CS6023 : GPU Programming

Assignment 3 (13 marks)

Submission deadline: Apr 4, 2021, 23:55 on Moodle

Problem Statement:

A highway has '*k*' toll tax zones and each zone has '*m*' toll tax points. On a particular day, '*n*' vehicles are passing through that highway. Each vehicle is taking '*x*' minutes for crossing a toll tax point and they have to pass one toll tax point in each toll tax zone. Only one vehicle can pass through each toll tax point at a time.

Each vehicle has its own speed. For simplicity, let's assume

1. All vehicles have the same entry time.
2. Vehicles are maintaining same speed 'S_i' (km/hr) in-between one zone to the next one.
3. Distance between all toll tax zones (including Start and End points of highway) are same.
4. All variables in Integer.

Note: Start and end points of highway is not a toll tax zone. So, any number of vehicles can start and exit at same time.

Tasks:

1. Find the time taken (in minutes) for crossing the highway for each vehicle.
2. For fun, all the toll tax zones (including exit point) maintain their first crossing vehicle and last crossing vehicle. Give the list of those such vehicles. (In case of conflict, they noted the lower vehicle number for first crossing vehicle and higher vehicle number for last crossing vehicle.)

Input constraints:

- $1 \le n \le 10^5$
- $1 \le k \le 100$
- $1 \le m \le 1000$
- $0 \le x \le 10$
- $50 \le dis \le 100$, distance between consecutive toll tax zones
- $10 \le S_i \le 100$, where $1 \le i \le n$

Format:

Input Format:

- First line contains n, k, m and x respectively.
- Second line contains distance between two consecutive toll tax zones (start and exit points of highway).
- Next k+1 lines contain speed S_i for n vehicles between two consecutive toll tax zones.

Output Format:

- First line contains time taken by each vehicles to pass the highway.
- Next k lines contain first passing vehicle and last passing vehicle for each toll tax zone.
- The last line contains first passing vehicle and last passing vehicle for exit point of highway.

Example:

Sample Input:

```
5 1 3 1

50

40 25 20 50 20

10 25 50 40 50
```

Sample Output:

```
376  241  211  136  211

4 5

4 1
```

Explanation:

Given 5 vehicles, 1 toll tax zone, 3 toll tax points in each zone and 1 minute for crossing each zone.

Distance between start point and the only toll tax point zone is 50km. So, vehicles take 75, 120, 150, 60 and 150 minutes respectively for reaching the toll tax zone.

1 minute is taken for crossing the toll tax zone. (Since there is no conflict between vehicles, there are 3 toll tax point and atmax. only 2 vehicles reached at a time, vehicle number 3 and 5.)

Distance between the toll tax zone and exit point of highway is also 50km. So, time taken in between this is 300, 120, 60, 75 and 60 minutes respectively.

The total time taken by each vehicle is 376, 241, 211, 136 and 211 minutes respectively.

For first toll tax zone, the first passing vehicle is vehicle number 4, which takes 60 minutes and last passing vehicle is vehicle number 3 and 5 (150 minutes), but higher vehicle number is maintained. So, vehicle number 5 is kept for record.

For exit point of highway, the first passing vehicle is number 4 (136 minutes) and last passing on is number 1 (376 minutes).

Points to be noted:

- The provided ‘main.cu’ contains the code, which takes care of file reading, writing and computation of time taken by ‘operations’ function. You need to implement the given ‘operations’ function.
- You can write your own code from scratch. In this case, only operation timing will be considered as execution time, not the file reading and writing time.
- You need to write your own kernels/parallelized code inside the ‘operations’ function. The sequential implementation leads to ‘0’ marks on the assignment. Please keep this in mind.
- Test your code on large input graphs.

Submission Guidelines:

- You can either use the code provided by us or can write the entire code on your own from scratch based on your choice.
- Compress the file ‘main.cu’, which contains the implementation of the above-described functionality to ROLL_NUMBER.tar.gz.
- Submit only ROLL_NUMBER.tar.gz on moodle.
- The name of file should strictly be of the format ROLL_NUMBER.tar.gz.
- For example, if your roll number is CS19M060, the name of the file you submit should be CS19M060.tar.gz.
- Make sure that the ROLL_NUMBER part of the filename is in upper case.
- Do not upload anything other than the ROLL_NUMBER.tar.gz file.
- After submission, download the file and make sure it was the one you intended to submit.

Learning Suggestions:

- Write a CPU-version of code achieving the same functionality. Time the CPU code and GPU code separately for large graphs and compare the performance.

Bonus Points:

- Those submission which pass all the test-cases and have an execution time of less than half of the whole set of submission’s average execution time will get one bonus mark.
- Total time taken by any particular test-case is the sum of time taken by all the kernels in the test-case.
- Total execution time for particular submission is an average of time taken by all the test-cases.