Elastic Dimer Automata: Discrete, Tunable Models for Complex Systems

Dustin Arendt¹ and Yang Cao¹
¹Department of Computer Science, Virginia Tech, Blacksburg, VA, USA

Abstract—Cellular automata, dimer automata, and other similar models are useful tools for understanding complex phenomena using simple rules and discrete states. They have simple implementations that run efficiently without problems such as numerical stability and round off errors that are common with continuous models like PDE’s. However, these discrete models, in general, lack a mechanism to tune the desired level of detail or accuracy, an important feature of PDE’s. We propose elastic dimer automata as a general way to reconcile the simplicity of discrete models with the tunable nature of continuous models. Furthermore, we present a simple method for measuring self-organization in elastic dimer automata which we use to aid in an exhaustive search for interesting rules. This search revealed simple rules for several interesting phenomena including multiple different types of wave phenomena as well as a mechanism for labyrinthine patterns and dislocation repairing.

Keywords: self-organization, continuous behavior, dimer automata, asynchronous cellular automata, graph

1. Introduction

The discrete nature of cellular automata [1], [2], boolean networks [3], and other models is both a blessing and a curse. Discreteness allows for a simpler implementation on a digital computer compared to their continuous counterparts, such as PDE’s. The rules describing the dynamics of discrete systems are often easy to understand and implement, and there is no cause to worry about round off error or numerical stability. In many cases, the state of the system can be represented with 8-bit integers, using less memory and CPU resources compared to the floating point representations needed by continuous models. Unfortunately, cellular automata and other models typically lack an important feature common to numerical solutions to PDE’s: a tunable level of detail. Adjusting the space and time step in PDE’s allows one to maximize accuracy within the constraints of available memory and CPU resources. This is not the case when we consider classical cellular automata; there is not usually an obvious way to add states to the rule in a way that “sharpen the picture.” Normally, we can increase the length or size of the simulation, but this does not necessarily equate to a more accurate result.

So, is there a way to reconcile the stable nature of discrete models with the tunable nature of continuous ones? There are many ad-hoc examples of this where cellular automata rules are accompanied by a parameter to determine the number of states used. This effectively tunes the model to exhibit smoother, more continuous behavior. For example, the Greenberg-Hastings [4] and cyclic cellular automata [5], [6] models for excitable media allow an arbitrary number of integer states, which affects the size of the spirals in the simulation. Other approaches have focused directly on cellular automata as a direct discretization of various PDE’s. For example, a coarse discretization of the Belousov-Zhabotinsky reaction was very effective in reproducing the important qualitative aspects of that phenomena, including wave curvature and dissipation [7]. Weimar developed a technique for a quantitatively accurate cellular automaton discretization of reaction diffusion systems [8]. A similar technique based on operator splitting is used by Narbel for cellular automaton discretization of the Fitzhugh-Nagumo equation [9]. Roughly speaking, these techniques carefully transform the continuous phase space of various PDEs into a discrete map to be used as the cellular automaton rule.

These techniques map well known continuous models into cellular automata, but is it possible to accomplish the reverse? Can we start with an arbitrary, coarse-grained, discrete rule and develop it into a continuous model? This has been accomplished for elementary cellular automata using the “inverse ultradiscretization technique” [10]. However, this technique essentially maps the cellular automaton rule into a set of continuous functions that exactly reproduce the discrete behavior of the rule. A more useful and intuitive approach would be to increase the automaton’s state space one state at a time so that in the infinite limit of states the rule behaves continuously, but is still modeled by discrete state transitions.

In previous work we used this approach to create rules for grain growth and spiral waves [11], [12]. We showed that the original 3-state rules could be grown in a straightforward manner preserving the topological properties of these rules. These generalized rules were engineered by studying the behavior of the original rules; we intuited a way to modify the rules to produce their appropriate behaviors. However, a general technique to produce continuous behavior from any seed rule would be very useful. For example, this would enable searching within a finite space of continuous behaviors with the potential to discover simple new explanations for complex phenomena, which is one of the main contributions.
of this paper. We propose a generic tunable discrete model for complex systems called elastic dimer automata that we use to search for simple rules exhibiting self-organization.

Elastic dimer automata allow stretching (i.e. increasing of the state space) to produce smooth, continuous behavior in a generic manner, overcoming the issues discussed above. We also develop a simple statistical method to measure self-organization in dimer automaton configurations having many states. Together, elastic dimer automata and this measurement let us perform an exhaustive search of the resulting behavior space, yielding several simple interesting rules. The rules discovered include several variants of excitable media phenomena with various wave behaviors and labyrinthine patterns that repair dislocations over time.

1.1 Dimer Automata

For this paper we only consider dimer automata, a fully discrete, asynchronously updating model for complex systems [13]. Space is represented as a graph $G = (V, E)$. Each vertex in the graph has a single state from a discrete state space $\Sigma$. Dimer automata are iterated by choosing a random edge in the graph and updating both endpoints of that edge\(^1\) according to a simple rule $R$. Assuming $G$ is undirected lets us assume the effect of $R$ is symmetric. In other words, if $(a, b) \rightarrow (c, d)$ then $(b, a) \rightarrow (d, c)$, which gives $R$ a slightly simpler form (i.e. $R : \Sigma^2 \rightarrow \Sigma$ instead of $R : \Sigma^2 \rightarrow \Sigma^2$), thus

$$(x, y) \rightarrow (R(x, y), R(y, x)).$$

This simplification allows $R$ to be considered a finite state machine whose input alphabet and state space are both $\Sigma$. In this paper we often refer to the rule’s topology, which is the directed graph defined by the state transitions of the finite state machine.

Dimer automata have several nice properties to justify their use here over traditional models such as cellular automata. Representing space as a general graph is much less restrictive than the uniform lattice often assumed (out convenience) for cellular automata. Since, by definition, all edges in a graph have exactly two endpoints, and the rule operates on these endpoints, any rule can be applied to any spatial structure without modification. This is only easily accomplished in cellular automata by assuming the rule is totalistic, or in boolean networks by assuming each vertex has the same degree. Also, the rule is a function of two states, the fewest number of states needed to model interactions, and one less state than elementary cellular automata [1]. Despite the challenges resulting from operating asynchronously over general graphs, in previous work we showed that dimer automata have efficient parallelizations assuming a low-dimensional graph [14].

\(^1\)This occurs by replacing the states of the endpoints of an edge with new states, simultaneously. So, $(a, b) \rightarrow (c, d)$ is to be read as “states $a$ and $b$ are replaced with states $c$ and $d$.”

1.2 Self-Organization

Since the goal is to search for dimer automata exhibiting self-organization, it is important to have a way to quantify self-organization for a number of reasons. Presumably, in the search we will be considering a large number (thousands or more) of dimer automata. The least sophisticated search is one in which a human views each output, and manually classifies them. This was the approach initially taken and still endorsed by Wolfram in his search of elementary cellular automata [1]. However, this approach quickly becomes impractical when one must consider many outputs. So, at the very least, a measurement of self-organization can be used to organize the output to make this task easier. For example, outputs can be sorted according to this value so that the best results are presented first, or a threshold can be set so that only the images exceeding this value are presented. More sophisticated techniques can build on this by replacing the role of the human with a computer algorithm.

Over the years, there have been many approaches towards understanding and measuring self-organization [15], [16], [17], [18], [19], [20]. This is partially due to the term “self-organization” itself being overly ambiguous, and partially due to the many different disciplines interested in the phenomena (e.g., physics, mathematics, computer science) [18]. It is generally accepted that a self-organizing system transforms an initially unstructured configuration in to one with more structure over time. There is no clear consensus on what good definitions of structure are, and how its increase over time should appear. However, there is general consensus that the thermodynamic concept of entropy is a very poor choice for measuring self-organization because self-organizing systems have elements of both order and disorder [17]. Neither a purely homogenous configuration (with zero entropy) nor a purely random configuration (with maximal entropy) are very interesting. Thus, an appropriate measurement will have large values for structured configurations and small values for fully ordered as well as fully disordered configurations.

Our measurement of self-organization makes several assumptions for simplicity and utility. Almost all measurements consider the rate of change of the complexity of the system over time, however, we only consider the final configuration of the dimer automaton. We let the dimer automaton be initialized with a fully random state (which contains zero structure), so we can assume any resulting structure is a sign of self-organization. Measuring the self-organization of dimer automata should also take into account the local, spatial distribution of states. The amount of information that one part of a configuration tell us about another part should depend on how far away those parts are, motivating our information theoretical approach.

Let $D_R$ and $D_G$ be random variables representing the distance between two vertices’ states and spatial locations respectively. If $i$ and $j$ are vertices chosen uniformly at
random from the graph \(G\), then
\[
\begin{align*}
D_R &= d_R(x_i^t, x_j^t), \\
D_G &= d_G(i, j).
\end{align*}
\]
(1)

The functions \(d_R\) and \(d_G\) refer to the finite state machine and graph shortest path distances\(^2\), respectively. We propose measuring local structure by their mutual information,
\[
\mathcal{L}_I = I(D_R; D_G).
\]
(2)

This measurement of local structure agrees well with our intuition about structure in the uniform and uniformly random cases. In either of these two cases, \(D_R\) and \(D_G\) are statistically independent, so the mutual information is 0. This measurement is similar to the concept of long range mutual information [16], and is a simplified and adapted version of Shalizi’s “light cone” approach [20]. Also, we note that our measurement is very general, as it applies to any configuration represented by a graph where vertices take discrete states. This means it can potentially be used for other complex systems models as long as \(d_R\) is appropriately defined.

A visualization of this distribution is shown in Figure 1 where each pixel \((i, j)\) in the image is proportional to \(Pr[D_R = i, D_G = j]\). This compares the joint probability distributions before and after the creation of spiral waves from an excitable media model. The first joint probability distribution shows no overall structure, which is intuitive since the initial condition is totally random. However, the second distribution has several interesting features. Most notably, there is a dark triangle in the upper left corner, implying that \(Pr[D_R > D_G] \approx 0\). Roughly speaking, this means that in this case the topological distance is an upper bound on the distance between any pair of states. Finally, using Equation 2, the mutual information is 0.021 in the initial configuration and 0.574 in the final configuration.

2. Methods: Elastic Dimer Automata

Our primary goal is to formally define a technique for creating arbitrary generalizable rules; the technique we develop here we refer to as elastic dimer automata. These are rules that are created through a two step process where an initial graph is stretched and its edges are labeled to produce the finite state machine for a dimer automaton rule. In the first step, an initial graph, \(G_0\) is stretched \(s\) times through an edge rewriting process to produce a sequence of graphs \(\{G_1, G_2, ..., G_{s-1}, G_s\}\). Edges are rewritten according to
\[
\begin{align*}
\overline{ij} &\rightarrow \{ik, kj\}, \\
\overline{ij} &\rightarrow \{ik, kj\},
\end{align*}
\]
(3)

\(^2\)Since \(G\) is potentially very large, it is not wise to compute \(d_G(i, j) \forall(i, j) \in V^2\). Instead, \((i, j)\) pairs can be sampled by repeatedly picking a random vertex, performing a breadth first search of fixed depth, and picking a random vertex from each depth.

![Initial Configuration](image1.png) ![Final Configuration](image2.png)

Fig. 1: Joint probability distribution (top) before and after simulation of excitable media (bottom).

![Fig. 2](image3.png)

Fig. 2: One iteration of the stretching operation for an example graph.

We let \(G_s\) define the set of allowable transitions in the finite state machine for each state. Let \(N[i]\) be the inclusive neighborhood of vertex \(i\) defined by \(G_s\). Thus \(N[i]\) is the set of states that \(i\) may transition to (including remaining as \(i\)). Deciding what causes this transition allows the edges of the finite state machine to be labeled, creating the dimer automaton rule \(R\). Suppose that for each \((i, j) \in \Sigma^2\) there is a configuration energy defined by \(J(i, j)\). The rule \(R\) is defined by picking a pair of states from the set of allowable
transitions that minimizes the configuration energy, thus

\[ (R(x, y), R(y, x)) = \min_{(i,j) \in N[x] \times N[y]} J(i, j) \tag{4} \]

However, this equation creates some ambiguity if there is more than one pair of transitions tied for the minimum. There are a number of potential ways to handle this tie. One way is to simply disallow transitions when there is not a unique minimum. Another option could be to swap \( i \) and \( j \) (we will refer to this as zero-swapping). The logic is that swapping \( i \) and \( j \) neither increases nor decreases that edge’s configuration energy. Swapping injects enough energy into the system to avoid totally frozen configurations, where no transitions are occurring. Yet another approach is to pick the \((i, j)\) corresponding to the smallest unique energy state. This is analogous to a ball placed on the top of a hypothetical mountain. If both slopes of the mountain are equally steep, the ball will roll down the ridge between the two slopes, even though this route is less optimal. We use a combination of this approach and swapping, where swapping is performed if there is no better unique energy configuration.

We let \( J \) be a function of the distance between \( i \) and \( j \) in \( G_s \) such that \( J(i, j) = F(\min[d_R(i, j), d_H(j, i)]) \). Note that \( J \) is symmetric, which is important since we are still assuming space is undirected. For simplicity we let \( F(d) = d \) so that the configuration energy of \((i, j)\) increases as \( i \) and \( j \) move farther apart in \( G_s \). Under the above assumptions, the rule \( R \) for an elastic dimer automaton is fully defined by \( G_0 \) and \( s \).

3. Results: Searching for Self-Organizing Elastic Dimer Automata

The previous assumptions give us a good starting point to search for interesting elastic dimer automata, with the search space being transformed to the set of all directed graphs. However, this space grows very quickly since there are \( 2^{|V|^2} \) possible adjacency matrices for a directed graph with \( |V| \) vertices. Fortunately, the search space can be further reduced by assuming that each graph contains no self loops, is strongly connected, and is isomorphic unique. It is not useful to allow graphs with self loops because a self loop would specify a state could transition to itself, which is already allowed by Equation 4, since \( N[x] \) includes \( x \).

A strongly connected graph is one in which there is a path of finite length between every pair of vertices. In other words, every vertex is recurrent, so no vertices are absorbing/transient (out/in-degree are zero). Because transient states have no incoming edges, the dimer automaton’s configuration may eventually contain no such states. On the other hand, once part of the dimer automaton’s configuration is absorbing, it can never change to another state. Finally, it is useful to discard rules that are isomorphically equivalent to others since they perform identical jobs.

First, we conducted an exhaustive search of the space of elastic dimer automata where \(|V_0| = 4\), which has 89 unique rules under the above assumptions\(^3\). For each of these rules, we start with random initial conditions run a simulation long enough for the system to exhibit its long term behavior\(^4\). In each simulation, the spatial topology used was an isotropic planar graph, roughly equivalent to a \( 600 \times 600 \) square lattice. From this set of 89 rules, we identified four interesting and unique forms of self-organizing behavior: rules\(^5\) 13, 18, 55, and 85. Figure 3 shows the information \( J \) and energy \( H \) of these four rules over time (with and without swapping) and Figure 4 shows the topology of each rule and its configuration after a long time. The configuration energy is the average energy of each edge in the graph using the above definition of \( J \), thus

\[ H = \frac{1}{|E|} \sum_{ij \in E} J(x_i, x_j). \tag{5} \]

Rule 13 produces fast moving spiral waves and swapping produces slightly larger waves in this case, but there is not a significant change in the qualitative behavior or in the information-energy curve. Rule 18 shows a rapid decrease in energy relative to the more gradual trends seen in the other rules. Furthermore, in both cases, there is a spontaneous increase in the configuration energy eventually following the initial sudden drop. This increase is then followed by yet another drop, where the system eventually settles down. Without swapping, rule 18 produces a combination of quickly moving wave fronts that leave a wake of slowly decaying uniform regions. Without swapping, the behavior is characterized by rounded homogeneous regions that spontaneously transition to a different state. Rule 55 appears to be a frustrated version of rule 13. The length scale of the waves are much larger, and spirals are less fully developed than those seen in rule 13. Swapping allows the waves to propagate more quickly, but does not appear to affect the qualitative behavior. Rule 85 is perhaps the most interesting, producing labyrinthine patterns without zero-swapping (an example of this is shown in Figure 5). Its configurations are characterized by dislocations, which gradually disappear over time, resulting in a low energy, high information configuration. Swapping allows the dislocations to be fixed more quickly, but causes the information to decrease eventually, perhaps because of the creation of large, nearly uniform regions. Once the labyrinthine pattern is formed, removing dislocations drives the system closer to a uniform state, decreasing its information content.

---

\(^3\)Without these assumptions, we would be considering \(2^{|V|^2}\) rules.

\(^4\)Experimentally, we found that \(300 \times |V| \) edge updates were sufficient.

\(^5\)The rule’s number refers to the order in which the algorithm generates each graph.
4. Discussion

Our intention in creating elastic dimer automata is to couple the tunable level of detail of continuous models such as PDEs with the simplicity, speed, and stability of discrete models such as cellular automata. The results thus far are promising, with elastic dimer automata producing a variety of interesting phenomena with continuum-like behavior. The stretching operations have the intended effect of tuning the level of detail in the automaton. Stretching the initial graph increases the characteristic length scale of the structures produced by the elastic dimer automaton. An example of this is shown in Figure 5 for rule 85 without swapping. Stretching causes structures take up more space, and require more time to develop. Finally, we note that random asynchronous updating is a closer approximation of continuous time than the synchronous method used by cellular automata since state changes are discrete. Synchronous updating in cellular automata moves the system forward in one large leap because state is discrete, as opposed to synchronous updating in PDE’s where state is continuous, and is updated by very tiny increments. So we conclude that stretching elastic dimer automata is analogous to simultaneously tuning their space and time step, which provides the tunable level of detail we have sought in their design.

Fig. 3: Information-Energy time series of rules 13, 18, 55, and 85 with (top) and without (bottom) swapping.

Fig. 4: Elastic dimer automaton 13, 18, 55, and 85 state transition topology (left) and resulting configurations (right) when swapping is allowed.
Fig. 5: Stretching increases the level of detail without qualitatively affecting its behavior (rule 85 without swapping is shown).

### 4.1 Rendering

Although the mutual information is useful in identifying configurations with structure, it is still important to render the configurations so they can be interpreted visually. The naïve approach of using a grayscale value directly proportional to each state will not work. The reason is that there may be little to no correlation between a state's value and its context in the finite state machine. So, it is necessary to develop an alternative way of rendering configurations. Recall that elastic dimer automata are constructed from an initial seed graph $G_0$, and that the indices of this original graph map directly to their corresponding indices in the final stretched graph $G_s$. Therefore, we let the color $c(i)$ corresponding to a state $i$ in $G_s$ be dependent on that state's distance to one of the original vertices such that

$$c(i) = \min_{j \in V_0} d_R(i, j).$$

In other words, a state's color will be proportional to that state's shortest distance to one of the original vertices from $V_0$. The effect is to accentuate regions of near-equal state, and for smooth state transitions in space to appear as gradients. An example of the usefulness of this transformation is show in Figure 6 compared to the naïve case where $c(i) = i$.

Fig. 6: A comparison of two rendering techniques for dimer automata.

### 4.2 Distribution of Behaviors

An interesting question to ask for this system is, what is the distribution of elastic dimer automata behaviors in the information-energy space? To approach this, we extended the search from the previous section by one, so that $|V_0| \leq 5$. This resulted in a total of 5137 suitable unique rules. The information and energy of the final configuration of each rule is plotted in Figure 7. Additionally, we used k-means clustering to pick 9 characteristic rules; the configurations of each of these rules are also shown in the plot. The information-energy distribution has several interesting features that become apparent as a result of the larger search space. First, there appears to be a clear upper bound on the ratio of information to energy, so we hypothesize that

$$H = O(L^{-1}),$$

as evidenced by the emptiness of the upper right portion of the plot. Furthermore, there appears to be another empty region where $L \in [0, 0.1]$ and $H > 1$. This suggests, at least for this experiment, a correlation between very low information and very low energy. In other words, the configurations in this region are mostly uniform, and not mostly random. The tail of the distribution where $L \geq 2.5$ corresponds to low energy and high information configurations that result from spiral waves (e.g. rule 13). The remaining rules that generate high information configurations in the distribution appear to be somewhat similar to rule 18.

### 5. Conclusion

We have presented a technique, elastic dimer automata, that when given a small (e.g. granular) state transition graph, creates a dimer automaton rule in a manner that shows successively more continuous like behavior. Additionally, we have developed a measurement of self-organization tailored towards these rules based on the mutual information between state and space distance distributions. Using these tools, we performed an exhaustive search of a simple class of elastic
dimer automata which revealed several interesting rules, and a rich behavior space.

Future work can take several directions. One question we may ask is, can we derive the PDE corresponding to a given elastic dimer automaton directly by considering the infinite limit of states? This may be simple for cases such as rules 13 and 85 where the transition graph is simply a cycle, allowing a direct mapping between the discrete state and its continuous phase angle. However, in the general case, deriving the PDE may be less trivial; perhaps other variants of elastic dimer automata would facilitate this better. Furthermore, in this paper we consider only a very simple energy function to minimize; future work may consider other classes of energy functions or even domain-specific functions to model a narrower range of physical phenomena. Finally it is useful and frequently challenging to demonstrate the connection between the abstract rules for complex systems and the corresponding physical phenomena (if such a connection exists). We encourage readers to do so for interesting phenomena revealed here or any that may be found in the future.

Acknowledgement

The work of Yang Cao was supported by the National Science Foundation under awards CCF-0726763 and CCF-0953590, and the National Institutes of Health under award GM078989.

References