VALIDATING AGENT-BASED MODELS OF LARGE NETWORKED SYSTEMS

Abhijin Adiga  Chris Barrett  Stephen Eubank  Chris J. Kuhlman
Madhav V. Marathe  Henning Mortveit  S. S. Ravi
Daniel J. Rosenkrantz  Richard E. Stearns  Samarth Swarup  Anil Vullikanti

Biocomplexity Institute and Initiative
University of Virginia
Charlottesville, VA 22904, USA

ABSTRACT

The paper describes a systematic approach for validating real-world biological, information, social and technical (BIST) networks. BIST systems are usually represented using agent-based models and computer simulations are used to study their dynamical (state-space) properties. Here, we use a formal representation called a graph dynamical system (GDS). We present two types of results. First we describe two real-world validation studies spanning a variety of BIST networks. Various types of validation are considered and unique challenges presented by each domain are discussed. Each system is represented using the GDS formalism. This illustrates the power of the formalism and enables a unified approach for validation. We complement the case studies by presenting new theoretical results on validating BIST systems represented as GDSs. These theoretical results delineate computationally intractable and efficiently solvable versions of validation problems.

1 INTRODUCTION

Validation and verification (V&V) are long sought after goals of every useful model (Forrester and Senge 1980; Robinson 1997; Kleijnen 1995; National Research Council 2008; Oreskes 2000; Oberkampf and Trucano 2002; Yilmaz 2006; Bharathy and Silverman 2013; Carley 1996). These two concepts can be easily defined. Model verification involves the assessment of the fact that the model implemented is actually what one intended to implement. Model validation is the assessment that the model, up to some measure of comparison, mimics the system it was intended to capture. It should be noted that a model is typically validated to some degree of fidelity or intended use (aka purpose).

BIST networks and agent-based models. In this paper, we focus on V&V issues as they pertain to networked biological, social, information and technical networks, also referred to as BIST networks. These systems lack inherent symmetries and are highly heterogeneous. Individual agents (represented as nodes in the network) are often selfish or adversarial and their behavior co-evolves in response to their perception of system dynamics. As a result, traditional methods rooted in physics and based on predictive validity are often not applicable. For one, in large simulations involving socio-technical networks, data matching exercises are usually postdictions of historical information such as matching epidemiological model output to an infection time series of a flu season. Although this is useful, it can also be misleading and is often inadequate. The configuration space (which captures the causal structure for an evolving system) is important for understanding the system being represented as well as the modeled representation of that system. However, any measured real world data is incapable of capturing this structural range—only those modes that took place in the real world appear in the measured data. Furthermore, this space is extremely large and is not enumerable in practice. Thus, the process of postdiction (retrospective validation) alone is inadequate. Additionally, while postdicting, the available information about the context is often insufficient to properly specify the initial and structural conditions that would allow the model to predict. As a result, high dimensional models are often fitted to relatively sparse data. In this sense, the occurrence of a fit can
be misleading because the inverse problem (i.e., the model matching the real data) does not have a unique solution. Predictive validity is also useful but again of limited value. Predictions based on past behavior that do not account for adaptive behavioral changes of individual agents often do not do well. Moreover, prediction alone does not provide insights into the underlying causal processes. For e.g., a simple time series model might provide accurate predictions of influenza dynamics but might have very little to say why a given season might have fewer cases than past few seasons. In both cases, explanatory power is really at issue. In decision-making, a causal basis for the choice of the best option is more relevant than any particular kind of detailed prediction of state. But, simply matching the model output with data collected in the field is of limited value, especially because such data is necessarily sparse, noisy, incomplete and not aligned in time.

Here we focus on V&V of agent-based models—such models have become popular for representing complex real-world BIST networks. Informally, such models are comprised of a collection of agents that represent the underlying BIST system at a certain level of granularity. Agents interact with other agents and modify their behavior as a result of the interactions. Usually the interaction is constrained by an underlying network that captures the agent neighborhood. Agent-based models are very expressive but come with two costs. First, computing all dynamical outcomes (the state-space) as specified by the agent-based models is usually expensive and as such one resorts to simulations. Second, the richness of agents and interaction structure is a strength but also leads to a more critical questioning and interrogation of such models as they pertain to believability and understandability. As a result, validation of agent-based models for BIST networks is even more important, and not surprisingly a subject of several papers (e.g., (Yilmaz 2006; Bharathy and Silverman 2013; Carley 1996)) in the last decade.

External and internal validation. Following (Carley 1996), we will use the term real data to mean information (including nominal and procedural information) gathered about the real system that is being modeled via experimental, field, survey or archival analysis. In this paper, we will focus on two distinct types of validation—internal and external. See (Yilmaz 2006; Bharathy and Silverman 2013; Carley 1996) for in depth discussions on various forms of validation. External validation aims to compare model output data with real life, in-situ and in-vivo measurements where the state-space data produced by the model are matched with measured data. Two key forms are: retrospective validity (matching with already measured data as used in machine learning for example) and predictive validity (matching with the outcomes that occur in the future). Furthermore, Carley (1996) classifies various levels of external validity, including: face, parameter, process, pattern, point and distributional validity. External validity connects model outcomes with observations pertaining to the real world problem that is being modeled. Internal (aka structural) validation aims to ensure that the model has been put together correctly. For agent-based models, this implies that interaction patterns (or networks), individual processes or rules for agents and model parameters are correct and adequate; furthermore, one ensures that the model is consistent with the specific and prevalent physical and social theories. In this sense, verification can be thought of as internal validation.

A mathematical framework. Here we use graph dynamical systems (GDSs) as a mathematical abstraction for agent-based models of BIST networks. Graph dynamical systems provide a powerful abstraction for a large class of BIST networks—these include transportation systems (Barrett et al. 2001), spread of epidemics (Eubank et al. 2004), systems biology (Shmulevich and Kauffman 2004), and immune systems (Alam et al. 2015). Such systems consist of a large number of interacting entities/agents, and the complex global dynamics of the system are the result of local interactions of each agent with its neighbors in an interaction structure, represented as a network. In this paper, we study a special form of GDSs, namely synchronous GDSs, where all agents compute and update their states in parallel. Such GDSs have been successfully used in a number of applications (Barrett et al. 2006). GDSs are universal, in that they can represent all other computing models, such as cellular automata (Wolfram 1987), Boolean networks (Ribeiro et al. 2007), neural networks, Hopfield networks, and graph automata; see (Barrett et al. 2007) for details. We will present the formal definitions associated with GDSs in Section 2.
We now define a formal model that enables us to develop rigorous formulations of issues related to validation of BIST systems. We follow (Barrett et al. 2006) in presenting these definitions. Let \( \mathbb{B} \) denote the Boolean domain \( \{0,1\} \). A synchronous Graph Dynamical System (GDS) \( \mathcal{S} \) over \( \mathbb{B} \) is specified as a pair \( \mathcal{S} = (G, \mathcal{F}) \), where (a) \( G(V,E) \), an undirected graph with \( n = |V| \), represents the underlying graph of the GDS, with node set \( V \) and edge set \( E \), and (b) \( \mathcal{F} = \{f_1, f_2, \ldots, f_n\} \) is a collection of functions in the system, with \( f_i \) denoting the local function associated with node \( v_i \), \( 1 \leq i \leq n \).

Each node (or agent) of \( G \) has a state value from \( \mathbb{B} \). Each function \( f_i \) specifies the local interaction between node \( v_i \) and its neighbors in \( G \). The inputs to function \( f_i \) are the state of \( v_i \) and those of the neighbors of \( v_i \) in \( G \); function \( f_i \) maps each combination of inputs to a value in \( \mathbb{B} \). This value becomes the next state of node \( v_i \).

At any time \( t \), the configuration \( C \) of a GDS is the \( n \)-vector \( (s'_1, s'_2, \ldots, s'_n) \), where \( s'_i \in \mathbb{B} \) is the state of node \( v_i \) at time \( t \) (\( 1 \leq i \leq n \)). In a GDS, all nodes compute and update their next state synchronously.

**Example:** Consider the graph of a GDS shown in Figure 1. Suppose the local transition functions at each of the nodes \( v_1, v_5 \) and \( v_6 \) is the OR function. The function at each of the nodes \( v_2, v_3 \) and \( v_4 \) is the 2-threshold function whose value is 1 iff at least two of the inputs are 1. Assume that initially, \( v_3 \) is in state 1 and all other nodes are in state 0. During the first time step, the states of nodes \( v_1, v_5 \) and \( v_6 \) change to 1 since each \( f_i \) \( (i \in \{1,5,6\}) \) is the OR function and each of these nodes has a neighbor (namely, \( v_3 \)) in state 1. Also, the state of \( v_3 \) changes to 0 since it requires at least two inputs with value 1. The states of \( v_2 \) and \( v_4 \) do not change; they continue to be 0. During time step 2, \( v_2 \) and \( v_3 \) change to 1 but \( v_4 \) remains at 0. Once the system reaches the configuration \( C = (1,1,1,0,1,1) \) at time step 2, it remains in that configuration forever; that is, \( C \) is a fixed point for this system.

**Additional dynamical systems terminology and notation.** If a given GDS can transition in one step from a configuration \( C' \) to a configuration \( C \), then \( C \) is a successor of \( C' \) and \( C' \) is a predecessor of \( C \). Since our local functions are deterministic, each configuration has a unique successor; however, a configuration may have zero or more predecessors. As mentioned above, a fixed point is a configuration \( C \) for which the successor is \( C \) itself. A configuration with no predecessors is called a Garden of Eden (GE) configuration.

The phase space \( \mathbb{P}_\mathcal{S} \) of a GDS \( \mathcal{S} \) is a directed graph defined as follows. There is a node in \( \mathbb{P}_\mathcal{S} \) for each configuration of \( \mathcal{S} \). There is a directed edge from a node representing configuration \( C \) to that representing configuration \( C' \) if there is a one step transition of \( \mathcal{S} \) from \( C \) to \( C' \). For a GDS with \( n \) nodes and a Boolean node state set (here, \( \{0,1\} \)), the number of nodes in the phase space is \( 2^n \); thus, the size of phase space

![Figure 1: An example of a GDS.](image)

<table>
<thead>
<tr>
<th>Initial Config.:</th>
<th>(0, 0, 1, 0, 0, 0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Config. at time 1:</td>
<td>(1, 0, 0, 0, 1, 1)</td>
</tr>
<tr>
<td>Config. at time 2:</td>
<td>(1, 1, 1, 0, 1, 1)</td>
</tr>
</tbody>
</table>
is exponential in the size of a GDS. Each node in the phase space has an outdegree of 1 (since our GDS model is deterministic). Also, in the phase space, each fixed point of a GDS is a self-loop and each GE configuration is a node of indegree zero. We use the symbol 0 (1) to denote a configuration in which every node is in state 0 (state 1).

**Some terminology regarding Boolean functions.** Throughout this paper, we will consider several classes of Boolean functions. We now define these classes. Given an integer $k$, a Boolean function $f$ is a $k$-threshold function iff $f$ is true when at least $k$ of its inputs are 1. A symmetric Boolean function (Crama and Hammer 2011) is one whose value does not depend on the order in which the input bits are specified; that is, the function value depends only on how many of its inputs are 1. Thus, any $k$-threshold function is symmetric. A symmetric function with $q$ inputs can be specified using a table with $q + 1$ rows, where row $i$ of the table specifies the value of the function when the number of 1-valued inputs is equal to $i$, $0 \leq i \leq q$. A Boolean function $f$ is $r$-symmetric (Barrett et al. 2007) if the inputs to $f$ can be partitioned into at most $r$ classes such that the value of $f$ depends only on how many of the inputs in each of the $r$ classes are 1. It can be seen that an $r$-symmetric function with $q \geq r$ inputs can be represented by a table with $O(q^r)$ entries. Note that any Boolean function with $d$ inputs is $d$-symmetric. We will be mainly concerned with $r$-symmetric functions where $r$ is a fixed integer. We say that a GDS is $r$-symmetric if each of its local functions is $r'$-symmetric for some $r' \leq r$.

We will explain in Section 5.1 how the GDS model can be used to formalize some research issues regarding validation of BIST networks.

## 3 SUMMARY OF CONTRIBUTIONS & RELATED WORK

### Summary of contributions.

In this paper, we describe an approach taken by our group to address issues as they pertain to validation of agent-based models of BIST networks. We present computational studies as well as theoretical results.

1. **Example case studies.** In Section 4, two case studies spanning a range of BIST networks are presented. Each study illustrates complementary types of validations as necessitated by the application. All of them use the GDS framework to represent the underlying agent-based models.

2. **Theoretical results.** We complement the discussion with new theoretical results in Section 5. Here, validation problems are formulated as model checking problems over GDSs and rigorous computational bounds are established. Each result compares properties of two BIST networks, represented as GDSs. One system can be considered as ground truth and the other as an inferred system built from data, and the goal is to evaluate the quality of the inferred system. An important practical consequence of these problem formulations and results is that if the similarity conditions do not hold between the two GDSs, then the inferred model does not fully capture the ground truth. Due to space considerations, only deterministic systems are considered for establishing rigorous results. These results are relevant for the following reasons. First, the results are applicable to real systems. The networked experiments of Centola (2010), Centola (2011), for example, seek to infer thresholds for humans in a health care setting. There are rigorous GDS representations for both the threshold model of Granovetter (1978) and the results of these aforementioned experiments that can be compared within our theoretical framework. Second, the results help us to delineate validation questions that are computationally intractable from those that can be solved efficiently. Finally, some of the results identify validation questions that can be answered in practice using open-source software SAT solvers (see, e.g., Gomes et al. (2008)).

### Related work.

Validation of complex systems has been a subject of extensive research in the modeling and simulation community; see (Balci and Nance 1985; Forrester 1971; Forrester and Senge 1980; National Research Council 1998; National Research Council 2008; Oberkampf and Trucano 2002; Robinson 1997; Sterman 2006). Philosophical discussions on these topics can be found in (Oreskes 2000; Carnap 1936; Popper 1963). The debate on validation can be summarized eloquently by a quote due to G. E. P. Box:
“Essentially, all models are wrong, but some are useful.” (Box and Draper 1987). As discussed above and in the recent reports of National Academies (National Research Council 1998; National Research Council 2008), validation is only meaningful when one states the purpose for which the model is used. Validation of agent-based models of BIST systems has also been actively studied. This includes our theoretical as well as practical work (Barrett et al. 2001; Eubank et al. 2004; Barrett et al. 2006; Barrett et al. 2007) as well as work done by others (Hahn 2013; Hahn 2017; Yilmaz 2006; Klügl 2008; Bruch and Atwell 2015; Bharathy and Silverman 2013; Bianchi et al. 2007; Lux and Zwinkels 2018; Macal 2016; Carley 1996). Additional references can be found in the above papers.

4 CASE STUDIES FOR MODEL VALIDATION

In this section, we describe two case studies to illustrate approaches for validating models of BIST networks. The first study develops models to capture human behaviors in a networked game. The second study models commodity flows among produce markets. The meanings of interactions vary from purposeful two-way communication, to crop flows, to unaided (i.e., human-oblivious) transmission, to passive observations of others’ actions. In all cases, the goal is to develop validated models to quantitatively predict system behavior for a wider range of conditions than were studied in experiments or obtained through observations.

4.1 Case Study 1: Validation of a Word Construction Model

An online game platform was designed and constructed for playing a word construction game. (Word construction games are often called anagram games. The board game Scrabble is one such example.) In our particular version of the game, (remote) players are recruited through Amazon Mechanical Turk (AMT) and play the game through their web browsers. Our game is motivated by the work of Charness et al. (2014). Each player \( v_i \in V \) \((n = |V|)\) is situated in a network \( G(V,E) \) of interacting players, where \( V \) is the set of players and \( E \) is the set of edges or communication channels \( \{v_i,v_j\} \in E \) between players \( v_i,v_j \in V \). Each player is assigned an initial set of alphabetic letters (called her own letters). Players work cooperatively to form words from their own letters and from those letters that they request, and then receive, from their distance-1 neighbors. In this way, players assist each other in forming more words, as a team, than individuals could produce on their own. A word corpus determines the validity of each word formed and submitted by players. The goal of the game is for the team to form as many words as possible over a 5-minute time window.

Observed data gathered from the games performed on this platform are the following actions and the time of each action by each player: (i) submit word; (ii) send letter request; (iii) receive letter request; (iv) send letter reply; and (v) receive letter reply. From these data, we built a set of action type-time sequence (ATTS) models that predict the time-ordered sequence of pairs (action type, time). Here, action types are three of the five actions identified immediately above: submit word, send letter request, and send letter reply. The ATTS model is the local function \( f_i \) for each \( v_i \in V \) (Section 2), and uses multinomial logistic regression that predicts each of the three action types, in time, for a player; see (Ren et al. 2018).

To evaluate our models, we use Kullback-Leibler (KL) divergence in the following way. We use initial conditions for each experiment of a particular class (e.g., the class of experiments where each player has degree \( d = 2 \) in the network, i.e., each player has two neighbors). We run the (agent-based) ATTS model, for each node, in time to simulate each game in the fashion shown in Figure 1. Thereafter, for each player, we sum up the counts of each of the five action types across all players in the computational games of a particular class. We form frequency distributions from these data—one for each action type—and compare these predicted distributions (which we convert to density distributions) against corresponding distributions from experimental data, using KL-divergence.
Figure 2 summarizes these ideas and some results for two models, M0 and M1, where M0 is based on a simple state transition matrix determined directly from the experimental data and M1 is based on a multinomial logistic regression model (Ren et al. 2018). The first two plots show results from experiments and from model predictions for \( d = 2 \) experiments, for the distributions of numbers of (letter) replies received and numbers of words formed. One can see by inspection that model M1 provides predictions (in distribution) that are in closer agreement with the data than are those from baseline model M0. This observation is made more formal with the KL-divergence values in Figure 2c. The lesser values of KL divergence for model M1 indeed confirm that this model is better than model M0, for not only replies received and form words, but also for the other three action types.

These results provide a combination of data and structural validation. Data validation is achieved through the favorable comparisons between model predictions and experimental results. At the same time, structural validation is achieved because our hierarchical logistic regression model (local function) generates the predictions. Note that structural validation does not preclude the possibility of other satisfactory formulations for local functions. Finally, these comparisons are self-consistency checks in that all data are used to build the models, and the predictions in Figure 2 are also over all data. Ten-fold cross validation is work in progress.

4.2 Case Study 2: Validation of a Model for the Spread of an Invasive Species

While trade and transport of goods is widely accepted as a primary pathway for the dispersal of invasive species that affect agricultural crops, these human mediated pathways are not well understood. Few, and often inaccurate, incidence reports and lack of knowledge of trade flows are some of the major hurdles in understanding their role. We modeled realistic spatio-temporal networks of seasonal agro-products between major markets (Venkatramanan et al. 2017) from diverse, multi-type, and noisy datasets. The methodology was applied to develop a spatio-temporal domestic tomato trade network in Nepal and investigate its role in the spread of the South American tomato leafminer or *Tuta absoluta*, a devastating pest of the tomato crop (Campos et al. 2017). Through dynamical analysis of the networks and a novel rank-based Bayesian inference approach, we showed that tomato trade has facilitated the rapid spread of the pest in the region.
We modeled the flow of agricultural produce among markets based on the following premise: major wholesale markets serve as key locations facilitating agricultural commodity flow, and the total outflow from a market depends on the amount of produce in its surrounding regions, and the total inflow is a function of the consumption linked to the market. These assumptions are driven by studies of the pest dynamics in other countries and the fact that tomato is a commercial crop in Nepal. The commodity flow is modeled as a temporal network with markets as nodes and directed weighted edges representing volumes of host crop being traded between the end points. The flows are estimated using a doubly constrained gravity model (Kaluza et al. 2010). The flow $F_{ij}$ from location $i$ to $j$ is given by $F_{ij} = a_i b_j O_i I_j f(d_{ij})$ where, $O_i$ is the total outflow of the commodity from $i$, $I_j$ is the total inflow to $j$, $d_{ij}$ is the time duration required to travel from $i$ to $j$, $f(\cdot)$ is the distance deterrence function, and coefficients $a_i$ and $b_j$ are computed through an iterative process to ensure flow balance.

For structural validation, we used yearly data from the largest wholesale market of Nepal. In Figures 3a–3c, we compare this data with the network flows. Given a set of network parameters $(\beta, \kappa, \gamma)$, we obtained the inflow from a particular district to Kathmandu as follows: We combined the weights of all edges of the corresponding network with destination node “Kathmandu” and source nodes belonging to that district. As seen in Figure 3a, for $\gamma$ values between 0.5 and 1, the flows from the networks are comparable to the Kalimati data except for two districts: Dhading (the top contributor) and Sarlahi (third highest). Upon further investigation we find that Dhading, which is a major producer west of Kathmandu, serves the Mid Hills and Terai regions of the Central Development Region in the flow networks (Figure 3b). While the gravity model predicts that these flows will be directly delivered to these regions, in reality, it is possible that Dhading’s produce is routed through the Kalimati market as there are several traders from Dhading registered in the Kalimati market. A similar argument holds for Sarlahi (Figure 3c).

Figure 3: Flow validation (from (Venkatramanan et al. 2017)).
may be different.) (ii) The nodes of each GDS are labeled using integers from 1 to $n$ so that the corresponding nodes in the two GDSs have the same number. (iii) The domain of state values for both the GDSs is $\{0, 1\}$. (iv) Each configuration in the phase space of one GDS corresponds to the same configuration in the phase space of the other GDS. Given two GDSs $S_1$ and $S_2$ satisfying the above assumptions, it is possible to formally represent some validation questions for the corresponding multi-agent systems using suitable similarity relationships between the two GDSs, expressed as logical predicates. We now present two examples of such predicates. (Several other examples are presented in (Adiga et al. 2019).) In Section 5.2, we discuss how such predicates are useful in studying validation issues for BIST networks.

(a) Let the predicate $PSE(S_1, S_2)$ be true iff the phase spaces of $S_1$ and $S_2$ are identical (i.e., for every configuration $C$, the successor of $C$ is the same configuration in both $S_1$ and $S_2$). We will refer to predicate $PSE$ as the phase space equivalence relationship.

(b) Let the predicate $CS-FP(S_1, S_2)$ be true iff the two GDSs have the same number of fixed points. A similar predicate $CS-GE(S_1, S_2)$ can be formulated for GE configurations. We will refer to $CS-FP$ and $CS-GE$ respectively as count similarity (CS) relationship with respect to fixed point and GE configurations.

5.2 Using Similarity Predicates for Validation

The literature on system verification (see, e.g., (Alur et al. 2015)), provides many examples where one wants to compare a model (e.g., a finite state machine) obtained from the specification of a system with one obtained from an implementation. The purpose of the comparison is to verify whether the implemented model correctly represents the specification model. This comparison is carried out by defining various forms of relationships between two models and determining whether those relationships hold. In this paper, we use a similar approach for validation. We use GDSs as models of both the ground truth and the implementation. As mentioned in Section 3, several studies reported in the literature (e.g., Centola (2010), Centola (2011)) use formalisms similar to GDSs. The relationships identified in Section 5.1 provide some ways by which the constructed model can be compared with the model obtained from ground truth. The most stringent of these relationships is phase space equivalence; when this relationship holds, the two systems become indistinguishable with respect to the desired behavior. Some of the relationships are likely to be easier to test in practice, especially due to the availability of tools such as public domain SAT solvers. When a relationship does not hold, it stands to reason that the inferred model does not fully capture the properties of the model derived from ground truth. When this happens, one can try to infer a better model using additional data and then proceed to validate the new model. In this manner, the similarity relationships identified in this paper serve as useful ways of carrying out validation.

5.3 Results for Phase Space Equivalence

Here and in the subsequent subsections, we consider the problems of determining the truth value of the predicates defined in Section 5.1. See (Adiga et al. 2019) for detailed proofs.

Recall that the predicate $PSE(S_1, S_2)$ is true iff the phase spaces of $S_1$ and $S_2$ are identical. We start with a general result on the complexity of the problem.

**Theorem 1** Given two GDSs $S_1$ and $S_2$ whose local functions are specified as Boolean expressions, the problem of finding the truth value of the predicate $PSE(S_1, S_2)$ is NP-hard. This result holds even if the underlying graphs of the two GDSs are identical.

**Proof (sketch):** We use a reduction from the Boolean Satisfiability problem (SAT) which is known to be NP-hard (Garey and Johnson 1979). Let an instance $I$ of SAT be specified using $p$ variables $X =
\{x_1,x_2,\ldots,x_p\}$ and $m$ clauses $Y = \{Y_1,Y_2,\ldots,Y_m\}$. We construct two GDSs $S_1$ and $S_2$ as follows. For both the GDSs, the underlying graph $G(V,E)$ is a star graph on $p+1$ nodes, with $V = \{v_0,v_1,\ldots,v_p\}$ and $E = \{(v_0,v_i) : 1 \leq i \leq p\}$. (Thus, the underlying graphs of the two GDSs are identical.) Let $s_i^j$ and $f_i^j$ denote respectively the state value of node $v_i$ and the local function at node $v_i$, $0 \leq i \leq p$, for GDS $S_1$. Likewise, let $s_i^2$ and $f_i^2$ denote respectively the state value of node $v_i$ and the local function at node $v_i$, $0 \leq i \leq p$, for GDS $S_2$. For $1 \leq i \leq p$ and $\ell = 1,2$, the local function $f_i^\ell$ at node $v_i$ is given by $f_i^\ell(s_0^i,s_1^i) = 0$. For $S_1$, the local function $f_0^1(s_0^1,s_1^1,\ldots,s_p^1) = 0$ and for $S_2$, the local function $f_2^2(s_0^2,s_1^2,\ldots,s_p^2) = \land_{j=1}^m Y_j$, with each variable $x_i$ in $Y_j$ replaced by the state variable $s_i^2$, $0 \leq i \leq p$ and $1 \leq j \leq m$. It is easy to see that this construction can be done in polynomial time. It can be verified that the predicate PSE $(S_1,S_2)$ is true iff the instance $I$ is not satisfiable.

We now mention a result that identifies that some special cases for which the PSE $(S_1,S_2)$ predicate can be evaluated in polynomial time. A proof the result appears in (Adiga et al. 2019).

**Theorem 2** The predicate PSE $(S_1,S_2)$ can be evaluated in polynomial time for the following special cases: (i) the local functions for both the GDSs are $r$-symmetric for a fixed $r$ and (ii) each local function in both the GDSs is linear (i.e., it is from the set \{XOR, XNOR\}).

When the degree of each node is bounded by a constant $d$, we note that every local function is $d$-symmetric. Thus, we obtain the following corollary from Theorem 2.

**Corollary 3** The predicate PSE $(S_1,S_2)$ can be evaluated in polynomial time if each of the underlying graphs has a degree of at most $d$, for a fixed $d$.

### 5.4 Results for Count Similarity

Here, we consider the count similarity relationships introduced earlier. Recall that the predicate CS-FP $(S_1,S_2)$ is true iff the the two GDSs $S_1$ and $S_2$ have the same number of fixed points. In proving the next theorem, we will use the following observation.

**Observation 1** Any GDS where each local function is NOR does not have any fixed point.

**Theorem 4** Let $S_1$ and $S_2$ be two given GDSs. The problem of evaluating the predicate CS-FP $(S_1,S_2)$ is NP-hard.

**Proof:** We will prove the result using a reduction from the fixed point existence (FPE) problem: does a given GDS $S$ have a fixed point? This problem was shown to be NP-complete in (Barrett et al. 2001). Let an instance of FPE be given by the GDS $S_1$. Let $S_2$ be the following GDS: the underlying graph of $S_2$ is the same as that of $S_1$ and the local function at each node of $S_2$ is the NOR function. By Observation 1, $S_2$ does not have any fixed point. Therefore, the predicate CS-FP $(S_1,S_2)$ is false iff $S_1$ has a fixed point. Thus, the problem of evaluating CS-FP $(S_1,S_2)$ is NP-hard.

We now present a result to show that the count similarity predicate can be evaluated efficiently for special cases.

**Proposition 1** Let $S_1$ and $S_2$ be two GDSs such that their underlying graphs are treewidth bounded\(^1\) and the local functions are $r$-symmetric for some fixed $r$. Then the predicate CS-FP $(S_1,S_2)$ can be evaluated in polynomial time.

**Proof:** It is shown in (Rosenkrantz et al. 2015) that if a GDS $S$ has bounded treewidth and each local function of $S$ is $r$-symmetric for a fixed $r$, the problem of counting the number of fixed points of $S$ can be

\(^1\)For the definition of treewidth, we refer the reader to (Bodlaender 1993).
solved in polynomial time. Using that algorithm, one can compute the number of fixed points of $S_1$ and $S_2$ and thus evaluate the predicate $\text{CS-FP}(S_1, S_2)$ in polynomial time.

6 SUMMARY, CONCLUSIONS AND FUTURE WORK

We observed that many V&V questions for MASs can be expressed using certain predicates on the GDSs corresponding to the MASs. We showed that many of these problems are computationally intractable in general. However, we observed that restricted versions of the problems can be solved efficiently. Also, some of the problems can be solved in practice using commercial or public domain SAT solvers. When the phase spaces of the GDSs modeling the given MASs are identical, the two MASs have identical global behavior. The other types of relationships defined in Section 5.1 provide useful ways of identifying certain 

**REFERENCES**


Adiga, Barrett, Kuhlman, Marathe, Mortveit, Ravi, Rosenkrantz, Stearns, Swarup, and Vullikanti


AUTHOR BIOGRAPHIES

Abhijin Adiga is a Research Assistant Professor in the Network Systems Science and Advanced Computing (NSSAC) Division of the Biocomplexity Institute and Initiative (BII) at the University of Virginia (UVA). His research interests include network science, modeling, combinatorial optimization, dynamical systems, machine learning and analysis of algorithms. Email: abhijin@virginia.edu.

Chris Barrett is the Executive Director of the BII and a Professor of Computer Science at the UVA. His research interests include modeling, simulation, dynamical systems, social science and network science. Email: ChrisBarrett@virginia.edu.

Chris J. Kuhlman is a Research Associate Professor in the NSSAC Division, BII at UV A. His research interests include modeling, simulation, software engineering, dynamical systems, computational social science and machine learning. Email: cjk8gx@virginia.edu.

Madhav Marathe is the Director of NSSAC Division, BII and a Professor of Computer Science at UVA. His research interests include modeling, simulation, machine learning, AI, data science, dynamical systems, network science, computational social science, and analysis of algorithms. Email: marathe@virginia.edu.

Henning Mortveit is an Associate Professor in the NSSAC Division, BII, and an Associate Professor of Engineering Systems & Environment at UVA. His research interests include rigorous methods to analyze and understand massively-interacting systems, modeling, simulation and software engineering. Email: Henning.Mortveit@virginia.edu.

S. S. Ravi is a Research Professor in the NSSAC Division, BII at UVA and a Distinguished Teaching Professor Emeritus of Computer Science at the University at Albany – State University of New York. His research interests include analysis of algorithms, data mining, dynamical systems, network science and fault-tolerant computing. Email: ssravi0@gmail.com.

Daniel J. Rosenkrantz is a Distinguished Institute Professor in the NSSAC Division, BII at UVA and a Professor Emeritus of Computer Science at the University at Albany – State University of New York. His research interests include analysis of algorithms, theory of computation, dynamical systems, machine learning, and fault-tolerant computing. Email: drosenkrantz@gmail.com.

Richard E. Stearns is a Distinguished Institute Professor in the NSSAC Division, BII at UVA and a Distinguished Professor Emeritus of Computer Science at the University at Albany – State University of New York. His research interests include analysis of algorithms, theory of computation, dynamical systems and game theory. Email: thestearns2@gmail.com.

Samarth Swarup is a Research Associate Professor in the NSSAC Division, BII at UVA. His research interests include multiagent systems, agent-based modeling, machine learning, AI and simulation analytics. Email: swarup@virginia.edu.

Anil Vullikanti is a Professor in the NSSAC Division, BII and a Professor of Computer Science at UVA. His research interests include analysis of algorithms, combinatorial optimization, machine learning, dynamical systems, network science and computational social science. Email: vsakumar@virginia.edu.