ASSA-PBN 2.0: A Software Tool for Probabilistic Boolean Networks

Andrzej Mizera[1], Jun Pang[1,2], and Qixia Yuan[1]

[1] Faculty of Science, Technology and Communication, University of Luxembourg
[2] Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg
firstname.lastname@uni.lu

Abstract. We present a major new release of ASSA-PBN, a software tool for modelling, simulation, and analysis of probabilistic Boolean networks (PBNs). PBNs are a widely used computational framework for modelling biological systems. The steady-state dynamics of a PBN is of special interest and obtaining it poses a significant challenge due to the state space explosion problem which often arises in the case of large biological systems. In its previous version, ASSA-PBN applied efficient statistical methods to approximately compute steady-state probabilities of large PBNs. In this newly released version, ASSA-PBN not only speeds up the computation of steady-state probabilities with three different realisations of parallel computing, but also implements parameter estimation and techniques for in-depth analysis of PBNs, i.e., influence and sensitivity analysis of PBNs. In addition, a graphical user interface (GUI) is provided for the convenience of users.

1 Introduction

Probabilistic Boolean networks (PBNs) [1, 2] are a modelling framework widely used to model gene regulatory networks (GRNs). A PBN model is capable to cope with uncertainties both on the data and model selection levels, allowing for systematic analysis of global network dynamics in the context of discrete-time Markov chains (DTMCs). It also provides means for quantifying relative influences and sensitivities of genes in their interactions and for characterising long-run behaviours of the whole genetic network. All these analyses are based on the steady-state probability distribution of the associated DTMC. Therefore, efficient computation of steady-state probabilities is a crucial foundation for analysing a PBN. Due to restricted computational resources, statistical methods are practically the only viable way to deal with large PBNs. However, the applicability of existing methods/tools for computing steady-state probabilities of PBNs are still limited by the network size, e.g., less than 100 nodes [3]. Moreover, to make the PBN framework generally accepted for mathematical modelling of biological systems, a user-friendly tool plays an important role but currently is still missing.

In this paper, we present a new release of ASSA-PBN, a tool designed for modelling, simulation and analysis of PBNs. ASSA-PBN in its previous version [4] has overcome the above mentioned network size limitation with the implementation of an efficient simulator and state-of-the-art techniques for steady-state analysis, e.g., the two-state Markov chain approach. The newly released version of ASSA-PBN contributes mainly

in three aspects. First, it speeds up the computation of steady-state probabilities of a PBN by using either multiple CPU/GPU core based parallelisation or structure-based parallelisation. Second, it implements parameter estimation and it supports in-depth analysis of PBNs, i.e., long-run influence and sensitivity analysis of PBNs. Third, it provides a graphical user interface (GUI) to ease user interactions with the tool.

A brief overview of the current functionality of ASSA-PBN is presented below. Items highlighted in bold are the new features added in version 2.0, and the tool is publicly available at `http://satoss.uni.lu/software/ASSA-PBN/`:

- modelling of PBNs in **high-level** ASSA-PBN format and converting a model from Matlab-PBN-toolbox format to ASSA-PBN format;
- random generation of PBNs;
- **efficient simulation** of a PBN;
- computation of steady-state probabilities of a PBN with either numerical methods or statistical methods (the two-state Markov chain approach, the Skart method, and the perfect-simulation method) [5, 6];
- **parallel computation** of steady-state probabilities of a PBN with either the two-state Markov chain approach or the Skart method;
- **parameter estimation** of a PBN;
- **long-run influence and sensitivity analysis** of a PBN;
- a command-line tool and a **GUI**.

2 Preliminaries

We briefly introduce the concept of PBN in this section. 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 selection 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. We focus on synchronous PBNs, where the values of all the nodes are updated simultaneously, while in asynchronous PBNs only one node is updated at a time step. 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 [7] and [2, page 4] for a detailed description of PBNs.

3 Tool Architecture and New Features

The architecture of ASSA-PBN consists of three main modules, i.e., a modeller, a simulator, and an analyser. The modeller provides a simple way to construct a PBN model

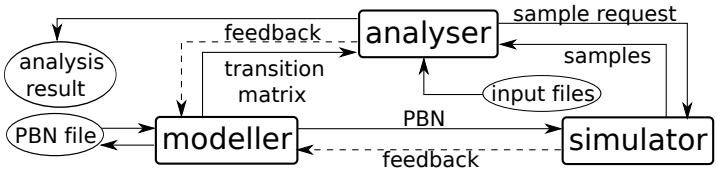

Fig. 1: Architecture of ASSA-PBN.

of a real-life biological system, e.g., a GRN. The simulator takes the PBN constructed in the modeller and performs simulation to produce trajectory samples. Based on the constructed model and generated samples, the analyser performs basic and in-depth analysis of the PBN. The analysis results can be used either to interpret the original system or to optimise the fitting of the system to experimental measurements. Figure 1 depicts the architecture of ASSA-PBN. The analyser requires different input files depending on the analysis task. While simulator and analyser rely on modeller as input, the simulation and analysis results can be used to optimise model fitting.

The newly released ASSA-PBN preserves the original architecture while making improvements to all the three modules. We proceed with describing the three modules in more details, while focusing on the newly implemented features.

Modeller. The PBN modeller can either load a PBN from a specification file or generate a random PBN (e.g., for benchmarking and testing purposes) complying with a given parametrisation [4]. In ASSA-PBN 2.0, a high-level PBN definition format is provided and visualisation of a PBN is supported in the GUI. The high-level PBN definition format provides a way to define Boolean function directly via its semantics instead of its truth table. The visualisation allows the user to check the details of each function and to interactively change the values of a predictor function. The modified PBN can then be used for the *long-run sensitivity with respect to function perturbation* analysis.

Simulator. Statistical approaches are practically the only viable option for the analysis of large PBNs. However, applications of such methods necessitate generation of trajectories of significant length. In order to make the analysis to execute in a reasonable amount of time, ASSA-PBN in its previous version applied the *alias method* [8] to sample the predictor function in each node. In this new version, the simulation process is sped up either with the use of the multiple CPU/GPU core parallelisation technique or with the structure-based parallelisation technique. The multiple core based technique [9] parallelises the simulation with the use of more cores while the structure-based parallelisation [10] achieves the same goal with the use of more memory. In order to parallelise the sample generation process, the algorithms for computing steady-state probabilities in the analyser have to be adjusted. More details are provided in the analyser section.

Moreover, visualisation of the simulation result is supported in ASSA-PBN 2.0 as well. Time-course evolution of selected node values can be displayed.

Analyser. The analyser of ASSA-PBN provides three main functionalities: basic computation of steady-state probabilities of a PBN, in-depth computation of long-run influ-

ences and sensitivities of a PBN, and parameter estimation of a PBN. In ASSA-PBN 2.0, the basic computation is largely improved with different parallelisation and multi-core techniques, while the other two are newly implemented.

Parallel computation of steady-state probabilities. The steady-state distributions can be computed either in an exact way or in a statistical way. Two iterative methods, i.e., the Jacobi method and the Gauss-Seidel method are implemented for exact computation; while three statistical methods, i.e., the perfect simulation, the two-state Markov chain approach, and the Skart method are implemented for the approximate computation [4]. Due to their large memory and time costs, the two iterative methods and the perfect simulation method are only suitable for analysing small-size PBNs [4].

Based on incremental sampling, the two-state Markov chain approach and the Skart method are capable of computing steady-state probabilities for large PBNs. In ASSA-PBN 2.0, we provide two types of techniques to speedup steady-state probability computation. Firstly, we implement our approach [9] of combining the Gelman & Rubin method with the two incremental methods to parallelise steady-state probability computation with multiple CPU/GPU cores. The Gelman & Rubin method is used to monitor that all the simulated chains have approximately converged to the steady-state distribution while the two-state Markov chain approach and the Skart method are used to determine the sample size required for computing the steady-state probabilities with specified precision. For a given precision, the lengths of trajectories used to estimate steady-state probabilities in the parallel approach are virtually the same as in the original two incremental methods. However, since the samples are generated with multiple cores in parallel, the processing time is significantly reduced. Details on the combined algorithms can be found in [9]. Secondly, we apply our structure-based parallel technique [10] to speedup the computation. This technique contributes in two aspects: reducing the network size by removing irrelevant nodes and by grouping nodes via merging their predictor functions. The key idea of this technique is to gain faster simulation speed with the use of larger memory.

Long-run influence and sensitivity. In a GRN, it is often important to distinguish which parent gene plays a major role in regulating a target gene and how sensitive the system is with respect to certain changes. PBNs feature quantification of the importances (formally known as long-run influences) and sensitivities [1, 7, 11]. ASSA-PBN 2.0 supports computation of long-run influences and sensitivities, i.e., the long-run influence of a gene on a specified predictor function, the long-run influence of a gene on another gene, the long-run sensitivity of a gene in a PBN, the long-run sensitivity of a gene with respect to function perturbation, and the long-run sensitivity of a gene with respect to selection probability perturbation. All these functionalities require the computation of several steady-state probabilities. For each probability, a trajectory of certain length has to be generated. Note that ASSA-PBN does not store the generated trajectory for the sake of memory saving. Instead, ASSA-PBN verifies whether the next sampled state of the PBN belongs to the set of states of interest and stores this information only. Therefore, a new trajectory is required when computing the steady-state probability for a new set of states of interest. ASSA-PBN 2.0 implements computation of steady-state probabilities of several sets of states in parallel with the two-state Markov chain approach [9], allowing the reuse of a generated trajectory. The crucial idea is that each time the next

state of the PBN is generated, it is processed for all state sets of interest simultaneously. Different sets require trajectories of different lengths and the lengths are determined dynamically through an iteration process. Whenever the trajectory is long enough for computing the steady-state probability of a certain set of states, the steady-state probability of this set will be computed and this set will not be considered in future iterations.

Parameter estimation. A key challenge in constructing a PBN model is the determination of the model parameters which make the model match the behaviour of the real system. A few algorithms [12, 13] have been proposed in literature for parameter estimation of biological systems. ASSA-PBN 2.0 applies the *particle swarm optimisation* algorithm for estimating parameters for a PBN. This algorithm is an iterative process for finding an optimal set of parameters. A set of parameters is called a particle and its fit to the experimental data is verified through a cost function. ASSA-PBN uses the *mean square error* (MSE) function, i.e., $MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2$, where n is the number of measurement data, $\hat{Y}_i$ is the ith measurement data point value, and Y_i is the computed steady-state probability corresponding to the ith data point. In each iteration, all the particles are updated and verified using the cost function. The particle that results in the minimum cost function value is the optimal particle. Particle values are updated based on the current values and the current best optimal particle values so that each particle is moving towards the direction of the current best optimal particle. Normally, the verification of the cost function for each particle requires the computation of steady-state probabilities of several sets of states. To make the computation as fast as possible, ASSA-PBN 2.0 provides the support for computing several steady-state probabilities in parallel see [9] for details).

4 Evaluation and Case Studies

As shown in [4], the first version of ASSA-PBN has shown a significant advantage in terms of speed compared to the Matlab tool optPBN [2]. We proceed to compare the performance of ASSA-PBN 2.0 with its previous version. This comparison is done both on randomly generated PBNs and a PBN constructed for a real-life biological system. The newly released version supports three types of parallelised computation of steady-state probabilities of a PBN. We show in [9] that for the multiple CPU core based parallelisation, the speed-up is approximately linear with the number of cores in our hardware environment (CPU cores up to 40). For the multiple GPU core based parallelisation, ASSA-PBN 2.0 can approximately achieve a speed-up of 200; while the structure-based parallelisation shows a promising performance for sparse networks with large percentage of leaves (for more details refer to [10]).

Moreover, we compared the performance of the three types of parallelisations with the sequential approach on an analysis of a 96-node PBN of apoptosis using the two-state Markov chain approach. In [5], the sequential version of the two-state Markov chain approach has been used for the long-run influence analysis on complex2 from each of its parent nodes. Seven steady-state probabilities are required to be computed in order to perform the analysis. We re-perform this computation with the three parallelised versions of the two-state Markov chain approach. Speed-ups of approximately 200 (GPU), 20 (multiple CPUs) and 10 (structure-based parallelisation) are obtained

with the use of the three different parallel computations. Detailed comparison of both the random networks and the real-life case-study can be found at `http://satoss.uni.lu/software/ASSA-PBN/benchmark`.

5 Future Developments

First, we plan to implement a suite of parameter estimation algorithms, e.g., genetic algorithms. Second, user-friendly improvements, such as support for the standard Systems Biology Markup Language (SBML) and graphical editing and visualisation of PBN models, will be introduced in the future releases of ASSA-PBN versions.

Acknowledgement. Qixia Yuan is supported by the National Research Fund, Luxembourg (grant 7814267). The authors also want to thank Gary Cornelius for his work on ASSA-PBN.

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. 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
4. Mizera, A., Pang, J., Yuan, Q.: ASSA-PBN: a tool for approximate steady-state analysis of large probabilistic Boolean networks. In: Proc. 13th International Symposium on Automated Technology for Verification and Analysis. Volume 9364 of LNCS., Springer (2015) 214–220
5. 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)
6. 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., Springer (2009) 120–134
7. Shmulevich, I., Dougherty, E.R.: Probabilistic Boolean Networks: The Modeling and Control of Gene Regulatory Networks. SIAM Press (2010)
8. Walker, A.: An efficient method for generating discrete random variables with general distributions. ACM Transactions on Mathematical Software **3**(3) (1977) 253–256
9. Mizera, A., Pang, J., Yuan, Q.: Parallel approximate steady-state analysis of large probabilistic Boolean networks. In: Proc. 31st ACM Symposium on Applied Computing, ACM Press (2016) 1–8
10. Mizera, A., Pang, J., Yuan, Q.: Fast simulation of probabilistic boolean networks. In: Proc. 14th International Conference on Computational Methods in Systems Biology. LNCS, Springer (2016) To appear.
11. Qian, X., Dougherty, E.R.: On the long-run sensitivity of probabilistic Boolean networks. Journal of Theoretical Biology **257**(4) (2009) 560–577
12. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proc. IEEE International Conference on Neural Networks. (1995) 1942–1948
13. Moles, C.G., Mendes, P., Banga, J.R.: Parameter estimation in biochemical pathways: a comparison of global optimization methods. Genome Research **13**(11) (2003) 2467–2474