Generating Application-Specific Benchmark Models for Complex Systems

Jun Wang and Gregory Provan*
Department of Computer Science, University College Cork, Cork, Ireland
{jw8, g.provan}@cs.ucc.ie

Abstract

Automated generators for synthetic models and data can play a crucial role in designing new algorithms/model-frameworks, given the sparsity of benchmark models for empirical analysis and the cost of generating models by hand. We describe an automated generator for benchmark models that is based on using a compositional modeling framework and employs random-graph models for the system topology. We choose the system topology that best matches the topology of the real-world system using a domain-analysis algorithm. To show the range of models for which this approach is applicable, we demonstrate our model-generation process using two examples of model generation optimized for a specific domain: (1) model-based diagnosis for discrete Boolean circuits, and (2) E.coli TRN networks for simulating gene expression.

Introduction

Creating benchmark model suites is becoming increasingly important, as such models are needed to validate a variety of algorithms, in domains including VLSI design (where models are intended to meet particular specifications) (Stroobandt 2001), the Internet (Mahadevan *et al.* 2006), model-based diagnosis (Provan & Wang 2007), and bioinformatics (Van den Bulcke *et al.* 2006; Mendes, Sha, & Ye 2003). Given the sparsity of benchmark models and the cost of generating models by hand, it is critical to design an automated generator for synthetic models and data.

To satisfy this need, we describe a *domain-independent* automated generator for benchmark models that is based on using a compositional modeling framework and employs random-graph models for the system topology. Compositional modeling (Keppens & Shen 2001) is the predominant knowledge-based approach to automated model construction. It assumes that a system can be decomposed into a collection of components, each of which can be defined using a functional model. These component models are then integrated into the full system model using a system topology graph, which describes the component interactions. Although there are *domain-specific* synthetic generators for certain domains, e.g., (Stroobandt 2001; Van den Bulcke *et al.* 2006; Mahadevan *et al.* 2006), this is the first *domain-independent* generator that enables users to adopt particular topology-generation algorithms best suited to the particular application.

In this article, we assume we have a library of functional component models for the domain in question, so the main focus of benchmark generation is on creating ensembles of random but "realistic" topologies. A range of methods exist to generate system topologies, each of which has a set of specific input parameters that must be optimized to create a model that accurately depicts a domain-specific topology. As we will show, the different generation methodologies produce quite different models, with different topological properties, such as degree distribution, etc. Since each application domain requires different topological properties, the key to generating good benchmark models is to match the generation methodology to the domain requirements.

Our contributions are as follows.

1. We describe a domain-independent synthetic model generator that can tailor high-fidelity models to arbitrary domains based on domain properties.
2. We describe the domain-analysis process, specifying the set Π of parameters and the set Φ of metrics that must be computed and how Π and Φ are used to select the best topology-generation algorithm.
3. We illustrate the domain-analysis and model-generation procedure on two quite different domains: (1) model-based diagnosis of discrete Boolean circuits (where we compare the topological fidelity of the generated models to that of real circuit models), and (2) simulation of transcriptional regulatory networks based on synthetic gene expression data.

Related Work

The topology-generation method we adopt was originally developed based on the theory of random graphs and complex networks–see (Boccaletti *et al.* 2006) for background. However, this method focuses solely on the system structure (as captured by the graph), and ignores the system functionality. We extend this approach by adopting the system structure based on the random-graph generators, and then encoding system functionality using a component library.

*Both authors are supported by SFI grant 04/IN3/I524.

This work is most closely related to domain-specific model generators, which exist for circuits (Stroobandt 2001) and biological interaction kinetics models (Van den Bulcke *et al.* 2006). Our approach is different from either of these approaches, in that we make no prior assumptions about domain properties, but rather compute the domain properties necessary for model generation. Our model generation approach differs from related work in VLSI auto-generation, e.g., (Stroobandt 2001), in several ways. The VLSI approach emphasizes circuit design for circuit optimization and simulation after placement and routing; in contrast, our approach focuses on topological and organizational principles of circuits, and can be used for a wider variety of applications, including the diagnostics applications we report. Existing biological network generators, e.g., (Van den Bulcke *et al.* 2006; Mendes, Sha, & Ye 2003) use random-graph models that have unsuccessfully reflected the underlying structure of biological networks (Chung *et al.* 2003; Hormozdiari *et al.* 2007). Van den Bulcke et al.(2006) proposed an alternative topology generation approach for transcriptional regulatory networks (TRNs) by selecting subgraphs from previously described TRNs. Although this approach captures some topological characteristics of TRNs, it is not scalable and depends on the availability of accurate data of existing TRNs. We improve the topology generator with more biologically plausible models.

This paper improves upon the model generation approach of (Provan & Wang 2007) in several ways. First, it explicitly defines a domain-analysis phase. Second, it extends and improves upon the topology generation algorithms for creating the underlying system structure, and examines a wider range of metrics for empirically evaluating synthetic networks.

Compositional modelling uses a set of functional component models, together with a specification of component interactions (called a "scenario" in (Keppens & Shen 2001)) to generate useful (mathematical) models. Our approach differs from that of (Keppens & Shen 2001) in that we create the system structure, or scenario using model generators instead of manual work. Further, although the model-generation (or compositional modeling) approach has primarily been applied to physical systems, it can be applied to other domains, such as socio-economic, ecological and biological systems (Van den Bulcke *et al.* 2006; Mendes, Sha, & Ye 2003).

Modeling Framework

This section describes our approach for generating benchmark models for compositional domains. A domain D is *compositional* if a system model from D can be composed from model components, each of which is defined by a component functional model. Our approach is applicable to any compositional domain, since (a) the underlying topological models can be optimized using appropriate parameters to approximate virtually the structure of real-world complex systems (Boccaletti *et al.* 2006), and (b) functionality is incorporated into the system model using a component-library, where components can be developed for any domain in which the system models are decomposable.

We assume that a model can be generated from the tuple $(G, \mathcal{B})$, where G denotes the topology graph, and $\mathcal{B}$ denotes the functionality descriptions for components. The topology graph $G = (V, E)$ consists of vertices V and edges E and specifies the topological relations among the system components. Each node $v \in V$ corresponds to a component or

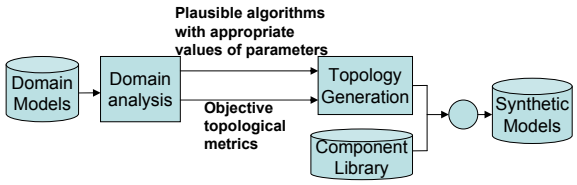

Figure 1: Automated model generation framework.

As shown in Figure 1, we generate benchmark models in a three-step process.

1. analyze existing domain models to extract important model properties;
2. generate the (topology) graph $\hat{G}$ underlying each synthetic model;
3. assign components to each node in $\hat{G}$, to create the system-level functional model $\hat{\Psi}$;

For example, electronic circuits can be viewed as graphs in which nodes are electronic components (such as logic gates in digital circuits) and edges are wires in a broad sense (Cancho, Janssen, & Solé 2001). In gene TRNs, nodes represent genes and edges correspond to regulatory interactions at transcriptional level between the genes (Van den Bulcke *et al.* 2006; Mendes, Sha, & Ye 2003).

As another example, in auto-generating TRN models, each node in $\hat{G}$ is instantiated as a gene, and the interaction kinetics between the genes are quantitatively modeled using a set of ordinary differential equations (ODEs) (Van den Bulcke *et al.* 2006; Mendes, Sha, & Ye 2003). For each combination of a gene and its regulators, a proper enzyme kinetic equation is selected, depending on the number of activators and repressors and on settings that control the fraction of complex interactions (Van den Bulcke *et al.* 2006).

Topological Model Selection and Generation

To generate a synthetic network $\hat{G}$ using an algorithm A, we provide to A a set Π of input parameters, and then measure the properties of $\hat{G}$ (e.g., degree distribution) using a set Φ of graph metrics (Mahadevan *et al.* 2006) to compare the properties of the real and synthetic networks. For example, the preferential attachment (PA) (Boccaletti *et al.* 2006) algorithm requires the number of nodes and edges of $\hat{G}$ as input parameters.

There is a wide range of generation algorithms available, e.g., (Boccaletti *et al.* 2006; Chung *et al.* 2003). Table 1 classifies the space of topology-generation approaches

that our model-generation tool supports in terms of the key properties of the approaches, together with their corresponding parameters, recommended applications, and associated model-generation computational costs. We classify the generator models into two main groups, as shown in column 1 of Table 1: *explanatory* models, which attempt to capture the underlying generation process of the complex system in the resulting model, or *descriptive* models, which capture the topology alone. For example, the explanatory Preferential Attachment model is designed to capture the growth process of complex systems, in which new network structure preferentially forms around existing sub-structures (Boccaletti *et al.* 2006). In contrast, the descriptive dK-series model (Mahadevan *et al.* 2006) just captures higher-order degree correlation distributions, independent of any complex system growth process. Another dimension in model selection encompasses trade-offs between: (1) complexity of a model and the number of metrics it tries to reproduce, and (2) its explanatory power and associated generality. The process of generating high-fidelity synthetic models differs based on this basic classification. In the following, we summarize our model selection process (using these two classes), and then review the different generation approaches.

Model Selection using Explanatory Models

To select an explanatory model, we must analyze the domain D to (1) select the most appropriate topology generation algorithm A from a set $\mathcal{A}$ of candidate algorithms, and (2) provide parameters for A that are best suited to generating high-fidelity networks. We select an explanatory model from a set $\mathcal{A}$ of possible generators (see Table 1) as follows:

1. analyze real-world network G, together with key properties in domain D, to specify a topological metric set Φ according to domain-specific requirements;
2. generate potential algorithm set $\mathcal{A}' \subseteq \mathcal{A}$ based on analytical results in step 1;
3. optimize parameters Π_i of each algorithm $A_i \in \mathcal{A}'$ to match G in terms of specified topological metrics Φ, and put the A_i into the result set $\hat{\mathcal{A}}$ if it can match G with appropriate values of Π_i;
4. if $\hat{\mathcal{A}}$ contains multiple algorithms, we compute additional metrics Φ', according to further requirements in D, and continue to evaluate and select algorithms in terms of Φ'.

When using an explanatory model, we first restrict the possible algorithms based on *Model Focus* (cf. column 2 of Table 1), i.e., whether the domain D provides information to to generate a model from topological parameters, or using an optimization approach given the system's global objective function. We briefly discuss these two approaches.

Topology-Based Generators: Given the wide range of graph generators defined in the literature, e.g., (Boccaletti *et al.* 2006; Chung *et al.* 2003), we have selected four of the most important approaches, i.e., the small-world graph (SWG), Preferential Attachment (PA), Spatial Preferential Attachment (SPA) and Partial Duplication (PD) models. Each approach has particular properties, which lend themselves to modeling particular domains with differing fidelity.

Optimization-Based Generators: Rather than explicitly replicate of statistical properties, the *Optimization approach* (OPT) use an optimization framework to model the mechanisms driving network growth. This approach gives rise to power-laws in graph degree distributions (D'Souza *et al.* 2007; Mathias & Gopal 2001). The OPT model formulates a weighted objective function over conflicting system properties ξ_i and weights λ_i, e.g., $f = \sum_{i=1}^{n} \xi_i \cdot \lambda_i$, and trades off the properties using the weights λ_i. For example, in circuit design we may trade off wire-length WL and logic-depth LD, where $f = \lambda LD + (1 - \lambda)WL$. We have used simulated annealing to search for the cost minimum of the objective function (Kirkpatrick, Gelatt, & Vecchi 1983).

An explanatory model with parsimonious parameters can capture the general principles or structures of real-world systems, but it is hard to match all topological metrics simultaneously and perfectly As a consequence, we need to identify and understand the essential metrics that are responsible for certain behaviors of certain applications, and focus on specified metrics necessary for capturing the domain-specific requirements of different applications. For instance, if we generate a model for evaluating the complexity of discrete MBD algorithms, we need first to focus on domain-specific (jointree) metrics (Provan & Wang 2007), which are more important than regular metrics.

In steps 3 and 4, after a plausible model is selected, we further optimize its topology by searching over appropriate values of input parameters to minimize the difference between G and $\hat{G}$ in the specified metrics. We automatically scan appropriate values in a specific range (within a reasonable interval); if the topological metrics are monotonic functions of input parameters (such as α in the SPA model and p_r in the SWG model), we can speed up the search process using strategies like the binary search.

Model Selection using Descriptive Models

The *dK-series Model* generator (Mahadevan *et al.* 2006) has as its primary input parameter an integer d, which allows one to specify all degree correlations within d-size subgraphs of a given graph G[1]. $1K$ captures the degree distribution P_k and is equal to the generalized random graph (GRG) (Boccaletti *et al.* 2006). $2K$-graphs reproduce the joint degree distribution, and $3K$-graphs consider interconnectivity among triples of nodes.

Given a descriptive dK-series algorithm, we generate a synthetic model $\hat{G}$ by increasing the input parameter (d) until the generated graph $\hat{G}$ matches the properties of the real-world graph G with sufficient fidelity. Increasing values of d capture progressively more properties of G, at the cost of more complex representation of the probability distribution and dramatically increasing computational complexity.

Although the dK-series model generally can capture regular topological metrics better than explanatory models due to the number of constraints imposed, the model doesn't provide insights into the driving force shaping the network, and

[1] Actually, a large number parameters are needed for every value of d in real implementations, but d is the governing parameter.

Table 1: Topology Generation Approaches. Input parameters for generation algorithms are as follows: n—node number; m—edge number; p_r—rewiring probability; α—spatial factor; g_s—seed network; p_d—duplication probability; λ_i—trade-off weight; d—subgraph size

Model Class	**Model Focus**	**Generation Algorithm**	**Key Properties**	**Parameters**	**Recommended Applications**	**Computational Cost**
Explanatory	Topological Properties	Small-world Graph (SWG)	Exponential degree distribution	n, m, p_r	Technological systems	Low
		Preferential Attachment (PA)	Power law degree distribution	n, m	WWW, social and citation networks	Low
		Spatial Preferential Attachment (SPA)	Power law degree distribution with cutoff	n, m, α	Spatial technological systems	Medium
		Partial Duplication (PD)	Power law degree distribution	n, m, g_s, p_d	Biological systems	Low
	Functional Optimization	Multi-constraint Optimization (OPT)	Power law degree distribution with cutoff	λ_i	Technological and transportation systems	High
Descriptive	Topological properties	dK-series	All degree correlations in d-sized subgraphs	d	Technological and biological systems	High

it lacks predictive and rescaling power for explaining network growth. Our experiments on diagnosis model generation also showed that the dK-series model is not flexible enough for fitting more complicated joint-tree metrics.

Summary of Topological Metrics

We assume that we have a correct set of functional components $\mathcal{B}$, meaning that it is the system topology G which is the source of model fidelity. In this case, we need to identify metrics for topology comparison, i.e., methods to define some topological distance measure $\delta(G, \hat{G})$. There are many metrics used to analyze and compare a system's topological structures (Boccaletti *et al.* 2006; Mahadevan *et al.* 2006). The following list is not complete, but we believe it is sufficiently diverse and representative to be used as good examples of topological similarity.

Standard Metrics: Most research on topological analysis of complex systems focuses on a subset of graph properties, in particular on the characteristic path length $\bar{L}$, average clustering coefficient $\bar{C}$ and degree distribution P_k (Boccaletti *et al.* 2006). The $\bar{L}$ measures the typical separation between two nodes in the network is given by the average shortest path length. The clustering coefficient C characterizes the degree of cliquishness of a typical neighborhood (a node's immediately connected neighbors), and the mean coefficient $\bar{C}$ is the average over C for all nodes in G. The degree distribution P_k specifies the probability of a node having degree k.

Extended Topology Metrics: We focus on the following extended metrics.

s-Metric: The s-Metric of graph G is defined as $s(G) = \sum_{edge(v_i,v_j)} d_i d_j$, where (v_i, v_j) is the edges in the graph, and d_i and d_j are the degrees of the node v_i and v_j respectively. The s-Metric is closely related to betweenness, degree correlation and graph assortativity(Mahadevan *et al.* 2006).

Subgraph Frequency Distribution: $P(F_x(G))$ defines that probability of subgraph of type x occurring in graph G. The distribution $P(F_x(G))$ enables us to analyze the frequencies of all sub-graphs with specified sizes.

Join-tree Metrics: In many applications involving inference over systems Ψ, e.g., probabilistic inference and model-based diagnosis, the inference complexity has been found to be dependent on parameters of the join-tree $\mathcal{T}$ of the graph G of Ψ (Darwiche 1998).[2] As a consequence, for applications involving system inference, we use appropriate join-tree metrics, such as the largest clique size $\mu(\mathcal{T})$(Darwiche 1998), which can be used to represent the inference complexity of the system.

Examples of High-Fidelity Model-Generation

To demonstrate the range of models for which this approach is applicable, we describe model-generation for two radically different domains, E.coli TRN networks for simulating gene expression, and MBD inference of discrete circuits.

TRN Inference Benchmark

The validation of algorithms used to infer the structure of gene regulatory networks, based on expression data from high throughput microarrays, requires benchmark data sets for which the underlying network is known. Since experimental data sets of the appropriate size and design are usually not available, there is a clear need to generate well-characterized synthetic data sets that allow thorough testing of learning algorithms in a fast and reproducible manner (Mendes, Sha, & Ye 2003; Van den Bulcke *et al.* 2006). So we need a network generator that creates synthetic TRNs and produces simulated gene expression data that approximates experimental data. TRN model generation provides a good example to demonstrate the applicability of our general model generator to biological domains, and we use the well-known TRN of E. coli collected by Shen-Orr et al. (Van den Bulcke *et al.* 2006) as the targeted domain model.

[2] Roughly speaking, the join-tree $\mathcal{T}$ of a graph G is a topological transformation of G into a tree of cliques, where a clique is a fully-connected subgraph (Darwiche 1998).

Explanatory Model Approach We generated the synthetic TRN model based on the first three steps in the process of model selection, as discussed in the previous section.

Step 1: We analyzed the E.coli TRN model and found that it displays a clear power law degree distribution as shown in Figure 2. Since the synthetic TRN models are used to generate gene expression data (on which the accuracy of reverse-engineering algorithms is evaluated), we only need to measure the model fidelity in terms of regular topological metrics. For this task, we use the degree distribution P_k, which is the most fundamental and widely-used metric. P_k can be simplified as an exponent β when following a power law; the β of the E.coli TRN model is about 2.5.

Step 2: According to key properties of the potential algorithms listed in Table 1, both the PA and PD model can generate a power law degree distribution, and thus are selected as candidate algorithms.

Step 3: The parameters in the $PA(n, m)$ and $PD(n, m, g_s, p_d)$ algorithms are optimized in terms of β; n and m are assigned as the numbers of nodes and edges in the actual TRN model respectively. We carefully sampled an appropriate subgraph of the TRN of E. coli using the method in (Van den Bulcke *et al.* 2006) as the seed graph g_s. Chung *et al.* (2003) showed that in the PD model, β is a monotonically decreasing function of p_d, so we can approximate the degree distribution of the TRN of E. coli by adjusting p_d. The PA model's value of β is fixed at about 3, but the PD model can generate β in a wide range $(1 \sim 3)$, consistent with various real biological networks(Chung *et al.* 2003). Figure 2 shows that the PD model ($p_d = 0.2$) closely matches the actual TRN, much better than can the PA model.

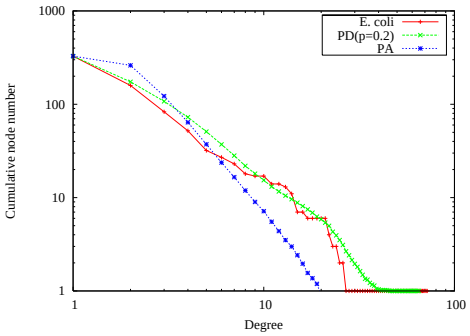

Figure 2: The cumulative degree distribution of E. coli and the corresponding graphs generated by the PD and PA model (averaged over 100 runs).

dK-series Approach Table 2 shows that the graphs generated by the dK-series model perfectly match common graph metrics of the E. coli TRN, including P_k when $d = 3$. The experimental results show that the dK-series is a good model for TRN benchmark generation. However, compared with the parsimonious PD model, the dK-series model is more computationally expensive and less flexible, since it requires as input parameters multiple degree correlations within d-sized subgraphs of an existing TRN.

Table 2: The statistics on the average clustering coefficient $\bar{C}$, characteristic path length $\bar{L}$, and s-Metric of E. coli and the graphs generated by the 3K-series model (averaged over 100 graphs).

Model	L	C	s-Metric
E. coli	4.83371	0.11018	26621
3K	4.64722	0.11018	26621

Model-Based Diagnosis Benchmark

The Model-based diagnosis (MBD) problem determines whether an assignment of failure status to a set of mode-variables is consistent with a system description and an observation (e.g., of sensor values). For the target domain of ISCAS85 benchmark circuits (Harlow 2000), we synthesized topological models having the identical numbers of nodes and edges, and tried to characterize average-case diagnosis inference complexity in real circuits.

Explanatory Model Approach We generated MBD models using the four steps shown below.

Step 1: We used as our primary metric the maximum clique size $\mu(\mathcal{T})$ in the compiled join-tree structure, which is a typical complexity measure for this type of model (Darwiche 1998), and is correlated to the tail length of degree distribution P_k (Provan & Wang 2007). As shown in Figure 3, the tail of P_k must be modeled well, since it defines the high-degree nodes that contribute to large cliques in the join-tree, and hence high complexities using join-tree metrics. We have empirically showed that most of the ISCAS85 circuits have power law degree distributions with sharp cutoffs, which can be well characterized by the SPA and OPT models in Table 1. The SWG model naturally has a sharp cutoff in its exponential degree distribution, and can vary the tail length of its degree distribution in a limited range.

Step 2: Based on the above analysis, the SPA, OPT, and SWG model can be selected as potential candidates.

Step 3: We automatically optimized parameters in each model to match the $\mu(\mathcal{T})$ of real circuits. Experiments showed that all selected models can match real circuits with appropriate parameters. For example, the typical circuit C432 can be matched by the SWG model with $p_r \simeq 0.28$ (Provan & Wang 2007), the SPA model with $\alpha \simeq 3.7$ (as shown in Figure 3). We, along with Barthelemy (2003) have found that, under appropriate parameters, the SPA model can generate structures similar to that of the OPT model. However, the computational cost of model-generation using the OPT model is significantly higher than that of using the SPA model, so we use the SPA model as an efficient alternative of the OPT model.

Step 4: Since both the SPA and SWG model fit the real circuits well in terms of $\mu(\mathcal{T})$, we can further refine the model selection by other topological metrics, such as degree distribution P_k. Based on P_k, the SPA model's power-law distribution can match real circuits better than can the SWG model's exponential distribution.

dK-series Approach When $d = 3$, the dK-series model can match almost all common circuit topological metrics perfectly, as also occurs in the case of the TRN and Internet

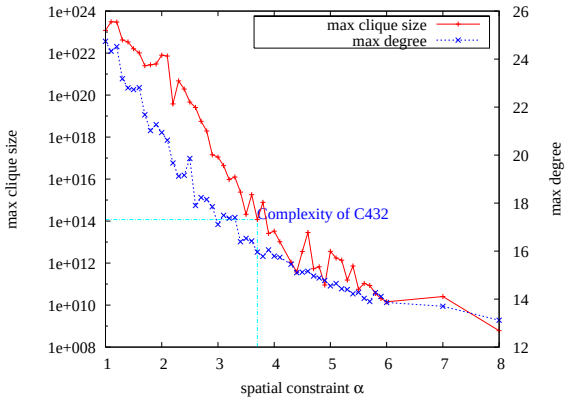

Figure 3: The inference complexity and maximal degree of the SPA model corresponding to the circuit C432 (averaged over 100 runs).

Table 3: The inference complexity of C432 and corresponding dK-series Models ($d = 1, 2, 3$). All values of three models are averaged over 100 graphs respectively.

Model	C432	1K	2K	3K
max clique size	1.4e14	8.9e17	3.3e16	1.8e16

modeling (Mahadevan *et al.* 2006). Table 3 shows, however, that $d = 3$ provides insufficient fidelity to match $\mu(\mathcal{T})$ metrics for MBD benchmark generation; it also shows that increasing d can generate random graphs with increasing levels of fidelity of inference complexity. For $d > 3$ the computational complexity increases dramatically, and the size of the generated random-graph ensemble decreases exponentially as well. In this case, the dK-series model is unsuitable for diagnosis benchmark generation, compared with the SPA and the SWG model.

Conclusions

We have described a model-generation tool that can be used for compositional systems in which we use a component library $\mathcal{B}$, together with a system topology G, to generate benchmark models for a wide range of systems. We assumed a library $\mathcal{B}$, and focused on the problem of topology generation. We described a domain-analysis algorithm that computes model properties for selecting the topological model generator best suited to creating high-fidelity networks.

We applied model-generation to two radically different domains, E.coli TRN networks for simulating gene expression, and MBD inference of discrete circuits, to demonstrate the range of models for which this approach is applicable. For each domain we showed how the topological properties, together with the functional requirements (e.g., simulation, diagnostic inference), enabled us to tune the generated network topology to the application.

Much work remains to be done in automated model generation. First, more component libraries need to be created to take advantage of this approach. Second, the domain analysis approach could be further improved through adoption of machine learning techniques. Third, further improvements in topology-generators are necessary to increase the fidelity of the synthetic models.

References

Barthlemy, M. 2003. Crossover from scale-free to spatial networks. *Europhysics Letters* 63:915–921.

Boccaletti, S.; Latora, V.; Moreno, Y.; Chavez, M.; and Hwang, D.-U. 2006. Complex networks : Structure and dynamics. *Physics Reports* 424(4-5):175–308.

Cancho, R. F. i.; Janssen, C.; and Solé, R. V. 2001. Topology of technology graphs: Small world patterns in electronic circuits. *Physical Review E* 64(4):046119.

Chung, F. R. K.; Lu, L.; Dewey, T. G.; and Galas, D. J. 2003. Duplication models for biological networks. *Journal of Computational Biology* 10(5):677–687.

Darwiche, A. 1998. Model-based diagnosis using structured system descriptions. *J. Artif. Intell. Res. (JAIR)* 8:165–222.

D'Souza, R. M.; Borgs, C.; Chayes, J. T.; Berger, N.; and Kleinberg, R. D. 2007. Emergence of tempered preferential attachment from optimization. *Proc. Natl. Acad. Sci. USA* 104(15):6112–6117.

Harlow, J. E. 2000. Overview of popular benchmark sets. *IEEE Design and Test of Computers* 17(3):15–17.

Hormozdiari, F.; Berenbrink, P.; Przulj, N.; and Sahinalp, S. C. C. 2007. Not all scale-free networks are born equal: The role of the seed graph in ppi network evolution. *PLoS Comput Biol* 3(7).

Keppens, J., and Shen, Q. 2001. On compositional modelling. *Knowl. Eng. Rev.* 16(2):157–200.

Kirkpatrick, S.; Gelatt, C. D.; and Vecchi, M. P. 1983. Optimization by simulated annealing. *Science, Number 4598, 13 May 1983* 220, 4598:671–680.

Mahadevan, P.; Krioukov, D. V.; Fall, K. R.; and Vahdat, A. 2006. Systematic topology analysis and generation using degree correlations. In *SIGCOMM*, 135–146.

Mathias, N., and Gopal, V. 2001. Small worlds: How and why. *Physical Review E* 63:021117.

Mendes, P.; Sha, W.; and Ye, K. 2003. Artificial gene networks for objective comparison of analysis algorithms. *Bioinformatics* 19 Suppl 2.

Provan, G. M., and Wang, J. 2007. Automated benchmark model generators for model-based diagnostic inference. In *IJCAI*, 513–518.

Stroobandt, D. 2001. Analytical methods for a priori wire length estimates in computer systems. *Ph.D. dissertation: Ghent University*.

Van den Bulcke, T.; Van Leemput, K.; Naudts, B.; van Remortel, P.; Ma, H.; Verschoren, A.; De Moor, B.; and Marchal, K. 2006. Syntren: a generator of synthetic gene expression data for design and analysis of structure learning algorithms. *BMC Bioinformatics* 7.