ASSA-PBN: An Approximate Steady-state Analyser of Probabilistic Boolean Networks

Andrzej Mizera, Jun Pang, and Qixia Yuan

Computer Science and Communications, University of Luxembourg

Abstract. We present ASSA-PBN, a tool for approximate steady-state analysis of large probabilistic Boolean networks (PBNs). ASSA-PBN contains a constructor, a simulator, and an analyser which can approximately compute the steady-state probabilities of PBNs. For large PBNs, such approximate analysis is the only viable way to study their long-run behaviours. Experiments show that ASSA-PBN can handle large PBNs with a few thousands of nodes.

1 Introduction

Probabilistic Boolean networks (PBNs) [1, 2], introduced by Shmulevich et al. in 2002 as an extension of Boolean networks, is a modelling framework widely used to model gene regulatory networks (GRNs). Inheriting appealing features of Boolean networks of being simple but effective, PBN additionally is capable to cope with uncertainties both on the data and model selection levels. The dynamics of a PBN with N nodes is governed by a discrete-time Markov chain (DTMC) with exponential in N number of states, i.e., 2^N states. Although qualitative in nature, PBNs provide means for quantifying the influences of genes on other genes and for characterising the long-run behaviour of the system, both based on the steady-state distribution of the associated DTMC. Therefore, the efficient computation of steady-state probabilities is of utter importance. It is well studied how to compute the steady-state distribution of small PBNs with numerical methods [2]. However, due to their computational cost, these methods are not scalable. In the literature, a few statistical methods such as Monte Carlo methods [3], are proposed to deal with large PBNs. A recent study analysed a 96-node PBN of apoptosis using the optPBN tool [4]. However, the existing methods/tools for PBNs are prohibited by the network size. For instance, optPBN can only analyse parts of the 96-node PBN due to its efficiency limits, leaving some properties of the network unconfirmed [4]. Therefore, there is demand for a tool which can handle large PBNs efficiently.

In this work we present ASSA-PBN, a tool which provides the means for efficient analysis of large PBNs. Firstly, it applies the state-of-the-art techniques to the computation of the steady-state probabilities of large PBNs. The current version supports three different statistical methods, i.e., the perfect simulation algorithm [5], the two-state Markov chain approach [6, 7], and the Skart method [8]. To the best of our knowledge, ASSA-PBN is the first tool to introduce the perfect simulation algorithm and the Skart method to the context of PBNs. Secondly, it contains a fast simulator, which is designed using the alias technique [9]. This makes it capable of handling the steady-state computations that require generation of trajectories consisting of billions of states. Experimental results show that ASSA-PBN can analyse PBNs with thousands of nodes.

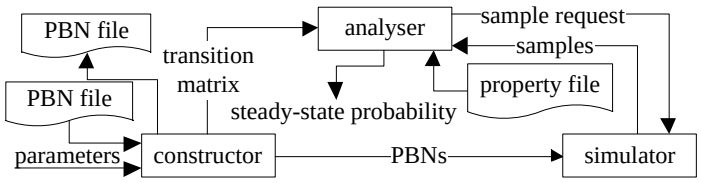

Fig. 1. Structure of ASSA-PBN.

2 Architecture and Usage

A PBN models a system such as a GRN with binary-valued nodes. For each node there is a certain number of Boolean functions, known as *predictor functions*, which determine the value of the node at the next time step. The selection of the predictor function is governed by a probability distribution: a probability parameter is associated with each predictor function of the node. Two variants of PBNs are considered: *instantaneous PBNs* and *context-sensitive PBNs*. In the former variant, the selection of a predictor function is performed for each node at each time step. In the latter variant, the PBN evolves in accordance with selected predictor functions and new selection is performed only if indicated by an additional random variable which is updated at each time step. Moreover, the so-called PBNs with perturbations, allow the system to transit to the next state due to random perturbations that are governed by a perturbation rate parameter. On top of that *dependent* and *independent PBNs* are considered as well as *synchronous* and *asynchronous* update schemes. The dynamics of a PBN can be viewed as a DTMC. In the case of PBNs with perturbations, the underlying DTMC is *ergodic*, thus having a unique *stationary distribution*, the so-called *steady-state* (or *limiting*) *distribution*, which governs the long-run behaviour of the system. Due to the space limitation, we refer to [10] and [2, page 4] for a formal definition and detailed description of PBNs.

For systems like GRNs, we are often interested in answering the following biological question: what is the probability that a gene or a number of genes will be expressed in the long run? This question can be addressed in the framework of PBNs by determining the steady-state probability for a PBN to be in a certain state or, equivalently, the steady-state probability of a state in the underlying ergodic DTMC. ASSA-PBN is designed to compute the steady-state probabilities for PBNs, especially for large PBNs.

ASSA-PBN contains three major parts (see Figure 1): a PBN constructor, a PBN simulator, and a PBN analyser. Based on the specified parameters or model file, the constructor can build a PBN. The simulator takes a PBN generated by the constructor as input and performs simulation of the PBN efficiently to produce trajectories (also called samples). The key function of ASSA-PBN is to compute the steady-state probability for a set of states of the PBN which is defined in a property file. This is achieved by the analyser in either a numerical manner (for small PBNs) or a statistical manner (for large PBNs). The implemented numerical methods require the transition matrix of a PBN as input, which is supplied by the constructor; while the implemented statistical methods require simulated trajectories of the PBN as input, which are supplied by the simulator. Simulation is not based on the transition matrix, as a consequence it does not suffer from the state-space explosion problem even for large PBNs. The ASSA-PBN program package can be found at `http://satoss.uni.lu/software/ASSA-PBN/`.

PBN constructor. The PBN constructor can either load a PBN from a specification file or generate a random PBN complying with a given parametrisation.

To load a PBN into ASSA-PBN, the user needs to provide a description of the PBN in a specification file. The description consists of four elements: the number of nodes (also known as genes), the definitions of predictor functions of each node, the selection probabilities for the predictor functions of each node, finally the perturbation rate parameter value. The details on the format can be found in the user guide of the program package. ASSA-PBN also supports importing PBNs defined in the optPBN Matlab toolbox [4] format. ASSA-PBN constructs the PBN based on its loaded specification and stores it in memory in a structure which contains the information from the specification. The stored information is sufficient for performing simulations of the PBN.

For the convenience of experiment and testing, the constructor provides a function for generating a random PBN complying with specified parameters, i.e., the number of nodes in the PBN, the perturbation rate, the maximal number of predictor functions for nodes in the network, and the maximal number of variables of the predictor functions. Given these parameters, the constructor will randomly generate for each node of the PBN the specified number of predictor functions and their selection probabilities. The generated PBN can be saved in a specification file or exported to the optPBN format.

In certain biological experiments, the environmental conditions of cells are kept constant, e.g., sustained activation of cell receptors. Therefore, in ASSA-PBN it is possible to disable the perturbations of certain nodes. This is provided by a filter storing the indices of these nodes that should not undergo perturbations. Note that this does not change the ergodicity of the PBN: since the values of those nodes are kept constant, the state-space of the underlying Markov chain is simply reduced, but it remains aperiodic and irreducible, thus ergodic.

Simulator. Statistical approaches are the only viable option for the analysis of PBNs characterised by large state space, e.g., PBNs that arise in the context of realistic biological study of GRNs. However, applications of such methods necessitate generation of trajectories of significant length. Therefore, the efficiency requirement is crucial for enabling analysis of large networks in a reasonable computational time. To achieve this goal, sampling of the consecutive state of the trajectory is performed with the use of an instance of the *alias method*, a class of efficient algorithms for sampling from a discrete probability distribution originally introduced by Walker in [9].

The ASSA-PBN simulator can operate in two modes: 1) the *global alias mode* and 2) the *local alias mode*. In the global mode, a joint probability distribution is considered on all possible combinations of predictor function selections for all the nodes and a single alias table for this distribution is constructed. In the local mode, the independence of the PBN is exploited: individual alias tables are constructed for each node of the PBN. In both cases the consecutive state is generated by updating the value of each node with the predictor function selected for this node. However, in the global mode predictor functions for all nodes are selected simultaneously with the use of only two random numbers, while in the local mode the number of random numbers used is twice the number of nodes. In consequence, the generation of the next state is faster in the global mode, but more expensive in terms of memory usage.

When simulating the next state, the simulator first decides whether perturbations are applied. If yes, the simulator updates the current state accordingly. If no, the simulator chooses the predictor functions for each node and update the current state according to the chosen predictor functions. Notice that the state transition matrix is not needed in the simulation process, which makes ASSA-PBN capable of managing large PBNs.

The simulator is called by the analyser to perform simulation when statistical methods are selected for the analysis of a PBN. We explain the interaction between the simulator and the analyser in the next section.

Analyser. The analyser provides three different classes of methods for the computation of exact/approximate steady-state probabilities. The first class consists of two iterative methods for exact computation of the steady-state distributions: the Jacobi method and the Gauss-Seidel method. However, both of them require the state transition matrix to be constructed by the constructor, which is expensive in terms of memory and time. Therefore, these methods are only suitable for analysing small-size PBNs (the focus of ASSA-PBN is on large PBNs). The two methods are available in probabilistic model checkers, e.g., PRISM [11]. We reimplement them for the comprehensiveness of the tool and the convenience of the user as there is no direct way to handle PBNs in PRISM.

The second class consists of the *perfect simulation algorithm* for sampling the steady-state of DTMCs. It is based on the ingenious idea of the *backward coupling scheme* originally proposed by Propp and Wilson in [12]. The *perfect simulation algorithm* allows to draw independent samples which are distributed exactly in accordance with the steady-state distribution of a DTMC. Thus, it avoids problems related to the speed of convergence to the steady-state distribution or non-zero correlation between consecutive samples. Given a confidence level, the number of required samples used for the approximation of the steady-state probability can be iteratively computed. Due to the uncorrelated, exact samples the use of this sampling method for statistical approximation of steady-state probabilities results in a smaller sample size than the statistical methods from the third class discussed below. The current implementation is in-line with the 'Functional backward-coupling simulation with aliasing' algorithm provided in [5], which highly improves the efficiency. The functional coupling shortens significantly the average coupling time in cases where the aim is to approximate the steady-state probability for a subset of states. In general, larger subsets of interest result in smaller coupling time. Due to the nature of this method, each state of the state space needs to be considered at each step of the coupling scheme. Therefore, this approach is suitable only for medium-size PBNs and is implemented for the comprehensiveness of the tool. Unfortunately, since PBNs with perturbations are non-monotone systems, the very efficient monotone version of perfect simulation [13] in which only a small subset of the state space needs to be considered is of no use in this context.

The third class of available methods consists of two *incremental sampling* methods for approximate computation of the steady-state probabilities, i.e., 1) the Skart method [8] and 2) the two-state Markov chain approach [6]. Both statistical methods operate in accordance with the following scheme. The analyser calls the simulator to generate a trajectory of initial length. The algorithms check whether the trajectory is long enough to compute estimates of average internal statistics, which satisfy a predefined confidence level and precision requirements. If the confidence level and precision are

node number	method	the two-state Markov chain			Skart		
	precision	0.01	0.005	0.001	0.01	0.005	0.001
1,000	trajectory size	35,066	133,803	3,402,637	37,999	139,672	3,272,940
	time cost (s)	6.19	23.53	616.26	7.02	24.39	590.26
2,000	trajectory size	64,057	240,662	5,978,309	63,674	273,942	5,936,060
	time cost (s)	20.42	67.60	1722.86	20.65	78.53	1761.05

Table 1. Approximate steady-state analysis of two large PBNs (confidence level 95%).

not reached, the simulator will be called again to generate more samples. This process is repeated until the confidence level and precision requirements are met. For a comprehensive description of these methods we refer to [6–8]. They are best suited for large networks which cannot be analysed with the approaches from the previous two classes. Our experiments show that both the Skart method and the two-state Markov chain approach in the current implementations can be used for the approximation of steady-state probabilities for networks consisting of thousands of nodes (see Section 3).

3 Comparison, Evaluation and Future Developments

Comparison. Both ASSA-PBN and optPBN are designed for steady-state analysis of PBNs. ASSA-PBN supports three statistical methods for computing the steady-state probabilities of a PBN, while optPBN only prototypically implemented the two-state Markov chain approach in Matlab. We compared our tool with optPBN for the analysis of the 96-node PBN model of apoptosis in [4] using the two-state Markov chain approach. As mentioned in [4], their analysis of the influence of RIP_deubi on complex2 was not handled completely due to the long computational time required by optPBN. With the use of ASSA-PBN, we performed the complete analysis of the influence and presented it in [7]. ASSA-PBN outperforms optPBN in terms of simulation speed. For example, the time cost for simulating 100,000 steps of the 96-node PBN model in optPBN is around 120s, which is almost 100 times more than that in ASSA-PBN.

Our tool is highly related to tools for statistical model checking [14, 15]. Existing statistical model checkers are either restricted for bounded properties or cannot directly deal with PBNs. The Skart method and the perfect simulation algorithm have been recently used for statistical model checking of steady state and unbounded until properties [13, 16]. To the best of our knowledge, ASSA-PBN is the first tool to introduce those two methods into the context of PBNs.

Evaluation. We have evaluated ASSA-PBN exhaustively on a large number of randomly generated PBNs of different sizes ranging from a few nodes to a few thousands nodes and with different characteristics (network structures being either sparse or dense). We show in Table 1 the trajectory sizes and the time costs for computing steady-state probabilities of two large PBNs using the two-state Markov chain approach and the Skart method for different precision requirements. We have compared the performance of these two methods in computing steady-state probabilities of PBNs on 882 randomly generated PBNs with nodes numbers ranging from 15 to $2,000$. We collected $5,263$ pairs of results. Based on this complete set of results, i.e., with nodes numbers ranging from 15 to $2,000$, the two-state Markov chain approach was faster than the Skart method in most of the compared cases (see [7, Section 4] for more details).

Future developments. One technique to improve the performance of ASSA-PBN is parallel computation. This can be applied to the Jacobi method, the perfect simulation algorithm and the simulation of a PBN. In the future, we plan to implement in ASSA-PBN other algorithms (e.g., see [17]) for approximating steady-state probabilities.

References

1. Shmulevich, I., Dougherty, E., Zhang, W.: From boolean to probabilistic boolean networks as models of genetic regulatory networks. Proceedings of the IEEE **90**(11) (2002) 1778–1792
2. Trairatphisan, P., Mizera, A., Pang, J., Tantar, A.A., Schneider, J., Sauter, T.: Recent development and biomedical applications of probabilistic Boolean networks. Cell Communication and Signaling **11** (2013) 46
3. Shmulevich, I., Gluhovsky, I., Hashimoto, R., Dougherty, E., Zhang, W.: Steady-state analysis of genetic regulatory networks modelled by probabilistic boolean networks. Comparative and Functional Genomics **4**(6) (2003) 601–608
4. Trairatphisan, P., Mizera, A., Pang, J., Tantar, A.A., Sauter, T.: optPBN: An optimisation toolbox for probabilistic boolean networks. PLOS ONE **9**(7) (2014) e98001
5. Vincent, J.M., Marchand, C.: On the exact simulation of functionals of stationary Markov chains. Linear Algebra and its Applications. **385** (2004) 285–310
6. Raftery, A., Lewis, S.: How many iterations in the Gibbs sampler? Bayesian Statistics **4** (1992) 763–773
7. Mizera, A., Pang, J., Yuan, Q.: Reviving the two-state markov chain approach (technical report). Available online at `http://arxiv.org/abs/1501.01779` (2015)
8. Tafazzoli, A., Wilson, J., Lada, E., Steiger, N.: Skart: A skewness- and autoregression-adjusted batch-means procedure for simulation analysis. In: Proc. of the 2008 Winter Simulation Conference. (2008) 387–395
9. Walker, A.: An efficient method for generating discrete random variables with general distributions. ACM Transactions on Mathematical Software **3**(3) (1977) 253–256
10. Shmulevich, I., Dougherty, E.R.: Probabilistic Boolean Networks: The Modeling and Control of Gene Regulatory Networks. SIAM Press, Philadelphia, USA (2010)
11. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In Gopalakrishnan, G., Qadeer, S., eds.: Proc. 23rd International Conference on Computer Aided Verification (CAV'11). Volume 6806 of LNCS., Springer (2011) 585–591
12. Propp, J.G., Wilson, D.: Exact sampling with coupled markov chains and applications to statistical mechanics. Random Structures & Algorithms **9**(1) (1996) 223–252
13. El Rabih, D., Pekergin, N.: Statistical model checking using perfect simulation. In: Proc. 7th Symposium on Automated Technology for Verification and Analysis. Volume 5799 of LNCS. (2009) 120–134
14. Younes, H.S., Simmons, R.: Probabilistic verification of discrete event systems using acceptance sampling. In: Proc. 14th Conference on Computer Aided Verification. Volume 2404 of LNCS., Springer (2002) 223–235
15. Sen, K., Viswanathan, M., Agha, G.: On statistical model checking of stochastic systems. In: Proc. 17th Conference on Computer Aided Verification. Volume 3576 of LNCS., Springer (2005) 266–280
16. Rohr, C.: Simulative model checking of steady state and time-unbounded temporal operators. Transactions on Petri Nets and Other Models of Concurrency **8** (2013) 142–158
17. Gelman, A., Rubin, D.: Inference from iterative simulation using multiple sequences. Statistical Science **7**(4) (1992) 457–472