УДК 004.272.26

АНАЛИЗ ЭФФЕКТИВНОСТИ ИСПОЛЬЗОВАНИЯ GPU ДЛЯ АВТОМАТИЧЕСКОГО СИНТЕЗА СИСТЕМЫ УПРАВЛЕНИЯ МОБИЛЬНЫМ РОБОТОМ

А.А. Сергеев, В.О. Клебан, А.А. Шалыто

Исследуется эффективность использования графических сопроцессоров (GPU) для синтеза систем автоматического управления (САУ) мобильным роботом. В ходе работы учитываются и анализируются их особенности применительно к данной проблеме. В качестве модельной задачи для оценки эффективности синтеза выбрана параллельная реализация генетических алгоритмов на примере синтеза САУ роботом для соревнований «Сумо роботов».
Ключевые слова: производительность, GPU, генетический алгоритм, мобильный робот, синтез системы управления, конечный автомат, нейрон.

Введение

Синтез САУ, например, мобильным роботом может быть выполнен с помощью генетических алгоритмов (ГА), которые требуют значительного объема вычислений и являются в высокой степени параллельными. Поэтому актуально использование GPU для решения данной задачи. ГА в настоящее время уже реализованы на GPU, и они позволили получить существенное ускорение по сравнению с процессорной реализацией, однако применительно к синтезу САУ мобильных роботов такие исследования не выполнялись.

Постановка задачи

Цель работы – исследовать эффективность реализации синтеза системы управления мобильным роботом с использованием GPU. В качестве критерия эффективности рассматривается время, затрачиваемое на расчет фиксированной задачи при помощи GPU. Для оценки возможностей GPU рассматриваются особенности вычисления на ней и приводятся теоретические оценки.

В качестве эксперимента исследуется эффективность использования GPU для ГА на примере задачи «Сумо роботов». Оценка может быть получена путем анализа количества выращенных поколений и рассчитанных особей на машине с использованием GPU и без. Разрабатываются модели робота, его системы автоматического управления, выясняются особенности реализации ГА. Для проведения эксперимента и получения его результатов создается визуальная среда. Таким образом, в задачу входит подтверждение эффективности GPU для синтеза системы управления роботом на практике.

Сумо роботов

«Сумо роботов» представляет собой антагонистическую игру. Участие в ней принимают два робота. Они устанавливаются на плоскую круглую арену в выбранные хозяевами позиции и начинают движение по сигналу. Целью игры является выталкивание противника за пределы арены. Продолжительность поединка, как правило, ограничена 30 с. В начале игры роботы должны быть установлены за пределами ограничительных линий.

Традиционно робот имеет два колеса, скользящий шарик в качестве третьей точки опоры, индикатор линии и дальномер, например, инфракрасную пушку. Передвигается такой робот за счет разности хода между колесами.

Обзор

Исследований эффективности GPU для задач синтеза САУ мобильными роботами обнаружено не было. Существуют работы, например [1], посвященные обучению системы управления роботом с использованием ГА, но в них не применяются вычисления на GPU. Также известны работы [2–3], в которых выполняется оценка роста производительности при использовании GPU для генетических алгоритмов.

Архитектура комплекса

Тестовая машина имеет следующую конфигурацию: Intel Core i3 2.93GHz, 2 GB RAM, nVidia GeForce gtx 275 1792 MB. Для тестирования используется также видеокарта gt 240 512 MB. Пиковая производительность gtx 275 по данным nVidia составляет 1070 GFLOP/s. Для i3 это значение достигает порядка 40 GFLOP/s.

На самом деле, быстродействие всегда заметно ниже пикового, поскольку необходимо осуществлять доступ к памяти, выполнять медленные операции, синхронизировать потоки. Реально достижимая производительность на процессоре на большинстве параллельных звдвя, использующих полную мощ-

ность видеокарты, составляет около 70% от максимальной. Для видеокарты это значение всегда меньше, около 50–60%. Если учесть эти оценки, то получим, что на данном стенде можно достичь ускорение порядка 20 раз, при этом максимальное ускорение оценим примерно как 27 раз (отношение пиковых производительностей).

Физическая модель робота

Физическая модель робота состоит из тела робота и нескольких колес. Тело представляет из себя цилиндр с основанием полигона. Для передвижения по поверхности арены роботы используют колеса, приводимые в движение двигателями. Таким образом, момент колеса ограничен по модулю. Колеса и тело робота взвешены и имеют ненулевой момент инерции.

Робот использует для идентификации себя в пространстве воображаемую камеру, а для наблюдения ему доступен угол обзора перед собой. Также робот снабжен датчиком цвета точки арены под собой. Это позволит ему немедленно среагировать и изменить поведение, находясь на краю арены.

Расчет физической модели

Импульс и момент импульса робота изменяются под воздействием сил трения колес о поверхность арены. Сила трения, действующая со стороны опоры на колесо, рассчитывается как

$$\left\|\mathbf{F}_{\parallel}\right\| = \mu \cdot \left\|\mathbf{N}\right\| \cdot \mathbf{f}((\boldsymbol{\omega} \times \mathbf{r}) - \mathbf{v}_{\parallel}) .$$

Здесь **F** – сила трения на колесо, **v** – скорость колеса относительно поверхности арены, **ω** – угловая скорость колеса, **N –** сила реакции опоры (рисунок). Сила трения, таким образом, определяется относительной скоростью двух поверхностей и реакцией опоры. Скорость колеса рассчитывается из скорости центра масс робота и его угловой скорости как $\mathbf{v} = \mathbf{v}_{cm} + \boldsymbol{\omega} \times \mathbf{r}_{cm}$. Ускорение робота рассчитывается из второго закона Ньютона, а угловое ускорение – из уравнения моментов: $\mathbf{F}=m\mathbf{a}$; $\mathbf{I}\beta = \sum_{i=1}^{n} \mathbf{M}_i$.

В случае столкновения на робота действуют добавочные силы. В последнем случае важно делать поправки для соблюдения закона сохранения энергии.

При малом проскальзывании по поверхности арены для двухколесного робота поворот происходит так, как изображено на рисунке. За счет разностей моментов сил создается разность угловых скоростей, обеспечивающая разность скоростей левого и правого борта робота. Тогда робот поворачивает вокруг обозначенной на рисунке точки *O*.

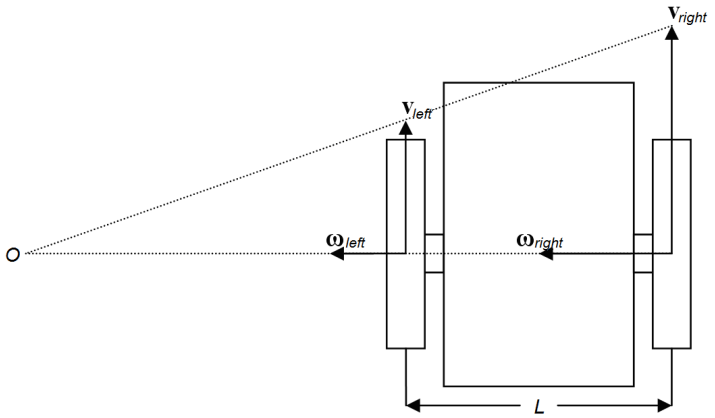

Рисунок. Поворот двухколесного робота

Используемая модель САУ

Единственное, что робот может использовать для изменения своего положение в пространстве, – это моменты сил, прикладываемые к колесам. Для реализации такого поведения необходимо на каждом шаге вычислять эти моменты. При выборе модели рассматривались следующие критерии для выбора модели системы управления:

1. Поведение робота должно в большинстве случаев непрерывно зависеть от входных параметров;
2. Робот избирает стратегию поведения в зависимости от действий противника;
3. Система управления должна иметь достаточно простую реализацию для обеспечения высокой скорости расчета действий и удобной для применения ГА.

Нейрон описывается линейной функцей от входных параметров:

$$output = f\left(\sum_{i=0}^{n-1} input[i] \cdot weight[i]\right).$$

Здесь f – передаточная функция. В частном случае положим ее тождественным оператором. Такой нейрон также называют звездой Гроссберга. Таким образом, вход нейрона есть фактически массив вещественных чисел, а выход – одно вещественное число.

Перцептрон есть модификация нейрона, реализующая пороговую функцию:

$$output = f\left(\sum_{i=0}^{n-1} input[i] \cdot weight[i]\right), \text{ где } f(x) = \begin{cases} 0, x <= threshold, \\ 1, x > threshold. \end{cases}$$

Помимо весов, перцептрон опционально хранит еще и пороговое значение. Выходом его является уже целое число – нуль или единица.

Традиционно нейроны и их модификации применяются для построения нейронных сетей, где выход одного нейрона подается на вход другому. В данной работе использована несколько другая схема, основанная на передаче управления между нейронами при помощи конечного автомата.

Чтобы сделать поведение робота достаточно естественным и удовлетворить приведенным требованиям, была разработана следующая модель:

1. Моменты колес рассчитываются из входных параметров как выходы нейронов;
2. Для определения последовательности действий используется конечный автомат.

Каждое из состояний автомата имеет свои нейроны для расчета моментов, по одному нейрону на колесо. На вход нейронам подаются входные параметры робота, представляющие из себя вектор чисел. Таковыми являются, например, расстояние до противника, данные детектора близости линии. В зависимости от наличия противника в поле зрения этот вектор имеет различную длину и подается на вход нейронам. Приведенная модель автомата является автоматом Мура [2], в котором значение функции выхода есть моменты колес.

САУ робота описывается автоматом с 32-мя состояниями. Состояние имеет 8 пар нейронов (максимальное число колес) и 3 пары перцептронов. Первый нейрон или перцептрон в паре имеет длину входа в 5 параметров, второй – в 9. В зависимости от видимости противника активны первый или второй из них.

Для перехода в новое состояние происходит расчет значений перцептронов с такими же входными параметрами. Номер нового состояния формируется исходя из выходов перцептронов и индикатора наличия противника в поле видимости и индикатора линии.

Визуализация

Для тестирования и оценки результатов была разработана визуальная среда Sumo Robot Simulation. Это приложение написано на языке C++ с использованием для визуализации OpenGL [4], для реализации GUI – библиотеки Qt [5], для расчетов на GPU – динамическую библиотеку на CUDA [6].

Окно просмотра статистики роста позволяет получать всю необходимую информацию о росте популяции: просматривать график фитнесс-функции и сохранять его, оценивать скорость генерации новых поколений, просматривать особей в таблице, сортировать их и сохранять в файл.

Используемая модель ГА

В данной работе будем использовать островную модель ГА. Выбранная реализация ГА использует также конкурентный элитизм. Для отбора особей применяется метод рулетки. В отборе на равных основаниях участвуют родители и их потомки. Хромосома представляет собой строку вещественных чисел одинарной точности, являющуюся последовательностью весов всех используемых нейронов и перцептронов.

Оценка приспособленности рассчитывается на основе результатов его поединков. Для этого создается набор из неизменных тестирующих особей, с каждой из которых происходит поединок, и после этого суммируются результаты. Всего в данной реализации предусмотрено 100 фиксированных особей для составления подобной оценки.

Особенности вычислений на GPU

Время доступа к регистрам и разделяемой памяти составляет четыре тактовых цикла против 400–600 для доступа к глобальной памяти [7]. На практике использование разделяемой памяти вместо глобальной может обеспечивать ускорение около 10–20 раз [8].

Код, выполняемый на устройстве, не имеет возможности напрямую выделять память на видеокарте или в оперативной памяти. Конфигурация робота и набор переменных, определяющих его движение, занимают более 10 Кбайт и достаточно велики, для того чтобы не рассматривать идею разместить их в разделяемой памяти. В таком случае задействуем как можно больше потоков.

Для расчета выходов нейронов, суммы моментов и суммы сил при движении будем использовать параллельную редукцию (reduction) [9].

Реализация ГА

Применительно к данной задаче наиболее требовательной к объему вычислений частью ГА является расчет фитнесс-функции. Поэтому именно ее следует выполнять на GPU и параллельно генерировать новых особей, пользуясь возможностью асинхронного запуска ядер на GPU.

Параллельно на видеокарте происходит симуляция битв, загруженных в буфер. Каждая битва рассчитывается на фиксированное число шагов. Генерация новых экземпляров происходит на процессоре в результате скрещиваний и мутаций.

Выполнение на CPU теоретически возможно путем компилирования кода, написанного на CUDA, в режиме эмуляции. Однако использование эмуляции в настоящее время не рекомендуется (deprecated). Поэтому аналогичный код переписан для выполнения на процессоре.

Теоретическая оценка прироста производительности

Для параллельно выполняемой задачи справедлив закон Амдала: максимально возможное ускорение от параллельного выполнения равняется (для N потоков параллельно)

$$S = \frac{1}{(1-P) + \frac{P}{N}}.$$

Оценка снизу распараллеливаемого объема вычислений P дает значение 0,99. Считая производительность видеокарты средней для математических задач, получим $S \approx 18{,}4$ (взято $N=20$ – средний коэффициент ускорения видеокарты относительно процессора). Значит, можно ожидать на порядок бóльшую производительность при использовании GPU.

Результаты

Сравнивая время достижения максимальных значений фитнесс-функции, полученное программно, можно судить о приросте производительности для CPU и разных моделей GPU. В табл. 1 приведены вычислительные мощности используемых устройств, цены приобретения и средние выигрыши в производительности.

	GFLOP/s	Цена, руб	Выигрыш
i3 2.93 GHz	40	4000	1,00
gt 240	330	3000	5,5
gt 275	1070	8000	13,5

Таблица 1. Сравнение производительностей и ценовых категорий устройств

	GPU	CPU	Средний прирост	Конфигурация
Простой ГА	4,08	81,88	20,1	AMD Athlon 2500+, nVidia GeForce 6800GT
ГА для оптимизации	0,001–0,1	0,25–0,26	30,0	Intel Core i7 920 3.2GHz, nVidia gtx 8800
Муравьиный алгоритм	1,06–0,85	22,49	6,0	Core 2 Quad Q9550, nVidia gtx 280

Таблица 2. Сравнение приростов производительностей для различных алгоритмов

Результат 20,1 (табл. 2) был получен при реализации на GPU простейшего ГА для фитнесс-функции – несложной функции вещественных переменных. Здесь вычисление фитнесс-функции имеет очевидные преимущества в скорости перед центральным процессором. Операции поиска лучших индивидуумов, кроссовер и мутация реализованы параллельно на GPU, что не представляет сложности для данной задачи. Здесь возможно эффективное использование разделяемой памяти (shared memory) в силу малого объема данных для отдельной подзадачи.

Результат 6,0 соответствует ускорению на муравьином алгоритме. Эта задача с высоким уровнем параллельности, тем не менее, не показала высокой степени ускорения.

Работа по использованию GPU для решения задач многомерной оптимизации (второй ряд) показывает высокий разброс результатов на различных задачах: выигрыш в скорости составил от 2,5 раза до тысяч раз. Это означает случайный разброс значений, поскольку показатели слишком высоки, а ускоре-

ние существенно превосходит соотношение производительностей устройств. В то же время средние показатели ускорения составляют 20–30 раз.

Для прироста системы управления роботом (табл. 1) был получен результат 13,5. Это несколько медленней лучших результатов для задач оптимизации и простого ГА. В то же время теоретический результат вполне подтверждается экспериментом. Потеря производительности наблюдается по следующим причинам:

1. Активное использование глобальной памяти. САУ и конфигурация роботов слишком велики для разделяемой памяти;
2. Часть потоков с подзадачами завершается довольно быстро, что позволяет процессору существенно экономить время.

Заключение

В работе проанализирована эффективность GPU для синтеза системы автоматического управления мобильным роботом. Рассмотрены особенности, характерные для реализации подобных задач с использованием GPU. Поставлен эксперимент на примере синтеза САУ для робота Сумо. Получена оценка прироста производительности по сравнению с решением без использования GPU. Согласно результатам анализа и поставленного эксперимента, использование GPU высокоэффективно для поставленной задачи.

В качестве главного недостатка GPU при решении задач синтеза САУ выделяется малый объем разделяемой памяти. Достоинствами являются возможность асинхронного запуска ядра и высокая вычислительная мощность GPU относительно CPU за счет большого числа параллельно рассчитываемых нитей.

Исследование выполнено по Федеральной целевой программе «Научные и научно педагогические кадры инновационной России на 2009–2013 годы» в рамках государственного контракта П2236 от 11 ноября 2009 года.

Литература

1. Jiming Liu, Shiwu Zhang. Multi-phase sumo maneuver learning // Robotica 22. – 2004. – № 1. – P. 61–75.
2. Qizhi Yu, Chongcheng Chen, Zhigeng Pan. Parallel Genetic Algorithms on Programmable Graphics Hardware. [Электронный ресурс]. – Режим доступа: http://www.cad.zju.edu.cn/home/yqz/projects/gagpu/icnc05.pdf, свободный. Яз. англ. (дата обращения 19.02.2011).
3. Pospichal P., Jaros J. GPU-based Acceleration of the Genetic Algorithm. [Электронный ресурс]. – Режим доступа: http://www.gpgpgpu.com/gecco2009/7.pdf, свободный. Яз. англ. (дата обращения 19.02.2011).
4. Wright R., Lipchak B. OpenGL superbible // SAMS publishing, 2005.
5. Земсков Ю.В. Qt4 на примерах. – СПб: БХВ Петербург, 2008.
6. Боресков А.В., Харламов А.А. Основы работы с технологией CUDA. – М: ДМК пресс. – 2010.
7. NVIDIA Corporation. Fermi Compatibility Guide for CUDA Applications. 2010. [Электронный ресурс]. – Режим доступа: http://developer.download.nvidia.com/compute/cuda/3_0/docs/NVIDIA_FermiCompatibilityGuide.pdf, свободный. Яз. англ. (дата обращения 19.02.2011).
8. NVIDIA Corporation. NVIDIA CUDA Programming Guide. 2010. [Электронный ресурс]. – Режим доступа: http://developer.download.nvidia.com/compute/cuda/2_3/toolkit/docs/NVIDIA_CUDA_ProgrammingGuide_2.3.pdf, свободный. Яз. англ. (дата обращения 19.02.2011).
9. NVIDIA Corporation. NVIDIA CUDA Best Practices Guide. 2010. [Электронный ресурс]. – Режим доступа: http://developer.download.nvidia.com/compute/cuda/2_3/toolkit/docs/NVIDIA_CUDA_BestPracticesGuide_2.3.pdf, свободный. Яз. англ. (дата обращения 19.02.2011).

Сергеев Антон Алексеевич – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, студент, anton.a.sergeev@gmail.com

Клебан Виталий Олегович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, аспирант, vk.developer@gmail.com

Шалыто Анатолий Абрамович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, доктор технических наук, профессор, зав. кафедрой, shalyto@mail.ifmo.ru