УДК 519.688

М. С. Косяков, Д. Н. Шинкарук, А. В. Торопов, Ю. А. Шполянский

ПРИМЕНЕНИЕ ТЕХНОЛОГИИ CUDA ДЛЯ УСКОРЕНИЯ РАСЧЕТА ЦЕН ОПЦИОНОВ ЕВРОПЕЙСКОГО ТИПА СЕТОЧНЫМ МЕТОДОМ

Дифференциальное уравнение Блэка—Шоулза решено численно по схеме Кранка—Николсона на графическом процессоре с применением технологии CUDA. Использование видеокарты NVIDIA GTX 580 позволило достичь ускорения более чем в 20 раз по сравнению с однопоточным расчетом на процессоре Intel Core i7 3,4 ГГц и в 2—3 раза по сравнению с наилучшими показателями многопоточной версии, основанной на технологии GCD на процессорах 2x Intel Xeon 3,06 ГГц с 24 ядрами. Результаты получены для нагрузочных параметров, представляющих практический интерес в системах алгоритмической торговли.

Ключевые слова: *CUDA, GPU общего назначения, параллельная циклическая редукция, алгоритмическая торговля, опцион, схема Кранка—Николсона*.

Введение. Одной из важнейших задач систем алгоритмической торговли, позволяющих совершать торговые операции на электронных финансовых рынках с помощью специализированных компьютерных систем, является расчет цен опционов в режиме реального времени с учетом постоянного изменения их параметров. Точные аналитические решения доступны не для всех типов опционов и вариантов постановки задач. Поэтому для теоретического ценообразования широко применяются численные методы [1]. В последние годы активно исследуются возможности применения современных видеокарт с мощными графическими процессорами (ГП) для ускорения расчетов [2—5]. В большинстве работ рассматриваются численные методы, обладающие естественным параллелизмом — метод Монте-Карло, биномиальная модель или явные разностные схемы [2, 3].

Для ускорения расчета цен опционов с использованием неявной схемы Кранка—Николсона второго порядка точности в настоящей работе применена технология CUDA-вычислений (*Compute Unified Device Architecture*) на ГП, разработанном компанией NVIDIA. Алгоритмы реализованы в виде CUDA-программы, исполняемой ГП видеокарты, а также в виде одно- и многопоточного приложения для выполнения центральным процессором (ЦП). Проведено сравнение времени вычислений и погрешности получаемых на современном оборудовании результатов в условиях, характерных для систем алгоритмической торговли. Выявлены условия и способы оптимальной организации вычислительного процесса.

Математическая постановка задачи. Модель Блэка—Шоулза — классическая для расчета цен опционов. Зависимость цены опциона $V(S,t)$ от цены базового актива S и времени t в этой модели описывается дифференциальным уравнением [1]:

$$\frac{\partial V}{\partial t}+\frac{1}{2}\sigma^2 S^2\frac{\partial^2 V}{\partial S^2}+rS\frac{\partial V}{\partial S}-rV=0, \qquad (1)$$

где r — безрисковая процентная ставка и σ — волатильность.

В настоящей работе уравнение (1) решается численно методом конечных разностей по схеме Кранка—Николсона. Рассматривается диапазон цен базового актива от нижней $S_{\min}$ до верхней $S_{\max}$ границы, для интервала времени от даты исполнения $t=T$ до текущего момента $t=0$. Введем равномерную сетку значений цены базового актива $S_i=S_{\min}+i\Delta S$, $i=0,\ldots,I$, с шагом $\Delta S=(S_{\max}-S_{\min})/I$, а также времени $t^k=T-k\Delta t$, $k=0,\ldots,K$, с шагом $\Delta t=T/K$, где I и K — количество шагов по S и t соответственно. В неявной по времени схеме Кранка—Николсона соотношение (1) аппроксимируется по S и t со вторым порядком точности [1]. На временных шагах решаются системы линейных алгебраических уравнений (СЛАУ) с трехдиагональной матрицей. Коэффициенты матрицы рассчитываются один раз, поскольку r и σ — константы.

Для полноты математической постановки задачи необходимо установить начальное распределение $V(S,T)=P(S)$, где $P(S)$ — функция выплаты опциона, а также граничные условия. В работе рассмотрение ведется на примере распространенных европейских put-опционов с ценой исполнения $X>0$:

$$P(S)=\max\{[X-S(T)],\,0\}.$$

Для put-опциона $S_{\min}=0$, $S_{\max}=\infty$. Граничные условия имеют вид:

$$V(S_{\min},t)=X\exp[-r(T-t)],\quad V(S_{\max},t)=0.$$

В разностной схеме используется значение $S_{\max}=3X$, так как $V(3X,t)\approx 0$ при $\sigma<0{,}8$.

СЛАУ с трехдиагональной матрицей на ЦП решалась традиционным методом прогонки [6]. Алгоритм этого метода не может быть эффективно распараллелен для архитектуры CUDA. Поэтому для ГП применен метод параллельной циклической редукции (ПЦР), также использован ряд идей по ускорению вычислений, предложенных нами ранее в работе [7].

В алгоритмической торговле опционы естественным образом объединяются в группы по общему базовому активу. Опционы внутри группы, как правило, имеют различные значения параметров X, T и σ. В результате решения задачи получают набор значений $V(S_0,0)$ для всей группы опционов при текущей цене актива $S=S_0$.

Архитектурные особенности реализации алгоритмов. Алгоритмы реализованы в виде CUDA-программы для выполнения на ГП видеокарты, а также в виде одно- и многопоточного приложения для выполнения на ЦП. Многопоточная версия построена с использованием технологии GCD (*Grand Central Dispatch*) [8]. Она позволяет назначать задачи в приложении, которые в зависимости от загруженности системы автоматически распределяются по потокам для параллельного выполнения. В настоящей работе под одной задачей GCD понимается расчет группы опционов с общим базовым активом.

Технология CUDA использует ГП в роли массово-параллельного сопроцессора к ЦП [9]. В задачи ЦП входит подготовка данных в оперативной памяти ЦП, их передача в память видеокарты для последующей обработки ГП и возвращение результатов. Код массово-параллельных вычислений — ядро CUDA-программы (*kernel*), выполняется ГП как набор многих одновременно работающих потоков (*threads* в терминах CUDA) [9].

В нашей реализации алгоритма ПЦР каждый поток занимается расчетом значения цены опциона в одном узле сетки S_i, взаимодействуя с другими потоками через разделяемую память [5]. Потоки, выполняющие расчет одного опциона, объединены нами в блок, размер которого совпадает с количеством узлов сетки, а количество блоков определяется числом опционов в одной группе. Расчет каждой группы опционов организован в специальные потоки команд (*streams*), которые выполняются одним ГП последовательно с использованием асинхронной передачи данных из оперативной памяти ЦП в память видеокарты и обратно. Доступ к значениям параметров опционов в оперативной памяти ЦП из ядра CUDA-программы осуществляется через механизм отображаемой памяти [10].

Операции с вещественными числами двойной точности (тип *double*) ГП исполняет заметно медленнее, чем операции с числами одинарной точности (тип *float*). Код ядра CUDA-программы разрабатывался с использованием типа *float* в отличие от реализованных нами для ЦП алгоритмов (тип *double*). Поэтому в работе изучено влияние использования типа *float* на погрешность итогового результата вычислений.

Для анализа реализованных в работе алгоритмов расчета использованы экспериментальные стенды (табл. 1).

Таблица 1

Характеристики экспериментальных стендов

Модель	SunFire x4170 M2	ПК Core i7
ЦП	2 x Xeon X5675, 3,06 ГГц, 24 ядра	Core i7-2600, 3,4 ГГц, 8 ядер
Объем ОЗУ	49 ГБ	4 ГБ
ОС	Solaris 10 x86_64 U10	Ubuntu 11.10 x86_64

Многопоточная версия приложения, использующая технологии GCD, запускалась на ЦП сервера SunFire x4170 M2. Персональный компьютер (ПК) Core i7 использовался для запуска однопоточного приложения на ЦП и программы с применением CUDA. Параметры видеокарты NVIDIA GTX 580, используемой в экспериментах: 16 мультипроцессоров, 32 ядра в мультипроцессоре, частота процессора — 1,564 ГГц, частота памяти — 2 ГГц, объем памяти — 1536 МБ, разрядность шины видеопамяти — 384 бит.

В работе измерялось общее время τ выполнения расчетов для всех M групп опционов. Для CUDA-программы значение τ включало также время передачи данных. Для многопоточного приложения измерялось время расчета в режиме холодного старта τ_c (*cold*) с учетом всех накладных расходов на запуск потоков и время расчета в режиме под нагрузкой τ_h (*hot*), определяемое как минимальное время расчета в серии последовательных экспериментов в работающем приложении.

Результаты были получены для европейских put-опционов с параметрами: $X = 100$ у.е., $\sigma = 25$ % в год, $r = 5$ % в год, $T = 0{,}3$ года. Если не оговорено отдельно, применялись следующие настройки сетки: шаг по времени $\Delta t = 0{,}02$ года, количество шагов по цене $I = 150$.

Оптимизация CUDA-программы. Объем регистровой и разделяемой памяти видеокарты ограничен, это влияет на число одновременно запущенных на ГП потоков выполнения, что определяет общую производительность системы.

Предложенный алгоритм требует относительно небольшого объема разделяемой памяти, поэтому основное внимание необходимо уделять работе с регистровой памятью. Был предпринят ряд мер для уменьшения количества регистров, используемых каждым потоком выполнения:

— объединение функций программы для уменьшения числа локальных переменных;

— применение спецификатора volatile, что в некоторых случаях позволяет уменьшить количество используемых регистров;

— использование числовых констант одинарной точности;

— использование библиотеки быстрой математики (use_fast_math).

Перечисленные меры по реорганизации ядра CUDA-программы, используемые далее, позволили сократить общее время выполнения расчетов τ более чем на 20 %.

Анализ времени вычислений. В табл. 2 приведены значения времени расчета одной группы опционов в разных реализациях. Для многопоточного GCD-приложения в режиме холодного старта $\tau_c \approx \tau_h$, поскольку параллелизм в GCD реализован на уровне групп опционов, т.е. расчет одной группы рассматривается как одна задача. Преимущество однопоточного приложения (τ_1), запущенного на ПК Core i7, по сравнению с GCD здесь обусловлено разницей тактовых частот процессоров экспериментальных стендов (3,4 ГГц у Core i7 против 3,06 ГГц у Xeon X5675).

Таблица 2

Сравнение времени расчета одной группы опционов

N	τ, мс	τ_1, мс	τ_c, мс	τ_h, мс	τ / τ_1	τ / τ_c	τ / τ_h
1	0,143	0,071	0,104	0,062	0,497	0,729	0,434
5	0,132	0,265	0,353	0,309	2,007	2,673	2,34
10	0,137	0,575	0,663	0,619	4,193	4,835	4,514
20	0,143	1,132	1,277	1,238	7,939	8,956	8,682
60	0,179	3,475	3,826	3,713	19,413	21,374	20,743
200	0,474	10,602	12,419	12,378	22,381	26,217	26,131
500	0,979	22,746	30,987	30,940	25,731	31,663	31,615
800	1,501	41,021	49,568	49,519	27,323	33,016	32,984
1000	1,871	51,822	62,053	61,88	27,686	33,151	33,059

Из табл. 2 видно, что видеокарта NVIDIA GTX 580 позволяет достичь более чем 20-кратного преимущества при одновременной обработке нескольких опционов в группе ($N \geq 60$ для данного случая).

Рассмотрим результаты исследований, полученных для нескольких групп опционов. На рис. 1 представлен график зависимости времени τ (*1*), τ_c (*2*), τ_h (*3*), τ_1 (*4*) обработки $M = 10$ групп опционов от числа N опционов в группе.

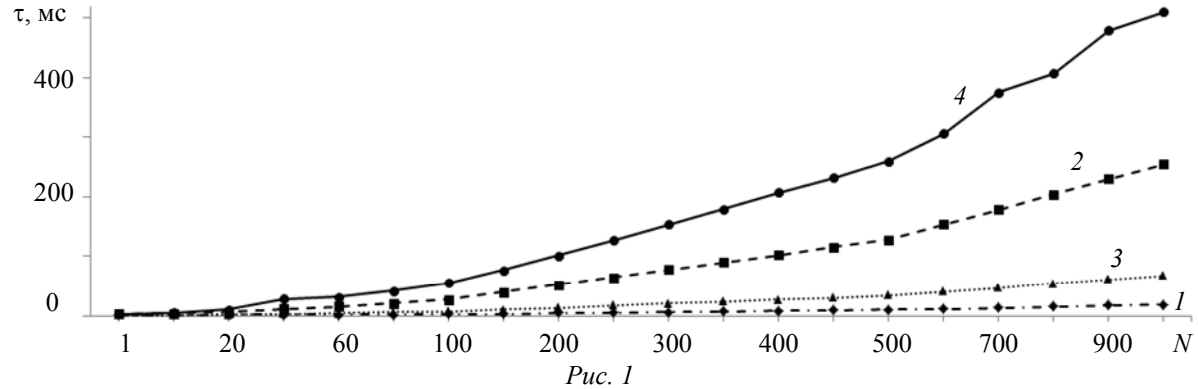

Рис. 1

Как видно из рис. 1, производительность многопоточного приложения GCD под нагрузкой значительно выше производительности однопоточного. Несмотря на это быстродействие CUDA-программы в 2,5—3 раза выше по сравнению с GCD в нагруженном состоянии и более чем в 9 раз по сравнению с GCD в режиме холодного старта (рассчитан важный для задач алгоритмической торговли случай $N \geq 100$). При увеличении N выигрыш в производительности CUDA-программы возрастает.

При увеличении количества групп M возрастает число операций передачи данных. Отсюда следует, что использование технологии CUDA должно давать большее преимущество при расчете малого числа групп большего размера. Для GCD, наоборот, с ростом M увеличивается количество используемых потоков, что повышает производительность.

На рис. 2 продемонстрировано преимущество CUDA-программы по сравнению с одно- и многопоточным приложением на ЦП. Точки кривой обозначают границы, ниже которых

использование соответствующей реализации для ЦП предпочтительней применения CUDA (GCD Hot (*1*), GCD Gold (*2*), Core i7 (*3*)).

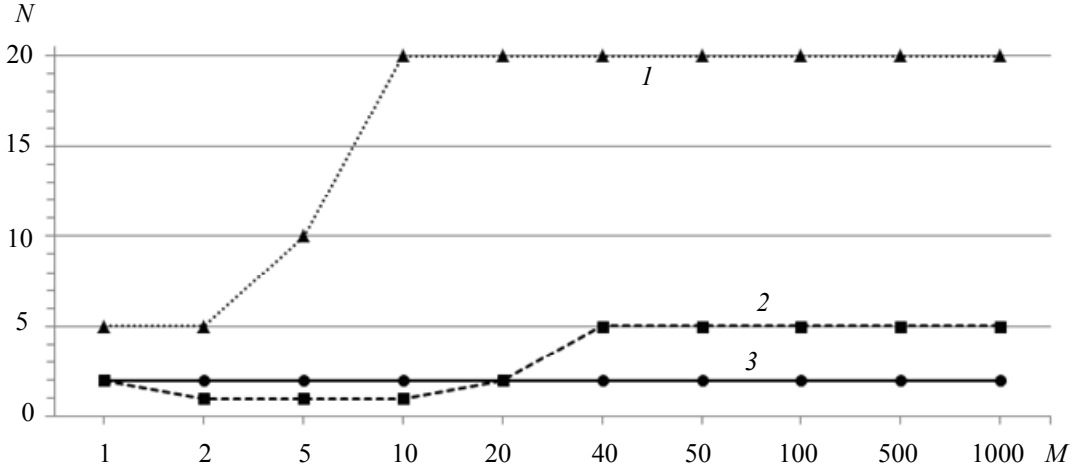

Рис. 2

Из рис. 2 видно, что при $N \geq 20$ CUDA-приложение выигрывает у GCD под нагрузкой для любого количества групп M.

Влияние особенностей работы графического процессора на время вычислений. В ходе экспериментов были отмечены некоторые особенности ГП и их влияние на время вычислений. Из-за архитектурно-функциональных особенностей CUDA потоки в блоке исполняются не отдельно, а группами по 32 потока (*warp* в терминах CUDA). Количество одновременно исполняемых блоков не может превышать числа мультипроцессоров видеокарты (16 для NVIDIA GTX 580), поэтому наблюдается скачкообразный рост времени вычислений при увеличении M и N. На рис. 3 представлено время расчета 10 (*1*), 50 (*2*) и 100 (*3*) групп опционов при различном размере групп. На рис. 4 приведена зависимости времени расчета одного опциона в от числа узлов сетки $I'=I+1$.

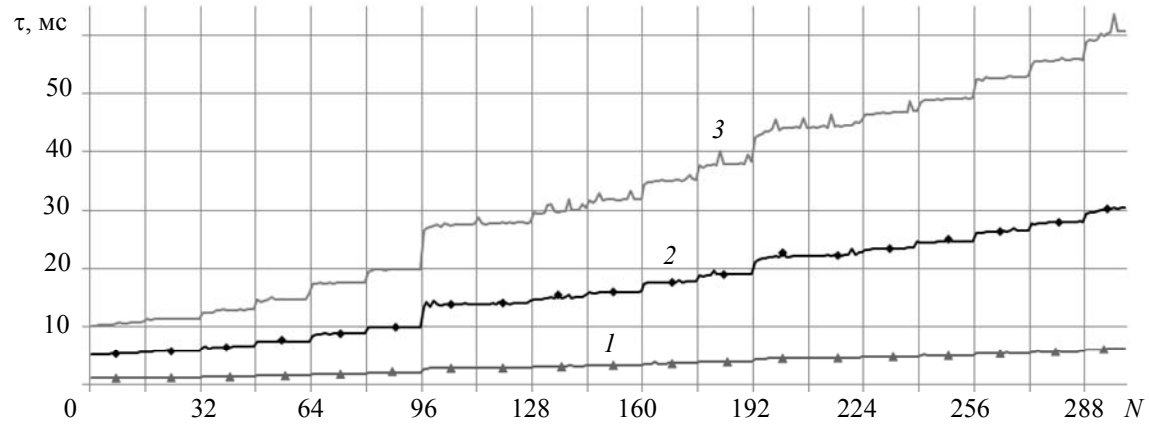

Рис. 3

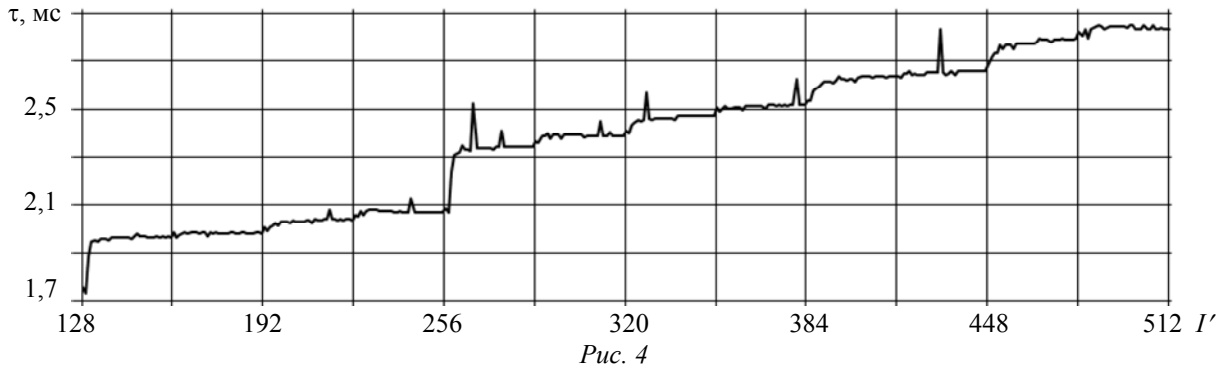

Рис. 4

Анализ погрешностей вычислений. Нами проанализированы погрешности численных алгоритмов, реализованных для разных архитектур. Анализ возможен благодаря наличию аналитического решения уравнения (1) для опционов европейского типа в отсутствие дискретных дивидендных выплат по базовому активу за время „жизни“ опциона. На рис. 5 приведены графики абсолютной погрешности (ЦП тип *double* (*1*), ЦП тип *float* (*2*), ГП тип *float*, ГП тип *use_fast_math* (*3*)), складывающейся из ошибки аппроксимации уравнения (1) схемой Кранка—Николсона и погрешности математических вычислений вещественных чисел с плавающей точкой. Отдельно рассматривался случай использования библиотеки быстрой математики, обладающей меньшей по сравнению со стандартной библиотекой точностью вычисления сложных функций для чисел одинарной точности (тип *float*) [10].

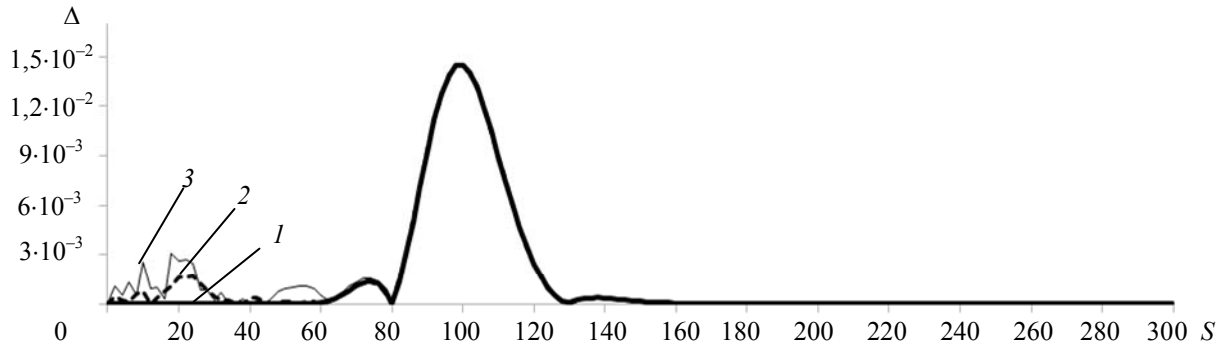

Рис. 5

Рис. 5 составлен для наиболее малого шага по времени (Δt = 0,001) и соответственно большого числа шагов (K = 300), в этом случае наблюдается максимальная погрешность вычислений. Значение погрешности, вносимой использованием *float* (область малых значений S) в 5 раз меньше максимальной погрешности схемы Кранка—Николсона в области $S \sim X = 100$ (относительная ошибка при этом порядка 0,3 %). Библиотека быстрой математики не снижает точность расчетов.

Выводы. В работе технология CUDA применена для ускорения расчета цен опционов европейского типа методом Кранка—Николсона. Соответствующие алгоритмы реализованы в виде CUDA-программы, исполняемой графическим процессором видеокарты, а также в виде одно- и многопоточного приложения, исполняемого ЦП. Многопоточная версия построена с использованием технологии GCD. Выявлены и сформулированы основные направления оптимизации ядра CUDA-программы, позволившие сократить время расчетов более чем на 20 %.

Отмечено влияние архитектурно-функциональных особенностей ГП на время вычислений. Сформулированы рекомендации по выбору параметров расчета цен опционов. Сделан вывод о допустимости использования вещественных чисел с одинарной точностью. Использование библиотеки быстрой математики ускоряет вычисления, но не приводит к увеличению погрешности.

Разработанная CUDA-программа, исполняемая на видеокарте NVIDIA GTX 580, позволяет достичь более чем 20-кратного преимущества перед однопоточной реализацией соответствующего алгоритма на современном ЦП Intel Core i7 3,4 ГГц. Быстродействие CUDA-программы по сравнению многопоточной реализацией выше в 2,5—3 раза с GCD в нагруженном состоянии и более чем 9 раз по сравнению с GCD в режиме холодного старта для практически важного для алгоритмической торговли количества опционов в группе. При увеличении числа опционов в группе выигрыш в производительности CUDA-программы возрастает.

СПИСОК ЛИТЕРАТУРЫ

1. *Wilmott P.* Paul Wilmott on Quantitative Finance. Wiley, 2006. 1380 p.
2. Monte Carlo Option Pricing [Электронный ресурс]: <http://developer.nvidia.com/cuda-cc-sdk-code-samples>.
3. Binomial Option Pricing [Электронный ресурс]: <http://developer.nvidia.com/cuda-cc-sdk-code-samples>.
4. *Lyuu Y. D., Wen K. W., Wu Y.* C. Performance of GPU for Pricing Financial Derivatives Convertible Bonds // To appear in J. of Information Science and Engineering [Электронный ресурс]: <http://www.iis.sinica.edu.tw/page/jise/FILE/AcceptedList/110/110444-POG.pdf>.
5. *Egloff D.* High Performance Finite Difference PDE Solvers on GPUs. Technical report. QuantAlea GmbH. February. 2010.
6. *Бахвалов Н. С., Жидков Н. П., Кобельков Г. М.* Численные методы. М.: Бином. Лаборатория знаний, 2003. 632 с.
7. *Шинкарук Д. Н., Шполянский Ю. А., Косяков М. С.* Анализ эффективности применения технологии CUDA для решения систем линейных уравнений с трехдиагональными матрицами в задачах расчета цен опционов // Наст. выпуск. С. 20—25.
8. Grand Central Dispatch (GCD) Reference. Apple Inc. 2011 [Электронный ресурс]: <http://developer.apple.com/library/ios/#documentation/Performance/Reference/GCD_libdispatch_Ref/Reference/reference.html>.
9. *Боресков А. В., Харламов А. А.* Основы работы с технологией CUDA. М.: ДМК Пресс, 2010. 232 с.
10. NVIDIA CUDA C Programming Guide. Version 4.2. 2011 [Электронный ресурс]: <http://developer.nvidia.com/nvidia-gpu-computing-documentation>.

Сведения об авторах

Михаил Сергеевич Косяков — канд. техн. наук; Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики, кафедра вычислительной техники; E-mail: mkosyakov@gmail.com

Дмитрий Николаевич Шинкарук — аспирант; Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики, кафедра вычислительной техники; E-mail: dimashink@gmail.com

Александр Владимирович Торопов — аспирант; Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики, кафедра компьютерных технологий; Тбрикс АБ; инженер-программист; E-mail: toropov@rain.ifmo.ru

Юрий Александрович Шполянский — д-р физ.-мат. наук, доцент; Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики, кафедра фотоники и оптоинформатики; Тбрикс АБ; ведущий математик; E-mail: shpolyan@mail.ru

Рекомендована кафедрой вычислительной техники

Поступила в редакцию 08.02.12 г.