагностики.

Ключевые слова: радионуклидная диагностика, компьютерная томография, графические процессоры, программный код.

Исследования в области биоинформатики требуют интенсивных вычислений до сотен петафлоп/сек [10]. Вычисления, необходимые в генной инженерии только для одного гена, требуют приблизительно 800 компьютеров на протяжении года [9]. Распределённые вычисления (distributed computing, grid computing, volunteer computing) – способ решения трудоёмких вычислительных задач геномики и биоинформатики с использованием двух и более компьютеров [8]. Распределённые вычисления являются частным случаем параллельных вычислений, то есть одновременного решения различных частей одной вычислительной задачи несколькими процессорами (или ядрами одного процессора) одного или нескольких компьютеров. Поэтому необходимо, чтобы решаемая задача была сегментирована: разделена на подзадачи, которые могут вычисляться параллельно. При этом для распределенных вычислений приходится учитывать возможное различие в вычислительных ресурсах, которые будут доступны для расчёта различных подзадач. Более того, не всякую задачу можно разделить на подзадачи, которые можно решать параллельно [8, 9].

В конце прошлого века высокая стоимость специализированных суперкомпьютерных решений и нарастающая потребность разных слоёв общества в доступных вычислительных ресурсах привели к широкому распространению компьютерных кластеров. Эти системы характеризует использование отдельных узлов на основе дешёвых и широко доступных компьютерных комплектующих для серверов и персональных компьютеров и объединённых при помощи мощных коммуникационных систем и специализированных программно-

аппаратных решений. Несмотря на кажущуюся простоту, кластеры довольно быстро заняли достаточно большой сегмент суперкомпьютерного рынка, обеспечивая высочайшую производительность при минимальной стоимости решений. Однако на компьютерных кластерах, массивно-параллельных компьютерах процесс удешевления и доступности распределённых числений не остановился, а с проникновением технологий параллелизации и многоядерности в процессорные устройства персональных компьютеров и рабочих станций даже ускорился.

Существенный скачок в этом неожиданно обеспечили современные графические процессоры (GPU – Graphics Processing Unit), первоначально используемые для ускорения трехмерной графики в компьютерных играх и ставшие мощным программируемым устройством параллельных вычислений для решения широкого класса задач, не связанных с графикой [1, 7]. Для вычислений с использованием GPU два основных производителя видеочипов NVIDIA и AMD разработали и анонсировали соответствующие программно-аппаратные платформы под названием CUDA (Compute Unified Device Architecture) и CTM (Close To Metal или AMD Stream Computing), соответственно. Перечисленные модели были выполнены с учётом прямого доступа к аппаратным возможностям видеокарт. Платформы несовместимы между собой, CUDA C – это расширение языка программирования C, а CTM – виртуальная машина, исполняющая ассемблерный код.

В математическом пакете программирования Matlab на основе Parallel Computing Toolbox реализована поддержка CUDA для графических процессоров NVIDIA и в версии 2013b содержит порядка 200 встроенных функций, использующих возможности GPU [11].

Приведем пример программного кода для вычисления быстрого преобразования Фурье без поддержки GPU:

```
a = rand(1000,'single');
fft = fft(a);
b = (real(Gfft) + Ga) * 6;
```

и с поддержкой GPU:

```
Ga = gpu Array.rand (1000,'single');
Gfft = fft (Ga);
Gb = (real(Gfft) + Ga) * 6;
G = gather (Gb);
```

Отличия кода минимальны и прозрачны.

Во втором случае наличие префикса *gpuArray* в первой строке кода означает передачу массива чисел в память GPU, а в последней строке посредством функции *gather* – возврат из памяти GPU в память среды Matlab [2].

Для сравнения ускорения вычислений приведем пример вычисления фрактала по алгоритму Мандельброта [5].

Программный код без поддержки GPU (*рис. 1*) и с поддержкой GPU (*рис. 2*):

```
maxIterations = 1000;
gridSize  = 1000;
xlim  = [-0.748766713922161,  -0.748766707771757];
ylim  = [ 0.123640844894862,  0.123640851045266];
% Setup
t = tic();
```

```
x = linspace(xlim(1), xlim(2), gridSize );
y = linspace(ylim(1), ylim(2), gridSize );
[xGrid,yGrid] = meshgrid( x, y );
z0 = xGrid + 1i*yGrid;
count = maxIterations*ones( size(z0) );
% Calculate
z = z0;
for n = 0:maxIterations
 z = z.*z + z0;
hi = abs( z )>2;
z(hi) = NaN;
count(hi) = n;
end
count = log( count+1);
% Show
cpuTime = toc( t );
figure
set(gcf, 'Position', [200 200 600 600] );
imagesc( x, y, count );
axisimage
colormap( [jet();flipud( jet() );0 0 0] );
title(sprintf( '%1.2fsecs (without GPU)', cpuTime ) );
%------------------------------------------------
% Working in CUDA
% Load the kernel
cudaFilename = 'pctdemo_processMandelbrotElement.cu';
ptxFilename = ['pctdemo_processMandelbrotElement.',parallel.gpu.ptxext];
kernel = parallel.gpu.CUDAKernel( ptxFilename, cudaFilename );
% Setup
t = tic();
x = parallel.gpu.GPUArray.linspace(xlim(1), xlim(2), gridSize );
y = parallel.gpu.GPUArray.linspace(ylim(1), ylim(2), gridSize );
[xGrid,yGrid] = meshgrid( x, y );
% Make sure we have sufficient blocks to cover the whole array
numElements = numel( xGrid );
kernel.ThreadBlockSize = [kernel.MaxThreadsPerBlock,1,1];
kernel.GridSize = [ceil(numElements/kernel.MaxThreadsPerBlock),1];
% Call the kernel
count = parallel.gpu.GPUArray.zeros( size(xGrid) );
count = feval( kernel, count, xGrid, yGrid, maxIterations, numElements );
% Show
gpuCUDAKernelTime = toc( t );
figure
set(gcf, 'Position', [200 200 600 600] );
imagesc( x, y, count )
axisimage
colormap( [jet();flipud( jet() );0 0 0] );
title(sprintf( '%1.2fsecs (GPU CUDAKernel) = %1.1fx faster', ...
gpuCUDAKernelTime, cpuTime/gpuCUDAKernelTime;
```

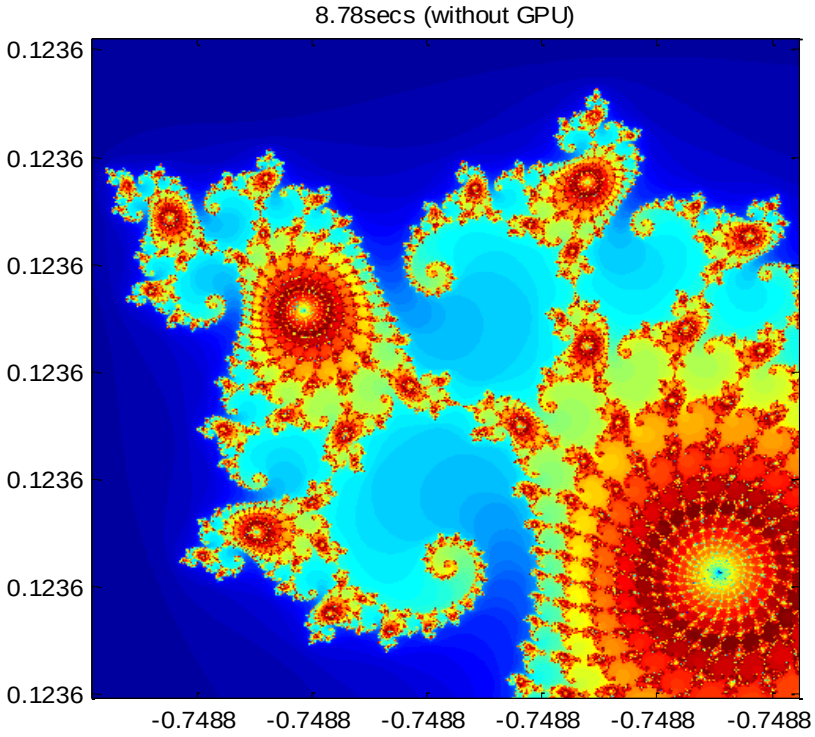

Рис. 1 – Расчет без GPU

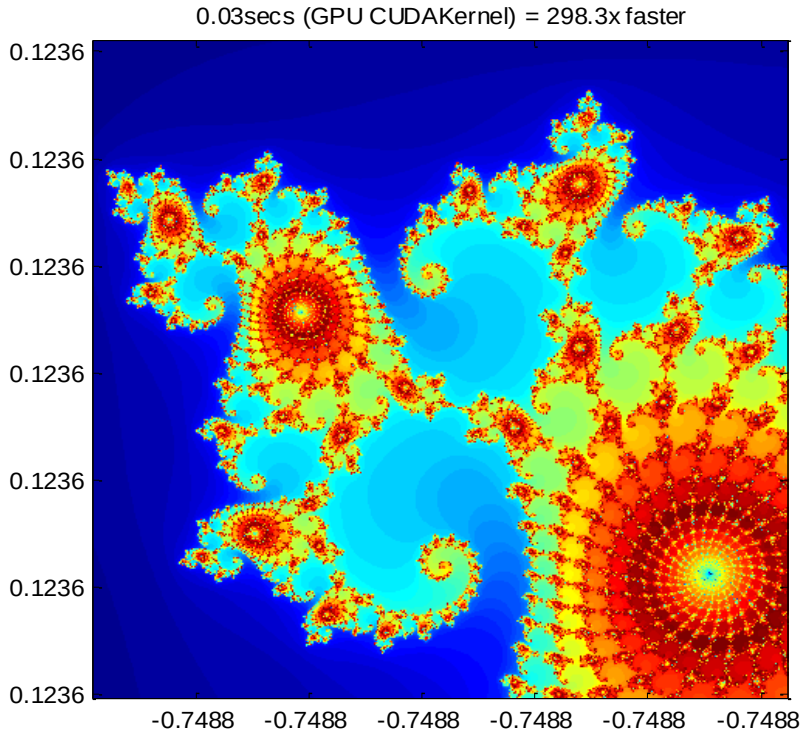

Рис. 2 – Расчет с GPU

Коэффициент ускорения составил 298.3.

Вычисления последнего примера были проведены на ПК с процессором IntelCorei7-3770KCPU@3.50 GHz и видеокартой NVIDIAGeForceGTX 670, объемом видеопамяти – 2 Gb.

Имеются и другие программные расширения сторонних производителей для поддержки CUDA в Matlab, такие как Jacket, GPUmat, ViennaCL и др. [11], которые также используются нами при выполнении проекта РФФИ (грант №13-07-00667) по созданию принципов компьютерного автоматизированного анализа совмещенных медицинских изображений на примере однофотонной эмиссионной компьютерной томографии и рентгеновской компьютерной томографии [6]. Были разработаны теоретические принципы создания и клинического применения телемедицинских компьютерных программ для анализа медицинских изображений. В качестве примера расширения Matlab [2] создано программное обеспечение задач анализа планарных сцинтиграмм скелета [10]. Главной целью комплекса разработанных программ является распознавание среди различных очагов гиперфиксации радиофармпрепарата (ОГРФП) очагов, обусловленных опухолевым (метастатическим) процессом. При этом развиваются на оригинальной основе методы автоматизированного компьютерного анализа (CAD – англ. [12]) и виртуального информационного моделирования [3, 10].

В качестве алгоритмов распознавания ОГРФП предполагается использовать, помимо фрактального анализа, также кластерный и дискриминантный анализ, метод опорных векторов и нейронных сетей. Особенностью комплекса программ САПР (КАД-анализа) [4] является создание базы данных обработанных изображений (БДОИ), являющейся основой для «обучения» программ распознавания изображений. Предполагается использование концепций облачных вычислений со свободным доступом к программам зарегистрированных медицинских пользователей. Принцип облачных вычислений позволит не только удовлетворить интересы специализированных пользователей, но и за счет расширения обучающей выборки БДОИ обеспечит возможность постоянно увеличивать дифференциально-диагностические характеристики программы. Возможно развитие идеологии фрактального анализа на облачные вычисления в иных задачах диагностической медицины.

Список использованной литературы

1. Боресков А.В., Харламов А.В. Основы работы с технологией CUDA. М.: Изд-во ДМК Пресс, 2010. 232 с.

2. Гонсалес Р., Вудс Р., Эддинс С. Цифровая обработка изображений в среде MATLAB. Пер. с англ. Москва: Изд-во Техносфера. 2006. 615 с.

3. Косых Н.Э., Савин С.З., Смагин С.И. Виртуальные информационные модели опухолевого роста // Информационные технологии и вычислительные системы. № 1, 2005. С. 117–129.

4. Малюх В.Н. Введение в современные САПР. М.: ДМК Пресс, 2010. 192 с.

5. Мандельброт Б. Самоаффинные фрактальные множества // Фракталы в физике. М.: Мир, 1988. С. 9–47.

6. Наттерер Ф. Математические аспекты компьютерной томографии: Пер. с англ.. – М: Мир, 1990. – 288 с.

7. Сандерс Д., Кэндрот Э. Технология CUDA в примерах. Введение в программирование графических процессоров. М.: Изд. ДМК Пресс, 2011. 232 с.

8. Суперкомпьютерные технологии в науке, образовании и промышленности / Под ред. Садовничего В.А., Савина Г.И. , Воеводина Вл.В. М.: Изд-во Московского университета, 2009. 232 с.

9. Суперкомпьютеры. [Электронный ресурс]. www.supercomputers.ru

10. Kosykh N.E., Gostuyshkin V.V., Savin S.Z., Vorojztov I.V. Designing the systems of computer diagnostics of medical images // Proc. of The First Russia and Pacific Conference on Computer Technology and Applications (RPC 2010). Vladivostok, Russia. 6–9 September, 2010. 4 p.

11. Mathlab. Url: http://www.mathworks.com/products/parallel-computing

12. Medical software and (CAD/CAE/CAM/EDA/PCB/GIS/FEA). Url: http://www.cd–soft.net

Стаття надійшла до редакції 28.02.14 російською мовою

© В.В. Гостюшкін, С.К. Полумієнко, С.З. Савін

ГРАФІЧНІ ПРОЦЕСОРИ В ЗАДАЧАХ БІОІНФОРМАТИКИ

Розробляються принципи використання графічних процесорів для вирішення задач комп'ютерного автоматизованого аналізу сполучених медичних зображень по даних однофотонної емісійної комп'ютерної томографії та рентгенівської комп'ютерної томографії. Наведено приклад програмного коду для обчислення швидкого перетворення Фур'є, що значно прискорює розрахунки при побудові сполучених зображень тривимірної графіки в задачах радіонуклідної діагностики.

© V.V. Gostuyshkin, S.K. Polumienko, S.Z. Savin

GRAPHICS PROCESSORS IN BIOINFORMATICS TASKS

Principles of Graphics Processing Unit using for computer-aided analysis of combined data on medical imaging of single photon emission computer tomography and x-ray computed tomography are considered. There is described an example of software code for computing the fast Fourier transform, significantly accelerating calculation when building composite 3D image in the tasks of radionuclide diagnostics.