Implementing the Game of Life in parallel

Conway’s Game of Life

- You implemented a serial version of the Game of Life in a previous programming course
- Reminder of the rules
  - Each cell has eight neighbours (vertical, horizontal and diagonal)
  - If a living cell has fewer than 2 neighbours it dies (not enough to breed)
  - If a living cell has 2 or 3 neighbours it survives
  - If a living cell has 4 or more neighbours it dies (over population)
  - If a dead cell has exactly 3 neighbours a living cell is born there

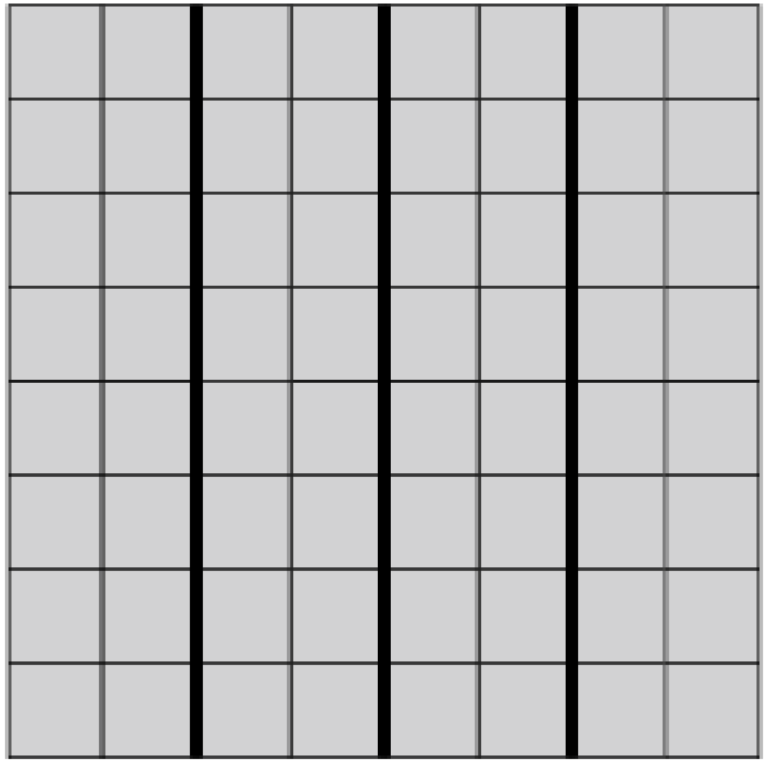
Periodic Simulation

Parallelise the Game of Life
Decomposition strategy

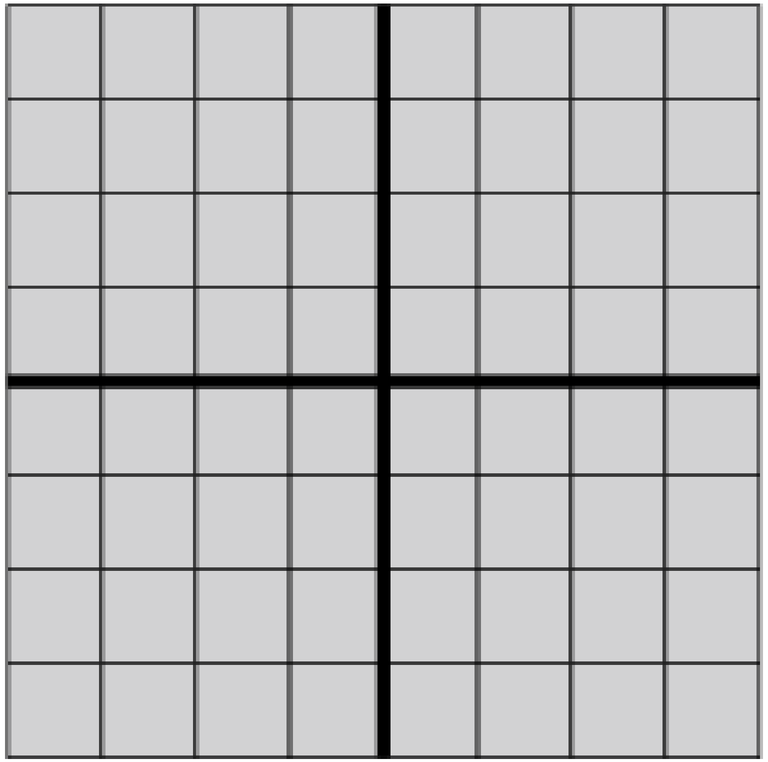
- Best decomposition tactic will be Domain Decomposition
  - Easiest to implement domain decomposition is to divided the domain into vertical or horizontal stripes
    - Only need to communicate with a maximum of two neighbouring processes
    - …,but lots of data needs to be exchanged as the edge length will be large relative to the area of the domain
  - More efficient is to divide the domain into rectangles that are as close to square as possible
    - Less data to exchange, same number of grid cell calculations
- This is exactly the same decomposition strategy as you would use for any other solution on a grid
  - E.g. Finite Difference or Finite Volume on a square grid

Domain Decomposition Strategies

- Striped decomposition



- Grid decomposition



What about the data?

- In order to make best use of the scalability of the problem each process should only store its portion of the grid
  - Don't allocate a grid of the full size of the problem!
  - This makes the problem far more scaleable – If you double the size of the system and the number of processes, the amount of memory required by each process remains the same
- In order for the new value in a cell to be calculated it needs to know all its neighbours
  - A process needs to obtain a layer of "ghost" cells from its neighbours for the points all the way around the edge of the domain
  - Easiest way to achieve this storage is to make the size of the grid 2 points wider and higher than the size of the domain that the process is responsible for

Communication strategy

- The best communication strategy to use will be peer-to-peer
- For the striped decomposition you just communicated with two other process
  - If ordered logically you communicate with the process with an id one higher and one lower than the id of the current process
  - A line of cells at the top and bottom of each region needs to be sent
- For a grid the communication needs to be with all the neighbouring processes
  - Line of cells sent to the vertical and horizontal neighbours
  - Single corner point sent to the diagonal neighbours
  - You might notice that the communication pattern required is exactly the same as that in Worksheet 2 – Exercise 3

Other notes…

- As this is going to be run peer to peer it is more efficient to have each process write an output file rather than transferring the data back to a single process
- You should therefore also write a separate post-processing code to combine these outputs
    - This can be in Python if you wish so that you can make use of the easy ability to do graphical output