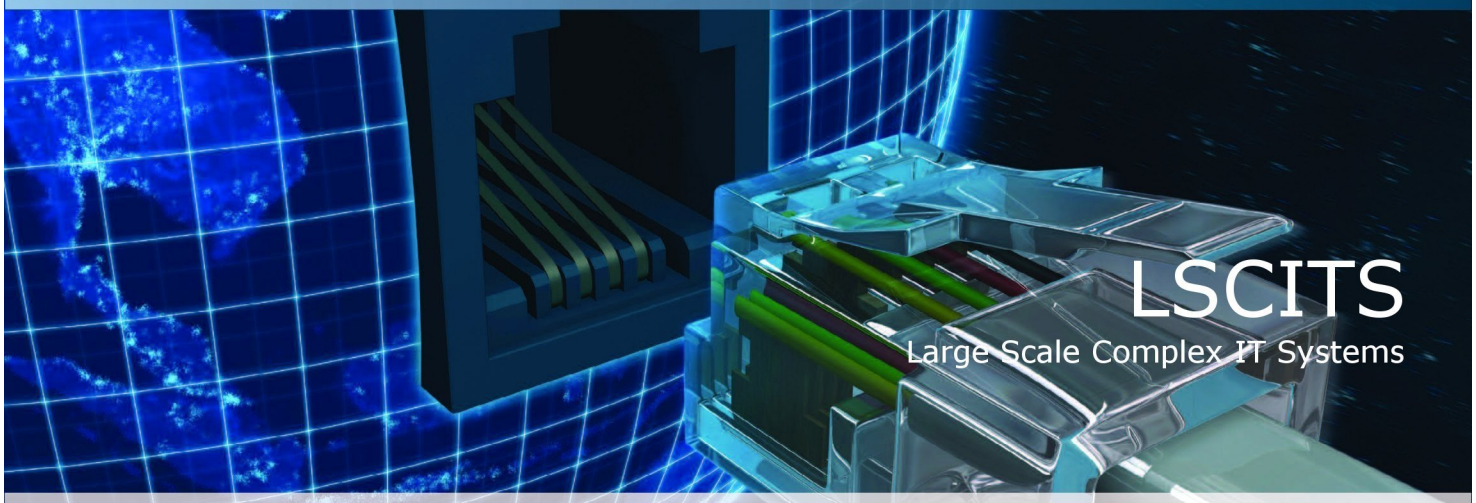

Quantitative Verification

Quantitative verification is a mathematically-based technique for establishing the correctness, performance and reliability of systems that exhibit stochastic behaviour. Given a precise mathematical model of a real-world system, and formal specifications of quantitative properties of this system, an exhaustive analysis of these properties is performed. Example properties include the probability that a fault occurs within a specified time period, and the expected power consumption of a complex IT system under a given workload.

Introduction

Many complex IT systems are characterised by random events and/or real-time delays that necessitate *quantitative* analysis in order to establish properties such as the probability of successful task completion within a given time limit. Probability is also used to quantify unreliable or unpredictable behaviour, for instance in fault-tolerant IT systems and computer networks, where properties such as component failure and packet loss can be described probabilistically.

Traditionally, quantitative evaluation of systems involved performance analysis: a probabilistic model of the system was derived and used to perform analytical, simulation-based or numerical calculations yielding the desired quantitative measures. In recent years, a complementary, automated verification technique for probabilistic models has been developed [1, 2]. Termed *probabilistic model checking*, this technique supports the calculation of the *likelihood* of the occurrence of certain events or the *expected* value of given measures within a complex IT system, e.g.,

- for a system prone to failures: "what is the *probability* of a critical fault occurring during execution?";
- for a communication protocol: "what is the *worst case* expected time for delivering a data packet?".

The Models

Probabilistic models used in quantitative verification are typically variants of discrete-space Markov chains [1]:

- *Discrete-time Markov chains* (DTMCs) consist of discrete states—representing the configurations of the system—and the transition probabilities between these states. They are particularly useful in modelling self-stabilising systems that employ randomisation as a symmetry breaker.
- *Markov decision processes* (MDPs) extend DTMCs by additionally allowing non-deterministic behaviour that is required, for example, to model complex IT systems comprising asynchronous concurrent components.
- *Continuous-time Markov chains* (CTMCs) can model the transition rates between the states of systems that have discrete states, but where time progresses continuously. They are suitable for the quantitative verification of the performance and dependability of complex IT systems.

Figure 1 illustrates a cluster of servers whose components can each break down and be repaired with known rates, and which can be modelled as a CTMC [3].

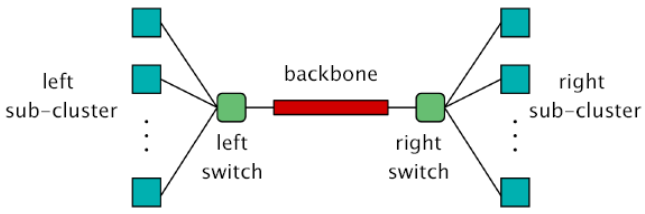

Figure 1: Cluster architecture

Quantitative Verification

The Probabilistic Model Checker PRISM

PRISM [4, 5] is a probabilistic model checker developed by the Quantitative Analysis and Verification group at the University of Oxford. It supports all three types of probabilistic models described in the previous section: DTMCs, MDPs and CTMCs. Given one of these probabilistic models described in its *modelling language*, PRISM parses the model description and constructs an internal representation of the probabilistic model, computing the reachable state space of the model and discarding any unreachable states. This represents the set of all feasible configurations which can arise in the modelled system. Next, the specification is parsed and appropriate model checking algorithms are performed on the model by induction over syntax.

In some cases, such as for properties which include a probability bound, PRISM will simply report a true/false outcome, indicating whether or not each property is satisfied by the current model. More often, however, properties return quantitative results and PRISM reports, for example, the actual probability of a certain event occurring in the model. Furthermore, PRISM supports the notion of experiments, which is a way of automating multiple instances of model checking. This allows the user to easily obtain the outcome of one or more properties as functions of model and property parameters. The resulting table of values can either be viewed directly, exported for use in an external application such as a spreadsheet, or plotted as a graph. For the latter, PRISM incorporates substantial graph-plotting functionality. This is often a very useful way of identifying interesting patterns or trends in the behaviour of a complex IT system.

Figure 2 shows the result of a PRISM experiment that analyses the probability that an N-server cluster with the architecture in Figure 1 takes more than T time units to recover from insufficient quality of service (QOS). The failure and repair rates for the components of the cluster are given, and the cluster is deemed to provide insufficient QOS if less than 75% of its servers are operational and connected via switches and the backbone.

PRISM has a graphical user interface, featuring a built-in text-editor for the PRISM modelling language. Alternatively, all model checking functionality is also available in a command-line version of the tool. PRISM is a free, open source application. It presently operates on Linux, Unix, Windows and Macintosh, and can be downloaded from [5].

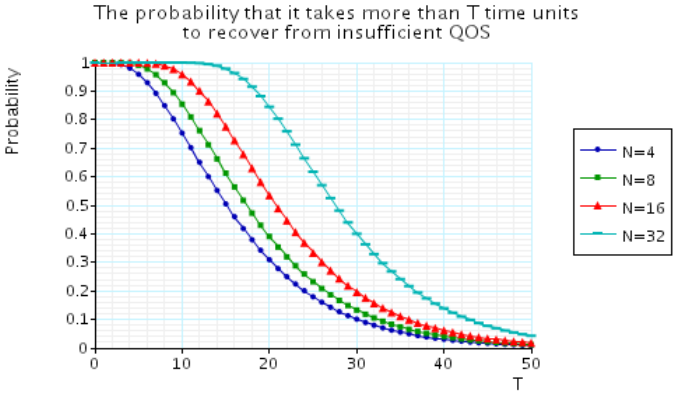

Figure 2: Quantitative verification results

References and Further Information

1. M. Kwiatkowska. Quantitative verification: Models, techniques and tools. In *Proc. 6th Joint Meeting of the European Software Engineering Conf. and the ACM SIGSOFT Symp. Foundations of Software Engineering*, pages 449-458. ACM Press, September 2007.
2. M. Kwiatkowska et al. Stochastic model checking. In M. Bernardo and J. Hillston, editors, *Formal Methods for the Design of Computer, Communication and Software Systems: Performance Evaluation*, volume 4486 of LNCS, pages 220-270. Springer, 2007.
3. B. Haverkort et al. On the use of model checking techniques for dependability evaluation. In *Proc. 19th IEEE Symp. Reliable Distributed Systems*, pages 228-237, October 2000.
4. M. Kwiatkowska et al. Quantitative analysis with the probabilistic model checker PRISM. *Electronic Notes in Theoretical Computer Science*, 153(2):5-31, 2005.
5. PRISM web site, http://www.prismmodelchecker.org/.

EPSRC Engineering and Physical Sciences Research Council