Łukasz Sękalski, 2015-05-31

Sprawozdanie

Techniki Programowania równoległego

Programowanie GPU/CUDA – mnożenie macierzy

1. Cel ćwiczenia

Celem ćwiczenia było przeprowadzenie testów wydajności mnożenia macierzy zaimplementowanego na zajęciach. Do tego celu należało użyć specjalnego profilera dla kart graficznych firmy Nvidia.

2. Konfiguracja

Konfiguracja urządzenia na którym były przeprowadzane eksperymenty to:

- procesor Intel 5960x,
- karta graficzna Nvidia Titan X

3. Realizacja zadania

W ramach zadania zaimplementowano jeden program (matrixMultiplication.cu), realizujący zarówno mnożenie macierzy na CUDA (GPU), jak i na CPU. Poniżej prezentowane są wyniki pomiarów czasów wykonania mnożenia macierzy o wymiarach 16x16.

```
a) GPU Elapsed time: 0.000016
a) CPU Elapsed time: 0.015447
b) GPU Elapsed total time: 1.283000
b) CPU Elapsed total time: 15.512000
```

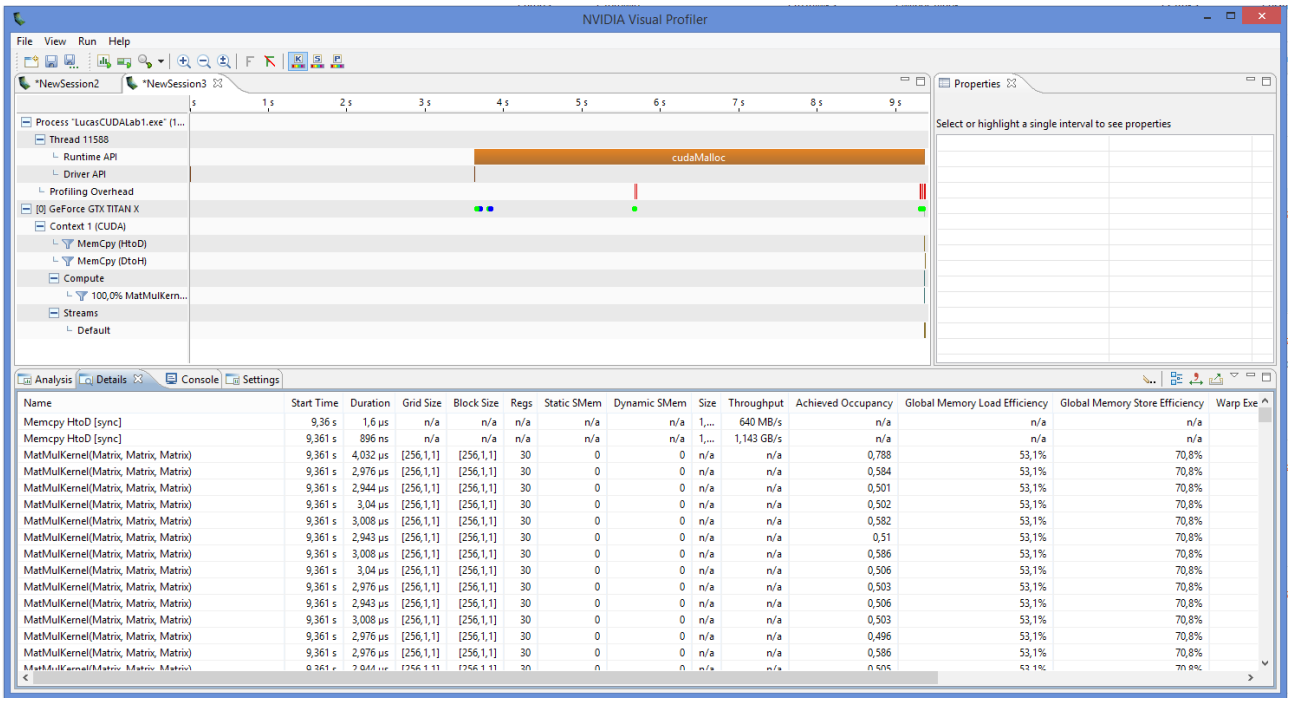
Wyniki podzielić można na 2 grupy:

- grupa a: jest to pomiar czasu jedynie faktycznej operacji mnożenia macierzy. Czas ten został pozyskany przez uśrednienie wyników (mnożenie macierzy zostało wykonane 1000-krotnie, i to łączny czas został podzielony przez 1000).
- grupa b: pomiar 1000 operacji mnożenia wraz z narzutem czasowym na operacje alokowania i przenoszenia pamięci. To właśnie te operacje ograniczają liniowy wzrost prędkości operacji na GPU.

Jak widać, wykonywanie operacji na GPU pozwala na niemalże tysiąckrotne przyspieszenie procesu mnożenia macierzy, jednak gdy uwzględnić narzut czasowy potrzebny na przeniesienie danych, zysk czasowy jest „zaledwie” 15-krotny.

4. Profiler

Użycie profilera dało następujące wyniki:



Podczas jego uruchomienia pojawiły się problemy z profiling'iem napisanej aplikacji, początkowo przypisane wyciekom pamięci. Okazało się jednak, że aby program należycie profile'owął, wystarczy dodać na końcu programu instrukcję *cudaDeviceReset();*.

W ramach poznania profiler'a, zauważyłem, że spełnia on, między innymi, poniższe funkcjonalności:

- Wyznaczanie statystyk GPU (zużycie pamięci, kernela, przepustowość maksymalna czy obciążenie).
- Analiza transferów pamięci.
- Monitoring funkcji CUDA (wywołania, czas trwania, etc.).