Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Lecture with Computer Exercises:
Modelling and Simulating Social Systems with MATLAB

Project Report

Insert Title Here
...

Name 1 & Name 2

Zurich
May 2008

Agreement for free-download

We hereby agree to make our source code for this project freely available for download from the web pages of the SOMS chair. Furthermore, we assure that all source code is written by ourselves and is not violating any copyright restrictions.

Name 1

Name 2

Contents

1 Abstract **4**

2 Individual contributions **4**

3 Introduction and Motivations **4**
3.1 What is Self-Organized Criticality? 4
3.2 Forest-Fire Model 4
3.3 Power Laws 5
3.4 Key Parameters in SOC 5
3.5 Motivation 6

4 Description of the Model **6**
4.1 Cellular automata 6
4.2 Modelling Approaches 7
4.2.1 Grid Based Updating 7
4.2.2 Random Based Updating 7

5 Implementation **8**

6 Simulation Results and Discussion **8**

7 Summary and Outlook **9**

8 References **9**

1 Abstract

2 Individual contributions

3 Introduction and Motivations

3.1 What is Self-Organized Criticality?

Self-Organized Criticality is the Concept that the dynamics of a physical system converge to a critical point independant of the bestowed boundary conditions. This critical point commonly is not the equilibrium point in a traditional, physical sense. The concept is best understood by example, so we shall bring up a very educative one introduced first by (P.Bak, C.Tang, K.Wiesenfeld, Phys. Rev. A 38, 1988). Imagine a sand pile in 2 dimensions. One can discretisze this pile into cells, which are denoted with the indices n. Each of those cells has an amount of sand grains on it, which is directly corresponding to the height z_n. Now we state the simple rule, that, if the difference in height between two neighboring cells becomes larger than a predefined limit, the higher cell will give grains to the lower cell (avalanche). Now every timestep, we drop a grain on cell one. Independant of the initial conditions, eventually, a straight slope will evolve. The critical point is reached if every difference between two neighboring cells is exactly the limit. What happens if we drop the next grain? Then the difference $z_1 - z_2 > p$ and one grain will drop on z_2. Subsequently, the difference $z_2 - z_3$ will be too big and the process repeats until it hits the last cell. We therefore call a state critical if a yet so small disturbance can propagate throughout the whole system. It is important to note here, that this critical state is *not* the equilibrium state of the system, since that would be a flat surface.

3.2 Forest-Fire Model

A Forest Fire Model is basically a cellular automata. Each cell has three possible states:

1. Empty Site
2. Alive Tree
3. Burning Tree

There are four rules defined for each cell, which are executed simultaneously:

- A burning tree turns into an empty site
- A tree starts to burn if at least one of its neighbors are burning

- A tree will grow on an empty site randomly with probability p
- A tree will burn randomly with probability f

It is important to note that the Forest Fire Model does not only apply to forest fires as its name would suggest. A whole class of problems follows the same basic rules and can be treated accordingly. An example would be the spreading of a disease, where one can directly transform the above rules and states.

3.3 Power Laws

Power laws are functions of the form $P(s) \propto s^{-\tau}$. These functions are found in many data sets, such as the size distribution of cities in a certain area. Take any country, one will find one very large metropolis, a few large cities, many medium sized towns and a huge lot of small towns. Power laws imply that the frequency of a general data point is inversely proportional to its magnitude.

In context to the FFM, it is proposed as a way of quantifying the frequency of Fires with respect to their cluster size.

Power laws have also prominently been found in data sets of earthquakes, where they apply almost perfectly, meaning that small earthquakes happen very often in respect to the frequency of big ones.

3.4 Key Parameters in SOC

To find out wether a basic FFM exhibits SOC behavior, we need to define a set of parameters which serves as indicators for the desired analysis.

- The time needed to burn down a forest cluster: This is heavily dependant on the form of the implementation. If we choose to implement it in a way like [reference needed], we will have an instantaneous fire which essentially takes no time to burn and just resets all the connected trees. If however, we choose the form of a visible fire, meaning that the ratio $f/p \ll 1$ is not 0 in limit and that it takes the fire one timestep to advance a grid cell, then the time needed to burn a forest cluster is an important variable we need to extract from the simulation.

- Number distribution of the size of clusters: Here we first need to define what a cluster is. A cluster is a set of neighboring cells all obtaining the same state. When we talk about forest clusters, we of course mean connected trees. In the implementation where the fire spreads infinitly fast, the forest cluster containing the ignitor cell is the same as the burnt area. The size distribution therefore is a very good indicator for the fire size distribution.

- The mean number of forest clusters in a unit volume $n(s)$, where s is the number of trees in a cluster.
- Number of fires per unit time step: This is a tuning parameter of the model.
- Correlation length.

We have tried to implement the model in a way that the measurment of these desired variables is possible with the least needed effort.

3.5 Motivation

Why is it important to study these effects? As mentioned before, forest fire models apply to a wide range of problems and are therefore helpful in predicting the behaviour of such systems. Self-Organized criticality is the concept that describes the dynamics of such models and the two fields are therefore closely connected. One has closely studied these effects to make better predictions about forest fires. The main problem was, that in most cases, even relatively small fires were extinguished by the authorities. Since that led to an overcritical state, the probability for a huge, devastating fire rose quickly. This has happended in the past and had a serios impact on the ecosystem of those areas. Since the problem of self-organized criticality has been understood, one has stopped to extinguish every little fire, therefore avoiding to reach an overcritical state at which the disturbance (a small fire) can propagate throughout the whole system (large-scale fire). The same ideas can be applied to a variety of problems such as disease spreading or urban planning.

4 Description of the Model

4.1 Cellular automata

A cellular automata is a very powerful way of simulating problems which are defined by a set of rules. It is usually simulated on a grid, but not necessarily restricted to those geometrical constraints. The main characteristics are:

- Every grid point has a state
- Grid points change their state depending on the neighbor states according to a set of rules
- Random actions may be introduced

In the example of the FFM, every grid point has three states (empty, alive and burning) and four rules, the most obvious of which is change state to burning if you are alive and your neighbor is burning.

Cellular Automata can exhibit very complicated behavior when fed with very simple rules, which is the main reason why they are so powerful. There are similarities to agent-based models and networks, but it is not to be mistaken for one of these.

4.2 Modelling Approaches

4.2.1 Grid Based Updating

In a grid based updating implementation, in every timestep, every cell is checked for the rules in the model in the following order:

1. Loop over all Grid points
    - If grid point is empty, grow a tree with probability p
2. Loop over all grid points
    - Check for burning direct neighbors, if yes, ignite current cell.
3. Loop over all grid points
    - If cell is burning, turn it into an empty site.
    - If cell is ignited, turn it into a burning cell.

This is the first grid-based updating implementation we came up with. It needs a workaround for the ignited/burning differentiation, because otherwise, the updating order has a direct impact on the fire propagation. It is also necessary to have multiple loops over the grid cell in order to not have instantaneous fire spreading.

The second approach to grid-based updating is with a second grid that stores temporary information. [Insert Description here]

4.2.2 Random Based Updating

In a random based updating implementation, at the beginning of every time step, one random cell is chosen. It then checks for the rules of the model and acts accordingly. In order to see fires, we need to have instantaneous burning, which is actually permitted by the model. However, it changes the dynamics of the system drastically. But more about that later. The following steps are taken for each timestep:

1. Choose a single cell

2. If cell is empty, grow a tree with probability p
3. If cell is a tree, with probabiltiy f, burn it and the connected cluster

This implementation of the forest fire model has several advantages:

- There are no problems due to updating processes
- Much shorter implementation
- Easier to extract data

The last point is to be explained more extensively: Every time a tree ignites, the whole connected cluster is burned down. Since the whole process happens in a single timestep, we were able to use a recursive function. This function would search every neighboring cell and pass the function on to it if it is alive, just after burning itself. Now with this function structure, we were easily able to retrieve much wanted data such as total function calls (Fire size) or recursion depth (Cluster size). Problems with this implementation will be discussed later.

5 Implementation

6 Simulation Results and Discussion

The first step in the simulation was to see if we could reproduce some of the results found in the references. For starters, we wanted to see if the Formula $\overline{s} \propto \Theta^{-1}$ would hold. This shold be explained: We define the factor $\frac{f}{p}$ as Θ and the mean number of trees burned by one fire as $\overline{s}$. Then the change in the Number of trees present in the system for some large time t is given as

$$\Delta N = p \cdot t - f \cdot \overline{s} \cdot t$$

If we reach a critical point, then this difference should be zero for large times. Then we can write

$$p \cdot t = f \cdot \overline{s} \cdot t \text{ or } p = f \cdot \overline{s}$$

Since we defined $\Theta = f/p$, we can now write

$$\overline{s} \approx \Theta^{-1}$$

What does it tell us if we find this to be true? It tells us, that the system dynamics have a stationary point. However, since we average over quite a large time, the result is not a strong one, but merely a check if the simulations work right. In a first attempt to recreate this result, it did not work. We plotted the curve $\overline{s} \cdot \Theta$ for values of very small to very large Θ (which shold be constant), and observed a small, but relevant slope with quite big differences for small values of Θ. Now there comes a problem with the implementation. Since we only plant a tree with probability p if the site is *empty* and only set a site on fire if the site has an alive tree (and no actions are taken if these requirements are not fulfilled), the above equation has to be rewritten in terms of the probabilities:

$$\Delta N = \frac{(G-N)}{G} \cdot p \cdot t - \frac{N}{G} \cdot f \cdot \overline{s} \cdot t$$

where G is the total Number of Grid Points. For a stationary point, this ΔN again has to be 0, therefore we can write

$$(G-N) \cdot p = N \cdot f \cdot \overline{s}$$

This means that

$$\Theta \cdot \overline{s} = \frac{G-N}{N}$$

should hold. Now $\frac{G-N}{N}$ is indeed a constant, which should not change. However, it is implicitly dependant on Θ, since this has a direct impact on the mean number N of alive trees in the system. Imagine for example a very small f, meaning that fires are very rare. Then this constant factor is very small because N approaches G. For larger values of Θ, N will become smaller and therefore, the factor $\frac{G-N}{N}$ will become larger. That was the result that we obtained from the simulation. It therefore suggests that the simulation is working as expected.

7 Summary and Outlook

8 References