Оптимизация алгоритмов вычисления опционов на гибридной системе

Д. С. Хмель[a], Э. А. Степанов[b], А. В. Богданов[c]

Санкт-Петербургский государственный университет,
Россия, 199034, Санкт-Петербург, Университетская наб. д.7-9.

E-mail: [a] dskhmel@cc.spbu.ru, [b] e.an.stepanov@gmail.com, [c] bogdanov@csa.ru

В статье рассмотрена проблема оптимизации алгоритмов ценообразования опционов на GPGPU. Основная цель добиться максимальной эффективности от использования гибридной системы. Авторами были предложены некоторые преобразования алгоритма, полученного из модели Блека-Шоулза, для ценообразования европейского и азиатского опциона по методу Монте-Карло с учетом особенностей архитектуры GPGPU. Основной идеей оптимизации является постоянная работа с одним большим массивом данных.

Ключевые слова: европейский опцион, азиатский опцион, гибридная система, GPGPU, CUDA, метод Монте-Карло

Работа выполнена при поддержке грантов РФФИ:
17-57-80111 FTRC-HPC: Высокопроизводительные вычисления с сетевыми реконфигурируемыми отказоустойчивыми вычислительными устройствами,
16-57-48005 Мульти-FPU на кристалле (MFoC) для приложений высокопроизводительных вычислений,
16-07-01111 Построение Unix-подобного окружения для управления виртуальным суперкомпьютером.

Введение

За последнее время происходит значительное увеличение объема торгов на рынке ценных бумаг. Задача поиска цены той или иной ценной бумаги с математической точки зрения имеет довольно специфический характер. Основным требованием является скорость вычислений, где важны даже миллисекунды, так как на практике они могут стоить больших денег участнику торгов. С другой стороны, из принципов функционирования рынка следует, что в данной задаче не требуется высокой точности вычислений.

При решении рассматриваемой задачи большую популярность сегодня имеет метод Монте-Карло. Он позволяет достичь необходимой точности вычислений, но при этом не обладает достаточной эффективностью. Эксперименты показывают, что в зависимости от точности, поиск цены опциона может занимать до нескольких десятков секунд или даже доходить до нескольких минут. Помимо увеличения вычислительной мощности используемых ЭВМ, в данной задаче присутствуют значительные возможности оптимизации на алгоритмическом и программном уровне, которые и будут рассмотрены ниже.

В представленной статье рассматривается возможность расчета цены опциона на гибридных системах с использованием архитектуры CUDA. В качестве основы для численных экспериментов были выбраны европейский и азиатский опционы. Для поиска их цены рассматривается модель, которая описывается континуальным интегралом.

Постановка задачи

В рамках данной статьи будут использоваться следующие обозначения: $S(t)$ — функция времени, описывающая динамику изменения цены базового актива. Будем рассматривать опционный контракт с периодом действия $[0, T]$ и ценой исполнения K. Предположим, что $S_0 = S(0)$, $x = \ln S$, $C_{eu}(S, t)$ — цена европейского опциона, $C_{as}(S, I, t)$ — цена азиатского опциона, где $I = \int_0^T S(\tau)d\tau$.

Исходный временной промежуток разобьем на равные части длиной Δt точками $t_0 < t_1 < \cdots < t_n < t_{n+1}$, причем $t_0 = 0$, $t_{n+1} = T$. При решении задачи ценообразования опционов на гибридной системе была применена модель, основанная на континуальном интеграле:

$$C_{eu}(S_0, 0) = e^{-rT} \int_{-\infty}^{+\infty} \dots \int_{-\infty}^{+\infty} dx_1 \dots dx_n dx_{n+1} \max\{e^{x_{n+1}} - K, 0\} \times \\ \times \frac{1}{(2\pi\sigma^2\Delta t)^{(n+1)/2}} \exp\left\{\sum_{i=1}^{n+1}\left[-\frac{1}{2\sigma^2\Delta t}[x_i - (x_{i-1} + \mu\Delta t)]^2\right]\right\} \tag{1}$$

$$\mu = r - \frac{\sigma^2}{2}$$

где r — безрисковая процентная ставка, σ — волатильность. Подробности можно найти в работах [Montagna, Nicrosini, Moreni, 2002], [Lientsky, 1997].

При помощи данного континуального интеграла определяется цена базового актива в момент исполнения опциона.

Для программирования приведенного выше выражения необходимо выполнить следующие основные стадии алгоритма:

1. Генерация набора случайных чисел $(x_0, x_1, \ldots, x_n, x_{n+1})$, таким образом, чтобы плотность распределения каждого x была равна:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2\Delta t}} \exp\left\{-\frac{1}{2\sigma^2\Delta t}[x - (x_{i-1} + \mu\Delta t)]^2\right\}. \quad (2)$$

2. Подставить x_{n+1} в подынтегральную функцию:

$$S(T) = \max\{e^{X_{n+1}} - K, 0\}. \quad (3)$$

3. Вычислить опцион по формуле:

$$C(S,t) = e^{-rT}\frac{1}{N}\sum_{i=1}^{N} S_i.$$

Для расчета интеграла необходимо сгенерировать набор случайных чисел таким образом, что плотность каждого из них зависит от предыдущего. В этом кроется основная проблема для параллелизма. В случае европейского опциона необходимо взять последнее случайное число из набора и подставить в функцию S(T).

Первый и второй этап являются одной итерацией Монте-Карло. Для точности вычислений необходимо выполнить как можно больше таких итераций. После чего подставить полученный результат в формулу вычисления опциона.

Для избавления от зависимости необходимо провести несколько математических преобразований.

Модернизация алгоритма

Из формулы плотности распределения случайной величины x_i (2) следует, что она распределена по нормальному закону

$$x_n \sim N(\mu\Delta t + x_{n-1}, \sigma\sqrt{\Delta t}),$$

с математическим ожиданием $M[x] = \mu\Delta t + x_{n-1}$ и дисперсией $D[x] = \sigma\sqrt{\Delta t}$.

Тогда x_i можно получить из случайной величины $\xi_i \sim N(0,1)$ следующим образом:

$$x_n = M[x] + D[x]\xi_n.$$

Или, подставляя выражения для $M[x]$ и $D[x]$, имеем:

$$x_n = \mu\Delta t + x_{n-1} + \sigma\sqrt{\Delta t}\,\xi_n. \quad (4)$$

Преобразуем данное выражение к более удобному виду, для этого заметим:

$$x_{n+1} = \xi_{n+1}\sigma\sqrt{\Delta t} + \mu\Delta t + x_n = \xi_{n+1}\sigma\sqrt{\Delta t} + \mu\Delta t + \xi_n\sigma\sqrt{\Delta t} + \mu\Delta t + x_{n-1} =$$

$$= (\xi_{n+1} + \xi_n)\sigma\sqrt{\Delta t} + 2\mu\Delta t + x_{n-1} = \sigma\sqrt{\Delta t}\sum_{i=1}^{n+1}\xi_i + (n+1)\mu\Delta t + x_0.$$

Таким образом имеем:

$$x_{n+1} = \sigma\sqrt{\Delta t}\sum_{i=1}^{n+1}\xi_i + (n+1)\mu\Delta t + x_0.$$

В конечном итоге, для получения случайной величины x_{n+1} требуется просуммировать сгенерированный массив случайных чисел со стандартным нормальным распределением, помножить его на $\sigma\sqrt{\Delta t}$ и прибавить значение выражения $(n+1)\mu\Delta t + x_0$, которое рассчитывается заранее.

Преимущество данной модификации заключается в том, что в ней отсутствуют зависимости, поэтому её можно легко запрограммировать и получить ускорение, используя CUDA или OpenCL на гибридной системе.

Отличие азиатского опциона состоит лишь в том, что его стоимость вычисляется по формуле:

$$C_{as}(S_0, I_0, 0) = e^{-rT} \int_{-\infty}^{+\infty} \dots \int_{-\infty}^{+\infty} dx_N \dots dx_1 \left(\frac{1}{2\pi\sigma^2\Delta t}\right)^{N/2} \times \\ \times \exp\left\{-\sum_{n=1}^{N} \frac{1}{2\sigma^2\Delta t}[x_n - (x_{n-1} + \mu\Delta t)]^2\right\} \times \\ \times \max\left(\frac{\Delta t \sum_{n=1}^{N} I(x_n, x_{n-1}, \Delta t)}{T} - K, 0\right), \tag{5}$$

где

$$I(x_n, x_{n-1}, \Delta t) = e^{x_n}\left(\frac{1}{\Delta x_n} + \frac{\sigma^2\Delta t}{2\Delta x_n^2} - \frac{\sigma^2\Delta t}{\Delta x_n^3}\right) + e^{x_{n-1}}\left(-\frac{1}{\Delta x_n} + \frac{\sigma^2\Delta t}{2\Delta x_n^2} + \frac{\sigma^2\Delta t}{\Delta x_n^3}\right),$$

Таким образом, все рассуждения для европейского опциона остаются справедливыми и для случая азиатского опциона, только формула (3) будет иметь вид:

$$S(T) = \max\left(\frac{\Delta t \sum_{n=1}^{N} I(x_n, x_{n-1}, \Delta t)}{T} - K, 0\right).$$

Эксперименты

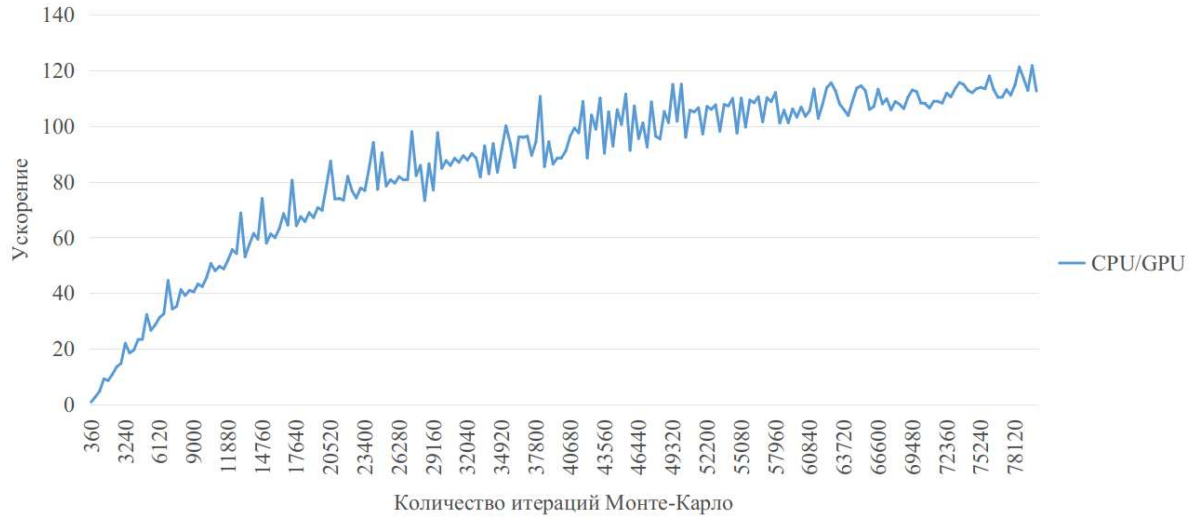

Рис. 1. Ускорение вычислений на GPU

Вопрос сходимости представленного алгоритма был исследован ранее в работе [Bogdanov, Stepanov, Khmel, 2015]. В данном эксперименте сравнивается время работы последовательного алгоритма на CPU (Intel Xeon) и параллельного на GPU (Tesla K40M). На рис. 1 приведено ускорение вычислений на GPU. Можно заметить, что график имеет типичное насыщение. Это означает, что на больших наборах ускорение сильно не изменится. В результате использования графической карты Tesla K40M удалось получить ускорение в 110 раз. Также проводился эксперимент по масштабированию этой задачи на несколько GPU (рис. 2). Результаты показали, что метод Монте-Карло можно перенести на несколько графических процессоров. Увеличение числа процессоров в 2 раза дает преимущество в скорости в 1.6 раз. Таким образом, масштабирование GPU можно использовать либо для ускорения вычислений, либо для увеличения их точности.

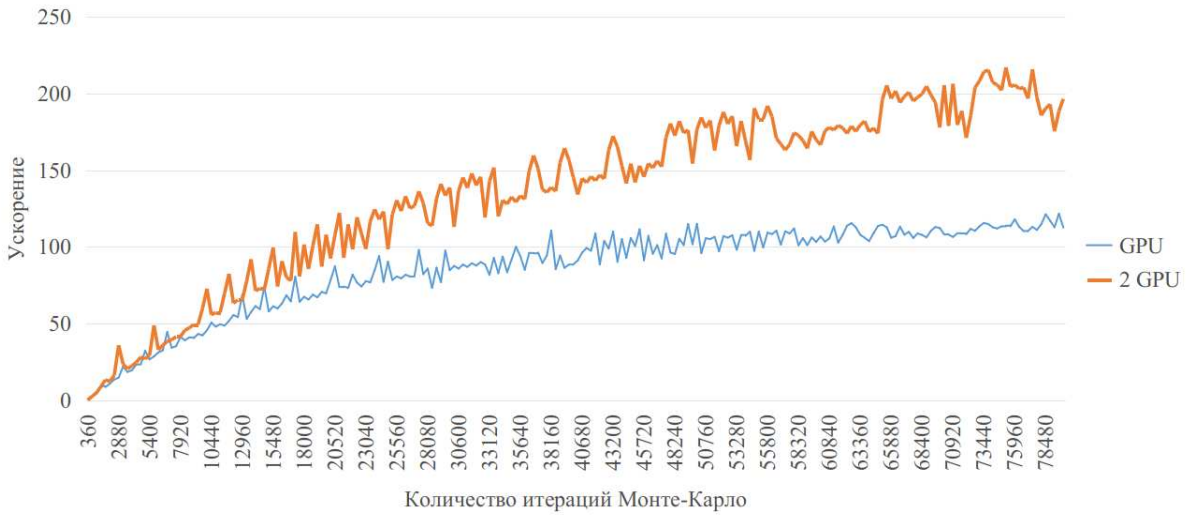

Рис. 2. Ускорение вычислений на двух GPU

Заключение

Подводя итог, следует отметить, что для некоторого класса задач использование GPGPU дает значительное преимущество.

Алгоритмы, написанные для GPU, требуют глубокого понимания принципов распараллеливания задач и особенностей архитектур графических процессоров

В будущем планируется завершить создание программно-аппаратного комплекса для решения поставленной задачи.

Список литературы

G. Montagna, O. Nicrosini, N. Moreni A path integral way to option pricing // Physica A: Statistical Mechanics and its Applications. — 2002. — Vol. 310, Issues 3–4. — P. 450–466

V. Linetsky The path integral approach to financial modeling and options pricing // Computational Economics. — 1997. — Vol. 11. Issue 1. — P. 129 – 163

A. V. Bogdanov, E. A. Stepanov, D. S. Khmel Assessment of the dynamics of Asian and European option on hybrid system // Journal of Physics: Conference Series. — 2015. — Vol. 681. — P. 121 —129

Optimization algorithm for computing options for the hybrid system

D. S. Khmel[a], E. A. Stepanov[b], A. V. Bogdanov[c]

Saint Petersburg University,
7/9 Universitetskaya emb., St. Petersburg, 199034

E-mail: [a] dskhmel@cc.spbu.ru, [b] e.an.stepanov@gmail.com, [c] bogdanov@csa.ru

In the article the problem of optimization of GPGPU option pricing algorithms. The main goal to achieve maximum efficiency from the use of the hybrid system. The authors offered some transformation algorithm derived from Blake-Scholes model for pricing European and Asian option on the Monte Carlo method based on GPGPU architecture features. The basic idea is the constant optimization of work with one large array of data.

Keywords: European option, Asian option, hybrid system, GPGPU, CUDA, Monte-Carlo method

This work was supported by RFBR grants:
17-57-80111 FTRC-HPC: High-Performance Computing with Networked Fault Tolerant Reconfigurable Computing Units,
16-57-48005 Multi FPU on Chip (MFoC) for HPC applications,
16-07-01111 Building Unix-like environment to control virtual supercomputer.