Network automata and the functional dynamic network framework

David M.D. Smith[1],[*] Jukka-Pekka Onnela[1,2,3], Chiu Fan Lee[2], Mark Fricker[4], and Neil F. Johnson[5]

[1]*Saïd Business School, Oxford University, Oxford OX1 1HP, U.K.*
[2]*Physics Department, Clarendon Laboratory, University of Oxford, Oxford, OX1 3PU, U.K.*
[3]*Laboratory of Computational Engineering, Helsinki University of Technology, FIN-02015HUT, Finland*
[4]*Department of Plant Sciences, University of Oxford, Oxford, OX1 3RB, U.K. and*
[5]*Physics Department, University of Miami, Coral Gables, Florida FL 33124, USA.*
(Dated: March 11, 2008)

We introduce and define Network Automata, a generalization of Cellular Automata, which relates the topological evolution of a network to its structure. This framework is capable of replicating many familiar network models. We also introduce the Functional Dynamic Network framework for dealing with networks in which the topology evolves according to some specified microscopic rules and, simultaneously, there is a dynamic process taking place on the network that both depends on its structure but is also capable of modifying it. As such it is a generic framework for dealing with the types of systems in which network structure, dynamics, and function are interrelated. At the practical level, these frameworks allow for easy implementation of the microscopic rules involved in such systems. To demonstrate the Functional Dynamic Network in action, we develop a class of simple biologically inspired models of fungal growth.

I. INTRODUCTION

The framework of complex networks has proved very successful in the study of various interacting systems. In the network description, the interacting elements are depicted as nodes and the interactions between the elements are represented by links connecting the corresponding nodes. The science of complex networks has progressed very quickly in the last few years, and some excellent reviews have been written covering both the methodology and key results [1–3]. The strength of the complex network paradigm lies in its ability to capture some of the essential structural characteristics of interacting systems while disregarding the details of both the elements and their interactions. Consequently, the early complex network literature was almost exclusively focused on structural properties of networks.

In many out-of-equilibrium growing networks, the evolution at a given time is dependent on the nature of the network at that time as exemplified in the preferential attachment model of Barabási and Albert [4]. In this paper, we develop the Network Automata (NA) framework, which can be seen as a natural extension of the Cellular Automata (CA) framework [5]. We describe different variants of NA in Sections II and III which are illustrated schematically in Fig. 1. To demonstrate the versatility of NA itself, we show how some familiar network models can be recast in the NA framework in Section IV. The NA framework not only removes ambiguity at an implementation level through exhaustive specification of the microscopic ruleset employed, it also provides a platform for comparison between apparently different network algorithms.
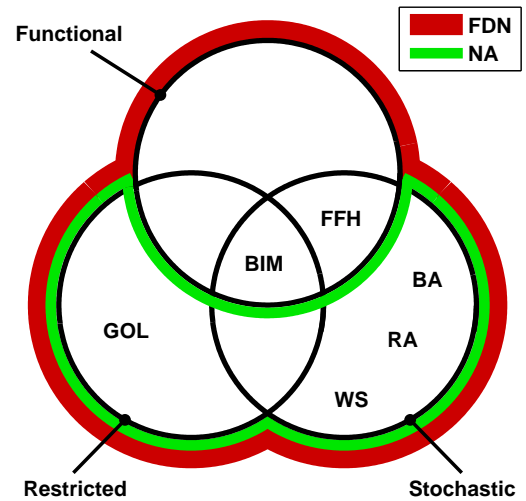

FIG. 1: (Color online) The functional, stochastic and restricted behaviors which can be encompassed by the FDN and NA frameworks. Also illustrated is the position of the Barabási-Albert (BA) network growth model, the random attachment (RA) network growth model, the Watts-Strogatz small world model (WS), the Game of Life cellular automata (GOL), the biologically inspired models (BIM) introduced in Section VI and the Self-Organised Criticality model of Fronczak, Fronczak and Holyst (FFH) [7].

While structural properties remain important in constraining the behavior of the system, there is a growing interest in dynamical processes taking place on networks [6]. Consider a situation in which the topology of a network evolves while there is simultaneously some process taking place on it. While its topology constrains the type

[*]Electronic address: d.smith3@physics.ox.ac.uk

of dynamics that may unfold on it, the dynamical process may be able to modify the topology of the network, meaning that its structural properties are coupled to its function. A real-world example might be the growth of transport links within a city. The dynamics of the human population within this network in turn affect the reinforcement and evolution of the those transport links and the feedback process is apparent.

There have been several clever attempts in the literature to inter-relate a network's structure, dynamics, and function – see, for example, the specific Self-Organised Criticality model of Fronczak, Fronczak and Holyst (FFN) [7] and the model of Zimmerman *et al* [8]. However, the present paper distinguishes itself by providing a *generic* framework for dealing with these types of systems. We call this the Functional Dynamic Network (FDN) framework and introduce it in detail later on in the paper. Its broad contribution to the network field lies in the fact that it enables a *precise* specification of the microscopic rules underlying the structural and functional evolution of a given network-based system, thereby enabling comparison between different classes of network.

II. NETWORK AUTOMATA

Consider an arbitrary weighted or unweighted, directed or undirected network at some time t which is to be grown to some size N_{tot}. Whilst nodes might be added to the system at each time step, the network might be considered as being of size N_{tot} where at time t many of the the nodes have no links. Information regarding the network's topology is entirely encompassed within the adjacency matrix $\mathbf{A}(t)$ which is of dimension N_{tot}. The matrix holds information about which links exist, their direction and, perhaps, weights. One might consider the evolution of the network as a process that alters the elements within this adjacency matrix, updating the states of the possible $N_{tot}(N_{tot} - 1)/2$ links which could exist in the system. We base a framework of network growth around this concept. If the microscopic ruleset governing the network's evolution is solely related to quantities which can be derived from the network's current topology (and hence from $\mathbf{A}(t)$) then the evolution of the network can be expressed in terms of some operation F acting upon the adjacency matrix:

$$\mathbf{A}(t+1) \ = \ F\big(\mathbf{A}(t)\big). \tag{1}$$

Thus, the update and consequential evolution of the network is entirely governed by the network itself [16]. **Definition:** *The Network Automaton (NA) is a network whose microscopic ruleset governing its topological evolution is determined by its own current topology* [17]. This ruleset could relate to any property of the nodes (their degree, betweeness, clustering and so on) or the links (weights or direction).

This concept is be achieved in practice by visiting all possible $N_{tot}(N_{tot} - 1)/2$ links within the adjacency matrix $\mathbf{A}(t)$ for a network comprising N_{tot} nodes (whether part of a component or not) once every time step. The update as to the nature of the link at the next time step (its existstence or its weight or direction) is then prescribed by the ruleset [18]. This process is analogous to the update of the state of a cell within a conventional Cellular Automaton [9]. To qualify as a Network Automata, the ruleset can only relate to quantities derived from $\mathbf{A}(t)$. This update then generates $\mathbf{A}(t + 1)$. This link-orientated update is a generic description of a dynamic network and all the essential features of that network's evolution are then contained within the exhaustive ruleset. No restriction has yet been made as to the directionality or weight of links.

III. RESTRICTED NETWORK AUTOMATA

Having defined Network Automata we shall now explore a simple example that can encapsulate some familiar Cellular Automata (CA) behavior [9]. We will impose some constraints such that this example is a small subclass of the NA framework. First, we shall look at the undirected graph with unweighted links such that the state of a link at time t can be described as the element $A_{i,j}(t)$. The ruleset governing the system's evolution determines the state of this link at the next time step $A_{i,j}(t+1)$. At the start of the update process, we let the information that the ruleset can act upon be simply the current state of the link and the degrees of the two nodes which it could possibly connect namely $k_i(t), k_j(t)$. We impose some simple rules to govern the evolution of the system, namely, that the state of a link at time $t + 1$ is a function of the combined degree of the adjacent nodes $k_i(t) + k_j(t)$ at the beginning of the time step and its own state $A_{i,j}(t)$.

We now restrict the network to an underlying lattice, $\mathbf{U}$ such that only those links that exist within the underlying structure can be formed in $\mathbf{A}$. Note that the underlying lattice is undirected such that $U_{i,j} = U_{i,j}$. For visual clarity, we shall make this an undirected degree 4 lattice with cyclic boundary conditions. Clearly this construction fulfills the criterion of being a Network Automaton, but since it is restricted to an underlying static network $\mathbf{U}$, we call it a Restricted Network Automaton (RNA) and this is illustrated in the schematic of Fig. 1. The evolution of the state of a specific link can be described as some operation

$$A_{i,j}(t+1) \ = \ F\Big(A_{i,j}(t), U_{i,j}, \sum_j A_{i,j}(t) + \sum_i A_{i,j}(t)\Big) \tag{2}$$

As an example of the RNA framework, let us construct the rules of the game in the nomenclature of the "life-like" CA models [9]. The survival and birth of a link is determined according to the value of $k_i(t) + k_j(t)$. We can express a ruleset for the survival process of the link in terms of a number (or a set of numbers) x_s such that if $k_i(t) + k_j(t) = x_s$ and the state of the

link $A_{i,j}(t) = 1$, then $A_{i,j}(t+1) = 1$ and is zero otherwise. Likewise, for link birth, we have a number x_b such that if $k_i(t) + k_j(t) = x_b$ and $A_{i,j}(t) = 0$, then $A_{i,j}(t+1) = 1$. These rules are conventionally expressed "survival set/birth set", and denoted as x_s/x_b. The rules in this particular example relate the number of neighboring links to the future existence of a link. For example, according to rule "3/2" a link will survive if the combined degree of the two nodes it connects is 3 (x_s), and a non-existent link will be "born" if the combined degrees of the two nodes between which it might exist is 2 (x_b). This ruleset is given explicitly in Table I.

time t				time $t+1$	
$k_i(t)$	$k_j(t)$	$A_{i,j}(t)$	$A_{j,i}(t)$	$A_{i,j}(t+1)$	$A_{j,i}(t+1)$
1	1	0	0	$U_{i,j}$	$U_{i,j}$
0	2	0	0	$U_{i,j}$	$U_{i,j}$
2	0	0	0	$U_{i,j}$	$U_{i,j}$
2	1	1	1	1	1
1	2	1	1	1	1
otherwise				0	0

TABLE I: The rules of game 3/2 on an arbitrary underlying network $\mathbf{U}$. A link will be born if it has 2 neighboring links and it must have only 1 neighbor to survive.

The top three rows of Table I refer to the birth of links and the next two refer to link survival. Clearly, if a link exists then the degree of the nodes at each end must be greater than zero. As such, the fourth and fifth lines of Table I cover all eventualities of link survival for this ruleset. The explicit inclusion of the (symmetric) underlying matrix $\mathbf{U}$ reflects the restricted nature of the automaton. Naturally, $k_i(t)$ and $k_j(t)$ can be expressed in terms of the network's adjacency matrix $\mathbf{A}(t)$ as the ith and jth element of $\mathbf{A}(t)\,\underline{1}$ where $\underline{1}$ is a vector with all elements equal to one and of dimension N_{tot}. More concisely, in terms of some operation F

$$\mathbf{A}(t+1) = F\big(\mathbf{A}(t), \mathbf{U}\big). \qquad (3)$$

We can now observe the evolution of the automaton on some initial configurations of $\mathbf{A}(0)$ using the rule set 3/2 and underlying lattice of order 4. We can observe "blinkers" which are motifs which return to their original position after some period. There are also motifs that replicate themselves after a number of steps but are spatially translated as shown in Fig. 2. These are known as "spaceships" in the CA nomenclature because they propagate through the space. There are many other interesting configurations and many rulesets to explore, even with the order 4 underlying lattice. There are a number of "still lifes" (objects that remain unchanged), blinkers of long periods, and "puffers" (debris leaving spaceships) which have been found [10]. As yet, there are no known "guns" (objects that replicate like blinkers but emit spaceships in the process). Implementation on the order 5 lattice would enable any of the Game of Life type objects to be emulated within the NA framework [19].
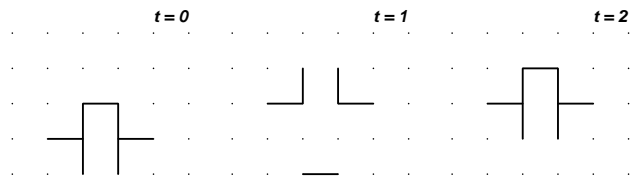

FIG. 2: A simple spaceship of period two in the 3/2 game with underlying lattice of order 4. Repeated application of the automaton rules will clearly perpetuate its motion.

IV. STOCHASTIC NETWORK AUTOMATA

We can also use the Network Automata framework to construct more conventional evolving networks. To do this we augment the NA framework, which consists of purely deterministic rules, by adding one or more stochastic rules and, thus, arrive at Stochastic Network Automata (SNA) as illustrated in Fig. 1. To implement SNA, we need two additional definitions. We denote the outcome of a Bernoulli trial with $\omega(x)$ defined as

$$\begin{cases} P(\omega(x) = 0) &= 1 - x \\ P(\omega(x) = 1) &= x. \end{cases}$$

We also define the Heaviside-like step function $\phi(x)$ as

$$\phi(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x \leq 0. \end{cases} \qquad (4)$$

Note that SNA can be restricted to a fixed underlying lattice (network) $\mathbf{U}$, resulting in Restricted Stochastic Network Automata (RSNA). We follow this approach in developing the biologically inspired model in Section VI, but emphasize that in this section there is no constraining underlying lattice structure imposed.

A. Random attachment model

The random attachment algorithm for building a single component network is very straightforward [12]. We consider a simple case in which at each time step a new node is connected to the existing network with one undirected link. Growing networks such as this one are known as non-equilibrium networks to distinguish them from equilibrium networks in which the number of nodes is constant [12]. The node in the existing network to which the new node is to be connected is chosen at random. The process is repeated until the required number of nodes is connected to the network. The conventional analysis for such a process is based on master equations [12]. Let there be $N(t)$ nodes at time t in the connected component of the network of which $X_k(t)$ are of degree k. For $k > 1$, the evolution of X_k can be written in terms of the probabilities of the new node connecting either to a node

degree $k-1$, thereby increasing X_k, or connecting to a k degree node, thereby reducing it. This can be written as

$$X_k(t+1) \ = \ X_k(t) + \frac{X_{k-1}(t)}{N(t)} - \frac{X_k(t)}{N(t)}. \qquad (5)$$

For $k = 1$, the new node (being of degree 1) will always increase $X_1(t+1)$, but through connecting to a degree 1 node in the existing network, it can decrease it too, such that

$$X_1(t+1) \ = \ X_1(t) - \frac{X_1(t)}{N(t)} + 1. \qquad (6)$$

We make a steady state approximation by assuming that the fraction of degree k nodes remains constant in time such that $X_k(t) = c_k N(t) \approx c_k t$, implying that $X_k(t+1) - X_k(t) \approx c_k$. It is interesting to note that the justification for treating the discrete process in this continuous manner is based on the assumption that, on average, one new node is connected to the network per time step with one undirected edge. The recurrence relation for the degree distribution of the system can be now written as

$$c_k = \frac{c_{k-1}}{2} \quad c_1 = \frac{1}{2}, \qquad (7)$$

which yields the degree distribution $P(k) = c_k = 2^{-k}$.

Next we will emulate this process in the SNA framework. Suppose we wish to grow the network to N_{tot} nodes, so that the adjacency matrix is of dimension N_{tot}. At each time step, we consider the update of all possible links in the network but only wish to update the links from nodes within the connected network component to nodes outside of it. Consequently, there are a total of $N(t)(N_{tot} - N(t))$ links that may be added, of which we wish that, on average, only one link will be added. Having identified those links which may be added, the required probability associated with one of them being added is $P(t) = [N(t)(N_{tot} - N(t)]^{-1}$.

time t				time $t+1$	
$A_{i,j}(t)$	$A_{j,i}(t)$	$\phi(k_i(t))$	$\phi(k_j(t))$	$A_{i,j}(t+1)$	$A_{j,i}(t+1)$
0	0	0	0	0	0
0	0	0	1	$\omega(P(t))$	$A_{i,j}(t+1)$
0	0	1	0	$\omega(P(t))$	$A_{i,j}(t+1)$
0	0	1	1	0	0
1	1	1	1	1	1

TABLE II: The stochastic rules to replicate random attachment network growth. The resulting distribution is shown in Fig. 3.

The explicit rules for this particular SNA which replicates random attachment are expressed in Table II. They state that the link can only be born if the node i is already part of the component and node j is not in the component (or vice-versa) *and* $\omega(P(t)) = 1$. If the link already exists, it stays. We seed the automaton with initial configuration of $A_{1,2}(0) = A_{2,1}(0) = 1$ reflecting a single component of 2 nodes. All links are only considered once in the update stage and both $A_{i,j}(t+1)$ and $A_{j,i}(t+1)$ are simultaneously updated [20]. The comparison of the degree distribution of the network generated by the SNA to that of the master equation analysis is shown in Fig. 3.
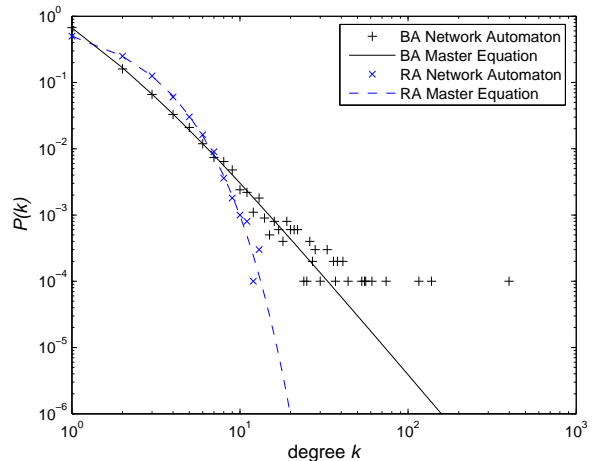

FIG. 3: (Color online) The resulting degree distributions $P(k)$ for the random attachment (RA) model and the preferential attachment (BA) model. The analytical results obtained using the master equation approach for these models are $P(k) = 2^{-k}$ and $P(k) = 2m(m + 1)/[k(k+1)(k+2)]$ respectively [1], with here $m = 1$. These are plotted with lines. Superimposed, are the corresponding distributions for one realization of the SNA, grown to only $10,000$ nodes. Clearly the two approaches are consistent.

It is clear from the binomial process governing the addition of new links (and nodes) to the existing component of the network that, on average, one new link and one new node are added, although clearly more than one new node could be attached with more than one new link allowing loops to be formed.

B. Barabási-Albert model

The SNA framework can also emulate a preferential attachment model such as that by Barabási and Albert [4]. In the BA-model the probability for a new node to attach to an existing node is proportional to the degree of the existing node, i.e., the attachment probability is a linear function of the node degree. This attachment mechanism is achieved in the SNA framework by simply modifying the probability of link birth, and the ruleset for this process is presented in Table III, where $K(t) = \sum_i k_i(t)$ denotes the sum of degrees over all nodes in the network.

The analytical result obtained using the master equation approach is given by $P(k) = 2m(m + 1)/[k(k+1)(k+2)]$ where m is the (fixed) degree of the

time t				time $t+1$	
$A_{i,j}(t)$	$A_{j,i}(t)$	$\phi(k_i(t))$	$\phi(k_j(t))$	$A_{i,j}(t+1)$	$A_{j,i}(t+1)$
0	0	0	0	0	0
0	0	0	1	$\omega\left(\frac{k_j(t)}{K(t)(N_{tot}-N(t))}\right)$	$A_{i,j}(t+1)$
0	0	1	0	$\omega\left(\frac{k_i(t)}{K(t)(N_{tot}-N(t))}\right)$	$A_{i,j}(t+1)$
0	0	1	1	0	0
1	1	1	1	1	1

TABLE III: The stochastic ruleset needed to replicate preferential attachment network growth. A small simulation distribution is shown in Fig 3.

new node entering the network [1]. Asymptotically this leads to $P(k) \sim k^{\gamma}$ with $\gamma = 3$. The analytically obtained distribution is plotted in Fig. 3 together with the corresponding distribution obtained from one realization of an SNA simulation. The match between the two approaches is very good.
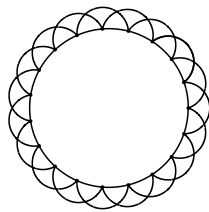
C. Watts-Strogatz model

FIG. 4: The initial network configuration of the Watts Strogatz model for $k = 4$.

We have demonstrated how non-equilibrium growing networks can be generated within the SNA framework. Here we turn to equilibrium networks and discuss the rules necessary to generate small world networks in the manner of Watts and Strogatz [13]. The WS model starts from considering a one-dimensional lattice comprising N nodes with all nodes having the same degree k (through connections to nearest neighbors, then next nearest neighbors etc) and cyclic boundary conditions [21]. An initial configuration for $k = 4$ is shown in Fig. 4, as an example. Each link in the network is visited and it is rewired with probability p. The original rewiring mechanism was such that one end of the link remained where it was and the other vertex was chosen at random from the rest of the network. In practise, the addition of shortcuts is the important aspect of this model so we choose a slightly simpler mechanism such that both ends of the rewired link are chosen at random. We aim to emulate this modified process within the SNA framework because it results in somewhat simpler ruleset than the original model.

Initially, there are $Nk/2$ links within the system. The expected number which are to be rewired is $pNk/2$. This process might be considered "link death". The number of nodes remains constant, and we wish the number of links to remain constant too. The expected number of links to be born is therefore set equal to the expected number of links which are removed. This is consistent with the notion of link rewiring in the model of Watts and Strogatz although we note that the number of links removed and those added are not necessarily equal. Assuming no loops of length one (melons) such that $A_{i,i} = 0$, the total number of links in the system that are not alive (and therefore capable of being born) is

$$\frac{N(N-1)}{2} - \frac{Nk}{2}. \qquad (8)$$

We can then describe the time-independent birth probability of links by

$$P(t) = p_b \; = \; \frac{kp}{(N-k-1)}. \qquad (9)$$

The ruleset for this system, which is is given in terms of the link removal probability p and the link birth probability p_b, is shown in Table IV. In an actual implementation of the model, one runs through all possible links, whether they exist or not, and updates their state according to the rules. The result of applying this ruleset is shown in Fig. 5, depicting the normalised clustering coefficient and mean shortest path for the networks generated. It is interesting to note that this implementation of the SNA requires only one time step. Successive applications yield networks comparable to the classic random graphs of Erdös and Rényi [11] in that all links will eventually be rewired and replaced randomly.

time $t=0$		time $t=1$	
$A_{i,j}(0)$	$A_{j,i}(0)$	$A_{i,j}(1)$	$A_{j,i}(1)$
1	1	$1 - \omega(p)$	$A_{i,j}(1)$
0	0	$\omega(p_b)$	$A_{i,j}(1)$

TABLE IV: The simple stochastic ruleset for replication of the Watts and Strogatz small world network model. The top row represents the removal of a link with probability p and the bottom, the rewiring of a link with probability p_b as defined in equation 9. The network is undirected, yielding a symmetric adjacency matrix $\mathbf{A}$.

V. FUNCTIONAL DYNAMIC NETWORK FRAMEWORK

Consider a situation in which the topology of a network evolves while there is simultaneously some process taking place on the network. At any given time the topology of the network constrains the type of dynamics that may unfold on it. However, the dynamical process may be
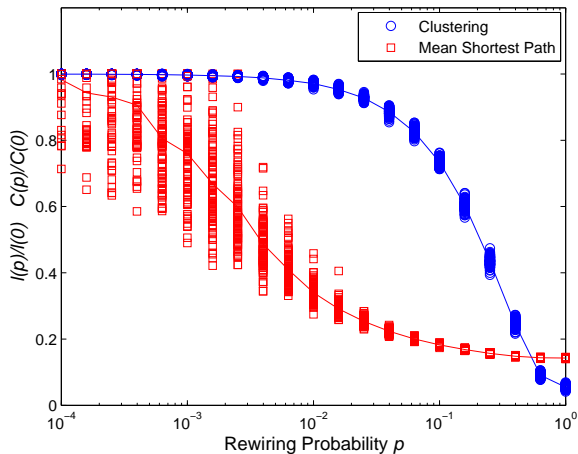
FIG. 5: (Color online) The normalised clustering coefficient and average shortest path of the networks generated using the Watts and Strogatz small-world mechanism implemented in the Network Automata framework. Here, $k = 10$ and $N = 400$. Each marker represents a simulated network, with the line being the mean values over 100 simulations. The normalising coefficients $C(0)$ and $l(0)$ are the clustering coefficient and average shortest path of the network prior to any rewiring.

able to modify the topology of the network, meaning that its structural properties are coupled to its function and vica-versa. A related real-world example is the growth of a city, where more affluent, well-connected areas are more prone to further expansion. A similar situation may arise in the context of social networks, where one's current social opportunities and dynamics are limited by the existing network structure, but they can be widened by extending the network.

The ruleset governing the topologival update process relates not only to network related quantities but also functional aspects of the nodes and/or links. This leads to the definition of the Funcitonal Dynamic Network. **Definition:** *A Functional Dynamic Network (FDN) is a network whose topological evolution is governed by a funtional process taking place upon the network as well as its own current topology.* Clearly, Network Automata are a subclass of the FDN as illustrated in the schematic of Fig. 1.

Since the functional process requires a network on which to perform, we can decouple the evolution of the network into two distinct phases, namely, that affecting its topology and that governing the functional process. Writing the functional information at some time t as a matrix $\mathbf{S}(t)$, the formal description of the evolution can be expressed in terms of some operators F and G as

$$\mathbf{A}(t+1) = F\big(\mathbf{A}(t), \mathbf{S}(t)\big)$$
$$\mathbf{S}(t+1) = G\big(\mathbf{A}(t+1), \mathbf{S}(t)\big). \qquad (10)$$

This expression states that the network evolves accord-

ing to some process, which is determined by its own current topology $\mathbf{A}(t)$, *and also* by some attributes of its nodes and links that includes function-based information $\mathbf{S}(t)$ [22]. The functional process then occurs on this network to generate the new set of information $\mathbf{S}(t+1)$. The system is synchronously updated by running through all possible links between all pairs of nodes and updating the state of each link in accordance to the ruleset employed.

VI. BIOLOGICALLY INSPIRED MODEL

In this Section we construct three simple increasingly realistic models of woodland fungi [14] to demonstrate the versatility of the Functional Dynamic Network framework. To model the growth of the fungi and distribution of resources within it requires pooling together concepts from both the Restricted Network Automata (Section III) and the Stochastic Network Automata (Section IV). Although the model is biologically inspired, its aim is not to incorporate a large number biological details. Instead, we adopt a minimalist approach to emulate fungal growth from a small set of microscopic rules. We start from a biologically naive but mathematically simple model (Model **a**). We then modify the model according to some basic physical and biological considerations (Model **b**). The end product (Model **c**) of fungal growth, may serve as a platform for more elaborate future models, demonstrating the effectiveness of using the framework.
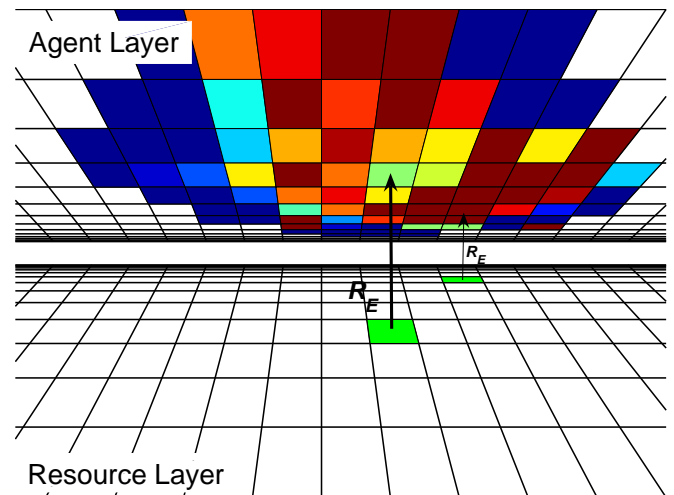

FIG. 6: (Color online) A biologically inspired multi-agent model whereby the agent layer is superimposed upon a resource layer. Agents above a resource can accumulate resources at some rate R_E

Consider a system of agents who might each be interpreted as a cell in a two dimensional lattice. The connectivity between agents is North, South, East and West reflecting a possible connectivity of 4, such that the agents

are restricted to local information as long range communication is assumed implausible in the biological system. The agent layer is superimposed on a resource layer as in Fig. 6. The rules of the system are very simple. If an agent is above a resource, it absorbs that resource at some rate R_E. The objective of each agent is two-fold: (1) become fully connected, and (2) redistribute excess resource. The process of becoming fully connected reflects growth into a neighboring cell. We allow each agent to grow only one new neighbor at a time step but only if the agent has resources to do so. To mimic active transport of resources, an agent passes resources to its neighbor cell provided that the neighbor does not pass resources to it. We now endeavor to categorize this simple multi-agent system it in the FDN framework. This serves not only to clarify any ambiguities that arise in the programming of a multi-agent system, but also as a potential aid to improving efficiency in that the required iteration and information storage/retrieval aspects are clearly defined by the rulesets imposed.

We start from a simple scenario (Model **a**) in which resources are neither consumed nor conserved. Let us first look at the growth (structural) stage. Each cell or agent represents a node and the boundary between two cells through which resource is passed is represented by a link, requiring a directed implementation of the RSNA. Consider that the information upon which the topological ruleset will act to update the state of a link in the network is simply the amount of resource that each of the two nodes has at each end of the link and their in and out degree. For clarity, instead of writing them in matrix form, we write them as distinct vectors such that $S_i(t)$ refers to the resource (functional variable) that agent (node) i has at time t. The element $k_i(t)$ is its total degree and $k_{o,i}(t)$ its out degree. The information available to the ruleset can be considered vectors of some dimension N_{tot} where N_{tot} is the number of nodes in the underlying lattice $\mathbf{U}$ that we take to be of order $d = 4$.

We will grow the FDN in an unweighted but directed adjacency matrix $\mathbf{A}$ such that if $A_{i,j} = 1$ the link exists and is directed from i to j, whereas if $A_{j,i} = 1$ the link exists and is directed from j to i. If neither $A_{i,j} = 1$ nor $A_{j,i} = 1$ then the link does not exist. Here $A_{i,j} = 1$ and $A_{j,i} = 1$ are mutually exclusive. The structural update process runs though all links by considering all nodes $i = 1 \rightarrow N_{tot}$ and for each value i through all possible neighbors $j = 1 \rightarrow i$ where $U_{i,j} = 1$. As such, each possible link between nodes i and j is considered once and the information $A_{i,j}$ and $A_{j,i}$ regarding its state and direction is updated at the same time.

We can now write the network update procedure for Model **a** in terms of an exhaustive truth table as in Table V. Note that if $A_{i,j}(t) = A_{j,i}(t) = 0$, the total degree of both nodes i and j is less than d. Also, the dependence on the underlying lattice is implicit and, as such, its cumbersome presence will not be explicitly included in the ruleset.

The resource distribution (functional) stage is slightly

more straightforward. We start by mapping the adjacency matrix $\mathbf{A}(t + 1)$ to a normalised transition matrix $\mathbf{T}(t + 1)$ such that an element $T_{i,j}(t + 1) = A_{i,j}(t + 1)/k_{o,i}(t + 1)$ for out degree of node i greater than zero at time $t + 1$.

We can write the update for the resource distribution process as

$$\underline{S}(t+1) \;=\; \mathbf{T}^{\dagger}(t+1)\underline{S}(t) + \underline{\xi}(t), \qquad (11)$$

where the vector $\underline{\xi}$ corresponds to agents who can accumulate resources. If we wish to impose the constraint that only "alive" (i.e. active) agents can accumulate resource through this process, then we write

$$\xi_i(t) \;=\; R_E \, \phi(S_i(t)) \, L_i \qquad (12)$$

where the vector $\underline{L}$ denotes the (binary) existence of resource at position of node (agent) i in the resource layer. This could be made time dependent (i.e. finite size resources) although here we will not consider this effect. For the example of Fig. 7, the amount of resources that
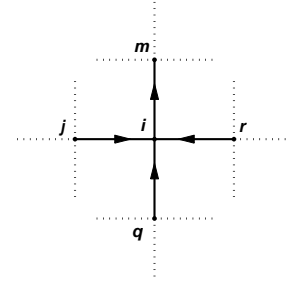

FIG. 7: The influx of resource into node labelled i. The amount this node receives from node j is related to how many out degrees node j has. This amount would be expressed as $S_j(t)/k_{o,j}(t + 1)$.

node i has at time $t + 1$ is

$$
\begin{aligned}
S_i(t+1) \;=\;& A_{j,i}(t+1)\frac{S_j(t)}{k_{o,j}(t+1)} + A_{m,i}(t+1)\frac{S_m(t)}{k_{o,m}(t+1)} \\
&+ A_{r,i}(t+1)\frac{S_r(t)}{k_{o,r}(t+1)} + A_{q,i}(t+1)\frac{S_q(t)}{k_{o,q}(t+1)} \\
\;=\;& T_{j,i}(t+1)S_j(t) \quad + \quad T_{r,i}(t+1)S_r(t) \\
&+ \quad T_{q,i}(t+1)S_q(t). \qquad (13)
\end{aligned}
$$

We can now observe the FDN in operation as shown in Fig. 8. Let us start with a single node η above a single food source such that at time $t = 0$ the agent has some resource. In the initial configuration the adjacency matrix is all zeros $A_{i,j}(0) = 0 \; \forall i, j$, and the resource information vector is all zeros except $S_\eta(0) = R_E$ such that the initial agent has amount R_E. The (static) resource accumulating vector $\underline{\xi}(t)$ is also all zeros except $\xi_\eta(t) = R_E \, \forall t$.

				Model **a** time $t+1$		Model **b** time $t+1$		Model **c** time $t+1$	
		time t							
$A_{i,j}(t)$	$A_{j,i}(t)$	$\phi(S_i(t))$	$\phi(S_j(t))$	$A_{i,j}(t+1)$	$A_{j,i}(t+1)$	$A_{i,j}(t+1)$	$A_{j,i}(t+1)$	$A_{i,j}(t+1)$	$A_{j,i}(t+1)$
0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	$\omega\left(\frac{1}{d-k_j(t)}\right)$	0	$\omega\left(\frac{1}{d-k_j(t)}\right)$	0	$\omega\left(\frac{g}{d-k_j(t)}\right)$
0	0	1	0	$\omega\left(\frac{1}{d-k_i(t)}\right)$	0	$\omega\left(\frac{1}{d-k_i(t)}\right)$	0	$\omega\left(\frac{g}{d-k_i(t)}\right)$	0
0	0	1	1	$\omega\left(\frac{1}{2}\right)$	$1-A_{i,j}(t+1)$	$\omega\left(\frac{1}{2}\right)$	$1-A_{i,j}(t+1)$	$\omega\left(\frac{1}{2}\right)$	$1-A_{i,j}(t+1)$
0	1	0	0	0	1	0	1	0	1
0	1	0	1	0	1	0	1	0	1
0	1	1	0	0	1	0	1	0	1
0	1	1	1	0	1	$\delta(k_{in,i},d)\omega\left(\frac{1}{d}\right)$	$1-A_{i,j}(t+1)$	$\delta(k_{in,i},d)\omega\left(\frac{1}{d}\right)$	$1-A_{i,j}(t+1)$
1	0	0	0	1	0	1	0	1	0
1	0	0	1	1	0	1	0	1	0
1	0	1	0	1	0	1	0	1	0
1	0	1	1	1	0	$1-A_{j,i}(t+1)$	$\delta(k_{in,i},d)\omega\left(\frac{1}{d}\right)$	$1-A_{j,i}(t+1)$	$\delta(k_{in,i},d)\omega\left(\frac{1}{d}\right)$

TABLE V: Different rulesets (the columns labeled 'time $t+1$') for three biologically inspired models: (**a**) the simplest scenario (Model **a**), (**b**) incorporating conservation of resources (Model **b**), and (**c**) implementing a delay factor (Model **c**). See Eq. A2 for the definition of δ and Appendix A for details.

We observe both the network and functional aspect of the system. The nodes (agents) are superimposed on the directed network, and the amount of resources a node has is indicated by its color ranging from blue (low concentration) to dark red (high concentration) [10]. Only nodes that have resources are included and the result is independent of the choice of R_E.

Note that under this ruleset and functional update stage resources are not conserved. As such, if an agent has no daughters, the resources held by the agent are "lost" in the functional update stage of the process. However, conservation and consumption of resources can also be incorporated by making further modifications to the functional update stage to construct Model **b**, a slightly more developed model. The details of these modifications are given in Appendix A and an example of the evolution is shown in Fig. 9. Emergent canalized flux channels are clear. Such channels have been observed experimentally in a wide class of real biological fungi [14].

A further development can be made to incorporate two different time scales in the model such that the growth and redistribution of resources take place at different rates, leading to Model **c** (see Appendix A for details). The ruleset of each progressively more detailed model is based on that of its predecessor as seen in Table V.

VII. CONCLUDING REMARKS AND DISCUSSION

In this paper we have developed the concepts of Network Automata (NA) together with its restricted (RNA) and stochastic (SNA) variants. This generic framework can encompass many familiar models. We have demonstrated its ability to reproduce Cellular Automata type models in Section III, and also non-equilibrium growing network models (random attachment, Barabási-Albert)
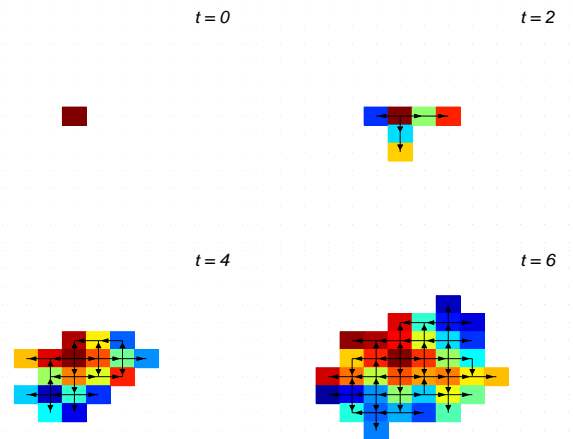

FIG. 8: (Color online) The evolution of the biologically inspired FDN. The colors represent the amount of resource $S_i(t)$ a node has (the functional aspect) superimposed on a directed network (the structural aspect).

and equilibrium (Watts-Strogatz) network models in Section IV. Whilst these pedagogical cases have comprised undirected, unweighted networks, these features can be incorporated within the framework.

The development of NA naturally leads to the introduction of the Functional Dynamic Network (FDN) framework, the main contribution of this paper, which couples evolution and function of complex networks by using simple microscopic rules at the level of nodes and links. We have demonstrated the practicability of the FDN framework by applying it to a class of biologically inspired models, which produce qualitatively similar structure to woodland fungi. The well defined and simple ruleset not only makes replication possible but also
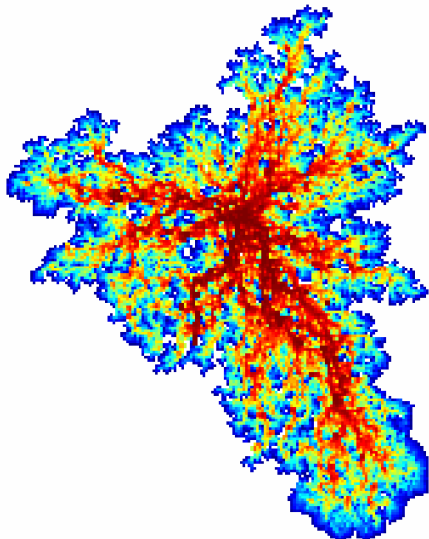
FIG. 9: (Color online) Incorporating conservation and consumption of resources. The simulation is seeded with one resource, one agent, $R_E = 20,000$, $R_C = 1$ and after 500 time steps there are 15,984 agents. The simulation used the ruleset of Model **b** of Table V, and the functional update is that of Eqs. A1, A3, A4. The lattice is of length 300.

aids implementation at the programming level. In the study of emergent phenomena involving networks, this framework enables concise and clear establishment of microscopic rules. One can easily think of more complex rulesets to more accurately model a real system, such as adding transport costs or finite resources, both of which can easily be accomplished, or even a time-dependent ruleset. In related biological systems the observations are usually restricted to the functional part of the network, as might be obtained by radioactive labeling of food sources. In designing models for such systems it seems sensible to retain this functional information not only because that function might be inherent to its development but also for comparative purposes. It is then interesting to ask what level of complexity is required to more accurately model real biological systems.

One can envisage applying the FDN framework also to discrete differential equation modelling and Diffusion Limited Aggregation (DLA) systems and, indeed, to any system in which the dynamics of network topology is related to the function performed thereon. There are certain biological systems which can quasi-solve increasingly complex problems in a constant time [15]. Given that it might be possible to model these systems within the FDN framework, it might suggest how to design a hardware based implementation to perform the similar calculations in constant time. It is interesting to pose the question as to what kind of problems could be solved by such a system and how complex the microscopic rules

would be for a given problem. This would reflect the minimum length of ruleset that would have to be employed by such a system in both the network and functional update stages. The physical replication into a digit system, and the length of the information processes thereon, alludes to a measure for the complexity of the system.

APPENDIX A: CONSERVATION AND CONSUMPTION OF RESOURCES

We can incorporate conservation of resources in the model by modifying the update stage of the functional process. This requires rewriting the transition matrix as

$$
T_{t+1}(i,j) = \begin{cases} \frac{A_{t+1}(i,j)}{k_{o,i}(t+1)} & \text{for } k_{o,i}(t+1) > 0 \\ 0 & \text{for } k_{o,i}(t+1) = 0 \end{cases}
$$
$$
T_{t+1}(i,i) = \begin{cases} 0 & \text{for } k_{o,i}(t+1) > 0 \\ 1 & \text{for } k_{o,i}(t+1) = 0. \end{cases} \tag{A1}
$$

Note that the above modification allows an agent to accumulate resource indefinitely if its in-degree is equal to the degree of the underlying lattice, i.e. $k_i(t) = d$. This undesirable feature can be overcome by having this agent flip the direction of, on average, one of its d links. To implement this we make use of the Kronecker delta function defined as

$$
\delta(x,y) = \begin{cases} 0 & \text{for } x \neq y \\ 1 & \text{for } x = y. \end{cases} \tag{A2}
$$

We can also incorporate consumption of resources in the model by making further modifications to the functional update stage, resulting in Model **b**. If an agent has more than some residual consumption amount, then R_C, the rate of consumption, is deducted from it. If the agent has less than this value, all of that agent's resources are removed such that the agent might be considered dead. The agent now distributes resources according to the number of its out-links if it has any, or retains its resources if it has none according to

$$
\underline{S}'(t+1) = \mathbf{T}^\dagger(t+1)\underline{S}(t), \tag{A3}
$$

where $\underline{S}'$ is the resource of every node prior to consumption. The transition matrix is as defined in Eq. A1. We can then write the update for $\underline{S}$ as

$$
\begin{aligned}
S_i(t+1) = {}& \phi(S_i'(t+1) - R_C)(S_i'(t+1) - R_C) \\
& + R_E \, \phi(S_i(t)) \, L_i,
\end{aligned} \tag{A4}
$$

which makes use of the step function of Eq. 4. Only agents (nodes) active in the network can accumulate resources from the resource layer. The effect of this "cost" of living clearly limits the potential size of the system.

In previous models, the physical transport of nutrient is of comparable speed to that of the growth, which is clearly not reasonable for most biological systems. To introduce two different time scales in the model, we can

delay the growth process by introducing a parameter g to the stochastic growth terms in the ruleset, such that an agent grows on average one neighbor every $1/g$ time steps. This is depicted explicitly in the ruleset of Model **c** of Table V, specifically in rules 1 and 2. The result of such a delay is the build up of resource towards the periphery of the object as one might expect (not shown).

[1] R. Albert and A.-L. Barabási, Rev. Mod. Phys **74**, 47 (2002).

[2] M.E.J. Newman, SIAM Rev. **45**, 167 (2003).

[3] S.N. Dorogovtsev and J.F.F. Mendes, Adv. Phys. **51**, 1079 (2002).

[4] A.L. Barabási and R. Albert, Science **286**, 509 (1999).

[5] S. Wolfram, *A New Kind of Science* (Wolfram Media, 2002).

[6] A.E. Motter, M.A. Matas, J. Kurths, and E. Ott (Eds.), Physica D **224**, 1 (2006).

[7] P. Fonczak, A. Fronczak and J.A. Holyst, Phys. Rev. E **73**, 046117 (2006).

[8] M.G. Zimmerman, V.M. Eguilez and M San Miguel, Phys. Rev. E **69**, 065102(R) (2004).

[9] J. von Neumann and A. W. Burks, *Theory of Self-reproducing Automata* (University of Illinois Press, Urbana, 1966).

[10] See http://www.physics.ox.ac.uk/users/smithdmd/FDN.pdf for details and examples.

[11] P. Erdös and A. Rényi, Publ. Math. Debrecen **6**, 290 (1959).

[12] S.N. Dorogovtsev and J.F.F. Mendes, *Evolution of Networks* (Oxford University Press, 2003).

[13] D. J. Watts and S. H. Strogatz, Nature **393**, 440 (1998).

[14] M. Tlalka, D. Hensman, P.R. Darrah, S.C. Watkinson, and M.D. Fricker, New Phytologist **158**, 325 (2003).

[15] T. Nakagaki, R. Kobayashi, Y. Nishiura, and T. Ueda, Proc. Biol. Sci. **271**, 1554:2305 (2004).

[16] This discrete time recurrence relation is more appropriate than the continuous time analogy $\frac{d\mathbf{A}}{dt} = f(\mathbf{A})$ in that it allows discontinous changes in the system's update, an inherent feature of network evolution.

[17] The concept of a Network Automaton has been alluded to before although not formally defined [5].

[18] Because each link can see the requisite information at the start of the time step, the system is synchronously updated. This implementation could then encompasses asynchronous updates through application over longer timescales.

[19] See http://mathworld.wolfram.com/Life.html for details on J.H.Conway's "Game of Life".

[20] This might be achieved in practice by running through all nodes $i = 1 \rightarrow N_{tot}$ and then potential neighbors $j = 1 \rightarrow i$ (the lower triangular portion of the adjacency matrix).

[21] Note that a two dimensional lattice of degree 4 would be inappropriate in that it has zero clustering.

[22] The discrete time recurrence relation is more appropriate than the continuous time coupled differential equation analogy $\frac{d\mathbf{A}}{dt} = f(\mathbf{A}, \mathbf{S})$, $\frac{d\mathbf{S}}{dt} = g(\mathbf{A}, \mathbf{S})$ in that it allows discontinous changes in the system's update.