УДК 681.52

ПАРАЛЛЕЛЬНЫЕ АЛГОРИТМЫ ОЦЕНКИ ИНФОРМАТИВНОСТИ ГЕОФИЗИЧЕСКОГО ПОЛЯ

В.Б.Костоусов
Институт математики и механики УрО РАН
Россия, 620219, Екатеринбург, ул. С.Ковалевской, 16
E-mail: vkost@imm.uran.ru

А.Е.Тарханов
Институт математики и механики УрО РАН
Россия, 620219, Екатеринбург, ул. С.Ковалевской, 16
E-mail: atarkhanov@imm.uran.ru

Ключевые слова: навигация по геофизическому полю, информативность геофизического поля, моделирование, параллельные вычисления, многопроцессорные графические ускорители
Key words: navigation on geophysical field, geophysical field informativeness, simulation, parallel computing, GPU

В докладе обсуждается задача оценки информативности геофизического поля. Точные оценки информативности поля позволяют качественно спроектировать корреляционно-экстремальную навигационную систему. Описывается способ оценки информативности, сочетающий точность оценок, получаемых с помощью имитационного моделирования, и скорость локальных оперативных оценок. Важной особенностью предлагаемых решений является применение параллельной технологии и новых вычислительных возможностей, предоставляемых современными многопроцессорными графическими ускорителями (GPU).
PARALLEL ALGORITHMS OF ESTIMATION OF GEOPHYSICAL FIELD INFORMATIVENESS / V.B.Kostousov (Institute of Mathematics and Mechanics, S. Kovalevskoy str. 16, Ekaterinburg 620219, Russia, E-mail: vkost@imm.uran.ru), A.E.Tarkhanov (Institute of Mathematics and Mechanics, S. Kovalevskoy str. 16, Ekaterinburg 620219, Russia, E-mail: atarkhanov@imm.uran.ru). This paper is devoted to the problem of estimation of geophysical field informativeness. Accurate estimates of field informativeness provide the design quality of extreme-correlation navigation system. One method of the informativeness estimation is proposed. This technique combines the accuracy of numerical simulation algorithms and computational speed of local estimation methods. Application of parallel technologies and computational power of modern GPUs are the main features of described method.

1. Введение

Оценка информативности геофизического поля является одной из задач, которая возникает при исследовании проблемы навигации по геофизическим полям [1]. Задача навигации по геофизическому полю состоит в определении местоположения движущегося объекта путём сопоставления измеренного фрагмента с эталоном поля. Информативность геофизического поля позволяет предсказывать возможную ошибку решения задачи навигации.

Среди существующих способов оценки информативности можно выделить следующие [1, гл.7]:

- локальные статистические характеристики поля (например, дисперсия и радиус корреляции, характеристики на основе градиентов поля),
- модуль информативности геофизического поля,
- статистическое моделирование.

Оценивание по локальным статистическим характеристикам поля достаточно легко вычисляется, но недостаточно точно. Результат оценивания трудно сопоставить непосредственно с ошибкой навигации. А если же это удаётся, то, тем не менее, нет гарантии, что полученная оценка навигационной ошибки будет достигнута. Соотношение, установленное между навигационной ошибкой и оценкой информативности по локальным статистическим характеристикам поля, определяется конкретным участком поля и, как правило, нарушается на всём поле.

Модуль информативности даёт гарантированную оценку ошибки и, хотя более затратен, чем предыдущий способ, но может быть приемлем по трудоёмкости. Однако, гарантированная оценка ошибки, в большинстве случаев, не может быть использована, так как чрезмерно пессимистична. Проблема состоит в том, что один плохой случай нарушения глобальной информативности испортит гарантированную оценку вне зависимости от своей статистической значимости.

На фоне этих способов оценивания информативности, статистическое моделирование выглядит гибкой, и во многом универсальной методикой. Оно позволяет моделировать процесс решения конкретной задачи навигации, моделировать процесс получения и обработки информации, в том числе, ошибки данных. Статистическое моделирование позволяет получать удобные оценки информативности, например ошибку навигации, которая не будет превышена с заданной вероятностью.

Главным недостатком метода статистического моделирования является его вычислительная сложность, поскольку требуется полностью решать задачу навигации достаточно большое число раз, создавая в каждом случае модель измеренного фрагмента геофизического поля.

В данной работе решается задача существенного ускорения статистического метода оценки информативности геофизического поля. Предлагается несколько вариантов решения задачи, в частности, вариант, использующий разработанный в начале 80-х годов В.Л. Гасиловым [1, гл.8] алгоритм специализированного градиентного спуска для решения задачи навигации по геофизическому полю. Важной особенностью предлагаемых решений является применение параллельных технологий и новых вычислительных возможностей, предоставляемых современными многопроцессорными графическими ускорителями (GPU). На примере задачи навигации по рельефу местности приводятся результаты вычислительного эксперимента, демонстрирующие значительное сокращение времени счёта.

2. Задача ускорения статистического моделирования

Для дальнейшего изложения, назовём алгоритм решения задачи навигации по рельефу местности алгоритмом привязки измеренного фрагмента к эталонной карте ГФП или, кратко, *алгоритмом привязки*. Алгоритм получения оценок информативности кроме алгоритма привязки включает: моделирование фрагмента поля высот рельефа, организацию серии испытаний, т.е. тестовых решений задачи навигации алгоритмом привязки, а так же обработку результатов. Но, тем не менее, основную вычислительную сложность представляет сам алгоритм привязки.

Рассмотрим конкретный пример алгоритма привязки, используемый в данной работе. Входом алгоритма являются матрица рельефа, которая выступает эталоном, и фрагмент, снятый во время движения летательного аппарата. Фрагмент имеет вид одномерного массива измеренной высоты. Выходными данными являются координаты летательного аппарата. Для решения задачи навигации, алгоритм привязки производит поиск положения фрагмента на эталоне, получая для каждого положения значение *функционала сравнения* и выбирая положение, которое соответствует экстремуму функционала. Вычисление функционала сравнения фрагмента с эталоном затратно. Время работы алгоритма определяется

количеством значений функционала сравнения, которое требуется вычислить в процессе поиска.

Базовая идея решения задачи поиска фрагмента на эталоне состоит в следующем. В каждом узле регулярной сетки, который задаёт *гипотезу* - предполагаемые значения искомых координат, происходит вычисление значения функционала сравнения фрагмента с эталоном. Выбрав гипотезу, которая доставляет экстремум функционалу, мы получим решение задачи.

Реализация данного алгоритма крайне неэффективна: если n - количество узлов сетки по каждой координате, то точность решения есть $O(1/n)$, а вычислительная сложность есть $O(n^2 \times m)$, где m есть число отсчётов фрагмента.

Первым шагом повышения вычислительной эффективности алгоритма является разбиение его на два этапа, называемых «грубый» поиск и «тонкий» поиск [1]. Для этапа «грубого» поиска характерна сетка с крупным шагом. Благодаря редкой сетке, количество вычислений функционала на этапе «грубого» поиска относительно невелико. Этап «тонкого» поиска использует результат «грубого» поиска таким образом, что проверяет гипотезы с мелким шагом в окрестности результата «грубого» поиска. Количество вычислений функционала значительно сокращается, но всё-таки остаётся большим, особенно на этапе «тонкого» поиска.

Время работы данного алгоритма приведено в Таблице 1 в колонке, обозначенной «без ускорения».

Следующим шагом в ускорении алгоритма является отказ от переборного «тонкого поиска» в пользу гораздо более быстрого специализированного градиентного спуска [1, гл.8]. Специализированный градиентный спуск, позволяет сократить время счёта на два порядка.

Время работы такого алгоритма приведено в Таблице 1 в колонке, обозначенной «ускорение: градиентный поиск». Для того чтобы получить на основе статистического моделирования метод построения оперативных оценок информативности, необходимо ускорить алгоритм ещё хотя бы на порядок.

К тому же текущий вариант алгоритма, выполняющего этап «тонкого» поиска посредством специализированного градиентного спуска, как обнаружилось, обладает значительным недостатком. На некоторых данных, график функционала становится сильно изрезанным (см. рис. 1), зона притяжения абсолютного минимума сужается и спуск не справляется с задачей поиска глобального минимума, застревая в одном из локальных минимумов.

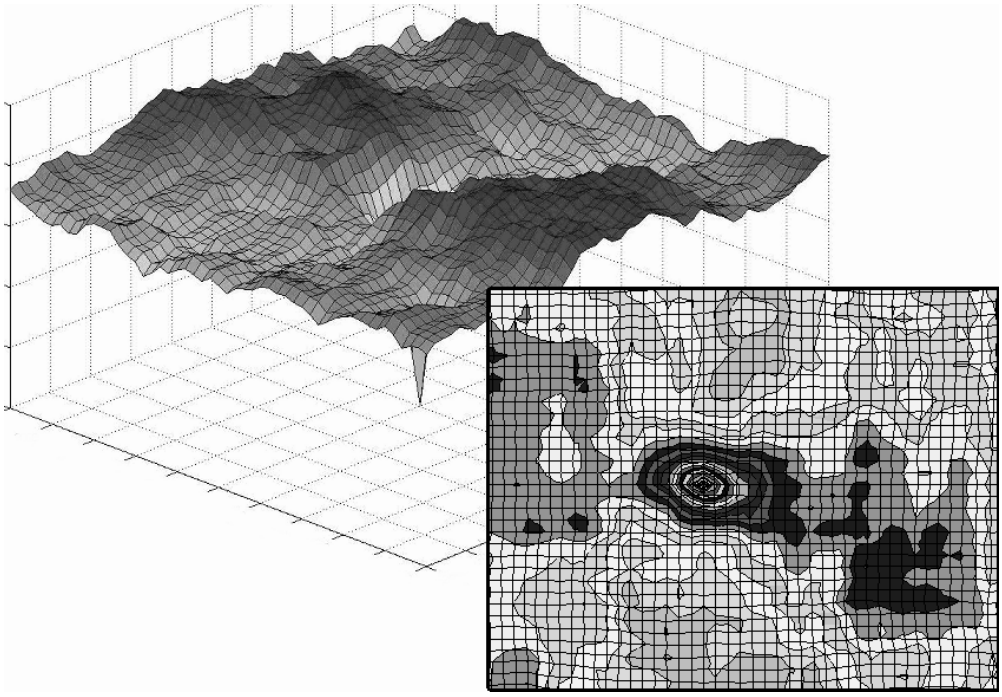

Рис.1. Пример графика функционала сравнения. По осям отложены искомые координаты. Абсолютный минимум соответствует истинному положению измеренного фрагмента ГФП.

3. Использование GPU для ускорения вычислений

Для достижения оперативности оценки требуется ускорить алгоритм привязки ещё на порядок. Исходя из предыдущего опыта [2], привлечение GPU способно дать ускорение на порядок и даже более. Возможности для этого в алгоритме привязки есть. Действительно, вычисления функционала сравнения, необходимые для этапа «грубого» поиска, независимы, а потому есть возможность производить счёт параллельно. Вычисления функционала для этапа «тонкого» поиска выполняются в строгой очерёдности, а потому сложно распараллелить этот этап.

Этап «грубого» поиска в реализации с градиентным «тонким» поиском занимает бóльшую часть времени вычислений. Для сравнения, на этапе «грубого» поиска происходит примерно 1500 вычислений функционала сравнения, когда специального вида градиентный спуск, применённый для этапа «тонкого» поиска, выполняет только 50-100 операций вычисления функционала.

Таким образом, если распараллелить вычисление значений функционала сравнения для «грубого» поиска, то этого должно быть достаточно для получения требуемого ускорения. Но 1500 потоков – это небольшое количество потоков для мощного GPU. Бóльшего параллелизма, и, соответственно, бóльшего коэффициента ускорения на мощном GPU, можно добиться, решая задачу грубого поиска сразу для всех измеренных фрагментов. Т.е. вычисляя параллельно значения функционала сравнения каждого измеренного фрагмента с каждым эталонным.

Есть возможность получить дополнительное ускорение, подготавливая эталонные фрагменты на GPU. Для получения эталонных фрагментов производится билинейная интерполяция на элементах матрицы рельефа, а эта операция является естественной для GPU и, в случае хранения данных в виде текстур, выполняется при текстурной выборке аппаратным способом, т.е. эффективно.

Для каждого измеренного фрагмента набор используемых на этапе «грубого» поиска эталонных фрагментов один и тот же. Поэтому есть возможность построить набор эталонных

фрагментов один раз и использовать его N раз, где N – кол-во измеренных фрагментов. По грубым оценкам для хранения всех фрагментов как измеренных, так и эталонных требуется несколько мегабайт памяти. Современный GPU располагает достаточным для этого объёмом памяти, а потому есть возможность насчитать заранее и хранить все требуемые фрагменты. Таким образом, разделение формирования эталонных фрагментов от вычисления функционала сравнения даёт существенный выигрыш по времени расчёта функционала сравнения.

Чтобы алгоритм не терял эффективность при большом числе измеренных фрагментов было решено остальные этапы алгоритма привязки выполнять на GPU параллельно для каждого измеренного фрагмента.

Ниже приведена схема параллельного алгоритма. В ней CPU и его память обозначены как *host*, а GPU и его память как *device*. Это является стандартным обозначением, принятым в CUDA [3], OpenCL [4] и прочих технологиях ускорения вычислений за счёт привлечения дополнительного вычислителя, в данном случае – GPU.

Алгоритм оценки информативности ГФП:

1. **Подготовка эталона ГФП.**
   Поместить в память *device* и *host* матрицу рельефа,
   для получения эталонных и измеренных фрагментов.
2. **Построение эталонных фрагментов.**
   Сформировать параллельно K_g эталонных фрагментов, используя
   аппаратную интерполяцию на *device*.
3. **Формирование измеренных фрагментов и «грубый» поиск.**
   3.1.Сформировать N измеренных фрагментов ГФП на *host* и перенести их
   в память device.
   3.2. Запустить счёт значений функционала для этапа «грубого» поиска
   на *device* параллельно для всех эталонных и всех измеренных фрагментов.
   3.3. Выбрать для каждого из N измеренных фрагментов решение
   этапа «грубого» поиска параллельно.
4. **Тонкий поиск и возврат результата.**
   4.1. Для N найденных решений в параллельном режиме осуществить
   градиентный спуск,
   4.2. Дождаться результата и перенести его в память *host*.
5. **Статистическая обработка результатов на *host*.**

Количество K_g эталонных фрагментов вычисляется по формуле $K_g = K_x \times K_y \times K_{vx} \times K_{vy}$, где K_x – количество гипотез по координате x, K_y – количество гипотез по координате y, K_{vx} – количество гипотез по проекции вектора скорости на ось x и K_{vx} – количество гипотез по проекции вектора скорости на ось y.

Обсудим перспективы ускорения за счет распараллеливания. Пусть M – это количество процессоров в параллельном вычислителе. Обозначим T_s - время работы последовательного алгоритма и T_p - время работы параллельного алгоритма. В случае «идеального» распараллеливания, который на практике редко реализуется, коэффициент ускорения $e = T_s/T_p$ равен M. В случае использования GPU ситуация, близкая к идеальной характеризуется следующими тремя свойствами [5]:

- объём передаваемой информации между *host* и *device* в пересчёте на количество вычислений на *device* мал,
- количество параллельно выполняемых потоков превышает количество процессоров на порядок или два,
- схема работы с памятью *device* учитывает особенности реализации обмена данными между процессорами и памятью *device*.

В предложенном алгоритме на *device* передаётся небольшой фрагмент матрицы рельефа, а возвращается набор из N векторов навигационных параметров. Обмен данными между *host* и *device* очень мал.

Высоким параллелизмом обладают подготовка эталонных фрагментов и счёт значений функционала для этапа «грубого» поиска, K_g и $N \times K_g$ потоков соответственно. Предположительно недостаточным параллелизмом обладают выбор решения этапа «грубого» поиска и этап «тонкого» поиска, имеющие по N потоков.

Можно считать, что запрос значений из памяти и запись происходят эффективно. Близкие в пространстве индексов потоки запрашивают данные из близко расположенных ячеек памяти и пишут данные в близко расположенные ячейки. Также используются выполняемые аппаратно текстурные выборки.

Счёт для этапа «грубого» поиска занимает в последовательном алгоритме бóльшую часть времени. В предложенном алгоритме этот этап эффективно распараллелен. Этап «тонкого» поиска распараллелен не столь эффективно, но он занимает гораздо меньше времени счёта. Таким образом, ускорение e должно отличаться от максимального, равного M, не более чем на 20-30%. При анализе фактического коэффициента ускорения нужно принимать в расчёт, что используются процессоры с разной скоростью выполнения операций.

Алгоритм реализован в виде программы на C++ в среде Microsoft Visual Studio 2008. В качестве технологии программирования задач для GPU использовалась OpenCL[4].

Для более ранней реализации алгоритма [2], был проведён вычислительный эксперимент, включающий 100 итераций моделирования. Результаты эксперимента представлены в Таблице 1. Характеристики использованных вычислителей представлены в Таблице 2.

Таблица 1. Время, затрачиваемое на 100 итераций моделирования.

Достигнутое ускорение			
	Без ускорения	Ускорение: «Градиентный поиск»	Ускорение: «Градиентный поиск» + GPU
Время счёта	16 м. 40 сек.	11 сек.	1 сек. 200 мс.
Ускорение	-	91x	810x (доп. 9x)

Таблица 2. Характеристики использованных вычислителей.

Вычислитель		Количество процессоров	Частота процессоров
CPU	AMD Turion	1	1,6 Ггц
GPU	nVidia GeForce 8600M GS	16	1,0 Ггц

Надо отметить, что хотя алгоритм реализован так, что практически полностью выполняется на GPU, в случае небольшого N (например, 100, как в проведённом эксперименте) существенным параллелизмом обладают только этапы получения эталонных фрагментов и вычисление значений функционала сравнения для этапа грубого поиска. Повышать параллелизм, выполняя несколько этапов алгоритма одновременно, также не представляется возможным из-за их строгой очерёдности: результат предыдущего этапа необходим для осуществления следующего.

4. Заключение

Совместное применение алгоритмических улучшений и современных технологий параллельных вычислений открывает путь к решению задачи получения оперативных оценок

информативности с помощью статистического моделирования. Но для полного решения задачи требуются дополнительные исследования.

В частности, градиентный спуск, выполняющий этап «тонкого» поиска, требует улучшения, которое позволит сохранить точность оценки.

Благодарности

Работа выполнена при финансовой поддержке программ фундаментальных исследований Президиума РАН: №09-П-1-1003 и №09-П-1-1013; а также гранта РФФИ №09-01-00523

Список литературы

1. Бердышев В.И., Костоусов В.Б. Экстремальные задачи и модели навигации по геофизическим полям. Научное издание. Екатеринбург: УрО РАН, 2007.
2. Тарханов А.Е. Использование GPU для оперативной оценки информативности геофизического поля. //Проблемы теоретической и прикладной математики: Труды 41-й Всероссийской молодёжной конференции., Екатеринбург, УрО РАН, 2010, С. 553-559.
3. NVIDIA CUDA Compute Unified Device Architecture. Programming Guide.: www.nvidia.com, 2010.
4. Khronos OpenCL – The open standard for parallel programming of heterogeneous systems.: www.khronos.org/opencl/, 2010.
5. NVIDIA OpenCL Best Practices Guide.: www.nvidia.com, 2010.