Game of Life

From BCCD 2.2

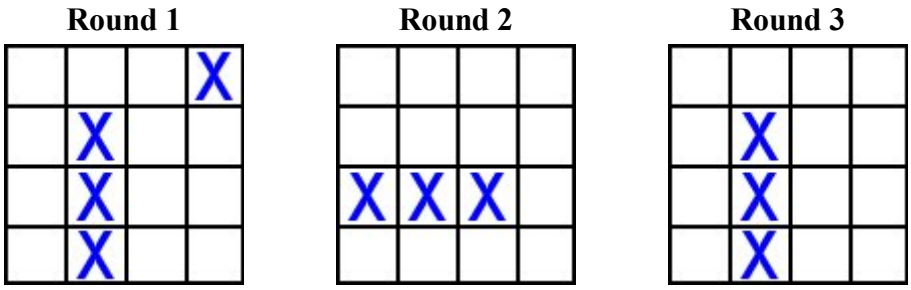
Jump to: navigation, search

This tutorial will walk you through an investigation of parallel computing using the Game of Life. For instructions on how to compile and run MPI programs, please see Compiling and Running. This tutorial assumes that you have booted up the BCCD and have X running (to do this, enter `startx`). It uses mpirun, which means you need an implementation of MPI installed; the BCCD comes with both LAM MPI and MPICH but with MPICH running by default. For more information, see Running MPICH and Running LAM MPI).

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 empty ("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 less than 2 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 this.



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.

To run the program, first move into the Life directory by executing `cd ~/Life`. Next the executable needs to be "made" by running `make`. This will create the executable `Life`.

Next we need to copy this executable to all the nodes that will be running it (If you have not yet set up your

nodes for remote access, make sure you have logged into each machine as bccd, started the heartbeat (pkbcast) program, run `bccd-allowall`, and run `bccd-snarfhosts`. You must do this before continuing.) A new automated script now exists to successfully copy executables across BCCD nodes without compromising other user's runs. It is called 'bccd-syncdir', and for GalaxSee, it is run with the following command:

```
bccd-syncdir ~/Life ~/machines
```

where `~/Life` is the directory which holds the executable, and `~/machines` is the machinefile created previously with 'bccd-snarfhosts', which contains a list of all the nodes in your cluster. This creates a unique directory in /tmp which holds your executable directory across all nodes. The name of this directory is unique and chosen by the first 8 characters of your current host's public key. Make note of this directory and move into with `cd <your directory>`.

Try running Life on just one processor:

```
time mpirun -np 1  -machinefile ~/machines  ./Life   100      100        10000
            #cpus                                   #nrows   #ncols   #niterations
```

Compare it to two processors (notice we've decreased the number of columns by 50 since each processor will only be responsible for half of the total number of columns):

```
time mpirun -np 2 -machinefile ~/machines ./Life 100 50 10000
```

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

What is the efficiency of this implementation on your cluster? (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)
```

Suppose you did the same thing with a 50x50 world? 1000x1000? What is meant by the efficiency of a parallel solution? If you calculate it once, does it apply to any parallel code running on that machine?

Retrieved from "http://bccd.net/ver2_2/wiki/index.php/Game_of_Life"

- Random page
- Help

Search

Go Search

Toolbox

- What links here
- Related changes
- Special pages
- Printable version
- Permanent link

- This page was last modified on 9 June 2009, at 14:59.
- This page has been accessed 3,699 times.
- Privacy policy
- About BCCD 2.2
- Disclaimers