Вестник Рязанского государственного радиотехнического университета. 2014. № 48. С. 68-76.

3. Корячко В. П., Перепелкин Д. А., Иванчикова М. А. Разработка и исследование алгоритма быстрой перемаршрутизации трафика между центрами обработки данных // Радиотехника. 2016. № 8. С. 133-139.

4. Koryachko V., Perepelkin D., Ivanchikova M. Adaptive rerouting of data flows in distributed data centers. Microprocessors and Microsystems, 2017, Vol. 52, no. 7, pp. 505-509. DOI: 10.1016/j.micpro.2017.05.009.

5. Perepelkin D., Ivanchikova M. Improved adaptive routing algorithm in distributed data centers. Advances in electrical and electronic engineering, 2017, vol. 15, no. 3, pp. 502-506. DOI: 10.15598/aeee.v15i3.2185.

Я.А. Игошев, Р.А. Ершов

ВЫЧИСЛИТЕЛЬНО ЭФФЕКТИВНЫЙ АЛГОРИТМ РАСЧЕТА ВЗАИМНОЙ ФУНКЦИИ НЕОПРЕДЕЛЕННОСТИ В ЗАДАЧЕ ОПРЕДЕЛЕНИЯ ВЗАИМНОЙ ВРЕМЕННОЙ ЗАДЕРЖКИ СИГНАЛОВ ПРИ БОЛЬШИХ ОБЪЕМАХ ВЫБОРОК

(ННГУ им. Н.И. Лобачевского)

Среди решаемых задач современной цифровой обработки сигналов огромное значение имеет проблема обнаружения и позиционирования излучающего объекта в реальном масштабе времени методами пассивной пеленгации. Наиболее широко применяемым методом определения местоположения источников излучения является разностно-дальномерный метод, требующий оценки взаимных временных задержек (ВВЗ) распространения сигнала [1], которые можно получить на основе построения и анализа взаимной функции неопределенности (ВФН) [2, 3]. Ввиду того что вычисление данной функции требует больших вычислительных затрат, могут быть использованы технологии параллельного программирования для реализации новых алгоритмов с целью применения последних в практических задачах. Одной из таких технологий является NVIDIA CUDA [4, 5], которая и рассматривается в настоящей работе при вычислении ВФН.

При определении частотно-временных параметров в системах связи с подвижными объектами, в частности с использованием космического сегмента, необходимо компенсировать искажение (масштабирование) спектра принимаемого сигнала, вызванного влиянием эффекта Доплера. Это можно достичь путем построения ВФН опорного и исследуемого сигналов. Взаимная временная задержка и доплеровский сдвиг оцениваются на основании метода максимального правдоподобия как аргументы данной функции, соответствующие ее максимальному значению.

В работе [6] предлагается вычислительно эффективный алгоритм расчета ВФН, основанный на разбиении принятых и оцифрованных сигналов на

неперекрывающиеся блоки. Функция неопределенности определяется следующим выражением:

$$A(n,m) = \sum_{k=0}^{\frac{N_1}{K}-1} p_{k.k-n} \exp\left(-j\frac{2\pi kKm}{N_1}\right), \quad (1)$$

где m – индекс, соответствующий временной задержке, N_1 – длина сигнала в опорном канале в отсчетах, n – индекс, соответствующий частотному смещению, K – шаг децимации. Значения $p_{k.k-n}$ представляют собой диагональные элементы матрицы $\hat{P}$, полученной в результате перемножения матриц $\hat{S}_1$ и $\hat{S}_2^H$, составленных из опорного s_1 и исследуемого s_2 сигналов [6], соответственно. Чтобы получить отсчеты функции неопределенности, необходимо преобразовать диагонали получившейся матрицы в строки, и затем выполнить преобразование Фурье каждой строки. Все описанные операции могут быть распараллелены и реализованы на графическом процессоре (GPU), что во много раз повышает вычислительную эффективность и дает возможность оценивать местоположение источника радиоизлучения в реальном масштабе времени.

Однако если сигнал представляет собой выборку большого объема, в свою очередь являясь узкополосным, или же имеет широкую спектральную полосу, т.е. оцифровывается с высокой частотой дискретизации, необходимо обрабатывать большое число комплексных отсчетов. Соответственно, увеличивается число элементов матрицы $\hat{P}$ и число отсчетов функции неопределенности. Возрастает объем памяти, необходимый для хранения требуемых матриц на GPU. Как известно, не все GPU способны хранить большие объемы выборок. Поэтому, чтобы оптимизировать работу с памятью, предлагается вычислять функцию неопределенности по независимым друг от друга блокам, размер которых соответствует объему памяти данного GPU. Также желательно, чтобы недиагональные элементы матрицы $\hat{P}$ не хранились в памяти и, по возможности, вообще не вычислялись, поскольку полезной информации при определении временной задержки они не несут.

Для реализации модифицированного алгоритма необходимо изменить операцию перемножения матриц так, чтобы модифицированная функция не вычисляла недиагональные элементы. Можно заметить, что для вычисления -го столбца матрицы $\hat{P}$ в диапазоне возможных временных задержек используется подматрица $\hat{B}$ исследуемого сигнала со сдвигом i и i-й столбец $\vec{a}$ матрицы опорного сигнала (рис. 1).

Математически данная операция выглядит следующим образом:

$$p_{ij} = \sum_{k=0}^{K-1} b_{i+j,k} \cdot a_{ki} = \sum_{k=0}^{K-1} s_1[(i+j)K+k] \cdot s_2^*[kK+j], \quad (2)$$

где p_{ij} – j-й значимый элемент -го столбца $\vec{p}_i$. Таким образом, задача сводится к параллельной реализации множества операций вида $\vec{p} = \hat{B} \cdot \vec{a}$.

Следует сказать, что для решения задач на GPU NVIDIA CUDA использует большое количество параллельно выполняемых потоков (нитей), при этом обычно каждой нити соответствует один элемент вычисляемых данных. Все запущенные на выполнение нити организованы в некоторую иерархию [4, 5]. Верхний уровень такой иерархии называется сеткой (grid) и представляет собой

одномерный, двухмерный или трехмерный массив блоков. Каждый блок в свою очередь является одномерным, двухмерным или трехмерным массивом нитей.

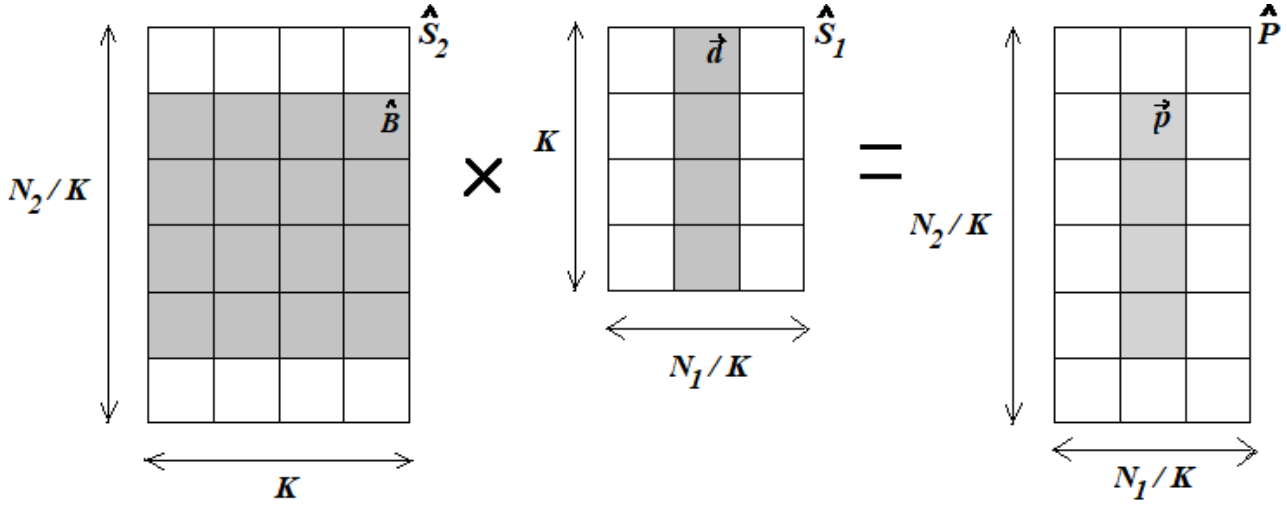

Рис. 1. Вычисление одного столбца матрицы $\hat{P}$

По аналогии с параллельной реализацией матричного перемножения, представленной в [6], можно реализовать ядро CUDA, каждый блок которого будет вычислять несколько элементов вектора $\vec{p}$. При этом подматрица $\hat{B}$ и вектор $\vec{a}$ разбиваются на небольшие фрагменты, соответствующие размеру сетки данного GPU, которые после чтения из глобальной памяти заносятся в разделяемую память (shared memory) GPU, поскольку она обладает значительно меньшей латентностью по сравнению с доступом к глобальной памяти.

В настоящей работе размер сетки равен $h \times N_1/K$ (h - высота вычисляемого блока ФН), а размер блока сетки выбирается исходя из организации разделяемой памяти данного GPU [4, 5]. В конкретном случае размер блока равен 16×16. На вход алгоритма, блок-схема которого представлена на рис. 2, поступают две матрицы $\hat{S}_1$ и $\hat{S}_2^H$.

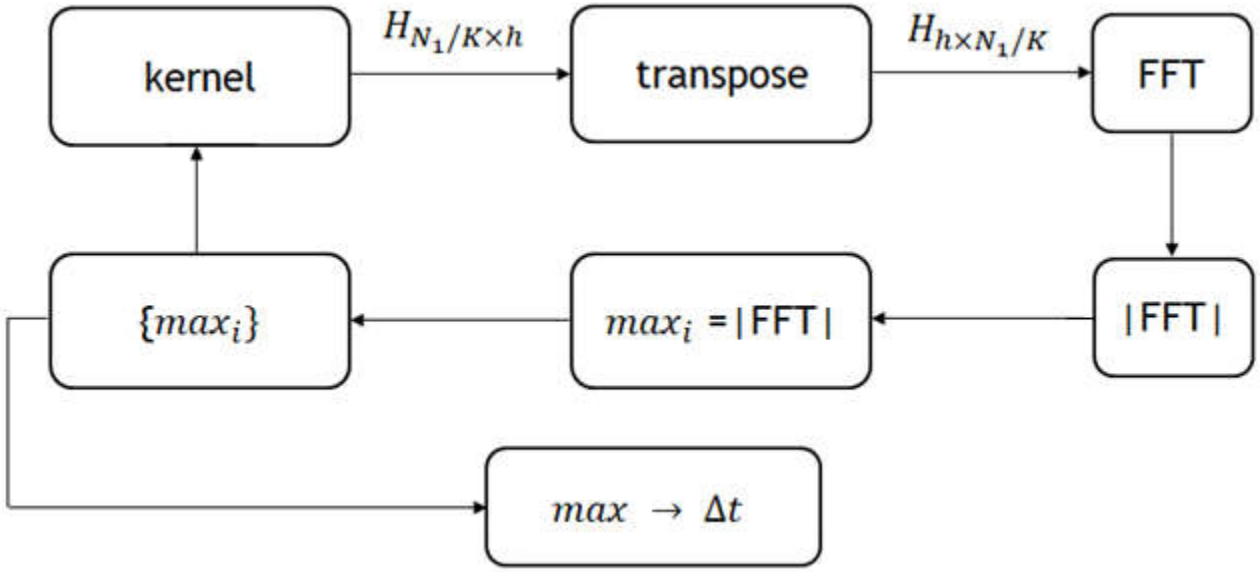

Рис. 2. Блок-схема модифицированного алгоритма

Поскольку диапазон возможных временных задержек равен $\frac{N_2-N_1}{K}$, то матрица $\hat{S}_2$ разбивается на блоки H_j высотой $h + N_1/K$ строк (со сдвигом $j \cdot h$, j – номер блока ВФН), число столбцов остается равным K. Далее, посредством реализованного ядра CUDA, производится перемножение матриц H_j и $\hat{S}_1$. Необходимо отметить, что элементы результирующего вектора $\vec{p}$ записываются в память последовательно (во избежание конфликта банков при обращении к разделяемой памяти GPU [4, 5]), в результате чего на выходе ядра получается блок ВФН размером $N_1/K \times h$, который необходимо транспонировать. Далее, над каждой строкой транспонированной матрицы выполняется преобразование Фурье. Затем вычисляется модуль преобразования, находится максимум и соответствующий ему индекс, которые заносятся в надлежащий массив; происходит переход к следующему блоку вычисления ВФН. Операции повторяются до тех пор, пока не будет вычислена ВФН для всего диапазона временных задержек. Из массива, содержащего максимальные элементы, определяется глобальный максимум ВФН и соответствующий индекс, который будет соответствовать ВВЗ.

Описанный модифицированный алгоритм позволяет применить блочную обработку для определения ВВЗ сигналов большой размерности. ВФН вычисляется по блокам в определенном диапазоне временных сдвигов, который эффективно помещается в глобальной памяти GPU.

В результате моделирования работы алгоритма на GPU NVIDIA GeForce GT 920M была получена зависимость времени вычисления ВФН от числа отсчетов в опорном сигнале $t(N_1)$, которая представлена в таблице ниже.

N_1, отсч.	8192	16384	32768	65536	131072	262144	524288
t, мс.	1134	1202	1412	1954	3524	9664	96615

Видно, что с увеличением N_1 существенно возрастает время расчета ВФН. Причина в том, что на одном из этапов алгоритма происходит операция транспонирования (вычислительная сложность которой, как известно, составляет $O(N^2)$) блока ВФН. К тому же каждый блок ВФН вычислялся на соответствующей ему итерации цикла.

Так, если независимые друг от друга блоки ВФН вычислять на разных GPU, можно в значительной степени повысить скорость расчета ВФН, а соответственно и определение ВВЗ. Поэтому данный алгоритм может быть использован в реальных задачах определения местоположения источника радиоизлучения.

Литература

1. Гришин, Ю.П. Радиотехнические системы. / Ю.П. Гришин, В.П. Ипатов, Ю.М. Казаринов. – М.: Высшая школа, 1990. – 496 с.

2. Johnson, J.J. Implementing the cross ambiguity function and generating geometry-specific signals. – Monterey: Electrical Engineering, 2001, P. 124.

3. Stein, S. Algorithms for ambiguity function processing // IEEE Transaction on acoustics, speech and signal processing. 1981. Vol. ASSP-19, №3. P. 588.

4. Боресков, А.В. Основы работы с технологией CUDA. / А.В. Боресков, А.А. Харламов – М.: ДМК Пресс, 2010. С. 232.

5. NVIDIA Corporation CUDA C PROGRAMMING GUIDE / NVIDIA Corporation, 2017. – 286 p.

6. Ершов, Р.А. Применение технологии NVIDIA CUDA в задаче определения взаимной временной задержки сигналов методом вычисления взаимной функции неопределенности / Р.А. Ершов, Я.А. Игошев // Труды XXII научной конференции по радиофизике, посвящённой 100-летию Нижегородской лаборатории, Н. Новгород: ННГУ, 2018. – с. 361-364.

М. Мирхомитов

ОСНОВЫ СОЗДАНИЯ РАСПРЕДЕЛЕННОЙ ИНФРАСТРУКТУРЫ НА ОСНОВЕ ГРИД ТЕХНОЛОГИИ

(Ферганский филиал Ташкентского университета информационных технологий имени Мухаммада аль-Хорезми)

Данная статья посвящена вопросам создания распределенной инфраструктуры на основе Грид технологии. Рассмотрены основные виды Грид технологии. Архитектура протоколов Грид разделена на уровни. Компоненты каждого уровня могут использовать возможности компонент любого из нижерасположенных уровней.

Создание корпоративных информационных систем со сложной распределенной структурой предполагает постоянный обмен данными между рассредоточенными компьютерными системами на высокой скорости, которая обеспечивается средствами телекоммуникации. Поэтому на современном этапе развития науки об информатизации процессы обработки данных на мощных компьютерных системах не мыслима, рассматривать отдельно, без применения средств телекоммуникации и наоборот.

Бурное развитие информационно-коммуникационной технологии и на её основе распределенных систем привело к появлению новой технологии, получившей название Грид – технологии.

Идейной основой технологии Грид является создание на основе сетевых технологий компьютерной инфраструктуры нового типа, обеспечивающей глобальную интеграцию информационных и вычислительных ресурсов, а также ресурсов памяти.

Грид технологии с помощью специального программного обеспечения промежуточного уровня (между базовым и прикладным программным обеспечением), а также набора стандартизованных служб обеспечивают надежный совместный доступ к географически распределенным информационным и вы-