Лабораторная работа №3

по курсу

«Программирование графических процессоров»

Выполнил:

Смирнов Александр

Группа 21225

Задание

Реализовать решение уравнение теплопроводности (пятиточечный шаблон) в двумерной области на  равномерных сетках. Использовать программу, размеченную директивами OpenACC,. из предыдущего задания. Для реализации операции редукции использовать вызовы из библиотеки cuBLAS. Сравнить скорость работы с исходной версией программы для разных размеров сеток. Произвести профилирование программы с использованием NsightSystems.

Решение

Исходный код расположен в репозитории <https://github.com/SmirAlex/gpu-programming> в папке lab3. Решение представлено в виде трех файлов: один хэдер файл с объявлением констант, функций и типов данных; один файл с основной логикой вычислений, содержащий работу с cuBLAS, и один файл с логикой чтения аргументов командной строки и проведения измерений времени.

Описание решения

Перед началом решения инициализируется сетка, у которой значения на гранях (начальные и граничные условия) задаются с помощью линейной интерполяции значения в углах сетки (они предопределены заранее).

Для вычисления ошибки создаются 2 версии сетки (под них выделяется память): grid1 и grid2, которые представляют собой состояния сетки соответственно нечетных и четных итерациях алгоритма. Значения в этих сетках предварительно устанавливаются как в начальной сетке. Ошибка для каждой ячейки сетки определяется по пятиточечному шаблону. Цикл прекращается, когда достигнута желаемая точность (минимизирована ошибка) или превышено максимальное число итераций. В конце значения начальной сетки обновляются в соответствии со значениями последней полученной версией сетки.

Описание работы с библиотекой cuBLAS

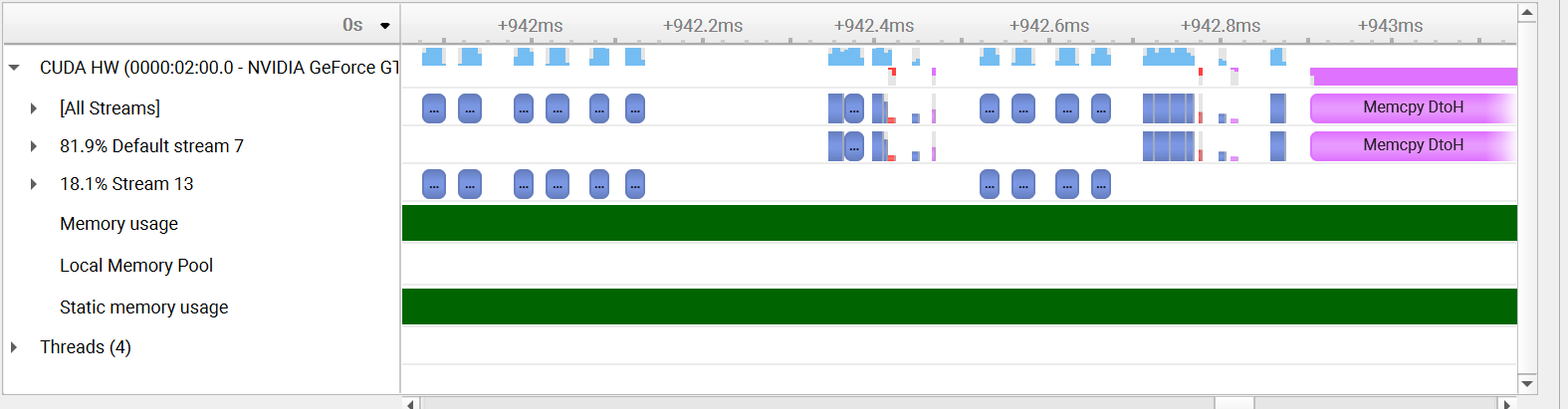
В данной лабораторной работе вычисление ошибки производилось с помощью библиотеки **cuBLAS (**CUDA Basic Linear Algebra Subroutine library). Использовалась вторая (более новая) версия библиотеки, соответствующая хэдеру «cublas\_v2.h». Перед началом работы алгоритма производится инициализация библиотеки с помощью вызова функции **«cublasCreate()»**, соответственно в конце работы все ресурсы, необходимые библиотеке, освобождаются с помощью **«cublasDestroy()»**.

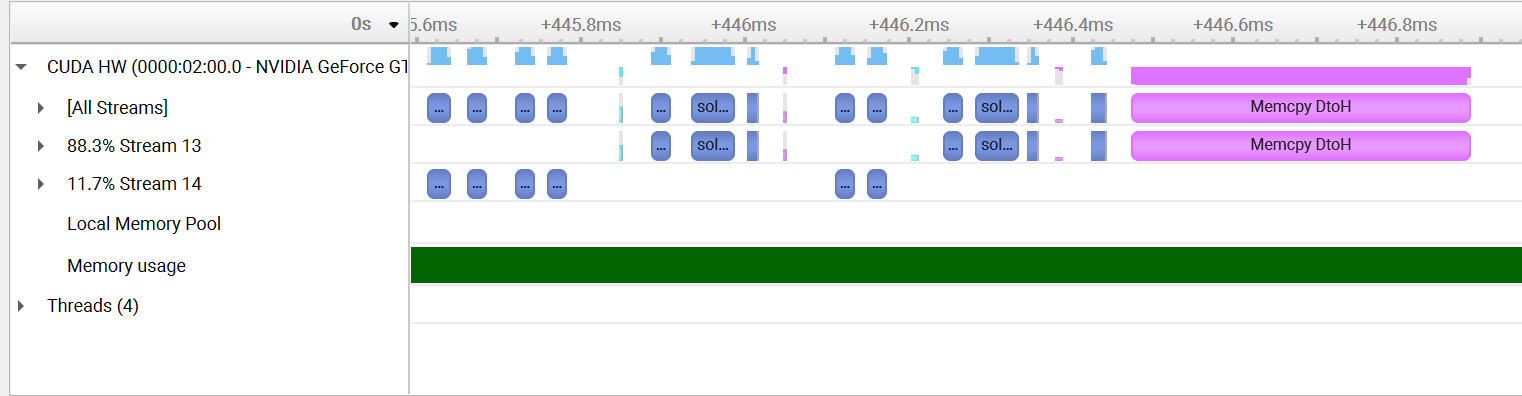
Вычисление ошибки производится в виде 4х последовательных вызовов функций **cublasDcopy, cublasDaxpy, cublasIdamax, cublasDcopy**:

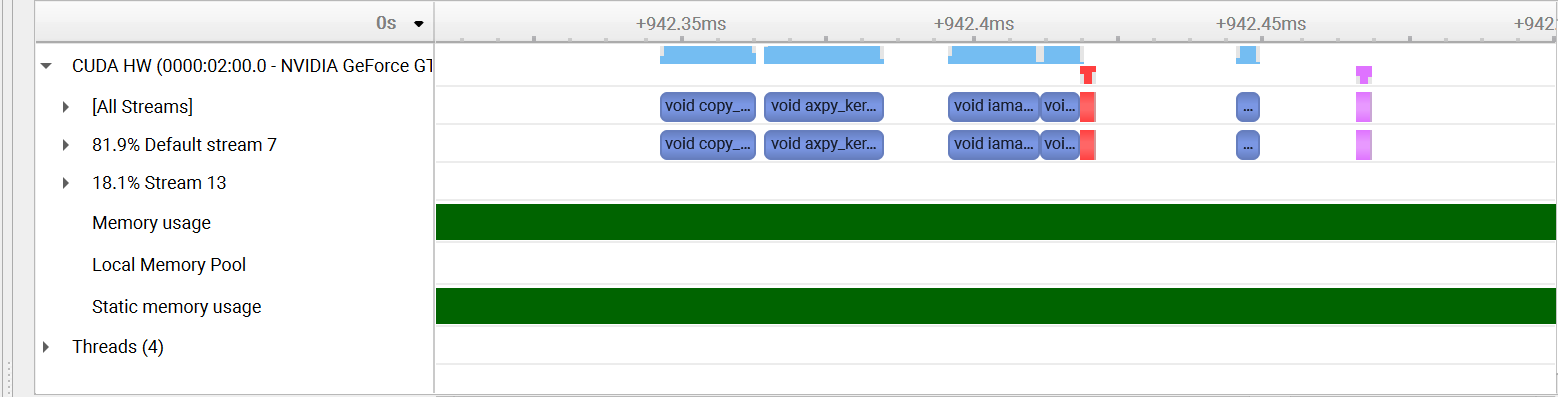
* **cublasDcopy** копирует значение первой сетки (напр. соответствующее текущему состоянию сетки) во временную сетку.
* **cublasDaxpy** записывает во временную сетку разность между второй и временной (первой) сеткой (tmp-grid = tmp-grid – grid2).
* **cublasIdamax** возвращает индекс, под которым во временной сетке лежит максимальное по модулю значение.
* **cublasDcopy** копирует значение ошибки, которая находится по полученному индексу, в соответствующую переменную. Далее ошибка копируется с GPU на хост (с помощью директивы OpenACC **#pragma acc data copyout(error)**, объявленной перед блоком вычисления ошибки).

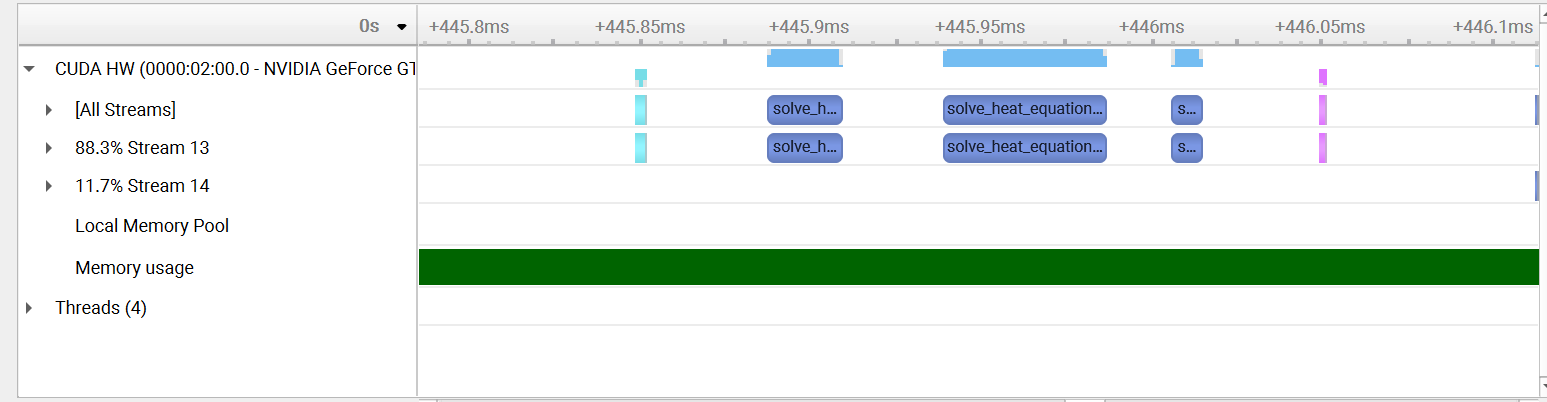
Другого способа оптимизировать вычисление ошибки через cuBLAS найдено не было (напр. вычислять ошибку одновременно с вычислением значений сетки, сразу находить максимум, а не делать это через индекс соответствующего элемента).

Профилирование NsyghtSystems

Версия c cuBLAS

Версия без cuBLAS

Вычисление ошибки с cuBLAS

Вычисление ошибки без cuBLAS (вместе с двумя итерациями алгоритма)

Можно видеть, что в варианте с cuBLAS у нас для вычисления ошибки ядро запускается 5 раз: 2 ядра на **cublasIdamax** и по одному на остальные вызовы функций, а также добавлено копирование (отображено красным).

В варианте же без cuBLAS у нас для вычисления ошибки запускается всего лишь одно дополнительное ядро.

Замеры времени

Сравнивались 2 версии компиляции программы: версия для GPU из предыдущего задания, а также версия, где ошибка измерения измерялась с помощью cuBLAS. Измерения проводились для двух размеров сетки: 256x256, 512x512. Для каждого размера сетки подбиралось такое количество итераций, чтобы алгоритм сходился с точность 10^(-6). Ошибка вычисляется на каждой 100-й итерации.

Исходя из профилирования, ожидаемо, версия с cuBLAS проигрывает в производительности версии с чистым OpenACC. Но с ростом объема данных (сетки), разница сглаживается.