Рецензия на выпускную квалификационную работу “Распараллеливание на GPGPU фильтров Forward и Backward-Forward в задаче поиска гомологов генов скрытыми марковскими моделями” Тарасенко Никиты Дмитриевича

Работа является попыткой ускорения части алгоритмов, входящих в пакет программ HMMer по поиску гомологов в цепочках белков и генов с использованием скрытых марковских моделей. Актуальность ускорения поиска последовательностей в цепочках обусловлена постоянным ростом многообразия и базы данных известных последовательностей, что ведет к нелинейному росту времени, требуемому для анализа цепочек.

Автор ставил целью разработать модель оптимизации алгоритмов “Forward” и “Forward-Backward”, широко использующихся в HMMer, а также сделать GPGPU реализацию этих алгоритмов на архитектуре NVIDIA CUDA и сравнить реализацию с существующими аналогами.

Реализация указанных алгоритмов была выполнена автором в существующем проекте CUDAMPF, который призван ускорить работу HMMer путем переноса части алгоритмов на CUDA, но не содержит CUDA версии алгоритмов “Forward” и “Forward-Backward”.

В конце работы представлено сравнение времен работы алгоритмов в CUDAMPF на графическом процессоре, и их вариантов из HMMer на центральном процессоре.

Замечания по данной работе стоит разделить на две части - замечания по оформлению и изложению работы, и на замечания по фактической части работы.

Замечания по изложению:

1. Структура изложения и разбивка на главы выполнены не совсем удачно. В Главу “Литературный обзор” включены подразделы с описанием технологий и алгоритмов, логично было бы их вынести в отдельные главы.
2. Введены обозначения, часть из которых в работе никак не фигурируют и не используются (стр. 7).
3. В главе “Модель оптимизации” отсутствуют, либо неясно изложены мотивы для принимаемых решений. “Необходимо делать векторизацию данных” - почему? Чем поможет “конвертирование данных в более удобный для взаимодействия вид”?
4. Из текста не понятна суть произведенных изменений в алгоритме. Например, в главе “Модель оптимизации”: “Каждая матрица упаковывается в векторы размера в 32 элемента” – не пояснено каким именно образом произведена упаковка и что это должно дать.
5. Непонятное использование терминов. Пример: – “в ядре CUDA выделяется 32 потока”. Ядро CUDA – исполняемая программа, а потоки для исполнения “выделяет” сам процессор.
6. В тексте периодически не хватает законченности мысли в предложениях, что приводит к утверждениям типа: “Однако главным недостатком его [инструмента] является тот [факт], что он реализован для CPU.” и

“Существующие аналоги на GPU дают прирост производительности в среднем в 2,5 раза, что не является пределом”.

В целом создается впечатление, что работа написана торопливо. Мало пояснений к изложенной информации, что влияет на ее восприятие и понимание логики.

Замечания по проделанной работе:

1. В процессе работы автором созданы 2 программы для архитектуры CUDA, в которых последовательно происходит несколько итераций двухступенчатого процесса перемножения матриц, куда входит параллельное суммирование элементов массива (prefix sum). Эти операции составляют основу алгоритмов “Forward” и “Forward-Backward”. Если требуются матричные операции, почему нельзя использовать уже готовые реализации sgemm из оптимизированной библиотеки NVIDIA cuBLAS?

2. Автор путает понятия оптимизации алгоритма и переноса алгоритма на другую архитектуру. Перенос существующего последовательного алгоритма в архитектуру CUDA требует создания схемы распараллеливания по данным и написания собственно кода. Под оптимизацией обычно подразумевается ускорение либо сокращение требуемых ресурсов готового алгоритма на той же архитектуре. При этом в целях работы отдельно заявлены оптимизация работы алгоритмов “Forward” и “Forward-Backward”, и создание параллельной версии алгоритмов для CUDA. В результатах работы про проведенную оптимизацию не говорится ничего, поэтому оптимизацию, видимо, стоит отнести к алгоритмам на CUDA. В таком случае стоило сделать сравнение времени работы оптимизированной версии GPGPU с неоптимизированной GPGPU.

3. Не сделана оценка ни качества получившихся алгоритмов, ни эффективности исполнения программ на графическом процессоре (что важно для оценки GPGPU алгоритмов). Не приведены затраты ресурсов графического процессора. Эти значения можно получить, прогнав программу через профилировщик nvprof из пакета разработки CUDA.

Из положительных моментов стоит отметить два момента. Вместо реализации параллельного суммирования с помощью стандартных инструкций, автор воспользовался советами по оптимизации программ от инженеров NVIDIA и использовал относительно новые инструкции shuffle, что требует более основательного подхода к написанию программ GPGPU.

Выбор в качестве платформы для реализации алгоритмов именно графического процессора вместо других альтернатив (FPGA и прочих) говорит об интересе автора к массивным параллельным вычислениям. Это выбор не самого простого пути, в силу многообразия архитектур графических процессоров, а также ограничений, налагаемых на алгоритмы в массивно-параллельных архитектурах.

В целом объем проделанной работы показался несколько меньше стандартного для выпускной квалификационной работы, однако это субъективное мнение. По совокупности факторов работа заслуживает оценки “хорошо”, а ее автор Тарасенко Никита Дмитриевич – присуждения степени бакалавра.

08.06.2018 г. Бычков А. Б.