CS6023: GPU Programming

Assignment 4: Game Simulation

Deadline: April 30, 2024

1 Problem Statement

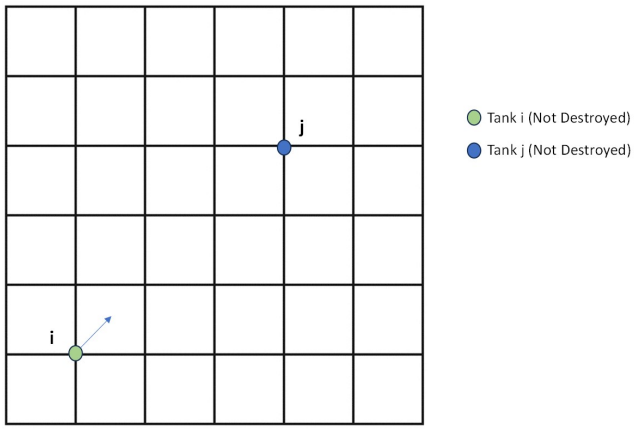
There is a battlefield which can be represented as a grid of size M x N.

There are 'T' no. of tanks placed randomly on the grid with ids (0 to T-1) at integer coordinates (x,y). Consider the tanks to be point sized objects.

Your task is to simulate a game between tanks with following rules and return the score of each tank at the end of the game.

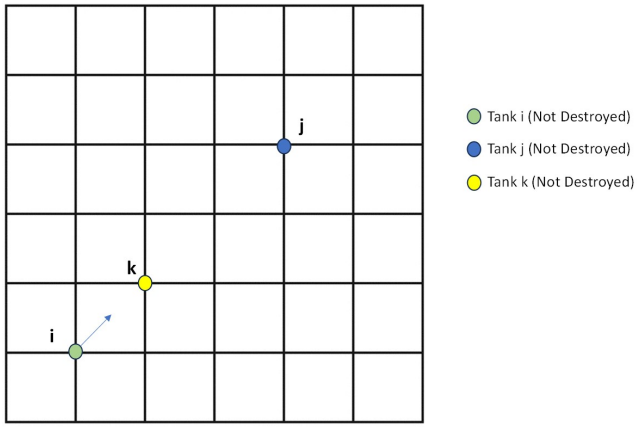
The rules are as follows:

- Each tank has Health Points (HP) allotted to them given in an array. A tank is considered destroyed at the end of a round if its HP dropped to $<= 0$.
- The game consists of multiple rounds, in each round the tanks which are not yet destroyed in the previous rounds fire in the direction of other tanks.
- In kth round the Tank i will fire in the direction of Tank (i+k)%T. Since a tank cannot fire at its own direction you can consider every round which is a multiple of T to be a null round i.e the tanks do not fire at each other.
- Each hit taken by a tank reduces its HP by 1 and increase the score of the tank which fired that shot by 1.
- The way hit detection works is that if in a round Tank i fires in the direction of Tank j the shot will hit the first tank it encounters in that direction which is not yet destroyed. So depending on the layout of the grid there can be following scenarios and their logical variations:
  - The simplest case in which there are no tanks between Tanks i and j and Tank j is not destroyed, so the shot hits Tank j.



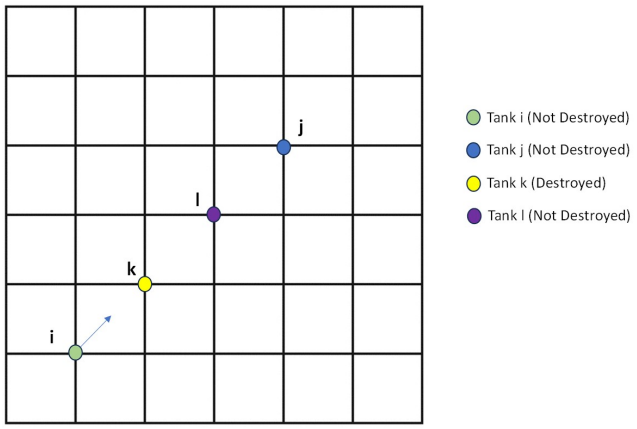
Tank i fires at Tank j and hits Tank j

– There was a tank Tank k between Tank i and j which was not yet destroyed in which case the shot will hit Tank k.



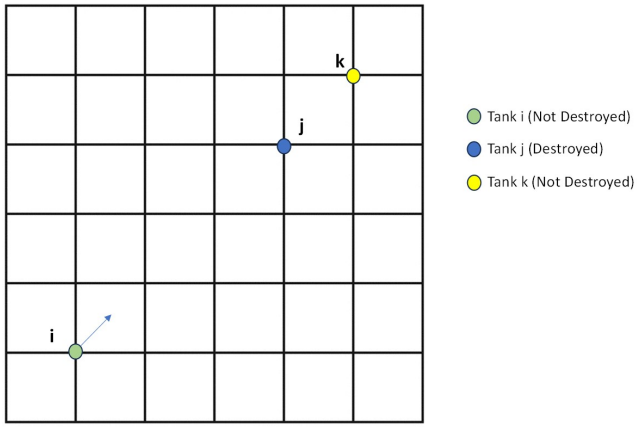
Tank i fires at Tank j and hits Tank k

– There were Tanks k and l between Tanks i and j in that order, Tank k was destroyed in some previous round but Tank l is not yet destroyed in which case the shot hits Tank l.



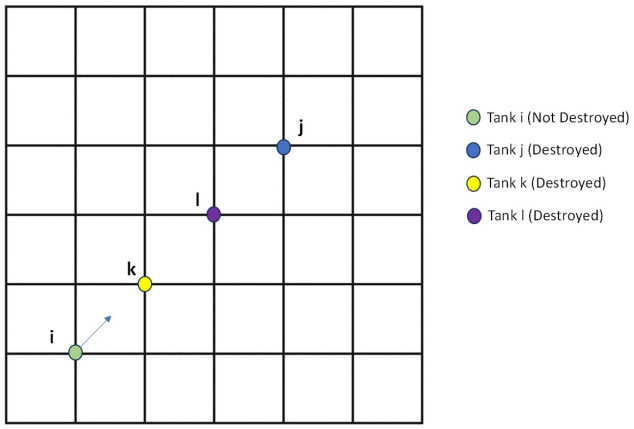
Tank i fires at Tank j and hits Tank l

– There are Tanks i, j & k in the i-j direction and Tank j was destroyed in some previous round but Tank k is still not destroyed in which case the shot will hit Tank k.



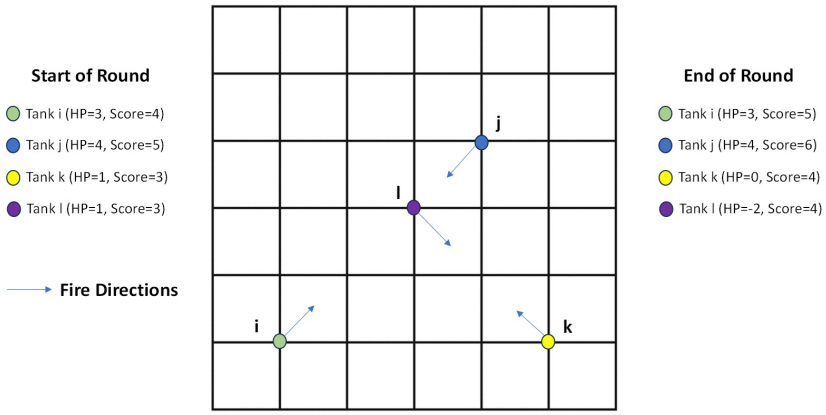
Tank i fires at Tank j and hits Tank k

– All the Tanks in the i-j direction except Tank i were destroyed in previous rounds in which case the shot will not hit any tank.



Tank i fires at Tank j and does not hit anything

- The game finishes when there is only one or no tanks left (i.e not destroyed) on the battlefield.
- A tank is considered destroyed only at the end of round in which its HP drops to <= 0, it will still fire its shot in that round and it will still take hits from all the shots which encounter it in that round and the scores of each of the tanks which fired those shots will increase by 1.



Hint: A line can be represented using equation y=mx + c. i.e. every unique combination of m & c represents a unique line. If you are taking this approach try to think of how you can avoid the use of c in the equation.

2 General Guidelines

- Starter code (main.cu file) is placed inside the submit folder. Edit only the required portion of the main.cu.
- You are allowed to use thrust library as long as your thrust related code execute on GPU i.e. device objects and functions.
- **All the calculations for the game logic have to be done on the GPU. CPU section of the code should only contain memory allocations ,kernel launches and thrust device objects/functions which get executed on GPU.**
- **You are not allowed to use any data types which represent decimals i.e float, double etc. If you feel the need to represent decimals try to store them in the form of fractions.**
- Test cases are to be put in the input folder and their expected outputs in the output folder.
- Run test.sh to compile and test your code. (Use command "bash test.sh" on terminal).
- The main.cu takes three command line arguments, the Input file name ,the Output file name and the Execution Time Logging file name.
- Execution time will be evaluated for this assignment. +3 points will be awarded for execution time lesser than 0.5*average and -4 points will be penalized for execution times greater than 2*average.
- Try to optimize your code as much as possible, the most obvious approaches might lead to long execution times resulting in penalty described above.
- This assignment is an individual effort, and it is expected that all work submitted is your own. Plagiarism, the act of using someone else's work, is a serious academic offence and will not be tolerated. Any instances of plagiarism will result in severe consequences, including receiving a grade of zero for the assignment or even facing academic disciplinary actions.

3 Input and Output Format

3.1 Input Format

- The first line contains integers M N T H where, M & N are grid sizes, T is the no. of tanks on the battlefield and H is the starting health points of each tank (all tanks have same starting HP).

- T lines follow with each line containing Tank coordinates, specifically line i contains 2 integers representing x & y coordinates respectively of Tank (i-1) .

3.2 Output Format

The final score of each Tank should be updated on the 'score' array in the host which will get written to output file such that Line i represents final score of Tank i.

3.3 Constraints

- $2 < M, N < 100000$
- $0 <= x <= M$
- $0 <= y <= N$
- $2 < T <= 1000$
- $1 <= HP <= 1000$
- No two tanks lie on the same point in the grid.

3.4 Sample Input

```
4 4 3 5
1 1
2 2
3 3
```

3.5 Sample Output

```
6
3
6
```

4 Submission Guidelines

- Fill in your code in the *main.cu* file. Rename it as your *ROLLNO.cu*. For example, if your roll number is *CS22M056*, your file should be named as *CS22M056.cu*.
- Zip this file as *ROLLNO.zip*. Submit on Moodle.