POLYTECHNIC UNIVERSITY
Department of Computer and Information Science

TWO DIMENSIONAL CELLULAR AUTOMATA

K. Ming Leung

Abstract: Dynamic properties of some two dimensional cellular automata are discussed.

Directory

- Table of Contents
- Begin Article

Last Revision Date: April 9, 2003

Table of Contents

1. Introduction
2. Counter and Growth Cellular Automata
3. Window Automata and Dendritic Growth
4. The Game of Life

1. Introduction

Two dimensional cellular automata have a relatively ancient history, from the self-reproducing automata of von Neumann and Ulam in 1948 to the "Game of Life" of Conway in 1960. These systems have proved to be valuable modeling tools in the fields of crystal growth, physical chemistry and hydrodynamics. Machines designed for specific computations have also been built. They are known as systolic arrays and are used for matrix computations, especially parallel matrix multiplications.

We will not discuss von Neumann's self-reproducing automaton since it is rather complex. It is an automaton that has 29 states, and the builder configuration contains 200,000 automata in the non-quiescent state. We will discuss instead three other two dimensional cellular automata which involve only boolean automata, simple neighborhoods and transition rules. The first one is used in simulating crystal growth, the second one simulates dendritic growth of crystals in solutions, and the last one is Conway's famous "Game of Life".

Common two dimensional lattices include the square, triangular and hexagonal lattices. The last two lattices are some times chosen for their better isotropy properties in the simulation of hydrodynamical systems. In the case of the square lattice, the most popular neighborhoods are the von Neumann neighborhood and the Moore neighborhood. In the von Neumann neighborhood each automaton has 5 inputs, consisting of itself and its 4 nearest neighbors. The Moore neighborhood has 9 inputs, consisting of itself and its 8 nearest neighbors. We will have a chance to use both them here.

2. Counter and Growth Cellular Automata

Suppose we want to grow a larger crystal by exposing a small piece of the crystal in a vapor of the same atoms. We want the atoms from the vapor to condense onto the surface of the crystal, thus building it up layer by layer. There are many controlling parameters in the growth process that influence the deposition rate. The temperature of the crystal which serves as a substrate for the condensation process is a very important controlling parameter.

To simulate such a process, we consider here a counter cellular automata whose transition rule depends only on the sum of the states of the automata in the neighborhood, S and a given fixed threshold value, T. Boolean automata are used. We identify automata's in state 1 as sites where an atom from the vapor sticks on the surface of the crystal. Lattice sites where the automata are in state 0 mark the locations where there is no crystal growth. The threshold T plays the role of the temperature of the crystal. The larger T is , the higher is the temperature. When the temperature is low, atoms in the vapor like to stick to the crystalline surface and stick to each other on the surface. One the other hand when the temperature is high, the atoms dissociate from the surface.

In the simulation, the threshold value is chosen to be either a real number or an integer between -1 and $k+1$, where k is the input connectivity of the automata. For any given automaton, the sum of the states in its neighborhood (including itself) is evaluated and compared to the threshold. The new state of the automaton is set to 1 if this sum is greater than or equal to the threshold, otherwise it is set to 0. Weak thresholds (negative or close to 0) favor the growth of clusters of automata in state 1, whereas strong threshold (close to k) favor the growth of clusters of automata in state 0.

To illustrate this phenomena, we follow the evolution of the dynamical behavior of a counter cellular automaton as the threshold is increased. A square lattice with the von Neumann neighborhood ($k = 5$) and a real threshold are used.

For negative thresholds, since the sum of the states in the neighborhood of any automaton cannot be negative, and so the sum is always larger than the threshold. Therefore in a single time step, the entire population of automata will be in state 1 no matter what the initial configuration is. Crystal growth occurs at an incredible pace.

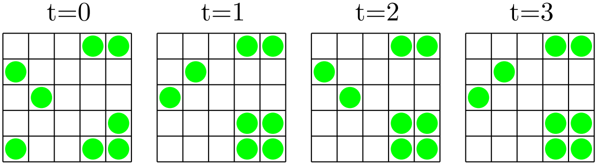
Next consider what happens if the threshold is increased so that it is positive but less than or equal to 1, that is $0 < T \leq 1$. There are only two possibilities, either the sum S, of the states in an automaton's neighborhood, is 0 or $S \geq 1$. If $S = 0$, meaning that the automaton as well as it nearest neighbors are all currently in states 0, then $S < T$, so the automaton will remain in state 0 the next time step. On the other hand, if $S \geq 1$, then $S \geq T$ and so the automaton will be in state 1 if it is not already. An automaton can have $S \geq 1$ because of two reasons. (1) It is in state 1 and at the same time some of its neighbors may also be in state 1, then it will remain in states 1, meaning that an atom already deposited on the crystal surface will remain stuck there. (2) It is in state 0 but one or more of its neighbors are in state 1, in that case it will switch to state 1 the next time step, meaning that a new atom is deposited. These neighboring state 1 automata act like seeds for the deposition of this new atom. Notice that at least one seed must be found in the neighborhood of an empty site in order for an atom to be deposited there.

The most interesting case is $1 < T \leq 2$. First we assume that

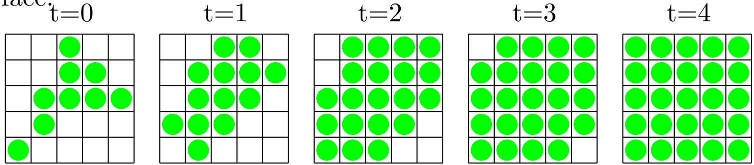
$S = 0$, which implies that $S < T$ and the automaton as well as all its neighbors are in state 0, and so the automata will remain at state 0. Therefore an empty site with no seed in its neighborhood will remain empty. Next we assume that $S = 1$ and so $S < T$ and therefore the state in the next time step is 0. In order that $S = 1$, an empty site must have a single seed in its neighborhood. But an empty site will remain empty. Thus the presence of a single seed is not enough to cause another atom to be deposited near it. Growth cannot occur even at the edge of a line of atoms. If the automaton is in state 1, then it must be an isolated atom (since we assume that $S = 1$). According to the rule, it will switch to state 0. Thus an isolated atom will dissociate from the crystal back into the vapor. Finally we assume that $S \geq 2$, and so $S \geq T$. In this case, the automata will switch to state 1 if not already in state 1. If the automaton is in state 0, then it must have 2 or more of its neighbors in state 1. Therefore crystal grows only at a facet or between 2 atoms. If the automaton is in state 1, then it must have one or more of it neighbors in state 1 also. Thus a cluster of atom will remain at the surface.

The situation is shown in the following figure. Notice that an isolated atom dissociates, a line of atoms is stable, a diagonal pair of atoms oscillates with a period of 2. The triplet evolves in one iteration to a stable square with 4 automata.

Notice that in displaying the states of the automata, instead of putting the automata at the nodes of the two dimensional grid, we can put them inside the squares formed by the grid. The two different ways of presenting the data are totally equivalent since the grid lines serve only as an aide to the eye, and in fact the lines are sometimes not even displayed.


t=0 t=1 t=2 t=3

The two figures depict the deposition of atoms at the facets of two different crystallites when $1 < T \leq 2$. The first surface has very few atoms on the top layer initially and the surface can only be covered partially by atoms (creating an atomically rough surface). The second surface has sufficiently high initial coverage and is eventually completely covered by a layer of atoms forming an atomically smooth surface.



For a threshold between 2 and 3, there is no growth even at a facet, which is instead stripped of its isolated atoms. For even larger thresholds, the roles of the 0's and 1's are reversed, and we observe the growth of the 0's in the opposite order.

For other neighborhoods and other lattices, we see similar behaviors, but not necessarily at the same thresholds. The shapes of the crystal deposited are also different. Instead of square crystals, we will see diamond- or hexagonal-shaped crystals.

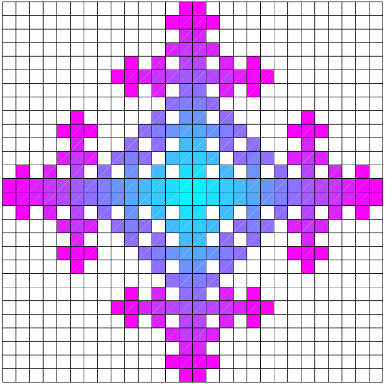
3. Window Automata and Dendritic Growth

A crystal, such as a snowflake, grown from a seed in solution can develop lacy dendritic shapes. This type of crystalline growth occurs when the seed is much colder than the surrounding solution. The heat dissipated by the crystallization process leads to the growth of dendrites which spread out into the solution to find colder zones of the liquid. This type of growth can be modelled by the use of "window" cellular automata. The need to dissipate the heat generated by crystallization is handled by not allowing an automaton to change to state 1 if the number of its neighbors in state 1 is too large. The growth process of course requires the presence of a seed, and so a transition to state 1 cannot occur if the number of neighbor in state 1 is too small. The combination of these two tendencies means that a state 0 automaton can change its state to 1 only if the number of its neighbors in state 1 is neither too large or too small.

Assuming the von Neumann neighborhood, the following transition rule will result in interesting dendritic growth patterns:

1. An automaton in state 1 will always stay in state 1.

2. An automaton in state 0 changes to state 1 only if exactly one of its neighbor is in state 1.



4. The Game of Life

The Game of Life is a rather simple two dimensional cellular automaton which exhibits rather diverse dynamical behaviors depending on the initial configurations. It is an example of a universal computing machine. That is, if we set up an initial configuration of life cells to represent any possible program and any set of input data, run the Game of Life, and in some region of the lattice the output data will appear. The proof of this result involves showing how various configurations of cells represent the components of a computer including wires, storage, and the fundamental components of a CPU - the digital logic gates that perform AND, OR and other logical and arithmetic operations.

The cellular automaton uses a square lattice and a Moore neighborhood with 9 inputs. An automaton is considered alive if it is in state 1, otherwise it is considered dead. The transition rules are as follows:

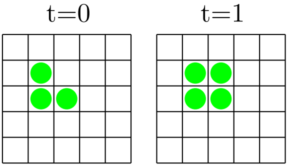
1. An automaton in state 0 switches to state 1 (it comes alive) if 3 of its neighbors are in state 1, otherwise it stays in state 0.

2. An automaton in state 1 remains in state 1 if 2 or 3 of its neighbors are in state 1. Having 2 or 3 of its neighbors alive is supposed to be an appropriate living environment for an automaton, and it remains alive as a result. It switches to state 0 in any other environments. That is an automaton dies if it has 1 or less living neighbor (because of loneliness) or if it has 4 or more living neighbors (because of over-crowding).

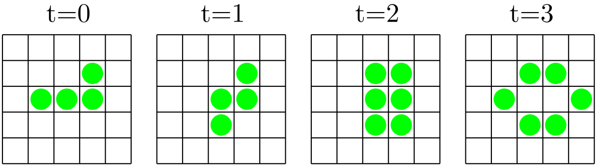
If the state of an automaton at time t is given by a boolean identifier s, and the sum of the states of its nearest neighbors is given by an integer identifier n, then the state at time $t+1$ can be conveniently computed as `(s&&(n==2))||(n==3))` in C or C++.

The following figures show some interesting configurations, assuming that they are isolated from other state 1 automata.

1. The first two configurations evolve rapidly toward simple attractors, called the block and the bee-hive, that are fixed points of the dynamics.

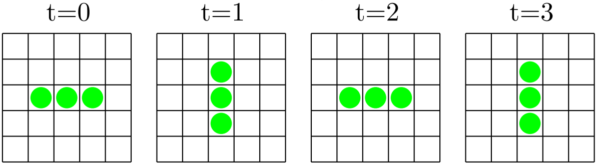


t=0 t=1



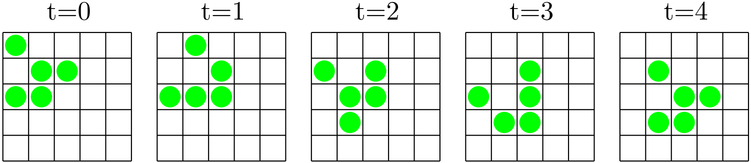
t=0 t=1 t=2 t=3

2. The third configuration gives an attractor configuration called the blinker that is a limit cycle of period 2.



t=0 t=1 t=2 t=3

3. The fourth configuration is very remarkable. After 4 iterations, it returns to its original form except that it has undergone a translation down the diagonal of the lattice. The attractor is called the glider. Clearly there are 4 types of gliders, each propagates either up or down and to the left or to the right, along the two diagonals of the lattice.



The glider is a basic component in the construction of a cellular computer. The Game of Life is actually equivalent to a Turing machine (a Universal computer). To build such a machine, one has to know how to construct 2 logic functions, including negation. The cellular computer is composed of the following components:

1. Gliders which act like binary signals that propagate down the diagonals of the lattice. The presents of a glider is interpreted

as the signal 1 and its absence 0.

2. The glider gun configurations that ejects gliders at regular time intervals. They allow the access to a signal in state 1.
3. The collision of 2 gliders destroys them both.

Even after the article by Martin Gardner about Conway's Game of Life that appeared more than 30 years ago in the Mathematical Games column of the October 1970 issue of Scientific American, the topic still attracts a tremendous interest among groups of non-professionals and professionals world wide. The Game of Life has very rich dynamical properties. New patterns are continuously being discovered. Many new questions are still being asked, and new discoveries are still being made. In particular, Paul Rendell reported recently the creation of a Universal Turing Machine with 3 states and 3 symbols. Any one interested should visit his http://hensel.lifepatterns.net/ *website* for details. In general in order to stay informed of the latest developments on this topic, the web often provides the best source of information. You may want to search related sites using *google* for example.

References

[1] G. Weisbuch, tr. by S. Ryckebysch, *Complex Systems Dynamics: An Introuction to Automata Networks*, Addison-Wesley publishing Company, 1990.

[2] S. Wolfram, Theory and Applications of Cellular Automata, World Scientific, 1986.

[3] Jörg R. Weimar, *Simulation with Cellular Automata*, Logos-Verlag, Berlin, 1998.

[4] H. Gould and J. Tobochnik, *An Introduction to Computer Simulation Methods: Applications to Physical Systems*, Second Edition, Ch. 11, Sec. 6. (Addison Wesley, New York, 1996).