VISUAL REPRESENTATION OF ENHANCED SAND PILE MODELS

Marta Pla-Castells
Ignacio García
Rafael J. Martínez
Instituto de Robótica
Universidad de Valencia
P.O. Box 2085
46071 Valencia, Spain
E-mail: [Marta.Pla;Ignacio.Garcia;Rafael.Martinez]@uv.es

Keywords Cellular automaton, simulation, sand pile model.

Abstract

This paper overviews some models based on the Chip Firing Game and propose a modification of the Abelian Sand Pile Model in a lattice in order to implement a realistic simulation of a pile of sand falling and finding an stable configuration. We have modified the update rule in the way that now is based on the comparison among neighbour nodes. We found that the simulations can be suitable for doing them in real time and propose as a future work a better solution to reduce the computational cost of the algorithms.

Introduction

Nowadays, simulation in real-time is applied in many fields concerning interactive games or in those activities related with instructional designs. In these cases, not only is essential to have dynamic models with good accuracy but also it has to be taken into account the visualization techniques used in order to obtain a good degree of realism.

Usually, when dealing with simple rigid objects, like cars, trains or containers, the main efforts of the designers are addressed to obtain dynamic models with good interaction with the virtual world. This is an easy task and forces can be calculated very well in real-time. By the other hand when complex objects or deformable bodies are being simulated, like the case of hair movement, cloth behaviour, water or fire simulation, modelling tasks are not so easy as before but however, their visualization properties have been deeply studied by many researchers. There are still another kind of elements whose properties and behaviour require to pay attention at the same time, both to its visualization properties and to its interactions with environment. This is the case, by example, of multiple particle systems subjected to human manipulation, like in the case of a terrain movement machine simulation or in a coal mobil crane simulation. In such systems, accuracy collision detection algorithms, complex models and enhanced visualization techniques have to cope with real time simplifications in order to satisfy the effects of the user actions over the simulation equipment.

When modelling the dynamics of a mountain of grain and its topplings, a good election, at least from the point of view of its theoretical properties and unestabilities, could be to use the sand pile model, since it has been deeply studied for many years (Bak et al., 1988). However, as it will be shown later, these classical algorithms exhibit a pour realism when the evolutions of the sand pile are being displayed in real-time using 3D visualization techniques.

This paper describes a new algorithm, which is a modification of the sand pile model, that represents very well the visual properties of such kind of piles when evolve until stability. In the first section an introduction to the Self-organized Criticality (SOC) is shown. Following, it relations with other models used to represent sandpiles, like the Chip firing game (CFG) and its variants, the Abelian Sand pile model in a lattice (ASPM) and the Sand pile model, are studied, and the results obtained from the computational implementation of ASPM in order to represent a mountain of grain are also shown. In second section we propose some modifications to the original models in order to achieve better accuracy taking into account that we are interested in its visual representation. Finally, third section shows some ideas for future work that will allow reduce the temporal cost of the algorithm at every step.

Self-organized Criticality

The concept of Self-Organized Criticality (SOC) is used to explain a particular behaviour that is observed in many natural systems and processes. These systems are generally characterized by the existence of a power law that rules their dynamics. The term self-organization denotes that in many critical systems tend to appear some structure or pattern at the macroscopical level. Examples of these kind of systems can be a sand pile, the frequency and magnitude of extinctions that take place in every ecosystem, the distribution of star clusters along the outer space, or the expansion process of Internet nodes. SOC related models have also been applied successfully in computational economy and social studies, like the work done by Scott Moss dealing with social simulation (Moss, 2001).

The Sand-Pile Model (SPM) was introduced by Bak, Tang and Wiesenfeld (Bak et al., 1988) in order to represent a self-organized critical system by means of a cellular automaton. This model received its name because of the analogies observed among the dynamical rules of the model and the way sand topples in a real life sandpile. Other studies, like the one done by Brylawski (Brylawski, 1973), use the properties of a one dimensional SPM to study integer partitions. SPM models are also useful to represent the information paths that take place in parallel architectures and distributed computing. In this sense, good performance is obtained when using the gradian properties of SPM to implement a dynamic load balancing algorithm (Subramanian and Scherson, 1994). In such case, the grains of the piles represent data or tasks, and the stacks formed by the grains, the set of available processors.

All the algorithms viewed before adjust the parameters of SPM to resolve a complex problem. In our contribution we want to take benefit of some of the models appeared during the study of this subject not for resolve a mathematical problem, but for testing its suitability to make a realistic visual representation of sand piles.

In next paragraphs we will review some of the different sandpile models appeared in literature during the last years that will be the basis of our algorithm.

Chip Firing Game

The Chip Firing Game (CFG) (Björner and Lovász, 1992) can be defined as a directed graph whose vertexes, v, have two parameters associated, a threshold, δ_v, and a load, $l(v)$, that intuitively represents the number of chips stored in v. A general rule is applied to decide the new load of the vertexes at every step. An SPM is a particular case of CFG.

In the model presented by (Björner and Lovász, 1992) the game evolves with respect to the following rule: if a vertex v contains more than δ_v chips, then δ_v of them are fired to its neighbours , i.e. the load of the vertex v is decreased by δ_v and the load of each of its neighbours is increased by δ_v/n_v, being n_v the number of neighbours of v. Usually the value elected for δ_v is equal to n_v.

Abelian Sandpile Model

Another particular case of the CFG is the abelian sand pile model, which is just an SPM over a rectangular grid (Dhar, 1990). In this case, the model is an undirected graph that forms a rectangular finite grid. The value of δ_v and n_v is always 4 for every vertex, except for an additional vertex, which receives one link from the vertexes located at the border and two links from the four corner vertexes. This special vertex also has an infinite threshold δ_v that makes it act like a sink; it collects the grains that leave the system and it never gives any of its grains.

All the models derived from the Chip Firing Game exhibit the property that are strongly convergent games (Eriksson, 1993). This property ensures that given an initial configuration, either a game can be played forever or it reaches a unique fixed state (where no firing is possible), called the *final configuration*. Furthermore, this state does not depend on the order in which the vertexes are fired.

Dhar (Dhar, 1990), based on this property, showed that in ASPM, for every initial configuration, some final configurations will be never reached (are forbidden), and that the others, the *allowed ones*, will happen with equal probability and in a finite number of steps.

In (Dorso and Dadamia, 2002) we can find a prediction study of every evolution step in an ASPM that represents an avalanche for a given size and in (Moore and Nilsson, 1999) it is demonstrated that the problem to know which will be the final state of an ASPM, given an initial configuration, is P-Complete.

1-Dimensional Sand Pile Model

Another special case of CFG is the Sand Pile Model (SPM) introduced by (Latapy et al., 2001). The graph, in this case, is an undirected chain, infinite on the right, whose vertex have a unique connection to the next. The threshold δ_v is equal to 2 for every vertex, except for the first one, which equals to infinity.

As in the previous models, most of the works related to this are quite theoretical (Durand-Lose, 1996; Goles and Latapy, 2002; Eric Goles and Phan, 2002; Novelli and Rossin, 2001; Latapy et al., 2001). However the novelty of them is that they introduce a variation in the updates of the sand pile. The general rule to decide whether a node must fire or not is calculated by means of a comparison with the load of the next neighbour, and not using the absolute value of the local node.

This variation will be used in our work to obtain similar 2-dimensional models more suitable for visualization purposes.

Implementation Details of ASPM

When working with ASPM based models, it is natural to have in mind the idea that we are representing a pile of sand, like those that exist in deserts or beaches, that evolve along the time. The question we are going to resolve in this section is whether these models represent with accuracy not the behaviour of such systems but their properties related with their visual appereance. The objective of this task is to achieve a good algorithm that can be used in a simulation environment where sand piles, or by extension particle systems, can be modelled according to the point view of their graphical shape.

For visualization purposes, an ASPM model can be displayed using a mesh whose nodes are loaded with a value, like the graph nodes of the theoretical model. This technique is used to represent the shape of the sandpile, not the total amount of sand grains, and therefore the problem is simplified. Also, we are going to suppose for our tests that adding sand to the pile causes the same effect than raising a zone of the mesh and leaving it to evolve to the stable state.

In order to implement the computational algorithm of an ASPM, we have used the method explained in (Dhar, 1990) which states that the number of allowed configurations is equal to the determinant of an integer matrix Δ that specifies the evolution rules.

Δ is a NxN matrix where $\Delta_{ii} = n_v$ and $\Delta_{ij} = -1$ for each node j connected to i, so that row i of Δ represents the amount of sand lost by every node when node i fires.

According to the rules of ASPM, the matrix takes the form

$$\Delta_{ij} = -1 + (n_v + 1)\delta_{ij}$$

where δ_{ij} is the Kronecker delta (δ_{ij} is equal to 1 if $i = j$ and 0 elsewhere) and n_v is equal to 4 since represents the number of neighbours considered.

Since ASPM is a strongly convergent game, it is guarantized that the pile will reach the same final configuration independently of the way the grid nodes are updated. This property is very useful since will allow parallelize the algorithm and therefore implement it using a multithreaded application with increases performance.

Test Results

The computational implementation of this matrix has allowed us perform some tests in order to discover the similarities among the real life and the modelled systems.

All the simulations have been done using a grid of size $30x30$. At the beggining of the simulation process, the cells in the rectangle $[10, 18]$ x $[16, 20]$ have been initialized randomly with $l(v) \in [\ 0\ ,\ 30\ ]$. With this initial conditions the simulation is run until the system reaches the *final configuration*.

Algorithm 1

for $i = 2 : N - 1$
  for $j = 2 : N - 1$
    if $(M(i,j) > \delta_v)$
      $M(i,j) = M(i,j) - \delta_v$;
      for $p = 1 : n_v$
        $k = neigh(p, 1)$;
        $l = neigh(p, 2)$;
        $M(i + k, j + l) = M(i + k, j + l) + \delta_v / n_v$;
      end
    fi
  end
end

Algorithm 1 shows a pseudo-code implementation of ASPM. In this case, as we have seen above, we will use $\delta_v = n_v$.

The experiment has been repeated for two different definitions of the neighbouring of a cell. The first one uses the following expression:

$$neigh_4(i,j) = \{(i+1,j), (i-1,j), (i,j+1), (i,j-1)\}$$

and therefore $n_v = 4$. This neighbouring is the most commonly used, and some examples can be found in (Dhar, 1990) or in (Dorso and Dadamia, 2002).

The second neighbouring tested has been defined as:

$$neigh_8(i,j) = neigh_4(i,j) \cup A$$

where

$$A = \{(i+1,j+1), (i+1,j-1), (i-1,j+1), (i-1,j-1)\}$$

In this case, $n_v = 8$.

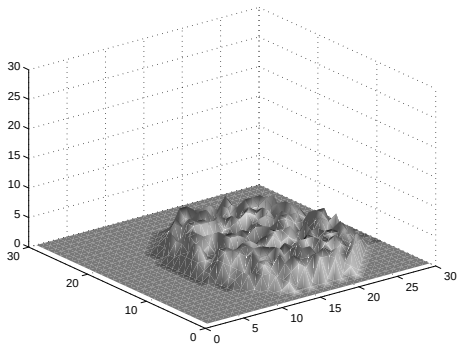

Figure 1: ASPM with $neigh_4$

The simulations were performed using $MATLAB^{TM}$. The cpu-time measured to implement one step was 0.023

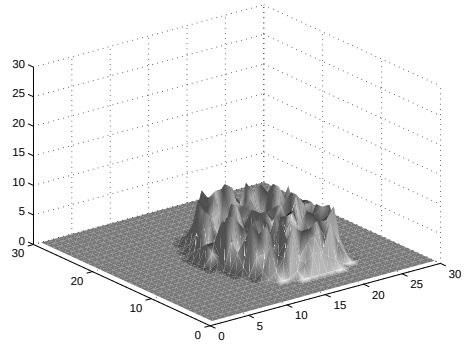

Figure 2: ASPM with $neigh_8$

for the $neigh_4$ case and 0.025 for the $neigh_8$ case. This includes the redrawn of the mesh. It is also worth noting that the number of steps required to achieve the final state is quite significative. In the case of $neigh_4$ was 104 and for $neigh_8$ was 57.

Figures 1 and 2 show the *final configuration* for the simulations. The results show that the ASPM, as it was originally presented, does not behaviour like grain piles or terrain mountain do. The main reason is that the height of any cell is bounded by the threshold of the model, and the system spreads wide until this requirement is satisfied.

For our purpose of simulate sand or grain falling and forming a mountain, this shape of a sand pile is not realistic at all. In the next section we propose some modifications to the original ASPM that will provide more realistic shapes suitable for visualization applications.

SPM for Visual Applications

As can be seen in the previous section, the original ASPM model does not suit very well with the purposes of simulating a realistic sand pile. Most physical grain flow models state that one of the properties of a sand mountain is the angle formed by the slope, which depends on the material properties. In this section some changes in the actualization rules of the models are proposed to cope with this characteristic.

Enhanced Sand Pile Model

We are going to consider a modification of the Abelian Sandpile Model by means of changing the update rule. In the original model an absolute threshold δ_v was used. Whenever the height of a cell overpasses δ_v it fires, causing sand topples.

The new model has to emulate the actual behaviour of sand, characterized by the occurrence of topplings only when the mountain shows high slopes. According to this, every cell will now be compared to its neighbours. If the difference in the height of two adjacent cells overpasses a certain threshold δ_t, the higher cell fires a number of grains z_+ into the lower one.

This model has two parameters, instead of one, and there exists no relation among the threshold and the number of neighbours, as there was in the ASPM. These parameters will allow control some aspects of the final state of the system, such as the angle of the slope or the frequency and length of topplings. Algorithm 2 shows its implementation details.

Algorithm 2

for $i = 2 : N-1$
  for $j = 2 : N-1$
    for $p = 1 : n_v$
      $k = neigh(p,1);$
      $l = neigh(p,2);$
      $dif = M(i,j) - M(i+k, j+l);$
      if $(dif > \delta_t)$
        $M(i,j) = M(i,j) - z_+;$
        $M(i+k, j+l) = M(i+k, j+l) + z_+;$
      fi
      if $(dif < -\delta_t)$
        $M(i,j) = M(i,j) + z_+;$
        $M(i+k, j+l) = M(i+k, j+l) - z_+;$
      fi
    end
  end
end

Some aspects must be considered within the new model. When talking about the ASPM, the concept of strong convergent model was introduced. In this case no guarantee exists that the new model will have even a *final configuration*. This point is not crucial for the objective of this work, though some later studies should be done to clarify it.

Another question, more closed to the subject of this paper, is the decision of how the new rule has to be run. Strictly, the new rule does not determine which of the cells in the neighbouring must be checked first. Two variants have been used in this paper, one with a fixed order of the neighbours and the other taking the neighbours randomly.

Test results

We repeated the simulation made with the ASPM model in order to compare the results of both methods for different values of δ_t and z_+.

We choosed values for δ_t between 2 and 8 and for z_+ between 1 and 8. The results of the simulation showed that if the difference among δ_t and z_+ was small, the visual appearance was better than if this difference was high. On the other hand as higher δ_t was, the slopes become more vertical and the cims less smooth.

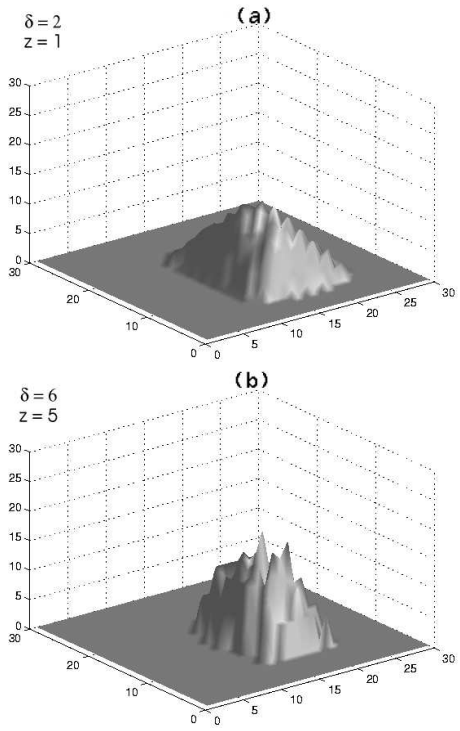

Figure 3: ESPM with $neigh_4$

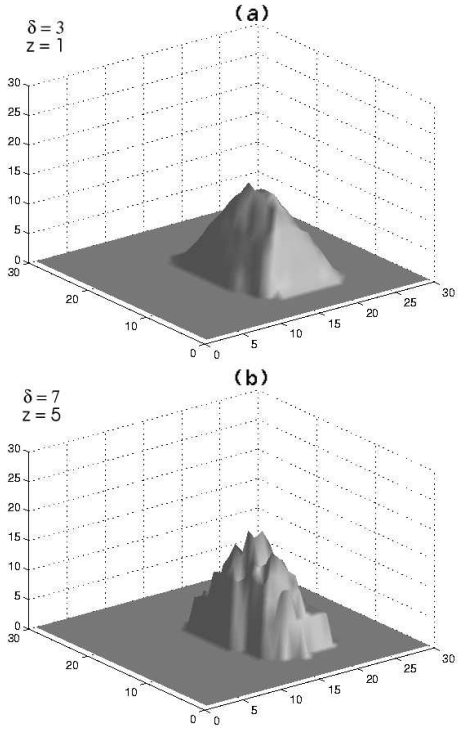

Figure 4: ESPM with $neigh_8$

One of the drawbacks of ASPM that have been resolved, according to our particular study, is that it is not possible to modify the visual aspect of the resulting pile by means of modifying the configuration parameters of the algorithm. Figures 1 and 2 will be always the same and it will not be possible to obtain different slopes nor smoother tops. In ESPM we can modify its parameters (δ_t and z_+) to obtain different kinds of piles. Figures 3 and 4 show the result of the simulation considering a neighbouring of 4 and 8 cells respectively. We can observe that the sand mountain is more realistic since the slopes are much more smooth.

The main difference among the two simulations using ESPM is that for the case of Figure 4 the surface is much more uniform because the sand is not going to the following cell in two-way traffic but in four.

Our tests exhibit a computational cost for every step of Algorithm 2 bigger than the observed in Algorithm 1. In both experiments, the cpu time elapsed for $neigh_4$ was 0.053s and for $neigh_8$ was 0.07s. However, the number of steps required to reach the stable state was 6 for $neigh_4$ and 5 for $neigh_8$, an order of magnitude lower than the used in the original algorithm, so it really outperforms its predecessor.

Conclusions and Future Work

The main idea of this work was to verify how the ASPM model behaves when used to represent the dynamics of a sand pile that evolves until reaching its *final configuration*.

We observed that the ASPM model can not be used to visualize a sand pile simulation and a modification in the model has been proposed, called ESPM, which resembles more our objective.

The test done showed that our model has a computation time of the same order that the original ASPM and that is suitable to be used in applications requiring real time.

As future work some improvements will be done in ESPM for using at every step only those nodes that are affected by avalanches, not the whole mesh.

Another interesting line of research following this work raises when considering not only the fact that we are building a pile of sand but the opposite operation of removing material from it. In simulation applications, it is usual to interact with granular materials, adding, subtracting and also colliding with them, and all those effects modify the way the pile has to be viewed.

Another aspect that we have no mentioned is the possibility of simulating piles of mixed materials, with different cohesion properties. In this case, the ESPM algorithm will have to take into account not only the neighbourhood of the nodes but the classes of them.

References

Bak, P., Tand, C., and Wiesenfeld, K. (1988). Self-organized criticality. *Phys. Rev. A*, 38:364–374.

Björner, A. and Lovász, L. (1992). Chip-firing games on directed graphs. *Journal of Algebraic Combinatorics*, 1:304–328.

Brylawski, T. (1973). The lattice of integer partitions. *Discrete Mathematics*, 6:210–219.

Dhar, D. (1990). Self-organized critical state of sandpile automaton models. *Physical Review Letters*, 64(14):1613–1616.

Dorso, C. O. and Dadamia, D. (2002). Avalanche prediction in abelian sandpile model. *Physica A: Statistical Mechanics and its Applications*, 308(1-4):179–191.

Durand-Lose (1996). Grain sorting in the one-dimensional sand pile model. *COMPSYSTS: Complex Systems*, 10.

Eric Goles, M. M. and Phan, H. D. (2002). The structure of a linear chip firing game and related models. *Theoretical Computer Science*, 270(1-2):827–841.

Eriksson, K. (1993). *Strongly convergent games and coxeter groups*. PhD thesis, Kungl Tekniska Hogskolan.

Goles, E. and Latapy, M. (2002). Complexity of grain-falling models. *Complexity International*.

Latapy, M., Mantaci, R., Morvan, M., and Phan, H. D. (2001). Structure of some sand piles model. *Theoretical Computer Science*, 262(1–2):525–556.

Moore, C. and Nilsson, M. (1999). The computational complexity of sandpiles. *Journal of Statistical Physics*, 96:205–224.

Moss, S. (2001). Competition in internal markets: Statistical signatures and critical densities. Technical Report 01-79, Manchester Metropolitan University. UK.

Novelli, J.-C. and Rossin, D. (2001). On the toppling of a sand pile. In *Discrete Models : Combinatorics, Computation, and Geometry*, pages 275–286. DM-CCG.

Subramanian, R. and Scherson, I. (1994). An analysis of diffusive load-balancing. In *ACM Symposium on Parallel Algorithms and Architecture*, pages 220–225.