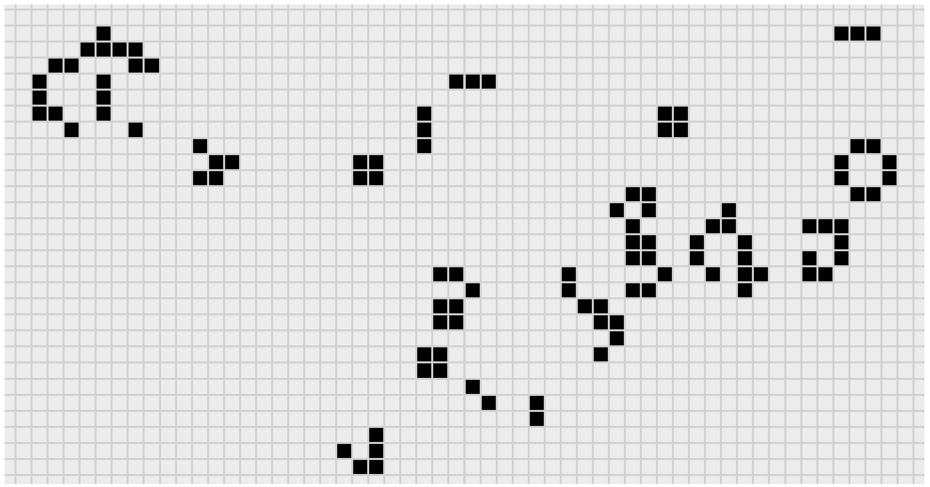
Chaos and Fractals

- Golly - A good cross-platform Game of Life Simulator
- Wikipedia article on Conway's Game of Life
- Wonders of Math - The Game of Life
- Patterns, Programs, and Links for Conway's Game of Life



Conway's Game of Life

The Game of Life (an example of a cellular automaton) is played on an infinite two-dimensional rectangular grid of cells. Each cell can be either alive or dead. The status of each cell changes each turn of the game (also called a generation) depending on the statuses of that cell's 8 neighbors. Neighbors of a cell are cells that touch that cell, either horizontal, vertical, or diagonal from that cell.

The initial pattern is the first generation. The second generation evolves from applying the rules simultaneously to every cell on the game board, i.e. births and deaths happen simultaneously. Afterwards, the rules are iteratively applied to create future generations. For each generation of the game, a cell's status in the next generation is determined by a set of rules. These simple rules are as follows:

- If the cell is alive, then it stays alive if it has either 2 or 3 live neighbors

- If the cell is dead, then it springs to life only in the case that it has 3 live neighbors

There are, of course, as many variations to these rules as there are different combinations of numbers to use for determining when cells live or die. Conway tried many of these different variants before settling on these specific rules. Some of these variations cause the populations to quickly die out, and others expand without limit to fill up the entire universe, or some large portion thereof. The rules above are very close to the boundary between these two regions of rules, and knowing what we know about other chaotic systems, you might expect to find the most complex and interesting patterns at this boundary, where the opposing forces of runaway expansion and death carefully balance each other. Conway carefully examined various rule combinations according to the following three criteria:

- There should be no initial pattern for which there is a simple proof that the population can grow without limit.

- There should be initial patterns that apparently do grow without limit.

- There should be simple initial patterns that grow and change for a considerable period of time before coming to an end in the following possible ways:

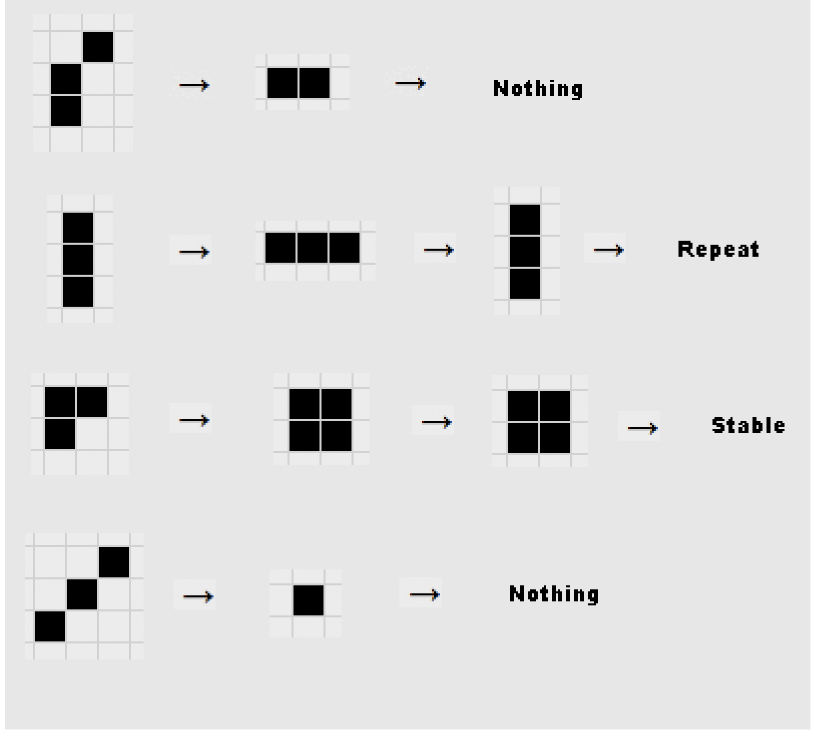
    1. Fading away completely (from overcrowding or from becoming too sparse)

    2. Settling into a stable configuration that remains unchanged thereafter, or entering an oscillating phase in which they repeat an endless cycle of two or more periods.

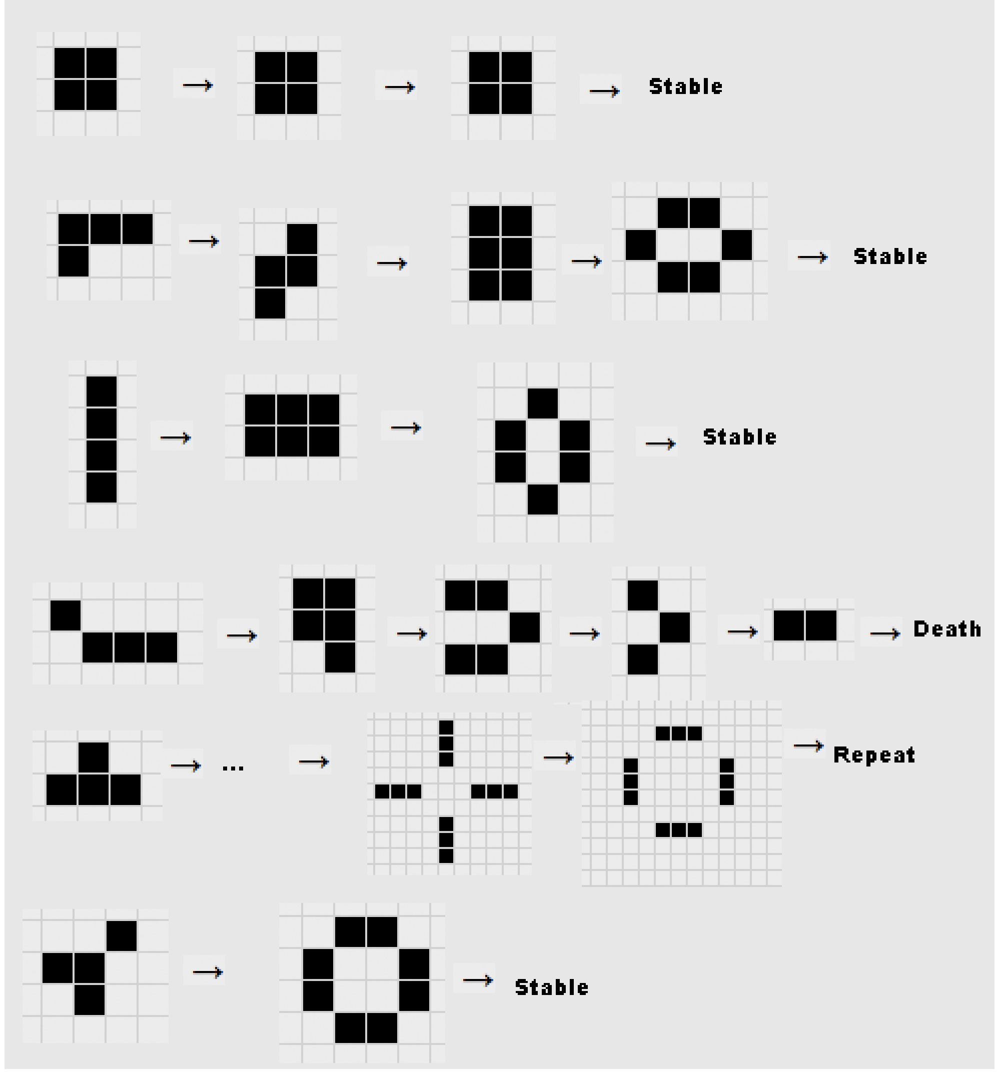
Example Patterns

Using the provided game board(s) and rules as outline above, the students can investigate the evolution of the simplest patterns. They should verify that any single living cell or any pair of living cells will die during the next iteration.

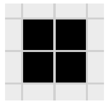
Some possible triomino patterns (and their evolution) to check:

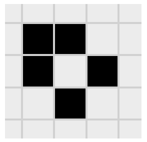


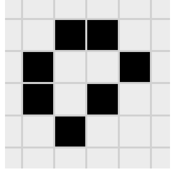
Here are some tetromino patterns (NOTE: The students can do maybe one or two of these on the game board and the rest on the computer):

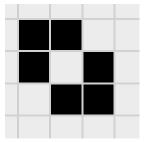


Some example still lifes:

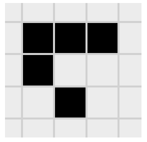
Square : 

Boat : 

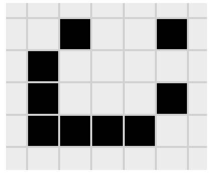
Loaf : 

Ship : 

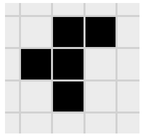
The following pattern is called a "glider." The students should follow its evolution on the game board to see that the pattern repeats every 4 generations, but translated up and to the left one square. A glider will keep on moving forever across the plane.



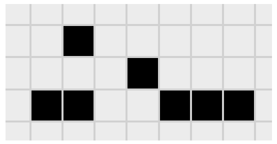
Another pattern similar to the glider is called the "lightweight space ship." It too slowly and steadily moves across the grid.



Early on (without the use of computers), Conway found that the F-pentomino (or R-pentomino) did not evolve into a stable pattern after a few iterations. In fact, it doesn't stabilize until generation 1103.



The F-pentomino stabilizes (meaning future iterations are easy to predict) after 1,103 iterations. The class of patterns which start off small but take a very long time to become periodic and predictable are called Methuselahs. The students should use the computer programs to view the evolution of this pattern and see how/where it becomes stable. The "acorn" is another example of a Methuselah that becomes predictable only after 5206 generations.



Alan Hensel compiled a fairly large list of other common patterns and names for them, available at radicaleye.com/lifepage/picgloss/picgloss.html.

Programs

Life32 is a full-featured and fast Game of Life simulator for Windows. You can download the Life32 program here. There are initial patterns that can be used only with Life32 that you can download here. Another extraordinarily fast program that can be installed on Windows, OS X, and Linux is Golly, which uses hashing for truly amazing speedups. Golly can be found at http://sourceforge.net/project/showfiles.php?group_id=139354. There is a brief description of how Golly achieves such amazing speed here. There are also many Java implementations of The Game that can be run under in most modern web browsers, though they are usually slower. One of these can be found at http://www.ibiblio.org/lifepatterns/. Jason Summers has compiled a very interesting collection of life patterns that can be run with either Life32 or Golly, which can be downloaded here.

If you're using Life32, then after installing, the students should navigate to the directory containing the initial patterns linked to above. In this directory are files with standard Life patterns predefined in them. The following patterns are provided (and the students should run the files in this order): a standard glider, a Queen shuttle bee, a lasting Queen shuttle bee, a Gosper glider gun (first example of a pattern growing indefinitely, won the creator $50), a LWSS (light-weight space ship), a pulsar, and a pentadecathlon. After looking at (and trying to understand) the easier examples, the students can play around with some of the files in this compilation by Jason Summers of popular and look at other interesting patterns. Some of the better files are located in the "applications" and "guns" directories.

If you're using Golly, then another list of initial patterns is prominently located on the left-hand side of the window.

Activity - Two-Player Game of Life

To call Conway's Game of Life a game is to stretch the meaning of the word "game", but there is an fun adaptation that can produce a competitive and strategic activity for multiple players.

The modification made is that now the live cells come in two colors (one associated with each player). When a new cell comes to life, the cell takes on the color of the majority of its neighbors. (Since there must be three neighbors in order for a cell to come to life, there cannot be a tie. There must be a majority)

Players alternate turns. On a player's turn, he or she **must** kill one enemy cell and **must** change one empty cell to a cell of their own color. They are allowed to create a new cell at the location in which they killed an enemy cell.

After a player's turn, the Life cells go through one generation, and the play moves to the next player. There is always exactly one generation of evolution between separate players' actions.

The initial board configuration should be decided beforehand and be symmetric. A player is eliminated when they have no cells remaining of their color.

This variant of life can well be adapted to multiple players. However, with more than two players, it is possible that a newborn cell will have three neighbors belonging to three separate players. In that case, the newborn cell is neutral, and does not belong to anyone.

Computation

It's possible even, to create patterns which emulate logic gates (and, not, or, etc.) and counters. Building up from these, it was proved that the Game of Life is Turing Complete, which means that with a suitable initial pattern, one can do any computation that can be done on any computer. Later, Paul Rendell actually constructed a simple Turing Machine as a proof of concept, which can be found here. Although Rendell's Turing Machine is fairly small, it contains all of the ideas necessary to create larger machines that could actually do meaningful calculations. One of the patterns in Jason Summers' collection will compute prime numbers, and another will compute twin primes (two primes that only differ by adding or subtracting 2).

A very far zoom out of Paul Rendell's Turing Machine: