GPGPU Assignment #1

TA: Yu Sheng Lin Instructor: Wei Chao Chen

March 1, 2016

1 Goals

You have to

1. Learn how to implement some parallel algorithms on GPU.
2. Learn how to use Thrust library.
3. Be creative!

in this assignment.

2 Requirements

This assignment is split into 3 parts. Part I and II is closed, and part III is opened.

2.1 Part I: Count the Position in Words (60pts)

In this part you are required to count the position of characters in a word. Listing 1. provides a sample input and output.

```
aaa aa  a aaaaa   aaaaaa (input)
123012001012345000123456 (output)
```

Listing 1: Example: Count the Position in Words.

You may easily come up with an $O(n)$ sequantial algorithm and an $O(nk)$ parallel algorithm, where n is the length of the input and k is the maximum length of a word. However in this assignment k is very large deliberately, so can you come up with an $O(n \ln k)$ algorithm? We provide some hints about the algorithm in a seperate PDF, and you can decide whether to read them liberally.

The input is generated pseudo-randomly, and you can assume that $k = 500$, $n \approx 2 \times 10^7$. Besides, the input only contains characters `[a-z]` and we use linebreak `'\n'` as the spaces.

You have to implement a function whose signature is Listing 2.. All pointers are device pointers and `text_size` is the n.

We also provide some hints not relating to the algorithm itself.

- A kernel may start before previous kernels have finished. To prevent this, insert `cudaDeviceSynchronize()` between them.
- `blockDim.x` cannot exceed 131072 if you don't use `-arch sm_30` compile flag.

```
void CountPosition(const char *text, int *pos, int text_size);
```

Listing 2: Example: Count the Position in Words.

2.2 Part II: Find the Heads (40pts)

Using the result of part I, you can find all heads of the characters, namely the position of 1's. You can only use `thrust::*` functions and CUDA API in this part. To be more precise, `__global__` functions are not allowed. We will give you correct answer of part I, so you can do this part even if you cannot finish part I.

Listing 3. is the sample input and output.

```
123012001012345000123456 (input)
^   ^   ^ ^       ^
0,4,8,10,18              (output)
```

Listing 3: Example: Extract the Heads

Here are some hints:

- Of course you need the document https://thrust.github.io/doc/modules.html
- I have already included some necessary headers.
- I used " " (white text).

The function signature of part II is Listing 4., and `head` is large enough to hold all heads (actually, the same size as `pos`). You should return the number of heads you find, and you should fill ALL HEADS IN-ORDER in `head`. The data after the heads can be left uninitialized.

```
int ExtractHead(const int *pos, int *head, int text_size);
```

Listing 4: Example: Count the Position in Words.

2.3 Part III: Be Creative! (20pts bonus)

With the results you got in previous parts, please do something interesting.

If you had no ideas about what to do, then Listing 5. shows some possible outputs. The first one is the "quite hard" part in assignment #0. I think that it's not that hard now (hopefully). The second one clamps the words which length are longer than 3.

```
abcdeffof abc abcd   abab (input)
badcfeoff bac badc   baba (possible output 1)
abc- abc abc-   aba-      (possible output 2)
```

Listing 5: Sample outputs for part III.

The points you will get is based on your creativity, and you will get 10pts if you implement the examples we provide.

3 Submission

- The submission deadline is 2016/3/24 23:59 (Thur.).
- The efficiency of part I will also be considered when grading, and pure CPU implementation is counted as cheating.
- The efficiency of part II will NOT be considered when grading, but if you call any `__global__` functions, you will get 0pt.
- For part III, [TODO: how to demo?].
- You can only modify `counting.cu`, and we will only copy this file from your repo.
- The compile flags are `--std=c++11 -arch sm_30 -O2`.
- Please also refer to assignment #0 for more details :P.