VESTA: A Statistical Model-checker and Analyzer for Probabilistic Systems

Koushik Sen, Mahesh Viswanathan, Gul Agha
Department of Computer Science
University of Illinois at Urbana Champaign
{ksen,vmahesh,agha}@cs.uiuc.edu

Abstract

We give a brief overview of a statistical model-checking and analysis tool VESTA.

1 Overview of VESTA

VESTA is a tool for statistical analysis of probabilistic systems. It supports statistical model-checking [5, 6] and statistical evaluation of expected values of temporal expressions. For model-checking VESTA uses a sequence of inter-related statistical hypothesis testing to check if a property specified in probabilistic computation tree logic (PCTL) [2] or continuous stochastic logic (CSL) is satisfied by a stochastic model. Furthermore, VESTA supports the statistical computation of expected values of expressions written in a query language called *Quantitative Temporal Expressions* (or QUATEX in short). We next describe the various components of the tool.

Models

The analysis algorithms of VESTA are independent of the modelling language for probabilistic systems. VESTA only assumes that it can perform discrete-event simulation of the input probabilistic model. It also assumes that the probability measure associated with the paths of the model is fixed. VESTA works for any probabilistic modelling language for which we provide a discrete-event simulator with the following three interface methods:

1. `State initialState()` : returns the initial state of the model,
2. `State nextState(State current)` : computes a next state of the model from the `current` state using discrete-event simulation and returns the computed state,
3. `State duplicate(State current)` : duplicates the current state and returns it.

In particular, VESTA currently includes interfaces for two modelling languages:

1. a language to specify discrete-time and continuous-time Markov chains. The syntax of this language is closely related to that in PRISM [4], and
2. PMAUDE, a executable algebraic specification language which allows to describe models in probabilistic rewrite theories [3].

Statistical Model Checking

VESTA supports statistical model-checking algorithms for the transient part of probabilistic computation tree logic (PCTL) and continuous stochastic logic (CSL). Some example of properties that can expressed using these logics are as follows:

- $\mathcal{P}_{\leq 0.05}[\Diamond\ \textit{full}\ ]$: probability that a message queue eventually becomes full is less than or equal to 0.05.
- $\textit{reboot} \Rightarrow \mathcal{P}_{\geq 0.9}[\neg \textit{drop}\ \mathcal{U}^{\leq 5} \textit{recover}]$: if a message processing server is rebooted then the probability that it recovers within 5 minutes without dropping any message is greater than or equal to 0.9.

The model-checking algorithm $\mathcal{A}$ of VESTA takes as input a probabilistic model $\mathcal{M}$, a formula ϕ in CSL or PCTL, error bounds α^* and β^*, and three other parameters δ_1, δ_2, and p_s. The result of model checking on these parameters, denoted by $\mathcal{A}^{\delta_1,\delta_2,p_s}(\mathcal{M},\phi,\alpha^*,\beta^*)$, can be either *true* or *false*. The algorithm provides the following correctness guarantees.

If the model $\mathcal{M}$ satisfies the following conditions

C1: For every subformula of the form $\mathcal{P}_{\geq p}\psi$ in the formula ϕ and for every state s in $\mathcal{M}$, the probability that a path from s satisfies ψ must not lie in the range $[\frac{p-\delta_1-\alpha^*}{1-\alpha^*}, \frac{p+\delta_1}{1-\beta^*}]$;

C2: For any subformula of the form $\phi_1\ \mathcal{U}\ \phi_2$ and for every state s in $\mathcal{M}$, the probability that a path from s satisfies $\phi_1\ \mathcal{U}\ \phi_2$ must not lie in the range $(0, \frac{\delta_2}{p^{(N-1)}(1-p_s)^{(N-1)}}]$, where N is the number of states in the model $\mathcal{M}$ and p is the smallest non-zero transition probability in the system.

Then the algorithm provides the following guarantees

$$Prob[\mathcal{A}^{\delta_1,\delta_2,p_s}(\mathcal{M},\phi,\alpha^*,\beta^*) = \textit{true} \mid \mathcal{M} \not\models \phi] \leq \alpha^*$$
$$Prob[\mathcal{A}^{\delta_1,\delta_2,p_s}(\mathcal{M},\phi,\alpha^*,\beta^*) = \textit{false} \mid \mathcal{M} \models \phi] \leq \beta^*$$

Condition C1 requires that the model be such that for any subformula $\mathcal{P}_{\geq p}\psi$, the probability of ψ being satisfied at a state be bounded away from p. Condition C2 requires that

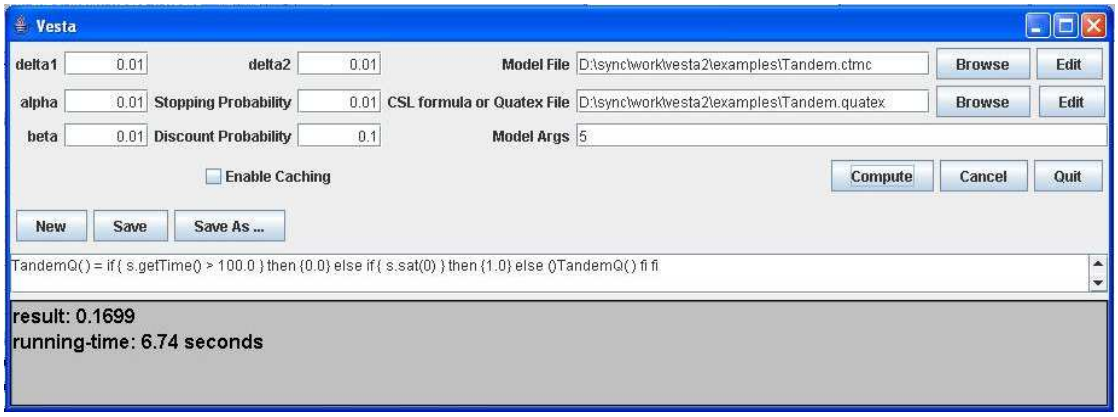

Figure 1. VESTA **Screenshot**

either an until formula does not hold in a state or it holds with some probability that is bounded away from 0. Under such circumstances, we guarantee that the probability of error of $\mathcal{A}$ is within the required bounds.

VESTA performs statistical model-checking by invoking a series of inter-dependent statistical hypothesis testing. The details of the algorithm can be found in [5].

Quantitative Temporal Expressions

To gain more insight into a probabilistic model, we realized that model-checking is not sufficient. For example, we wanted to know the expected number of clients that get connected to a server in the presence of SYN flood attack. Therefore, in addition to model-checking, VESTA supports statistical evaluation of the expected values of expressions written in a query language called *Quantitative Temporal Expressions* (or QUATEX in short). In QUATEX, one can easily query about expected time, expected power consumption, expected queue size etc. Some example queries that can be encoded are as follows:

1. What is the expected fraction of clients that successfully connect to a server under DoS attack?
2. What is the probability that a client connected to a server within 10 seconds after it initiated the connection request?

The expected value of a QUATEX expression is statistically evaluated with respect to two parameters α and δ provided as input. Specifically, we compute the expected value iteratively by sampling until the size of $(1-\alpha)100\%$ confidence interval for the expected value gets bounded by δ. A detailed discussion of the QUATEX is provided in [1].

2 Implementation

We have implemented VESTA in Java 1.5. The tool is available for download from `http://osl.cs.uiuc.edu/~ksen/vesta2/`). We have successfully used VESTA to analyze several DTMC (discrete-time Markov chains), CTMC (continuous-time Markov chains), and probabilistic rewrite theory models. The details of these case studies can be found in [5, 1].

Acknowledgements

This research has strongly benefited from collaboration with José Meseguer, particularly in the development of QUATEX and PMAUDE. The second author was supported in part by DARPA/AFOSR MURI Award F49620-02-1-0325 and NSF 04-29639. The other two authors were supported in part by ONR Grant N00014-02-1-0715.

References

[1] G. Agha, C. Gunter, M. Greenwald, S. Khanna, J. Meseguer, K. Sen, and P. Thati. Formal modeling and analysis of dos using probabilistic rewrite theories. In *Workshop on Foundations of Computer Security (FCS'05)*, 2005.

[2] H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Comp.*, 6(5):512–535, 1994.

[3] N. Kumar, K. Sen, J. Meseguer, and A. Agha. A rewriting based model for probabilistic distributed object systems. In *Proc. of Formal Methods for Open Object-Based Distributed Systems*, volume 2884 of *LNCS*, pages 32–46, 2003.

[4] M. Z. Kwiatkowska, G. Norman, and D. Parker. Prism 2.0: A tool for probabilistic model checking. In *1st Int. Conference on Quantitative Evaluation of Systems (QEST'04)*, pages 322–323. IEEE, 2004.

[5] K. Sen, M. Viswanathan, and G. Agha. On statistical model checking of probabilistic systems. In *17th conference on Computer Aided Verification (CAV'05)*, LNCS (To Appear). Springer, July 2005.

[6] H. L. S. Younes and R. G. Simmons. Probabilistic verification of discrete event systems using acceptance sampling. In *Proc. of Computer Aided Verification (CAV'02)*, volume 2404 of *LNCS*, pages 223–235, 2002.