Complex Networks from Simple Rules

Richard Southwell
Jianwei Huang
Information Engineering Department
The Chinese University of Hong Kong
Shatin, New Territories, Hong Kong
richardsouthwell254@gmail.com
jianweihuang@gmail.com

Chris Cannings
School of Mathematics and Statistics
University of Sheffield
Sheffield, S3 7RH, UK
c.cannings@sheffield.ac.uk

Complex networks are all around us, and they can be generated by simple mechanisms. Understanding what kinds of networks can be produced by following simple rules is therefore of great importance. This issue is investigated by studying the dynamics of extremely simple systems where a "writer" moves around a network, modifying it in a way that depends upon the writer's surroundings. Each vertex in the network has three edges incident upon it, which are colored red, blue, and green. This edge coloring is done to provide a way for the writer to orient its movement. The dynamics of a space of 3888 of these colored trinet automata systems are explored. A large variety of behavior is found, ranging from the very simple to the very complex. Our systems are studied using simulations (with appropriate visualization techniques) and selected rules are analyzed mathematically. An empirical classification scheme is arrived at, which reveals a lot about the kinds of dynamics and networks that can be generated by these systems.

1. Introduction

Many advances in complex systems research have come from identifying simple deterministic systems that can generate complex behavior. For example, cellular automata are used to model many systems in physics and biology [1], string rewrite systems are used to model plant development [2], and chaotic differential equations are used in many areas [3]. The most popular models of complex network growth are probabilistic [4, 5]. These models have yielded many insights, but they do not offer an explanation as to *where* the complex-

ity in networks comes from. The complexity in probabilistic models comes from the randomizing mechanisms they are based on.

In this paper, we study a class of simple deterministic network growth models. Our systems can produce a wide variety of behavior, ranging from the very simple to the very complex. They offer insights into how complex structures can be generated, because each aspect of their behavior can be traced back to a deterministic cause. The simple nature of our models allows us to quickly generate exact pictures of their dynamics. This gives us the ability to easily explore the behavior of large numbers of systems. By taking an unbiased look at the dynamics of many simple rules, we see what types of network growth are easily generated by simple computational processes.

In Turing machines [6], a writer moves along a one-dimensional tape and rewrites symbols using local rules. Our *network automata* systems are like a generalization of this idea—where the writer moves around a network and rewrites it on a local level. The way the writer moves and modifies its local structure is determined by its surroundings. The network the writer runs on is a *trinet* (i.e., each vertex has three connections). Trinets are also known as cubic graphs and three-regular graphs. There are many natural examples of trinets, such as two-dimensional foams [7] and polygonal networks formed by biological cells and cracks in the soil [8]. We focus on trinets because they are easy to manipulate and expressive. Any network can be represented as a trinet by replacing vertices that have more than three connections with roundabout-like circles of vertices with three connections [9].

One of our goals is to find simple deterministic rules that generate complex dynamics. An obstacle to this goal is the inability of deterministic rules to break symmetries. For example, if the writer is at one of the corners of a cube, then it has no deterministic way to select which of its similar-looking neighbors it should move to next. We circumvent this obstacle by supposing the edges of our trinet are colored red, blue, and green in such a way that touching edges have different colors (such an edge coloring is also known as a Tait coloring [10]). This allows us to specify how the writer should move by stating which colored edge it should move along in different situations. (We could remove the reliance of edge color in our systems by replacing the red, blue, and green edges with different uncolored structures and modifying our rewrite rules accordingly—although this would make the systems look more complicated.)

Our main goal is to understand what kinds of behavior can be generated by colored trinet automata. To achieve this, we do a thorough exploration of the dynamics of a space of simple rules. Using pictures and analytic techniques, we sort these rules into three classes (fixed

points, repetitive growth, and elaborate growth) according to their behavior. We discuss these classes in Sections 2 through 4. In Section 5, we exhibit more general rules, which can produce other exotic behavior such as persistent complex behavior and periodically changing networks.

We have tried to keep the main text as free from equations as possible. This was done to increase readership, and because complex phenomena are often better explained using pictures. Proofs of our theorems and extra technical details about these systems will be included in our upcoming paper, "Complex Networks from Simple Rules: Technical Details," which will be published in *Complex Systems* soon.

1.1 Related Literature

Many interesting deterministic network growth models have been previously considered. These include the growth models discussed in [11], the fascinating deterministic network growth systems considered in [12], and the biologically inspired models considered in [13–16], which can produce rich and complicated self-replicating structures, despite extremely simple rules. Self-replication in adaptive network models was also considered in [17]. In each of these cases, every part of the network gets updated in parallel. In our systems, by contrast, only a small piece of the network gets updated at any time. This makes our systems easier to implement because there is no need for distant parts of the network to have synchronized clocks. It is true that many real-world networks grow in parallel; however, evolving our systems for many time steps often leads to structural modifications that are equivalent to global/parallel rewrite operations (see Sections 4.1 and 4.4).

Our work was inspired by the work of Tommaso Bolognesi [9], who studies the dynamics of planar trinet automata. Like our models, these systems involve a writer that moves around the trinet, making structural modifications as it goes. Unlike our systems, the models considered in [9] circumvent the symmetry-breaking problem by supposing the trinets are embedded in the plane. The way the writer behaves in these systems is governed by the structure of the faces formed by the planar embedding. Although these models are fascinating, the fact that the dynamics depend on how the network is embedded makes it difficult to get a full picture of what is going on at any given time. Our approach (of considered edge-colored trinets, rather than planar embedded trinets) allows us to easily picture the complete dynamics (and rules) behind the systems we consider. Our systems also have similarities to the generalized mobile automata considered in [18].

The experimental approach we use was pioneered by Stephen Wolfram [18], who uses simulations to reveal a vast array of simple

programs that can generate complex dynamics. Wolfram finds many network growth models that can generate complex dynamics and explores the idea that the physical universe could be generated by a simple network growth model. The idea that the universe can be generated in exact detail, by a simple program, is one of the most interesting conjectures from digital physics [19–21]. Wolfram suggests that the correct model could be found by doing an automated search of the simplest possibilities [22]. In order to do such universe hunting, intuition is needed about the kinds of things that simple adaptive network mechanisms can do. One of our aims is to provide some of this intuition.

1.2 How Our Systems Work

A colored trinet automata is a dynamical system where a writer moves around the vertices of an edge-colored trinet, applying rewrite rules as it goes. For every time step, some rewrite operation is applied about the writer's current location, and then the writer moves. We start by considering only extremely simple rewrite operations, where the writer's current vertex is either replaced with a triangle or left unaltered. The action the writer takes on a given time step is determined by the colors of the edges interlinking its neighbors.

The rules behind a colored trinet automata specify how the writer should move and modify the network in response to its surroundings (i.e., the colors of the edges interlinking the writer's neighbors). We show the rules at the top of our figures, with the writer represented by a black vertex. For example, the rule for the system shown in Figure 1 can be described as follows.

1. If there are no edges linking the writer's neighbors, then the writer moves along a blue edge, and the writer's previous location is replaced with a triangle.
2. If there is exactly one edge linking the writer's neighbors, which is red, then take no action.
3. If there is exactly one edge linking the writer's neighbors, which is blue, then the writer moves along a red edge and the network is left unaltered.
4. If there is exactly one edge linking the writer's neighbors, which is green, then take no action.

Notice how these instructions correspond to the pictures in the four boxes at the top of Figure 1. In general, a rule is just a specification of which structural modifications and movements the writer should perform in response to the four types of surroundings (depending on the colors of edges interlinking the writer's neighbors) that the writer can have when there is no more than one edge linking

its neighbors. Our rules do not specify which actions the writer should take in other situations, where there are two or more edges linking its neighbors. In these cases we suppose that no action is taken. Actually, the rules we consider never generate vertices with two or more linked neighbors and so these situations never occur anyway.

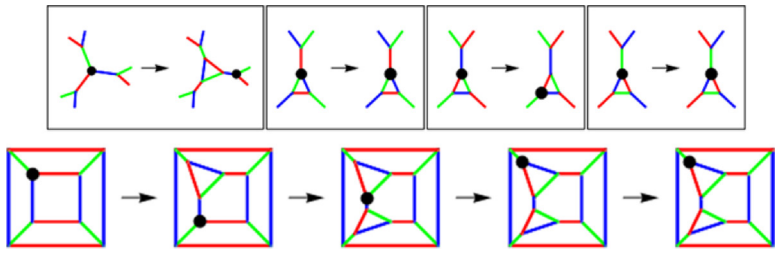

Figure 1. An example colored trinet automata. At the top, we show the rules of the system by indicating how the writer (the black vertex) moves and modifies the network in response to its surroundings. Underneath, we show the system evolving for four updates, starting from the cube. The system reaches a fixed point after three updates. The red, blue, and green edges (which are aligned horizontally, vertically, and diagonally within the cube shown at the bottom left) are more clearly visible in the online color version of this document.

1.3 The Rule Space

In each of our investigations we used the cube (shown at the bottom left of Figure 1) as our initial condition. We studied the dynamics of the space of $12 \times 18 \times 18 = 3888$ rules depicted in Figure 2.

Consider the collection of all rules where the writer's reactions consists of possible triangle replacement and movement across two or fewer links. Our rule set consists of the members of this collection such that (1) the writer's location is always replaced with a triangle when the writer has no interlinked neighbors, and (2) no action is taken when there is a green edge linking the writer's neighbors. We insist upon condition (1) because rules that do not satisfy this will never change the initial cube network, and so will have uninteresting dynamics. We insist upon condition (2) to reduce the number of rules we must consider. If we drop constraint (2), we find new kinds of behavior, but the rule space becomes too large to thoroughly explore. Without constraint (2), the set has $12 \times (18)^3 = 69\,984$ members, which makes it small enough to explore with a computer, but too large to examine all the interesting cases in detail.

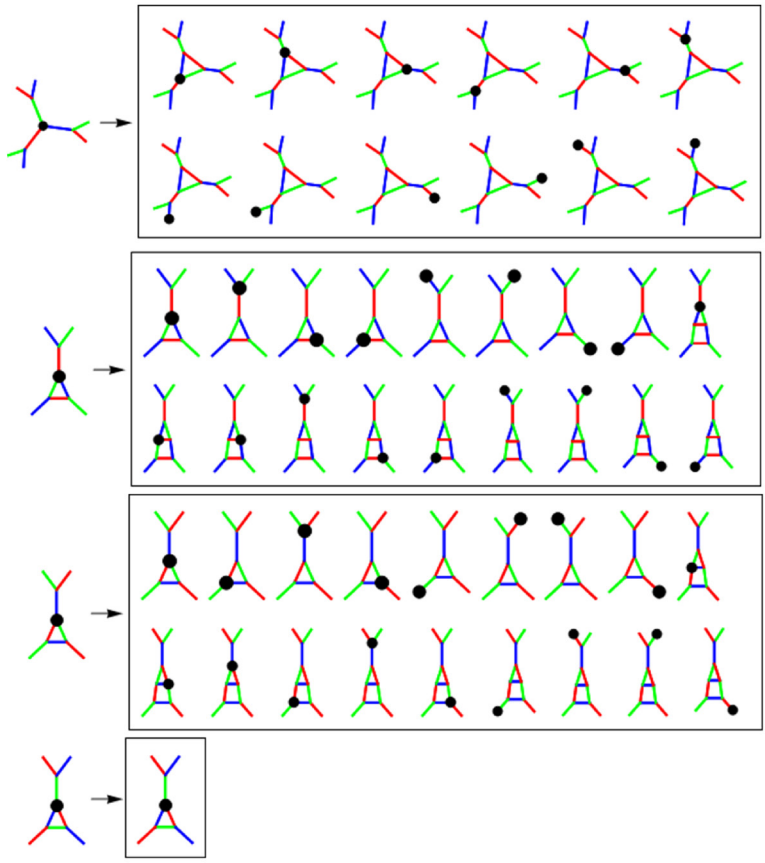

Figure 2. An illustration of the space of rules we explored. A rule is specified by associating each of the four different types of surroundings with one of the images to its right.

By severely limiting the kinds of operations our trinet automata can perform, we reduce the rule space to a manageable size and ensure that it contains minimal systems that generate certain types of behavior. We sort our 3888 rules into three classes (fixed points, repetitive growth, and elaborate growth) according to the long-term dynamics they generate, starting from the cube (see Figure 3). The following three sections are devoted to describing the behavior of the rules in these three classes.

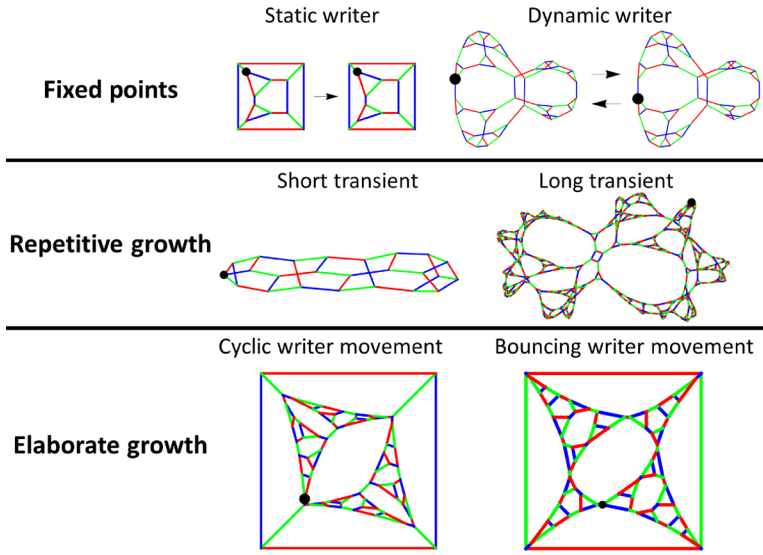

Figure 3. Our rules are sorted into three classes, and each class is divided into two subclasses. We picture a network produced by a rule from each subclass. Fixed point type rules produce small static networks. Repetitive growth with short transients quickly generates repeating substructures, while the rules with long transients exhibit complex growth for many time steps. Rules exhibiting elaborate growth tend to create networks with fractal structures.

2. Fixed Points

We say a system has reached a fixed point where there comes a time after which the network no longer changes. Of the rules, 2918 (about 75 percent) eventually reach a fixed point. Our example system from Figure 1 reaches a fixed point after three updates. One reason such a large number of rules end up at a fixed point is that our rule space is such that whenever a green edge links a pair of the writer's neighbors, a system becomes fixed. Indeed, 2562 of our rules halt their evolution because of this effect. These include the rule that grows the largest fixed structure, shown in Figure 4.

In addition to the fixed points where the writer halts, there are *dynamic-writer* fixed points where the writer continues to move forever, even after the network has become static. Only 162 of the rules in class 1 evolve into dynamic-writer fixed points. In 130 of these rules, the writer ends up doing a period two orbit (see Figure 5); in the others, the writer ends up doing a period four orbit.

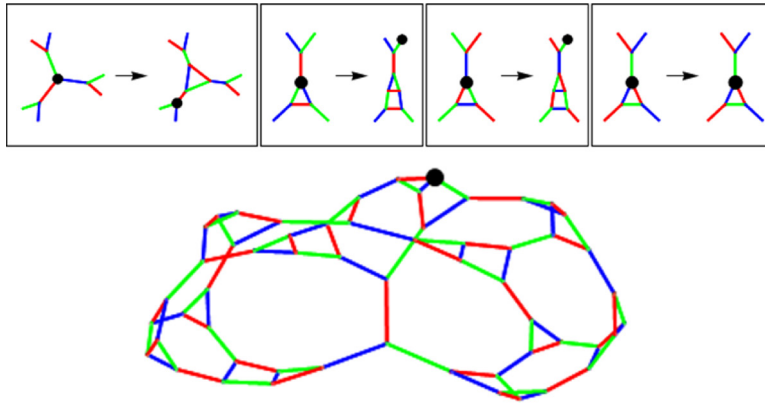

Figure 4. Of all our rules, this one generates the largest static network (with 56 vertices). It takes 26 updates for the cube to change into this network.

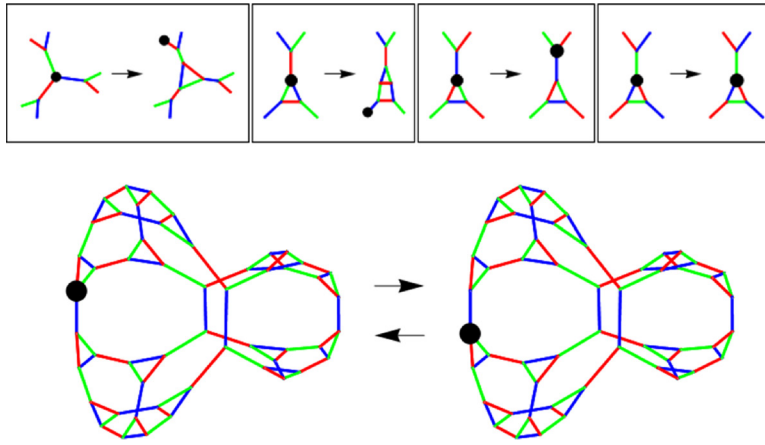

Figure 5. Of all our rules, this one takes the longest (34 updates) to reach a static network (which has 52 vertices). The writer continues to move in the period two orbit shown after this network has been generated.

3. Repetitive Growth

The next most complex type of behavior observed is repetitive growth. This is characterized by the feature that the structure continues to grow, while the writer is trapped within a particular region of the network—with the form of its surroundings changing periodically. Repetitive growth occurs because the writer's surroundings induce it to generate more structure around itself of the same form.

Repetitive growth is eventually generated by 840 of our rules (see Figure 6).

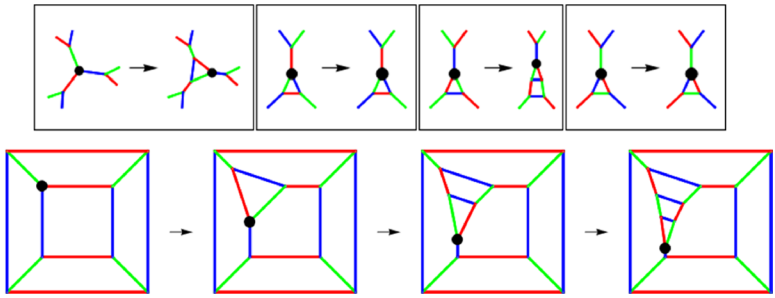

Figure 6. A rule that generates simple repetitive growth evolving over the first four updates. Once the writer's surroundings are such that a pair of its neighbors are linked by a blue edge, the writer performs an update that leads to similar surroundings occurring again on the next time step.

Let us define repetitive growth more precisely. Period p repetitive growth is occuring at time t when the network grows arbitrarily large eventually, and there is a distance r such that the network on vertices within a distance r of the writer's position at time t looks identical to the network on vertices within a distance r of the writer at time $t + p$, and the writer never moves to an old vertex more than a distance $r - 1$ from where it was at time step t during the interval $[t, t + p]$. The rule shown in Figure 6 falls into period one repetitive growth because after the first update, the structure within a distance one of the writer always looks similar. Evolving this rule for $t > 0$ time steps results in a network with $8 + 2\,t$ vertices and a single triangle on the end of a long ladder-like substructure.

Although whether dynamics fit our definition can be tested using a computer, there are more practical ways to spot repetitive growth (see Figure 7). Networks undergoing repetitive growth tend to have an elongated linear or circular shape because they are composed of a series of repeating substructures. Plotting the index of the writer over time is an excellent way to see dynamics. However, it does depend on the way the vertices are indexed within the computer program, and this inevitably depends on more than just the pure topological dynamics of the system. In our case, when a vertex with index v in an L vertex network is replaced with a triangle, we give the vertex of this triangle with a red external edge an index v, and we give the other two vertices of this triangle, with green and blue external edges, indices $L + 1$ and $L + 2$, respectively.

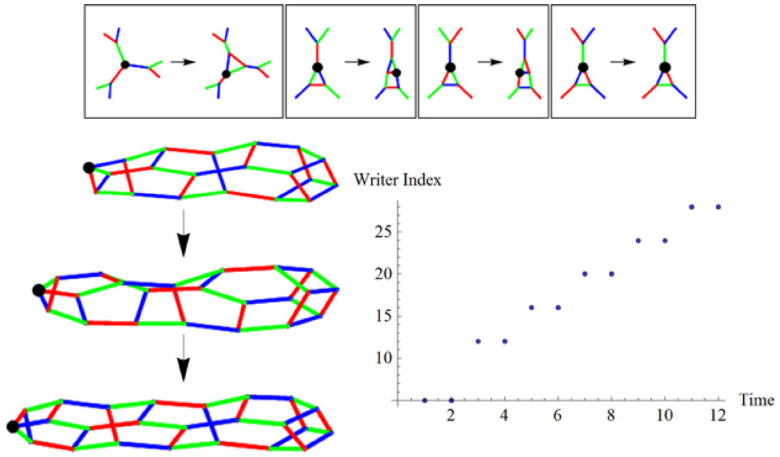

Figure 7. On the left, we show the networks generated on time steps 6, 7, and 8 (reading downward) when evolving from a cube. On the right, we plot the index of the writer's position over time. Notice how the differences between the indices of the writer's position form a periodic sequence. This is because, although the network keeps growing, the writer keeps moving in the same relative way. This system has period two repetitive growth because the network induced on vertices within a distance two of the writer on time step 6 looks identical to that induced on vertices within a distance two of the writer at time step 6 + 2.

3.1 Repetitive Growth with Long Transients

All but four of our repetitive growth rules settle into low (i.e., less than 12) period repetitive growth quickly (in less than 40 time steps). The other four rules eventually produce repetitive growth (see Figure 8), although it would perhaps be best to describe their behavior as complex because the systems have extremely long transients within which the structures appear to grow in a pseudorandom way (this is reminiscent of elementary cellular automaton 110 [18]). Interestingly, evolving the rule shown in Figure 8 from other small networks produces different behavior. The network known as $K_{3,3}$ is obtained by taking two clusters of three vertices and linking each vertex in one cluster to each vertex in the other. Initiating this system from $K_{3,3}$ (instead of the cube) leads to dynamics that (again) eventually settle in period 454 repetitive growth, although this time the transient lasts for 29 964 time steps. Another of the four rules that induce repetitive growth after a long transient is symmetrically equivalent to the rule shown in Figure 8, because it can be transformed into it by swapping the roles of the red and blue colors.

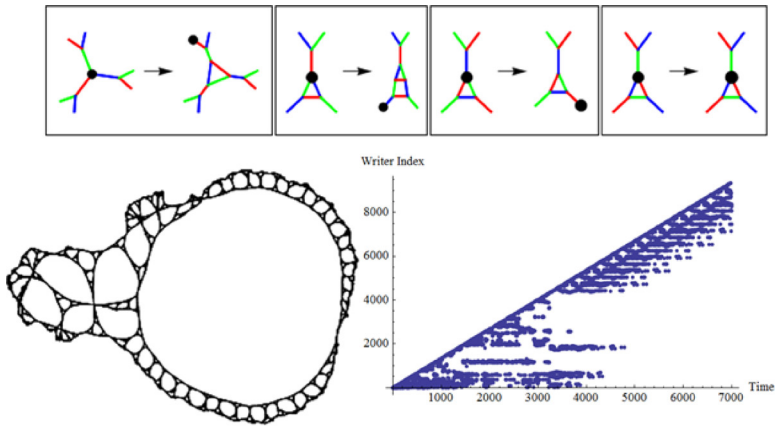

Figure 8. This system has complex behavior for the first 4994 time steps and then settles into period 454 repetitive growth. We show the rule at the top. On the left, we show an uncolored plot of the network present on time step 7000. The large "handle" developing on the right side of this picture is a symptom of the repetitive growth. The plot on the right shows the writer index over the first 7000 time steps.

The other two rules that induce repetitive growth, after a long transient, are not symmetrically equivalent to the system shown in Figure 8, although they are symmetrically equivalent to each other. One of these rules is shown in Figure 9.

The ability of these systems to alter their behavior after large amounts of time leads to the production of heterogeneous substructures. The fact that these rules produce a complex "ball" of network followed by a long one-dimensional structure is reminiscent of the way plants grow by complicating the structure around the initial seed and then growing out a long stalk (see the figures from Section 5).

4. Elaborate Growth

The remaining 130 rules produce elaborate patterns of growth, leading to self-similar networks. It turns out that in each of these cases, the way the writer moves is effectively one-dimensional. The writer repeatedly traverses a one-dimensional "track," applying rewrite rules as it goes, and lengthening the track with each traversal. In most cases, the fact that the writer is confined to a one-dimensional substructure can be inferred directly from the rules. In the other cases, this behavior is revealed by using appropriate visualization techniques.

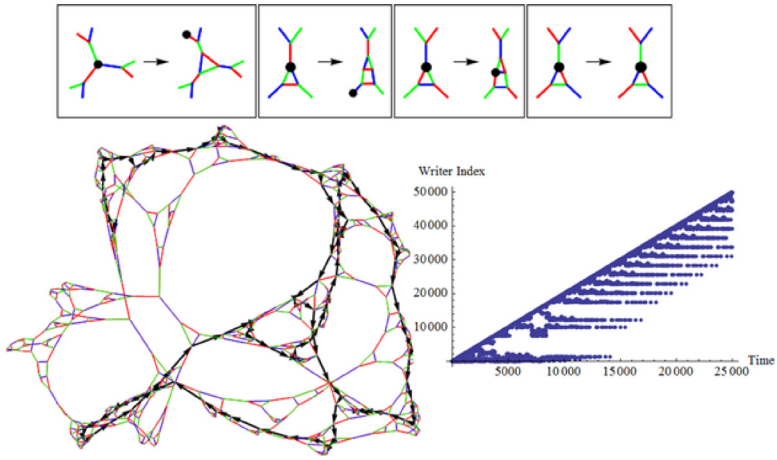

Figure 9. On the left, we show the network obtained after evolving this rule for 300 time steps. The black arrows track the course that the writer has taken over the last 150 time steps (note that the positioning of these arrows depends on how we indexed our vertices). The writer's movement looks unpredictable during this early transience. On the right, we plot the writer's index over time. The transient behavior lasts 17 615 time steps; then the system settles into period 1355 repetitive growth. Similar behavior ensues when evolving from other initial conditions.

The writer can move backward and forward on the one-dimensional track it is confined to. Although the writer can move in complex ways, we can sort the rules into two subclasses according to the general nature of the writer's movement. Either the writer moves around and around a closed loop, in which case we say the rule has *cyclic writer movement*, or the writer moves backward and forward over a line, in which case we say the rule has *bouncing writer movement*.

4.1 A Simple Case with Cyclic Writer Movement

Only 104 of the rules with elaborate growth have cyclic writer movement. We show one of the simplest in Figure 10. By looking at the rules of this system, is it clear that the writer must move in a one-dimensional fashion. Whenever there are no red or green edges interlinking the writer's neighbors (i.e., whenever the system is not at a fixed point), the writer moves along a red edge and then a blue edge, and the writer's previous location is replaced with a triangle. The fact that the writer's movement can be expressed as a combination of steps along edges of *only two* different colors implies that the writer's movement is effectively one-dimensional. In particular, since any suffi-

ciently long path along edges of alternating red/blue color must eventually return to its starting point, the writer must always remain confined to a cyclic track. In this case, the cyclic track starts out as the inner face of our cube (see Figure 11). On each update, the writer replaces its current vertex with a triangle and moves two edges clockwise around the developing track of edges with alternating red/blue colors. Theorem 1 describes the global rewrite operation, which the writer effectively performs with each complete traversal of its cyclic track. The amount of time successive traversals take doubles.

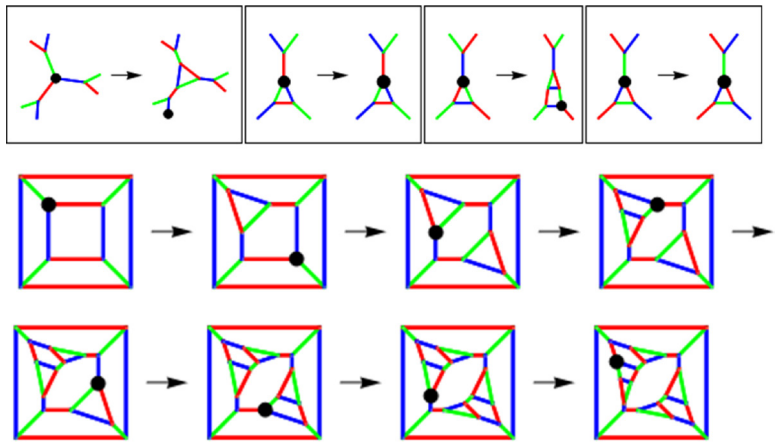

Figure 10. One of the simplest rules with cyclic writer movement, evolving for seven updates.

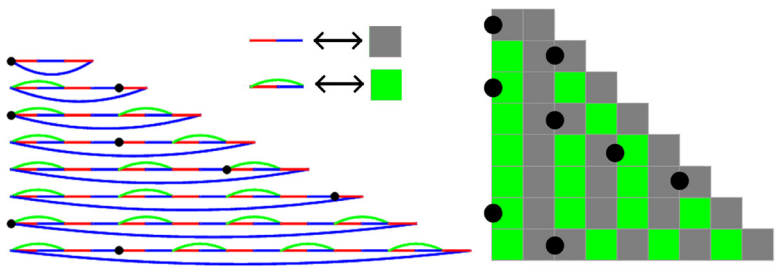

Figure 11. An alternative "one-dimensional" representation of the dynamics of the rule shown in Figure 10. Successive rows of the picture on the left (reading downward) show the relevant part of the network on successive time steps. Here the relevant part of the network consists of network induced on the vertices that can be reached by moving along red or blue edges from the writer. We do not show green edges that are not part of triangles, since these

have no effect on dynamics. For each time step, the writer replaces its current vertex with a triangle (effectively adding a red edge and a blue edge below a green arc) and moves two edges to the right. On the right, we illustrate how this system can be viewed as a string rewrite system.

Theorem 1. The way the rule shown in Figure 10 evolves (with the cube as the initial condition) is such that for each $n \geq 2$, the network present on the $\left(2^{n+1} - 2\right)^{\text{th}}$ time step can be obtained by taking the network present on the $(2^n - 2)^{\text{th}}$ time step and then simultaneously replacing each vertex, with a red or blue edge interconnecting its neighbors, with a triangle (see Figure 12).

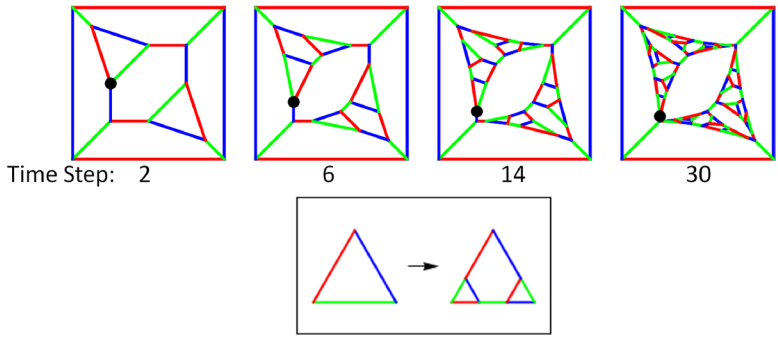

Figure 12. The network generated by the system with rules shown in Figure 10 on the first few time steps of the form $2^n - 2 : n \geq 2$. Between times of this form, the writer makes a complete traversal of its cyclic path and effectively performs the rewrite operation shown at the bottom globally.

4.2 A Rule Related to the Golden Ratio

In Figure 13, we show a rule with cyclic writer movement with a complicated-looking growth rate. It turns out that the growth rate can be described exactly in terms of the golden ratio $\phi = \left(1 + \sqrt{5}\right)/2$.

Theorem 2. The number of vertices in the network obtained by evolving the rule shown in Figure 13 for $t \geq 0$ time steps (starting from the cube) is

$$8 + 2\left\lceil \frac{1}{\phi^2} \left\lfloor \frac{t}{2} \right\rfloor \right\rceil + 2\left\lfloor \frac{t+1}{2} \right\rfloor.$$

It is pleasing that this seemingly complex growth rate can be described simply in terms of the golden ratio, because this number ap-

pears in so many interesting places in the natural world. This colored trinet automata has similar qualitative behavior to the one considered in Section 4.1. Once again the writer goes around and around a cyclic track consisting of edges with alternating red/blue color. Again the system can be reduced to a one-dimensional string rewrite system. The proof to Theorem 2 is based on relating this system to the binary rewrite system with rules $0 \to 01$, $1 \to 011$. This proof will be given in our upcoming paper "Complex Networks from Simple Rules: Technical Details."

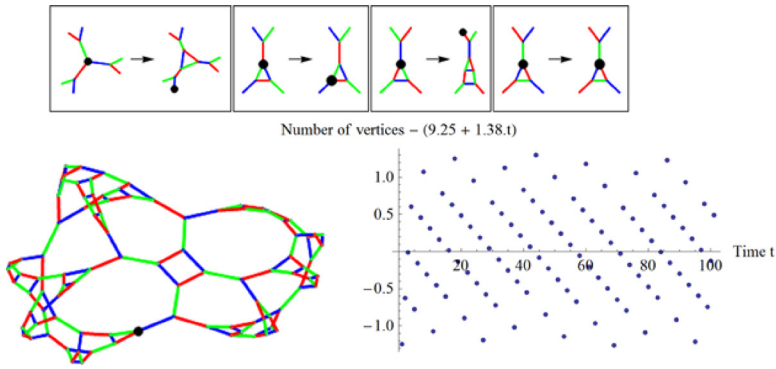

Figure 13. On the left, we show the network obtained by evolving this rule for 62 time steps. The number of vertices in the network at time step t grows approximately linearly with t. However, the way the number of vertices deviates from its best linear fit looks complicated (as shown on the right).

4.3 A Rule with Complex Behavior

The rule with cyclic writer movement shown in Figure 14 has a very complicated-looking growth rate. Although this system can be transformed into a one-dimensional rewrite system (in a way similar to the systems considered in Sections 4.1 and 4.2), we have not been able to derive a formula for its growth rate. This means this system, together with its equivalent red/blue reflection, stands out as the most complex rules in our entire set. We are unable to predict the long-term dynamics of these systems, although it appears the pseudorandom growth pattern continues forever. (All of the rules that generate fixed points and repetitive growth have trivial long-term behavior. Plotting the writer index over time reveals significant regularities in all the other rules exhibiting elaborate growth.)

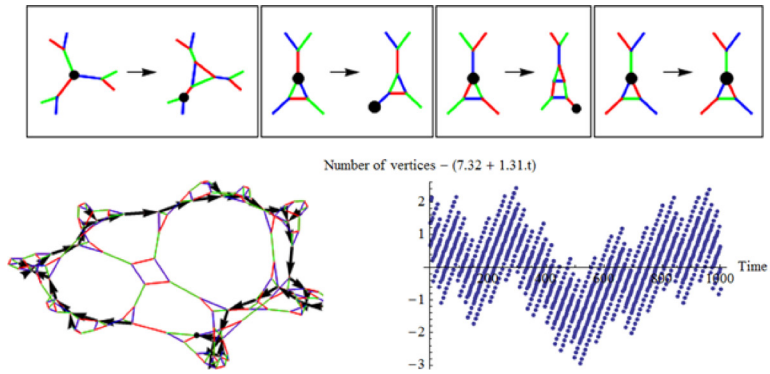

Figure 14. On the left, we show the structure obtained by evolving this rule for 100 time steps. The black arrows track the positions of the writer over the previous 45 time steps. On the right, we show how the growth rate of the number of vertices deviates from its best linear fit. This plot reveals considerable complexity.

Rules could be designed to directly emulate Turing machines by allowing the writer to perform more different kinds of structural rewrite operations (in addition to triangle replacement). In this way, a computationally universal colored trinet automata could be created by emulating the simplest universal Turing machine. Wolfram's Principle of Computational Equivalence [18] (which asserts that almost every system with behavior that does not seem obviously simple corresponds to a computation of equivalent sophistication) suggests that some of the systems we have encountered (such as the ones shown in Figures 9 and 14) should be computationally universal. However, it can sometimes be very difficult to prove that a given system is computationally universal.

4.4 A Simple Case with Bouncing Writer Movement

The 26 remaining rules with elaborate growth exhibit bouncing writer movement. In these rules, the writer moves forward and backward along a one-dimensional track, which grows with successive traversals. The rule in this class with the simplest behavior is shown in Figure 15. This rule can be represented in a one-dimensional manner similar to Figure 11, except that this time the writer is confined to moving on the red and green edges.

It appears that the writer effectively performs a global rewrite operation every time it makes a traversal of the one-dimensional track it is confined to. Simulations suggest that the network present on time step $2^{n+3} + n$ (where $n \geq 0$) can be obtained by taking the network

present at time step $2^{n+2} + n - 1$ and then simultaneously replacing each vertex with a triangle that has no red or green edges interlinking its neighbors and is not part of the external face (see Figure 16).

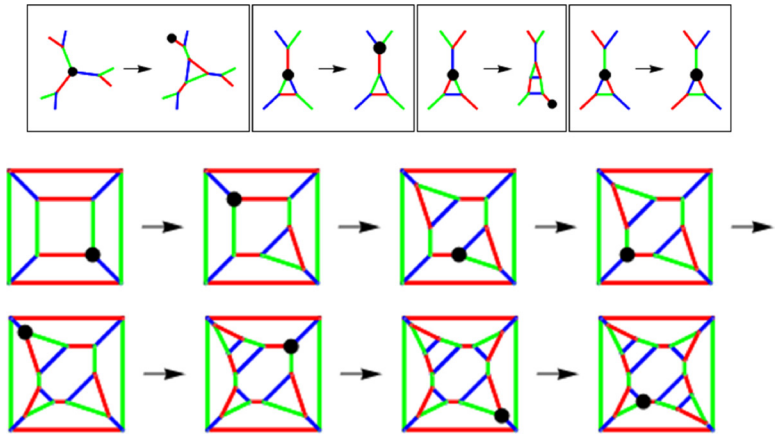

Figure 15. A rule with bouncing writer movement evolving for eight updates. The writer travels around the red and green edges of the track that start out as the central face of the cube. The writer keeps bouncing off the red edge at the bottom of this face and reversing its direction of movement.

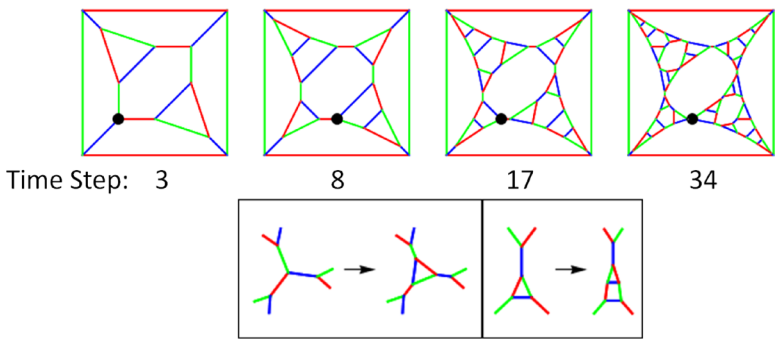

Figure 16. The network generated by the system with rules shown in Figure 15 on the first few time steps of the form $2^{n+2} - n - 1 : n \geq 0$. Between times of this form, the writer makes a complete traversal of its linear path and both rewrite operations shown at the bottom globally.

Not all of the rules with elaborate growth can be reduced to one-dimensional rewrite systems as directly as the examples we have con-

sidered, although the effectively one-dimensional nature of the writer's movement can be revealed by plotting the trail followed by the writer over several time steps (as on the left of Figure 14).

5. More General Rules

The behavior of colored trinets with more general rules does not always fall into the classes listed in Section 4. The set of 3888 rules we enumerated and previously discussed did not allow the writer to take any action when the edge between its neighbors is green. If we remove this restriction, we can find rules with four *active parts* such as the one shown in Figure 17. In this rule, the writer continues to move in a random-looking way (that is not "one-dimensional" as in the rules with elaborate growth discussed in Section 4) for at least the first 100 000 time steps. It is an open question whether this rule eventually settles into repetitive growth.

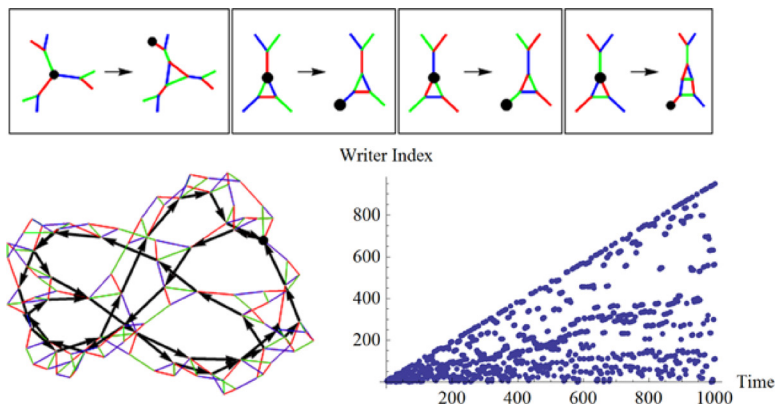

Figure 17. A rule with four active parts that has complex behavior. On the left, we show the structure obtained by evolving this rule for 100 time steps. The black arrows track the positions of the writer over the previous 45 time steps. On the right, we plot the index of the writer over the first 1000 time steps.

We can also consider rules that include more general kinds of rewrite operations, such as the one shown in Figure 18. In this system, triangles can be replaced with vertices. Unlike all of the systems in our previously considered rule set (which either has approximately linear growth rates or reached fixed points) the number of vertices in this case grows sublinearly over time.

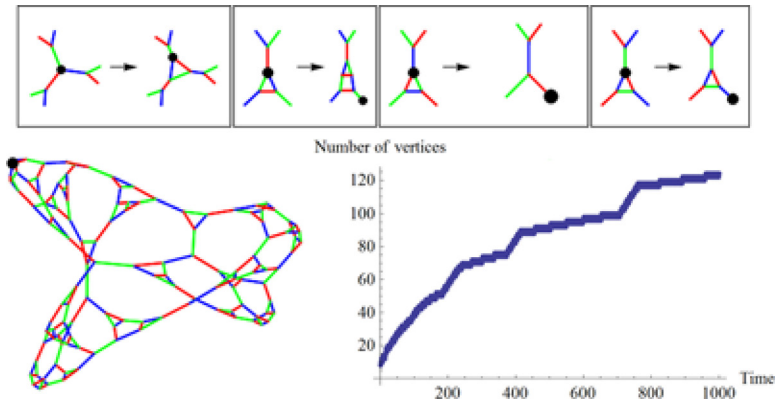

Figure 18. A system with more general rules that may involve replacing a triangle with a vertex. On the left, we show the network present on time step 1000. On the right, we plot the number of vertices over time.

When exploring these more general kinds of rules, many cases can be found with sublinear growth rates that exhibit repetitive or elaborate growth patterns. In many cases with sublinear growth, the number of vertices grows like the square root of the number of time steps elapsed. Other rules involving triangle shrinkage can lead to networks with shapes that oscillate over time. A trivial example is shown in Figure 19 (although rules exist with much higher period oscillations).

We can also consider rules including the so-called *exchange* operation, which rewires an edge [17]. In Figure 20, we show a rule that uses this type of exchange operation.

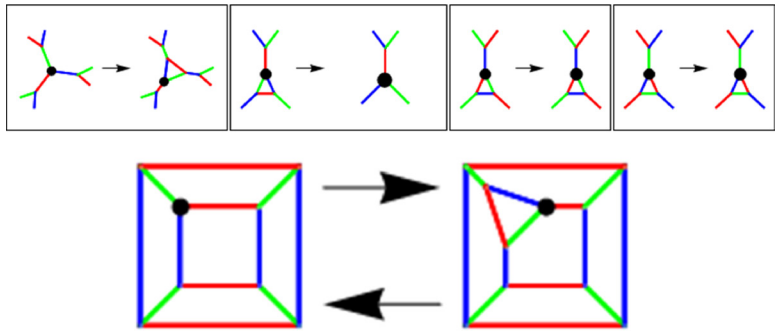

Figure 19. In this system, the network shape changes periodically.

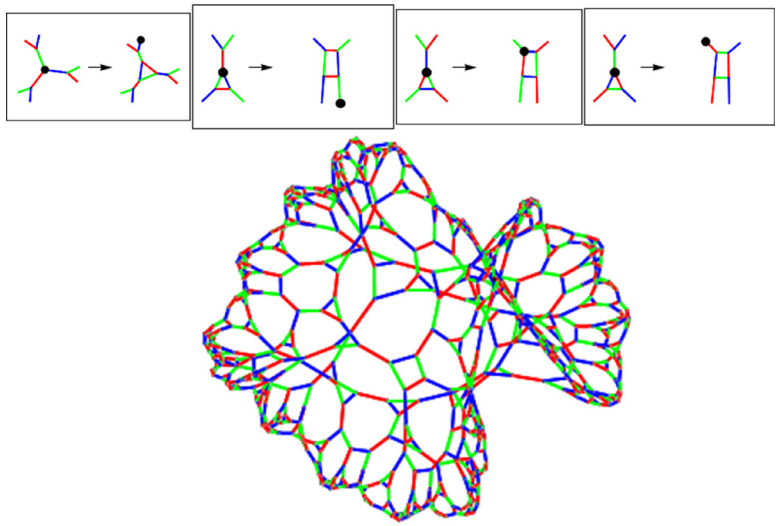

Figure 20. A system that includes the exchange rule, which effectively replaces a triangle with a square. We show the rule at the top. The structure is generated by running this system for 500 time steps.

6. Conclusion

We have explored many aspects of the behavior of colored trinet automata. We found that the systems in our initial rule space could be sorted into three classes—fixed, repetitive growth, and elaborate growth. We have shown that each class can be described in terms of writer movement. When we look at the dynamics of more general rules, we find other types of behavior. In particular, we find rules that generate sublinear growth, persistent complex behavior, and periodically changing networks.

As minimal models capable of growing complex structure, these systems should have some applications. The systems could be realized by creating a robot that pulls itself along chains, ropes, or silk and drops lines (like a spider) to triangulate vertices. This type of realization could perhaps be useful for weaving or construction work. The ability of these systems to produce exotic network structures with relative ease could make them useful in network design. In existential graph theory, there are many open questions as to whether large trinets exist with particular properties, and it will be interesting to see whether our systems can resolve any of these issues.

It will also be interesting to see if these systems have any applications in modeling. The way structures grow under repetitive growth is

reminiscent of the way plants grow. Many of the networks produced by rules with elaborate growth rules seem reminiscent of polygonal networks in cracked soil or foams. The general idea of having a single writer that rewires a complex network is reminiscent of brains and databases where the connectivity is altered to store information.

There many directions that this work can be taken in the future. We have explored the dynamics of many colored trinet automata and identified many kinds of behavior; however, a more general classification scheme is evidently required to deal with the systems we described in Section 5. Exploring more general rules will be exciting, and will surely yield systems with other interesting kinds of behavior and applications. There are also many unanswered questions about the systems presented here, for example: is there a simple formula for the growth rate of the system shown in Figure 14? Does the complex behavior in Figure 17 persist forever? How can rules be characterized according to the structural properties of the networks they produce?

Acknowledgments

This work is supported by the General Research Funds (Project Number 412509) established under the University Grant Committee of the Hong Kong Special Administrative Region, China.

References

[1] S. Wolfram, "Statistical Mechanics of Cellular Automata," *Reviews of Modern Physics*, **55**(3), 1983, pp. 601–644. doi:10.1103/RevModPhys.55.601.

[2] P. Pruskinkiewicz and A. Lindenmayer, *The Algorithmic Beauty of Plants*, New York: Springer-Verlag, 1990.

[3] K. Alligood, T. Sauer, and J. Yorke, *Chaos: An Introduction to Dynamical Systems*, New York: Springer-Verlag, 1997.

[4] D. Watts and S. Strogatz, "Collective Dynamics of 'Small-World' Networks," *Nature*, **393**, 1998 pp. 440–442. doi:10.1038/30918.

[5] A-L. Barabasi and R. Albert, "Emergence of Scaling in Random Networks," *Science*, **286**(5439), 1999 pp. 509–512. doi:10.1126/science.286.5439.509.

[6] A. Turing, "On Computable Numbers, with an Application to the Entscheidungsproblem," *Proceedings of the London Mathematical Society*, **242**, 1937 pp. 230–265.

[7] S. Cox, M. Vas, and D. Wearie, "Topological Changes in a Two-Dimensional Foam Cluster," *The European Physical Journal E: Soft Matter and Biological Physics*, **11**(1), 2003 pp. 29–35. doi:10.1140/epje/i2002-10126-9.

[8] P. Pina, J. Saraiva, L. Bandeira, and J. Antunes, "Polygonal Terrains on Mars: A Contribution to Their Geometric and Topological Characterization," *Planetary and Space Science*, **56**(15), 2008 pp. 1919–1924. doi:10.1016/j.pss.2008.09.020.

[9] T. Bolognesi, "Planar Trinet Dynamics with Two Rewrite Rules," *Complex Systems*, **18**(1), 2008 pp. 1–41. http://www.complex-systems.com/pdf/18-1-1.pdf.

[10] E. Gottlieb and K. Shelton, "Color-Induced Subgraphs of Grünbaum Colorings of Triangulations of the Sphere," *Australasian Journal of Combinatorics*, **30**, 2004 pp. 183–192.

[11] F. Comellas, "Complex Networks: Deterministic Models," *NATO Security through Science Series - D: Information and Communication Security*, Vol. 7, Amsterdam: IOS Press Ebooks, 2006 pp. 275–293.

[12] K. Morrow, T. Rowland, and C. Danforth, "Dynamic Structure of Networks Updated According to Simple, Local Rules," *Physical Review E*, **80**(1), 2009. doi:10.1103/PhysRevE.80.016103.

[13] R. Southwell and C. Cannings, "Games on Graphs that Grow Deterministically," in *Proceedings of the International Conference on Game Theory for Networks (GameNets 2009)*, Istanbul, New York: IEEE, 2009 pp. 347–356. doi:10.1109/GAMENETS.2009.5137420.

[14] R. Southwell and C. Cannings, "Some Models of Reproducing Graphs: I Pure Reproduction," *Applied Mathematics*, **1**(3), 2010 pp. 137–145. doi:10.4236/am.2010.13018.

[15] R. Southwell and C. Cannings, "Some Models of Reproducing Graphs: II Age Capped Vertices," *Applied Mathematics*, **1**(4), 2010 pp. 251–259. doi:10.4236/am.2010.14031.

[16] R. Southwell and C. Cannings, "Some Models of Reproducing Graphs: III Game Based Reproduction," *Applied Mathematics*, **1**(5), 2010 pp. 335–343. doi:10.4236/am.2010.15044.

[17] K. Tomita, H. Kurokawa, and S. Murata, "Graph Automata: Natural Expression of Self-Reproduction," *Physica D: Nonlinear Phenomena*, **171**(4), 2002 pp. 197–210. doi:10.1016/S0167-2789(02)00601-2.

[18] S. Wolfram, *A New Kind of Science*, Champaign, IL: Wolfram Media, Inc., 2002.

[19] E. Fredkin, "Five Big Questions with Pretty Simple Answers," *IBM Journal of Research and Development*, **48**(1), 2004 pp. 31–45. doi:10.1147/rd.481.0031.

[20] T. Bolognesi, "Causal Sets from Simple Models of Computation," *International Journal of Unconventional Computing*, **6**(6), 2010 pp. 489–524.

[21] A. Lamb, "Dense Graphs, Node Sets, and Riders: Toward a Foundation for Particle Physics without Continuum Mathematics," *Complex Systems*, **19**(2), 2010 pp. 115–130. http://www.complex-systems.com/pdf/19-2-1.pdf.

[22] H. Zenil, *Randomness through Computation*, Singapore: World Scientific, 2011.