VSDB

Protokoll

GPGPU

Dominik Backhausen Alexander Rieppel

17. Januar 2014

5AHITT

Inhaltsverzeichnis

1 Aufgabenstellung

GPU Computing oder GPGPU(= General Purpose Computing on GPUs) bezeichnet die Verwendung eines Grafikprozessors (engl. Graphics Processing Unit oder GPU) für allgemeine Berechnungen im wissenschaftlich-technischen Bereich. Übersetzt bedeutet GPGPU in etwa Allgemeine Berechnung auf Grafikprozessoren.

Informieren Sie sich über die Möglichkeiten der Nutzung von GPUs in normalen Anwendungen. Zeigen Sie dazu im Gegensatz den Vorteil der GPUs in rechenintensiven Implementierungen auf [1Pkt]. Gibt es Entwicklungsumgebungen und in welchen Programmiersprachen kann man diese nutzen [1Pkt]? Können bestehende Programme (C und Java) auf GPUs genutzt werden und was sind dabei die Grundvoraussetzungen dafür [1Pkt]? Gibt es transcompiler und wie kommen diese zum Einsatz [1Pkt]?

Präsentieren Sie an einem praktischen Beispiel den Nutzen dieser Technologie. Wählen Sie zwei rechenintensive Algorithmen (z.B. Faktorisierung) und zeigen Sie in einem Benchmark welche Vorteile der Einsatz der vorhandenen GPU Hardware bringt [12Pkt]! Um auch einen Vergleich auf verschiedenen Platformen zu gewährleisten, bietet sich die Verwendung von OpenCL an.

Diese Aufgabe ist als Gruppenarbeit (2) zu lösen. Es ist zu beachten, dass diese Aufgabe mit der Aufgabe05 gekoppelt ist, d.h. nur eine der beiden Aufgaben wird verpflichtend bewertet! Zusätzliche Abgaben erhöhen die Gesamtpunkte und können somit zur Notenverbesserung dienen.

2 Recherche

2.1 Nutzung von GPUs und Vorteil gegenüber CPU

GPUs können vor allem sehr gut mit parallelisierbaren Algorithmen Arbeiten, da sie weit mehr Kerne besitzen, als eine CPU. Die GPU ist daher besonders für wiederholende und gleichzeitig rechenintensive Aufgaben geeignet und bei diesen auch deutlich schneller. Die GPU zieht erst bei oft wechselnden und ausschließlich sequentiell abzuarbeitenden Aufgaben den Kürzeren.

2.2 Entwicklungsumgebungen

Vor kurzem wurde von AMD die Ati Stream SDK 2.0 veröffentlicht, diese ist nicht nur auf GPUs sondern auch auf CPUs lauffähig (auch von Intel). Neben dem Arbeiten in C, ist es allerdings auch möglich mit Eclipse, in Java zu arbeiten. Man benötigt dazu lediglich, die Bibliothek javaCL.

2.3 Transcompiler

Ja es gibt transcompiler die bespielsweise von OpenCL auf CUDA kompilieren können.

3 Arbeitsaufteilung

Name	Arbeitssegment	Time Estimated	Time Spent
Alexander Rieppel	Dokumentation	1h	1h
Alexander Rieppel	Recherche	1h	1h
Alexander Rieppel	Algorithmensuche	1h	1h
Alexander Rieppel	Kernel	2h	2h
Dominik Backhausen	Kernel	2h	3h
Dominik Backhausen	Benchmarkprogramm	2h	3h
Gesamt		7h	11h

4 Arbeitsdurchführung

5 Testbericht

Nachdem unser Programm fertig geschrieben war haben wir mit den Benchmarks begonnen.

5.0.1 Test 1

Wir haben einen Test auf den Heimrechner von Dominik Backhausen durchgeführt.
In diesem Computer befindet sich folgende HW:
CPU: AMD Phenom(tm) II X6 1100T Processor (6 Kerne, jeweils 3,30 GHz) GPU: AMD Radeon HD 6900 Series

```
Tests run on:
Cayman
AMD Phenom(tm) II X6 1100T Processor

Starting tests
Starting factor test 1 with ArraySize: 10
Loading factor test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 88.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 0.0ms

Starting factor test 2 with ArraySize: 50
Loading factor test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 85.0ms
Starting CPU program!
```

```
Working...
CPU program finished!
Time needed: 3.0ms

Starting factor test 3 with ArraySize: 100
Loading factor test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 102.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 1.0ms

Starting factor test 4 with ArraySize: 500
Loading factor test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 112.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 19.0ms

Starting factor test 5 with ArraySize: 1000
Loading factor test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 83.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 150.0ms

Starting factor test 6 with ArraySize: 5000
```

```
Loading factor test program...
Loading program finished!
This took 1ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 86.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 30.0ms

Starting factor test 7 with ArraySize: 10000
Loading factor test program...
Loading program finished!
This took 1ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 81.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 3.268s (3268ms)

Starting sort test 1 with ArraySize: 10
Loading sort test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 104.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 0.0ms

Starting sort test 2 with ArraySize: 50
Loading sort test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
```

```
GPU program finished!
Time needed: 108.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 0.0ms

Starting sort test 3 with ArraySize: 100
Loading sort test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 99.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 0.0ms

Starting sort test 4 with ArraySize: 500
Loading sort test program...
Loading program finished!
This took 1ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 86.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 4.0ms

Starting sort test 5 with ArraySize: 1000
Loading sort test program...
Loading program finished!
This took 2ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 83.0ms
Starting CPU program!
Working...
CPU program finished!
```

```
Time needed: 1.0ms

Starting sort test 6 with ArraySize: 5000
Loading sort test program...
Loading program finished!
This took 10ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 98.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 25.0ms

Starting sort test 7 with ArraySize: 10000
Loading sort test program...
Loading program finished!
This took 2ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 76.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 104.0ms

All tests finished!
Showing results for factor tests..
Test 1. with arraysize 10: CPU: 0.0ms GPU: 88.0ms
Test 2. with arraysize 50: CPU: 3.0ms GPU: 85.0ms
Test 3. with arraysize 100: CPU: 1.0ms GPU: 102.0ms
Test 4. with arraysize 500: CPU: 19.0ms GPU: 112.0ms
Test 5. with arraysize 1000: CPU: 150.0ms GPU: 83.0ms
Test 6. with arraysize 5000: CPU: 30.0ms GPU: 86.0ms
Test 7. with arraysize 10000: CPU: 3.268s (3268ms) GPU: 81.0ms
Showing Results for Sort Tests..
Test 1. with arraysize 10: CPU: 0.0ms GPU: 104.0ms
Test 2. with arraysize 50: CPU: 0.0ms GPU: 108.0ms
Test 3. with arraysize 100: CPU: 0.0ms GPU: 99.0ms
Test 4. with arraysize 500: CPU: 4.0ms GPU: 86.0ms
Test 5. with arraysize 1000: CPU: 1.0ms GPU: 83.0ms
Test 6. with arraysize 5000: CPU: 25.0ms GPU: 98.0ms
```

```
Test 7. with arraysize 10000: CPU: 104.0ms GPU: 76.0ms
Benchmark finished!
```

Aus diesem Test kann man schließen das der Algorithmus Faktorisieren eindeutig besser Parallelisierbar ist. Sowie das der Wechsel dieses Algorithmus erst einer Anzahl von 1000 Elementen in einer Liste passiert. Nach diesem Wechsel benötigt die GPU nur einwenig länger um den Algorithmus zu beenden, die geschwindichkeit der CPU nimmt allerdings deutlich ab.
Jedoch ist deutlich zu beachten das bei dem anderen Algorithmus bei einer anzahl von 10.000 Elementen die CPU gerade mal gleich schnell ist wie die GPU.

5.0.2 Test 2

Wir haben einen Test auf den Laptop von Dominik Backhausen durchgeführt.
In diesem Computer befindet sich folgende HW:
CPU: Intel(R) Core(TM) i5 (2 Kerne , jeweils 2,50 GHz) GPU: NVIDIA GeForce GT 330M

```
Tests run on:
GeForce GT 330M

Starting tests
Starting factor test 1 with ArraySize: 10
Loading factor test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 51.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 0.0ms

Starting factor test 2 with ArraySize: 50
Loading factor test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 10.0ms
Starting CPU program!
Working...
```

```
CPU program finished!
Time needed: 1.0ms

Starting factor test 3 with ArraySize: 100
Loading factor test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 12.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 9.0ms

Starting factor test 4 with ArraySize: 500
Loading factor test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 10.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 46.0ms

Starting factor test 5 with ArraySize: 1000
Loading factor test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 9.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 33.0ms

Starting factor test 6 with ArraySize: 5000
Loading factor test program...
```

```
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 8.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 3.35s (3350ms)

Starting factor test 7 with ArraySize: 10000
Loading factor test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 11.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 11.66s (11660ms)

Starting sort test 1 with ArraySize: 10
Loading sort test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 8.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 0.0ms

Starting sort test 2 with ArraySize: 50
Loading sort test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
```

```
Time needed: 13.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 0.0ms

Starting sort test 3 with ArraySize: 100
Loading sort test program...
Loading program finished!
This took 1ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 8.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 1.0ms

Starting sort test 4 with ArraySize: 500
Loading sort test program...
Loading program finished!
This took 0ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 8.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 11.0ms

Starting sort test 5 with ArraySize: 1000
Loading sort test program...
Loading program finished!
This took 1ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 8.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 10.0ms
```

```
Starting sort test 6 with ArraySize: 5000
Loading sort test program...
Loading program finished!
This took 6ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 33.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 83.0ms

Starting sort test 7 with ArraySize: 10000
Loading sort test program...
Loading program finished!
This took 4ms
Starting GPU program!
Working...
GPU program finished!
Time needed: 24.0ms
Starting CPU program!
Working...
CPU program finished!
Time needed: 355.0ms

All tests finished!
Showing results for factor tests..
Test 1. with arraysize 10: CPU: 0.0ms GPU: 51.0ms
Test 2. with arraysize 50: CPU: 1.0ms GPU: 10.0ms
Test 3. with arraysize 100: CPU: 9.0ms GPU: 12.0ms
Test 4. with arraysize 500: CPU: 46.0ms GPU: 10.0ms
Test 5. with arraysize 1000: CPU: 33.0ms GPU: 9.0ms
Test 6. with arraysize 5000: CPU: 3.35s (3350ms) GPU: 8.0ms
Test 7. with arraysize 10000: CPU: 11.66s (11660ms) GPU: 11.0ms
Showing Results for Sort Tests..
Test 1. with arraysize 10: CPU: 0.0ms GPU: 8.0ms
Test 2. with arraysize 50: CPU: 0.0ms GPU: 13.0ms
Test 3. with arraysize 100: CPU: 1.0ms GPU: 8.0ms
Test 4. with arraysize 500: CPU: 11.0ms GPU: 8.0ms
Test 5. with arraysize 1000: CPU: 10.0ms GPU: 8.0ms
Test 6. with arraysize 5000: CPU: 83.0ms GPU: 33.0ms
Test 7. with arraysize 10000: CPU: 355.0ms GPU: 24.0ms
```

Benchmark finished!

Hier findet der Wechsel jedoch früher statt. Da ab einer Anzahl von 500 Elementen die CPU auf einmal langsamer ist als die GPU. Bei dem Sort Algorithmus findet ein Wechsel diesmal auch erst bei 5000 Elementen statt.

5.0.3 Tests Vergleich

Die die AMD Radeon HD 6900 eigentlich stärker ist als die NVIDIA GeForce GT 330M und trotzdem die NVIDIA um gut 70ms schneller ist beim verarbeiten dieser Algorithmen kamen wir zu dem Entschluss das entweder der Treiber von der AMD nicht ganz aktuell ist oder das AMD GPU's der Readon Serie nicht besonders gut mit OpenCl kompatible sind.
Des Weiteren sieht man allerdings das speziell bei diesen Algorithmen eine CPU mit 4 Kernen mehr und um 0.7 GHz mehr pro Kern nicht nenenswert bessere Ergebnisse erzielt.

6 Quellen

$http://ht4u.net/news/21412_erste_finale_opencl-entwicklungsumgebung_von_amd_verfuegbar/$

https://code.google.com/p/javacl/