Cellular Automata
The Game of Life

The Game of Life is an iterative process set up on a square grid. Cells on the grid are either “alive” or “dead”. If a cell is “dead” and has exactly 3 neighbors, it has enough resources to be born without being overcrowded, and the next turn will be “alive” If a cell is alive and has 3 or 4 neighbors, it has resources without being overcrowded and will stay “alive”. If a cell has 2 or fewer neighbors, it cannot get enough resources to survive and the next turn will be “dead”. If a cell has more than 4 neighbors, it will be overcrowded and the next turn will be dead.

A typical progression might look like:

Turn 1

			X
	X		
	X		
	X		

Turn 2

X	X	X	

Turn 3

	X		
	X		
	X		

These simple rules can lead to many complicated phenomena, some of which seem quite stable, and some of which seem almost chaotic.

Running the Game of Life on a large scale can require a lot of memory. The amount of storage scales as the side length of your grid squared.

This problem is ripe for exploitation by parallel programming. You could break up a larger grid into smaller subgrids. Since each cell only needs information about its nearest neighbors, you only have to communicate among subgrids at the edges of the subgrids.

The MPI Life example is set up to run as “side by side” subgrids. You enter in the number of rows and columns of each subgrid, and the number of iterations to be solved.

Try the following

time mpirun –np 2 Life 100 50 1000 1
#cpus #nrows #ncols #niterations do_display

Compare it to

time mpirun –np 1 Life 100 100 1000 1

Does using more CPUs allow you to solve the same problem faster?

What is the efficiency of this implementation on your cluster?

#cpus	Real time	efficiency
1		XXXXXXXXXXXXXXX
2		
3		

(efficiency can be measured in many ways, but typically can be expressed by taking the running time with 1 processor, and dividing it by the running time with P processors *P)

efficiency = time(1)/(time(P)*P)

What is meant by the efficiency of a parallel solution? If you calculate it once, doe sit apply to any parallel code running on that machine?