Повышение эффективности параллельных вычислений при моделировании задач молекулярной динамики на основе технологии CUDA

Стиренко С.Г.; Грубый П.В.; Грибенко Д.В., НТУУ «КПИ»

Задача эффективной реализации параллельных алгоритмов всегда считалась не простой хотя бы потому, что программная реализация должна учитывать архитектурные особенности конкретной вычислительной системы. Оптимизация вычислений с учетом характерных аппаратных особенностей на уровне программных кодов еще более усложняет эту задачу.

Авторами предлагается оптимизированное программное решение g_correlation[1] из пакета GROMACS[2] под архитектуру NVidia CUDA[3].

Основные вычисления в g_correlation составляет алгоритм Краскова[4] для нахождения взаимной информации. Реализация алгоритма выполнена под архитектуру CUDA и состоит из 2-х частей:

1. Выполняется поиск взаимной информации между двумя точками.
2. Выполняется редукция найденных значений взаимной информации для всех пар точек.

1-я часть плохо поддается распараллеливанию, поэтому мы использовали принцип крупнозернистого параллелизма: каждый поток на графическом процессоре выбирает пару точек из памяти и выполняет над ними операции в соответствии с алгоритмом Краскова. Результат помещается в разделяемую память.

2-я часть занимает основную часть вычислений – это, по сути, редукция массива – задача с естественным параллелизмом. Здесь были использованы все доступные методы оптимизации для CUDA: использование разделяемой памяти, минимизация ветвлений, устранение конфликтов при доступе к банкам памяти, объединение запросов к памяти. Это позволило существенно сократить время выполнения алгоритма. В результате тестирования были получены следующие результаты:

Таблица. Время выполнения (мс)

Размерность задачи / Тип устройства	128	256	512	1024	2048	4096
Pentium D 2.8Ghz (1 ядро)	5,44755	18,11402	58,93398	212,52848	850,60590	3857,10374
GF8600GT (32 ядра)	3,01961	5,26315	12,50407	20,46570	59,68219	199,34439
Tesla C2050 (448 ядер)	0,95200	1,22700	2,37500	3,50100	5,91100	13,47200

Как видно из таблицы, ускорение растет прямо пропорционально размерности задачи. Максимальный достигнутый коэффициент ускорения – 286. Т.к. Tesla C2050 имеет 14 мультипроцессоров, каждый из которых может одновременно поддерживать до 1536 потоков – теоретическое насыщение системы будет достигнуто при размерности задачи в 21504. Для увеличения этого порога, g_correlation был распараллелен на N видеокарт при помощи библиотеки MPI. Запуск программы производится таким же образом и раньше, но небходимо указывать количество узлов в соответствии с количеством доступных GPU в системе. При этом, видеоадаптер будет выступать в качестве сопроцессора, беря на себя наиболее трудоемкие, с точки зрения вычислительной сложности, участки кода.

[1] T.M. Cover and J.A. Thomas, Elements of Information Theory (Wiley, New York 1991).
[2] Lindahl E, Hess B, Van der Spoel D. GROMACS 3.0: a package for molecular simulation and trajectory analysis. J Mol Model 2001;7: 306–317; http://www.gromacs.org.
[3] NVIDIA's Compute Unified Device Architecture; http://www.nvidia.com/cuda
[4] Kraskov A, Stogbauer H, Grassberger P. Estimating mutual information. (Julich, Germany February 2, 2008).