Frontal Cellular Automata for the Study of Non-Equilibrium Lattice Models

Hayk, Poghosyan
Institute for Informatics and Automation Problems,
NAS of Armenia
Yerevan, Armenia
e-mail: haykpoghos@gmail.com

Vahagn, Poghosyan
Institute for Informatics and Automation Problems,
NAS of Armenia
Yerevan, Armenia
e-mail: povahagn@gmail.com

ABSTRACT
This paper provides frontal cellular automata algorithms for abelian sandpile models. Also we calculate the height probabilities on a 2D lattice with fractal boundaries.

Keywords
frontal cellular automata, lattice models, stochastic processes, sandpile model, loop erased random walk, fractals

1. INTRODUCTION
The Abelian sandpile model.[6, 9](ASM) also known as the Bak Tang Wiesenfeld(BTW model originally proposed by Bak at.[1] is a dynamical system with essential properties of self-organized criticality (SOC). The characterization of the SOC state has two aspects, dynamical and structural. The first one, implies a description of avalanches, their duration, mass, linear extent, perimeter, etc. The structural characteristics of the sandpile are fractional numbers of sites having a given height and correlations between heights at different sites in a typical allowed configuration. The model is a cellular automata, which allows using CA algorithms for research. The most natural formulation of a 2D lattice ASM is in terms of discrete height variables, take on four integral values 1 to 4 and are located at the sites of a grid. The configuration of the sandpile, formed by the values of the heights, evolves under a stochastic dynamics, which eventually reaches a stationary regime, where the configurations are weighted according to an invariant measure.
The height properties have been calculated for an infinite lattice, also for upper half plane and various other elementary boundaries[2, 3, 4, 5, 8]. In this paper we take fractals as boundaries taking into account their self repeating nature.
ASM is strongly correlated with the rotor-router model and loop-erased random walks. The researches of the late decades showed the connection between the ASM and diverse phenomena such as: earthquakes, luminosity of stars, river flows, coagulation, relaxation phenomena in magnets, and neural networks. Another interesting feature is its connection to conformal field theory (CFT) models. In particular, for 2D ASM, the associated conformal field theory is suggested to be symplectic fermions with central charge c=-2. Furthermore, some operators in the logarithmic conformal field theories(LCFT) model which correspond to different clusters in ASM are found. LCFT are believed to describe the continuum limit of certain non-equilibrium lattice models, like dense polymers, sandpile models, dimer models and percolation, as well as the infinite series of the lattice models recently defined.

2. CELLULAR AUTOMATA
Cellular automata are consisted of a grid whose nodes are finite automata. Each node has a finite set of states and transition rules which depending on the node, the state of the node, the state of its neighbors and the moment of time change the state of the node. In the general case the neighbors of a node can be chosen arbitrarily. The general CA is a loosely defined model which allows representing a great number of models and systems as CAs. In our case we assume that the CA is frontal cellular automaton. The difference in general CAs and FCAs is in their transition rules and the number of active nodes (if a node acted by a transition rule changes its state it's named active). A general FCAs transition rules do not depend on the time, also the number of active nodes is far smaller then the number of nodes. The latter criterion ensures a much faster speed in comparison to general CAs. The following is the classical algorithm for FCAs relaxation.

Algorithm 1 The relaxation of the asynchronous frontal cellular automata

```
 1: for all x ∈ V do
 2:     if x.isUnstable() then
 3:         stack.push(x)
 4:         bitmap[x] ← 1
 5:     else
 6:         bitmap[x] ← 0
 7:     end if
 8: end for

 9: while stack.containsElement() do
10:     x ← stack.pop()
11:     bitmap[x] ← 0
12:     while x.isUnstable() do
13:         computeNewValue(x)
14:         for all y ∈ adj[x] do
15:             sendMessage(x, y)
16:             if y.isUnstable() ∧ bitmap[y] = 0 then
17:                 stack.push(y)
18:                 bitmap[y] ← 1
19:             end if
20:         end for
21:     end while
22: end while
```

3. ABELIAN SANDPILE MODEL

We denote by $G = (V, E)$ a directed finite graph where V is the set of vertices and E is the set of directed edges. In this model we allow self loops and multiple edges. We denote by $E(v)$ and d_v(out degree) the set of edges whose first or second component is v and the number of edges whose first component is vertex v accordingly. A vertex is a sink if its $d_v = 0$.
We label the vertices of G by $v_1, v_2, ..., v_n$ ($|V| = n$). The Laplacian Δ of G will be an $n \times n$ matrix with the following components:

$$\Delta_{ij} = \begin{cases} -a_{ij} & \text{for } i \neq j \\ d_i - a_{ii} & \text{for } i = j \end{cases} i, j \in 1, 2, ..., n,$$

where a_{ij} is the number of (v_i, v_j) edges.
We associate every vertex with, a non negative integer $h(v)$ which is the number of sand grains on the vertex (if it is a sink, we consider it to be always 0). A configuration or state of sand piles is an one line (or row) matrix of all $h(v)$. We denote it by $h = \{h(v_1), h(v_2), h(v_3), ..., h(v_n)\}$. The sand piles are in a stable configuration if for any $h(v) < d_v$. If for the given vertex v, the height $h(v) > d_v$ then we consider this vertex as active and in that case we can topple it. The new configuration after toppling will be $h' = h - \Delta_v$, where Δ_v is a row associated to the vertex v in the Laplacian matrix. We say there is a path from v_i to v_j if there exists a sequence of $e_i, e_{i+1}, ..., e_j$, where for every non ending edge his second component is equal to the firs component of the next edge. If for every vertex there is a path to a sink then this state can always be toppled to a stable state. We refer such graphs to be graphs with global sink and from here we will deal with such graphs.
We denote by E_{v_i} the sand grain addition operator which acts on the state as follows: $E_{v_i}h$ is equal to the stabilized $h = \{h(v_1), h(v_2), h(v_3), ..., h(v_i+1), ..., h(v_n)\}$. It's proven that in the sandpile models with a global sink the sand grain addition operator is commutative, this property is called the abelian property and a sandpile model with this property abelian sandpile model.
The number of stable configurations is $\prod_i \Delta_{ii}$, $i \in 1, 2, 3..., n$ where the product is over all non-sink vertices i. If a stable configuration can be reached from any configuration by using the operator E_v then we say it's a recurrent configuration[6]. The number of recurrent configurations is $\det \Delta$. The recurrent configurations form a group. A natural question arises: what is the probability of a concrete vertex having a concrete number of sand grains if it's in a recurrent configuration? The most trivial method is to generate all the stable configurations, then separate recurrent configurations and calculate the above mentioned probabilities. Generating stable configurations isn't difficult. If σ is a recurrent configuration, then $(\sigma + \epsilon)^\circ = \sigma$ equation is true, where $\epsilon = (2\delta) - (2\delta)^\circ$. Here the δ configuration is given by $\delta(v) = d_v$, and with this formula we can determine if it is stable or not. For a 100×100 quadratic lattice the number of stable configurations will be 4^{10000} which is far grater than the number of elementary particles in the observed universe (10^{86}). A solution to this dilemma might be a statistical approach where we don't generate all stable configurations but only a randomly chosen subset. The second problem is connected to $(\sigma + \epsilon)^\circ = \sigma$ equation which computes relatively slow and for large lattices is unusable. Fortunately, there is a one to one mapping between recurrent configurations and spanning trees.

4. LOOP ERASED RANDOM WALK AND WILSON'S METHOD

Wilson's method[11] is a way to generate spanning trees on a given graph. We choose this particular method due to uniform generation of the trees, also the resolution is computationally and resource efficient. A walk in $G(V, E)$ is a sequence of vertices $v_0, v_1, ...$ where for any v_i and v_{i+1} there is an (v_i, v_{i+1}) edge. We define a path as a walk where all vertices are distinct. Lawler[10] defined the loop-erasure $LE(v)$ operator as the path obtained by deleting the cycles in chronological order from walk v as follows:

$$LE(v) = (v_{s(0)}, v_{s(1)}, ..., v_{s(J)}),$$

where $s(0) = 0$ and for $j \geq 0$,

$$s(j+1) = 1 + max\{i | v_i = v_{s(j)}\}$$

A loop erased random walk is obtained by using the operation LE on a random walk.
The following is Wilson's method. Let $G = (V, E)$ be a finite connected graph. Pick up any vertex $r \in V$, and name it as "root". We define a growing sequence of subtrees $T(i), i \geq 0$. We let $T(0) := \{r\}$ and we let $< v_1, v_2, ..., v_{n-1}, r >$ be an enumeration of V. Suppose $T(i)$ has been generated. Start a LORW at v_{i+1} and stop when it hits $T(i)$

$$T(i+1) := T(i) \cup \text{LE walk from } v_{i+1} \text{ to} T(i)).$$

5. HEIGHT PROBABILITIES OF A FRACTAL BOUNDED LATTICE

We use as boundaries two kinds of fractals. These are created by acting on a square graphs vertices with fractal operations. The deforming operation which creates fractals we name fractal operation. The fractal operations we used are illustrated in the following figure: We

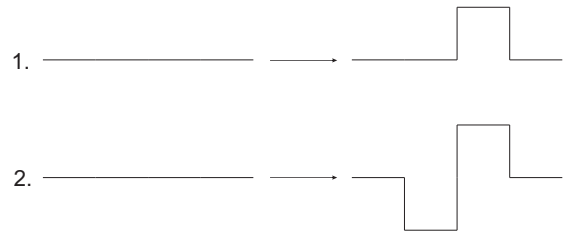

Figure 1: Illustration of the two fractal operations.

used four lattices, two of 500×500 and two of 150×150 with $4, 3$ fractal operations on a square as boundaries, respectively. We denote the number of adjacent vertices of v whose unique path to the root passes through v by $pred_v$, and name it the number of predecessors of v. Its proven that after a sufficient times of experiments the probability for vertex v to have i ($i \in \{0 - 3\}$) number of predecessors is the same as the probability p_v of having i number of sand grains. Since the number of predecessors depends on only neighboring vertices then

if we are interested in only vertex v probabilities we can generate only the subtrees which have the vertex v and its neighbors. According to Wilson's method, the sub tree T for vertex v can be generated in the following process: $T(1) \leftarrow v$
$T(2) \leftarrow$ the upper adjacent vertex of v
$T(3) \leftarrow$ the left adjacent vertex of v
$T(4) \leftarrow$ the lower adjacent vertex of v
$T(5) \leftarrow$ the right adjacent vertex of v
Considering the fact that vertices whose distance is more than 4 or 5 have constant probabilities, we calculated the near border vertices with fare more accuracy (1.6 billion times) in comparison to further away vertices (100 million times).

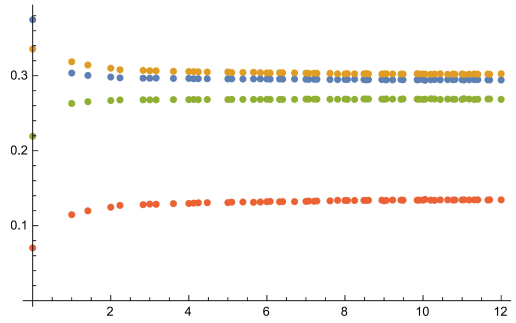

Figure 2: Illustration of the dependence of probabilities p_i from the distance from boundary. Here blue=0, orange=1, green=2, red=3 is the number of predecessors.

6. CONCLUSION

As was expected the further away vertices don't depend on the distance from the border, therefore have constant probabilities. The state is different for near border vertices which have a logarithmic dependance on the distance from the border. We try to find a dependence between this logarithmic expression and the dimension of fractal boundaries. Despite their complicated shapes, the fractals have a very weak effect on the probabilities. The results for all four lattices are almost the same and furthermore we experimented with other fractals as boundaries and the results indicate the same. We think that considering the almost invisible effect of 2D fractals numerous experiments are needed to evaluate the height probabilities. The latter is a considerable difficulty taking into account that the size of the lattice grows with the dimension of fractals.

7. ACKNOWLEDGEMENT

The authors are grateful to Prof. Yu.H. Shoukourian for important discussions and critical remarks on all stages of the work. This work was supported by the State Committee of Science MES RA, in frame of the research project No. SCS 13-1B170.

REFERENCES

[1] Bak P., Tang C. and Wiesenfeld K., *Phys. Rev. Lett.*, 59(4):381384, 1987.

[2] Poghosyan V. S., Priezzhev V. B., Ruelle P., *Phys. Rev. E*, 77 041130, 2008.

[3] Priezzhev V. B., Ruelle P., *Phys. Rev. E*, 77 061126, 2008.

[4] Poghosyan V. S., Grigorev S. Y., Priezzhev V. B., Ruelle P., *J. Stat. Mech.*, P07025, 2010.

[5] Poghosyan V. S., Priezzhev V. B., *Acta Polytechn.*, 51 59, 2011.

[6] Dhar D., *Phys. Rev. Lett.*, 64 1613, 1990.

[7] Bak P., *Oxford: Oxford University Press*, 1997.

[8] Priezzhev V. B., *J. Stat. Phys.*, 74 955, 1994.

[9] Mahieu S., Ruelle P., *Phys. Rev. E*, 64 066130, 201.

[10] G. F. Lawler, *Duke Math. J.*, 47(3):655-693, 1980.

[11] Wilson D. B., *Proc. 28th Annual ACM Symp. on the Theory of Computing*, p 296, 1996.