COMS3008A Course Project

William Vusumzi Booi(2109688)

Project part 1 : Parallel Conway's game of life

- *There are two parts in the project , I have pushed them to my github repository and made it private , to view it I can send an invite (it has a nice look) but will also submit both projects here .*

Quick description of the serial approach

After compiling the program using ***g++ Conway.cpp -o c*** *you have to run it using* ***./c ,*** *you will then be asked to enter the dimensions of your game(this is for user freedom , UX) , they should be integers separated by space* ***"row col".*** *You will then be asked to enter an initial state , you can manually do it or use the* ***input.txt*** *in the previous folder as it contains few samples with specified dimensions. Finally you will be asked to specify the number of generations you want to be displayed. After the program is done , the output will be in the* ***serial_output.txt*** *textfile that will be generated.*

Parallelization approaches :

For efficiency and freedom I used process 0 to read from stdin .

- *Process 0 broadcasted all the information it took from the user .*
- *The number of iterations that each process have to perform were calculated and also ,where each should start(start index) and stop(stop index) was calculated . The idea is that , different rows are distributed to different processing elements and then computations are done in parallel.*

- *Each processor broadcast all the data it modified to all other processing elements so that all have the same data (in preparation for the next iteration(s)).*
- *Finally , processor with rank 0 take the results to the textfile file* ***parallel_output.txt****.*

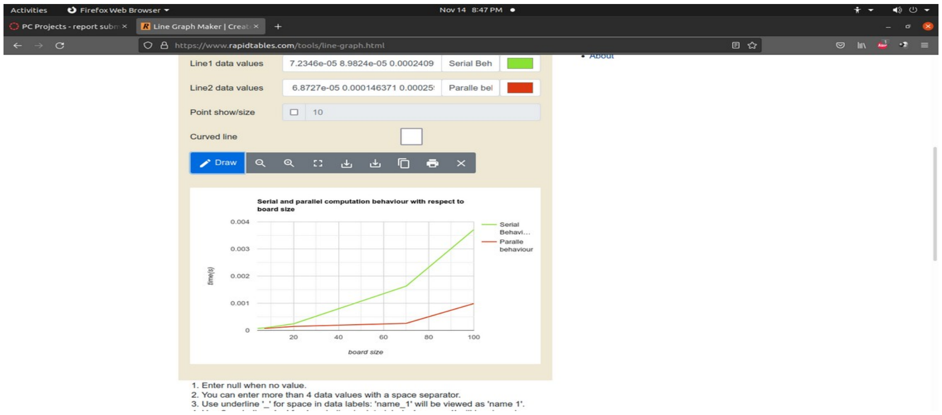
Experiments and performance evaluation :

(1)*The table below shows the parallel and serial computation time when processing elements and generations are fixed in the parallel program (fixed to 2 and 1 , respectively).*

Board size	Serial time(s)	Parallel time(s)
2 x 2	7.2346e-05	6.8727e-05
4 x 3	8.9824e-05	0.000146371
10 x 10	0.000240999	0.000259304
50 x 20	0.00163123	0.000986596
50 x 50	0.00371088	0.00174344

(2)*The table below shows the parallel and serial computation time when processing elements and generations are fixed in the parallel program (fixed to 2 and 10000 , respectively).*

Board size	Serial time(s)	Parallel time(s)
2 x 2	0.130561	0.213993
4 x 3	0.302079	0.58842
10 x 10	1.54333	2.15206
50 x 20	13.0663	10.8431
50 x 50	29.1406	24.2888



Line1 data values 7.2346e-05 8.9824e-05 0.0002409 Serial Beh 🟩

Line2 data values 6.8727e-05 0.000146371 0.00025 Paralle bel 🟥

Point show/size □ 10

Curved line □

1. Enter null when no value.
2. You can enter more than 4 data values with a space separator.
3. Use underline '_' for space in data labels: 'name_1' will be viewed as 'name 1'.

Conclusion

From above , the serial computation is effective to use when the size is small , this is because it won't account time for exchange of information but for large sizes , it is efficient and gives better performance when using parallel program , this is because for large data a large chunk of work is given to each process hence overriding the exchange of data.

Project part 1 : Parallelization of Dijstra's single source shortest path(sssp) algorithm.

There exist only the parallel program for this implementation , I started with the parallelized version so no serial implementation.
Compile and run as the previous program.
The code will be attached.