СПИСОК ЛИТЕРАТУРЫ

1. Подиновский В.В. Математическая теория выработки решений в сложных ситуациях. – М.: Министерство обороны СССР, 1981. – 212 с.
2. Дубов Ю.А., Травкин С.И., Якимец В.Н. Многокритериальные модели формирования и выбора вариантов систем. – М.: Наука, 1986. – 296 с.
3. Машунин Ю.К. Методы и модели векторной оптимизации. – М.: Наука, 1986. – 144 с.
4. Вентцель Е.С. Исследование операций. Методологические аспекты. – М.: Наука, 1972. – 105 с.
5. Степанов А.В. Человеко-машинная процедура принятия решений в задачах векторной оптимизации // Математическое моделирование. – 1991. – Т. 3. – № 5. – С. 61–73.
6. Соболь И.М., Статников Р.Б. Выбор оптимальных параметров в задачах со многими критериями. – М.: Наука, 1981. – 112 с.
7. Михалевич В.С., Волкович В.Л. Вычислительные методы исследования и проектирования сложных систем. – М.: Наука, 1982. – 278 с.
8. Рыков А.С. О диалоговых методах деформируемых конфигураций // Доклады РАН. – 2000. – Т. 375. – № 2. – С. 39–46.
9. Рыков А.С., Калашников А.Е. Диалоговый метод деформируемых конфигураций для многокритериальной оптимизации технологических процессов // Современные сложные системы управления (СССУ/HTCS 2003): Труды Междунар. конф. – Воронеж: ВГАСУ, 2003. – Т. 2. – С. 185–188.
10. Кейперс Л., Нидеррайтер Г. Равномерное распределение последовательностей. – М.: Наука, 1985. – 408 с.
11. Орлов В.А., Рейзлин В.И. Новое семейство квазислучайных последовательностей // Известия Томского политехнического университета. – 2012. – Т. 320. – № 2. – С. 24–26.

Поступила 14.03.2013 г.

УДК 519.6:004.93

АНАЛИЗ ПУТЕЙ ПОВЫШЕНИЯ ЭФФЕКТИВНОСТИ РАСЧЕТОВ ЧАСТОТНО-ВРЕМЕННОЙ КОРРЕЛЯЦИОННОЙ ФУНКЦИИ

Е.Е. Лунева, В.С. Аврамчук

Томский политехнический университет
E-mail: lee@tpu.ru, avs@tpu.ru

Рассмотрены технологии повышения эффективности математических расчетов на многопроцессорных системах с использованием графических процессоров NVIDA CUDA. Показано, что графические процессоры превосходят по скорости процессоры общего назначения при решении задач, связанных с вычислениями. Эффективность использования технологии Microsoft.NET Framework целиком зависит от числа вычислительных ядер в процессоре. Применение технологии CUDA позволяет привлечь уникальную вычислительную архитектуру графических процессоров и существенно сократить общее время выполнения расчетов.

Ключевые слова:
Цифровой сигнал, корреляционный анализ, Microsoft.NET Framework, NVIDIA CUDA.

Key words:
Digital signal, correlation analysis, Microsoft.NET Framework, NVIDIA CUDA.

Цифровая обработка сигналов в настоящее время является важным и перспективным направлением развития современной науки и техники. По мере развития вычислительных мощностей современных ЭВМ увеличиваются объёмы обрабатываемой информации, создаются совершенно новые алгоритмы обработки сигналов, расширяются сферы применения цифровой обработки сигналов. К создаваемым способам и программному обеспечению предъявляются более жесткие требования по быстродействию, возможности использования в режиме реального времени и точности. Удовлетворение этих требований невозможно без привлечения передовых вычислительных устройств и технологий. Цель данной работы заключается в исследовании возможных путей повышения эффективности использования аппаратных ресурсов ЭВМ при решении задач корреляционного анализа сигналов.

Корреляционный анализ сигналов находит широкое применение при решении задач неразрушающего контроля и диагностики, определения характеристик электрических систем, обработки цифровых изображений [1]. Функции корреляции достаточно просто определяются через дискретное преобразование Фурье (ДПФ) [1]. Эффективность вычисления в данном случае зависит от способа реализации ДПФ. Максимальная эффективность достигается при использовании быстрого преобразования Фурье (БПФ). Основным недостатком такого способа расчета корреляционных функций является отсутствие информации о связи сигналов в частотной области. Этого недостатка лишен метод частотно-временного корреляционного анализа [2], использование которого позволяет существенно повысить информативность проводимого анализа. Однако применение данного подхода сопряжено с большими вычислительными затратами, так как данный метод вычисления требует многократного выполнения процедур БПФ. Количество необходимых преобразований напрямую зависит от количества формируемых копий сигнала. Для оценки трудоемкости вычислений указанным

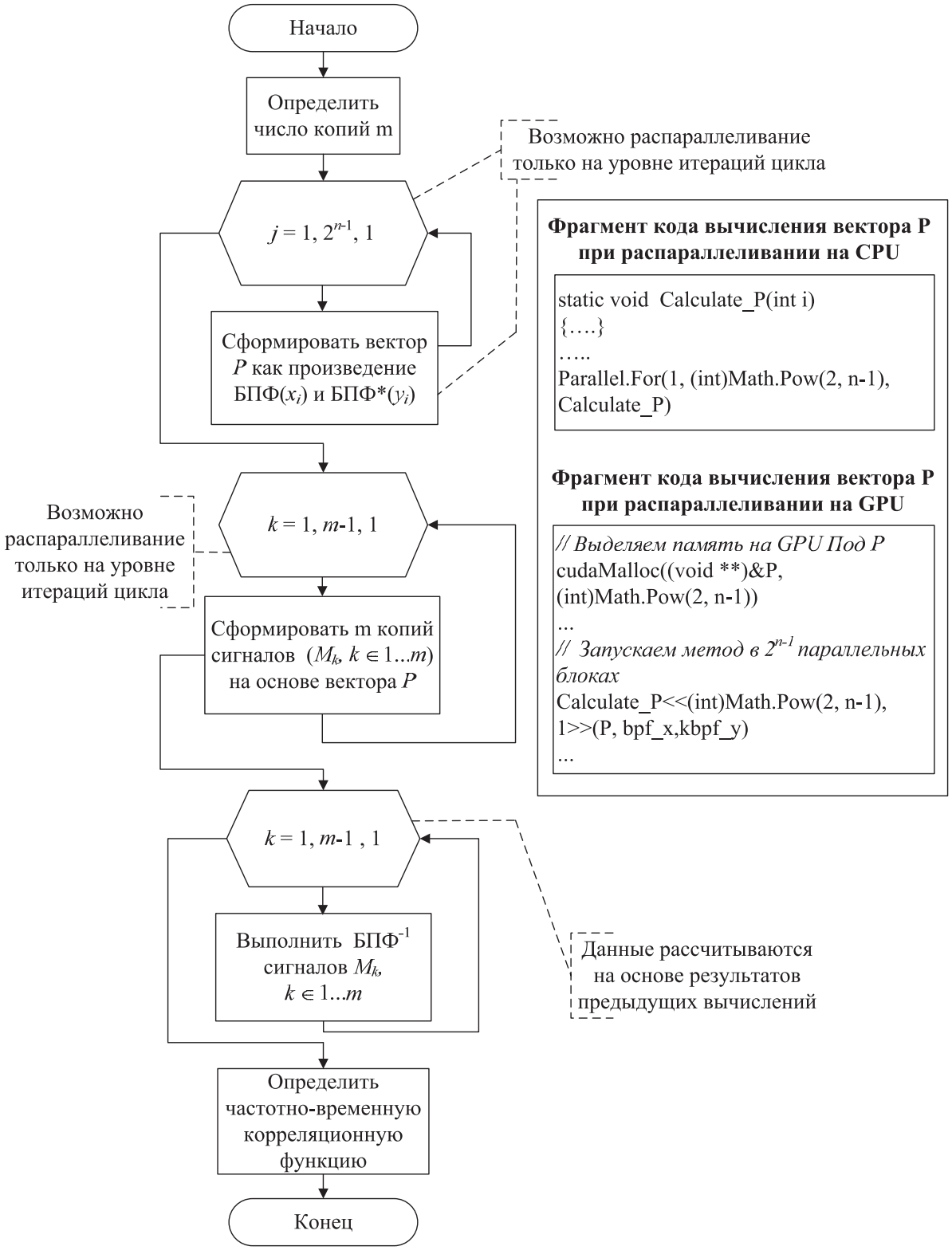

Рисунок. *Алгоритм вычисления частотно-временной корреляционной функции: m – число формируемых копий сигнала; x_i, y_i – дискретные отсчеты сигналов; БПФ* – комплексно-сопряженное значение результатов прямого ДПФ; БПФ$^{-1}$ – обратное ДПФ; P – вектор произведения результатов прямого ДПФ сигнала x_i с комплексно-сопряженными значениями прямого ДПФ сигнала y_i*

методом на рисунке приведен укрупненный алгоритм необходимых математических расчетов.

Несмотря на хорошую алгоритмизуемость и наличие большого числа оптимизированных библиотек вычисления БПФ, отмеченная выше проблема является узким местом при использовании данного подхода и создаваемого на его основе программного обеспечения. В этом случае эффективность создаваемых программ зависит от использования специализированных технологий.

Для многопроцессорных систем, в которых все вычисления выполняются исключительно на центральном процессоре (CPU), повысить эффективность можно с использованием технологий распараллеливания вычислительных процессов. Корпорацией Microsoft разработан универсальный набор инструментов параллельного выполнения заданий ParallelExtensions, входящих в состав набора Microsoft.NET Framework 4.0 [3]. Данный инструментарий позволяет автоматически использовать все доступные процессоры для выделенных блоков кода, пригодных для распараллеливания, в существующем последовательном коде. Разметка задач осуществляется посредством вызова предназначенных для этого статических методов класса Parallel, являющегося частью инструмента ParallelExtensions. В частности, методы «Parallel.For ()», «Parallel.Foreach ()» преобразуют обычный последовательно исполняемый цикл в параллельный. Произвольные участки кода размечаются путем оформления их в отдельные методы и последующего вызова с помощью метода «Parallel.Invoke ()».

Однако сложность разработки приложений на основе инструментария ParallelExtensions увеличивается при распараллеливании задач с использованием одних и тех же данных, что приводит к необходимости детального управления распределением массивов и витков циклов между потоками, а также обменом сообщениями между ними. Например, это возможно при разработке наиболее популярного типа приложений «WindowsForms», при обработке данных в фоновых потоках и графической визуализации этих данных в основном потоке. Также использование универсального инструментария ParallelExtensions не позволяет максимально эффективно использовать преимущества конкретной параллельной архитектуры. Кроме этого, учитывая, что в настоящее время повышение производительности центральных процессоров затруднено фундаментальными ограничениями при производстве интегральных схем, следует рассматривать и другие решения повышения вычислительной мощности системы.

В литературе [4] отмечено, что приложения, использующие графические процессоры (GPU) NVIDIA для неграфических вычислений, продемонстрировали существенный прирост эффективности вычислений, по сравнению с реализациями, построенными на базе одних лишь центральных процессоров. Технология CUDA (Compute Unified Device Architecture) представляет собой программно-аппаратную архитектуру, позволяющую применять графический процессор компании NVIDIA для неграфических расчетов на основе Runtime API.

Runtime API представляет собой расширение к языку C/C++. Данное расширение позволяет выделить память на GPU при помощи функции «cudamalloc ()», передать GPU на обработку участки кода (функция «cudaMemcpy ()»), оформленные как функции. Для параллельной обработки массивов данных может быть использован механизм блоков, позволяющий параллельно обрабатывать элементы массивов. Использование типа «dim3» CUDA позволяет работать с сетками – параллельными потоками для обработки матриц. Значимым достоинством CUDA является возможность прямого обращения к OpenGL и DirectX [4], что, несомненно, необходимо использовать в задачах с выводом изображений.

На основе рассмотренных технологий было создано программное обеспечение расчета частотно-временных корреляционных функций. В качестве алгоритма вычисления БПФ был выбран наиболее распространенный алгоритм Кули–Тьюки с фиксированным основанием 2, обладающий простотой реализацией, наглядностью и поддающийся эффективному распараллеливанию [5]. Сравнение эффективности рассмотренных технологий было проверено на ряде тестовых примеров. Результаты экспериментов были получены на размерах выборок: 2048, 4096, 8192, 16384, 32768 отсчетов. Количество формируемых копий m=1121. Результаты проведенных экспериментов сведены в таблицу. Максимальное время выполнения преобразований было получено на процессоре без реализации задач параллелизма вычислений. Для наглядности оценки эффективности применения рассмотренных технологий в таблице приведено отношение затраченного времени выполнения расчетов к максимальному времени вычислений, полученному на фиксированном размере выборки, выраженное в процентах.

Таблица. *Результаты экспериментов*

Размер выборки, отсчет	CPU Intel Celeron E1500 2 Core без Framework, %	CPU Intel Celeron E1500 2 Core + Framework, %	CUDA NVIDIA GeForce GTX640, %
2048	100	67	3,2
4096	100	61	2,9
8192	100	54	2,7
16384	100	52	2,5
32768	100	51	2,4

Исходя из полученных результатов можно констатировать, что применение технологии Microsoft.NET Framework на двуядерном процессоре позволило сократить время выполнения преобразований. Максимальная эффективность была получена при длине выборки 32768 отсчета, а минимальная – при 2048. Это различие легко объясняет-

ся тем, что время создания потока может быть сопоставимо, а иногда и превосходить время выполнения преобразования. Параллельная обработка данных в соответствии с представленным на рисунке (фрагменты кода) алгоритмом может быть выполнена только на уровне итераций цикла, а также при вычислении прямого и обратного преобразования Фурье, что требует привлечения значительных ресурсных затрат CPU на создание и завершение потоков.

Использование технологии CUDA привело к существенному сокращению времени обработки сигнала за счет высоких скоростных характеристик создания сеток и блоков потоков.

Выводы

Рассмотрены технологии увеличения эффективности проведения расчетов, позволяющих вычислить частотно-временную корреляционную функцию. За счет своей специализированной вычислительной архитектуры графические процессоры превосходят по скорости центральные процессоры общего назначения при решении задач, связанных с вычислениями.

Показано, что использование технологии Microsoft.NET Framework целиком зависит от числа вычислительных ядер в центральном процессоре. Предельное сокращение времени выполнения расчетов будет соответствовать количеству этих ядер, что подтверждается полученными результатами.

Применение технологии CUDA позволяет привлечь уникальную вычислительную архитектуру графических процессоров и существенно сократить общее время выполнения преобразований. Значительный эффект достигается при обработке большого количества данных, так как время создания потока может превосходить время выполнения преобразования. В то же время эффективность вычислений будет зависеть от конкретной реализации алгоритма быстрого преобразования Фурье.

Технология CUDA в настоящее время дает возможность создания эффективного программного обеспечения за счет своей уникальной вычислительной архитектуры.

СПИСОК ЛИТЕРАТУРЫ

1. Айфичер Э.С., Джервис Б.У. Цифровая обработка сигналов: практический подход. 2-е изд. – М.: Вильямс, 2008. – 992 с.
2. Аврамчук В.С., Чан Вьет Тьяу. Частотно-временной корреляционный анализ цифровых сигналов // Известия Томского политехнического университета. – 2009. – Т. 315. – № 5. – С. 112–115.
3. Рихтер Д. CLR via C#. Программирование на платформе Microsoft.NET Framework 4.0 на языке C#. 3-е изд. – СПб.: Питер, 2012. – 928 с.
4. Сандерс Д., Кэндрот Э. Технология CUDA в примерах: введение в программирование графических процессов. – М.: ДМК Пресс, 2011. – 232 с.
5. Нуссбаумер Г. Быстрое преобразование Фурье и алгоритмы вычисления свертки. – М.: Радио и связь, 1985. – 248 с.

Поступила 25.09.2013 г.