Эффективное использование архитектур вида CPU+GPU в библиотеке ttgLib

М.А. Кривов, С.А. Гризан, М.Н. Притула

В настоящее время вычислительная мощность современного персонального компьютера может достигать значений, более свойственных суперкомпьютерам[1]. К сожалению, использование всех возможностей графических ускорителей, обеспечивающих большую часть этой вычислительной мощности, сопряжено с рядом трудностей, замедляющих массовое распространение обычных компьютеров как высокопроизводительных вычислительных систем. Одной из таких проблем является одновременное использование множества графических процессоров, что не только усложняет решаемую задачу, но и заставляет реализовывать методы балансировки вычислительной нагрузки. С другой стороны, ускорение, получаемое при использовании графического процессора, сильно зависит как от алгоритма, так и от исходных данных. В результате этого созданная программа может оказаться даже медленней, чем её аналог для CPU.

В настоящей работе предложен метод, позволяющий частично решить подобные проблемы. Он реализован в библиотеке ttgLib, предоставляющей примитив, с использованием которого достигается эффективное распределение задач по доступным вычислительным устройствам, представленным графическими ускорителями и ядрами центрального процессора. Основная идея заключается в представлении программы в виде графа, каждому ребру которого сопоставляется поток данных, а каждой вершине которого - обработчик данных. Основным отличием данного параллельного примитива от других потоко-ориентированных подходов является возможность задания для конкретного обработчика данных дополнительных версий кода обработки, использующих соответствующие вычислительные устройства. Это, с одной стороны, позволит системе времени выполнения автоматически распределить нагрузку по доступным устройствам, а с другой — позволит для каждой задачи подобрать наиболее оптимальный вычислитель.

Для демонстрации работы данного подхода была решена классическая задача NBody, в которой требуется провести моделирование системы из N тел, каждое из которых обладает ненулевой массой и притягивает другие тела данной системы. Было создано четыре реализации, использующие как библиотеки Intel TBB и NVidia CUDA, так и прототип разрабатываемой библиотеки ttgLib. Все они были протестированы на наборах из 8192 и 32768 тел на двух различных машинах[2]. Как показало тестирование, в зависимости от выбора параметров алгоритма использование гибридной архитектуры вида CPU+GPU позволяет получить ускорение более чем 50% над обычными CPU-ориентированными реализациями.

Список литературы

1. Воеводин В.В., Воеводин Вл. В., Параллельные вычисления, СПб.: БХВ-Петербург, 2002.
2. Адинец А.В., Анализ эффективности задачи N тел на различных вычислительных архитектурах, Параллельные вычислительные технологии (ПаВТ'2009): Труды международной научной конференции (Нижний Новгород, 30 марта – 3 апреля 2009 г.). – Челябинск: Изд. ЮУрГУ, 2009.
3. Электронный ресурс www.nvidia.ru/page/link_cuda.html

1 Например, пиковая производительность кластера «СКИФ-Мономах» равна 4.7 Tflops, в то время как производительность персонального компьютера с двумя графическими ускорителями может достигать 2 Tflops.

2 Конфигурации тестовых систем: 1) Intel Core 2 Quad @ 2.66GHz + NVidia GeForce 9600GT и 2) Intel Atom @ 1.6GHz + NVidia GeForce 9300