As larger and more complex biological pathways are being modeled, long simulation results can be obtained. However, this implies that the modeling framework is able to perform model checking as a checker (MC2). MC2 has the advantage of being independent from piecewise-linear differential equations, which also includes the format for models to adhere to. Clarke et al. (2008) introduced BioLab, an algorithm to verify properties written in probabilistic temporal logics might need to refer to the values of previous states in order to determine the truth-value of a property holds. The above paragraph describes how MIRACH decides the truth-value of properties for a single simulation run. However, for stochastic models, each simulation run produces different results.

2 METHODS

To build a model checker, one of the first steps is defining a formalism to express the rules to be checked. For MIRACH, we have decided to use PLTL (Probabilistic Linear Temporal Logic) because it is sufficient for probabilistic model checking in general and is easy to write and interpret. Due to space constraints, PLTL syntax and semantics are supplied as supplementary material (Supp. Doc. 1).

To address the former question, the sample efficient hypothesis testing approach where the probability that a property holds is computed using a predetermined number of samples and compared with the \( \theta \).

Received and revised on December 2, 2010; accepted on December 27, 2010

1 INTRODUCTION

As larger and more complex biological pathways are being modeled, the manual validation of these models becomes tedious if not impossible. Therefore, there is a growing interest in the development and application of model checking algorithms for biological pathway models.

PRISM is a probabilistic model checker that is widely used in many different domains (Heath et al., 2008). As PRISM is meant for a wide range of domains, it has its own specific PRISM format for models to adhere to. Clarke et al. (2008) introduced BioLab, an algorithm to verify properties written in probabilistic boundless linear temporal logic, using the BioNetGen modeling (rule-based) framework. Genetic Network Analyzer (GNA) is software for the modeling and simulation of qualitative models in the form of piecewise-linear differential equations, which also includes the ability to do model checking (Batt et al., 2005). Donaldson and Gilbert (2008a) developed a Monte Carlo offline-based model checker (MC2). MC2 has the advantage of being independent from the modeling framework and is able to perform model checking as long simulation results can be obtained. However, this implies that the full simulation needs to be completed and all traversed states recorded before model checking can commence. This wastes CPU and storage resources if the decision of validity or rejection for the simulation can be determined early in the simulation run. On-the-fly model checking does the exactly this. It carries out model checking during the simulation run and results need not be recorded as simulation runs are only executed for as long as a decision needs to be made.

In this article, we present an on-the-fly probabilistic model checker, MIRACH, for quantitative pathway models that supports popular formats such as SBML (Hucka et al., 2003) and CSML (http://www.csml.org/). This quantitative model checker, MIRACH, would certainly be a valuable addition to the available arsenal of qualitative (GNA) and rule-based (BioLab) model checkers.

734 © The Author 2011. Published by Oxford University Press. All rights reserved. For Permissions, please email: journals.permissions@oup.com
Table 1. MIRACH versus MC2 (PLTLc) using Levchenko model

<table>
<thead>
<tr>
<th></th>
<th>100 samples</th>
<th>1000 samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIRACH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Initialization</td>
<td>6.85 (0.24)</td>
<td>6.86 (0.31)</td>
</tr>
<tr>
<td>Simulation and Checking</td>
<td>5.34 (0.20)</td>
<td>40.74 (0.90)</td>
</tr>
<tr>
<td>Total time</td>
<td>12.19</td>
<td>47.6</td>
</tr>
<tr>
<td>MC2 (PLTLc)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Run simulation and log results</td>
<td>12.14 (0.40)</td>
<td>107.95 (1.52)</td>
</tr>
<tr>
<td>Load results and check</td>
<td>10.13 (0.29)</td>
<td>88.58 (1.11)</td>
</tr>
<tr>
<td>Total time</td>
<td>22.27</td>
<td>196.5</td>
</tr>
</tbody>
</table>

One hundred samples indicate that 100 simulation runs were executed (similarly for 1000 samples). Results shown are in seconds and are the average of 20 repeated runs. The number in brackets is the SD. All runs are performed on a laptop with 1.6 GHz dual core processor and 2 G RAM running on Linux.

As for the latter, we implemented Wilson interval (Wilson, 1927) to estimate the confidence interval of the probability that holds. We have chosen to use Wilson interval instead of the simpler normal approximation interval because normal approximation is known to perform poorly when the sample probability is close to 0 or 1 and fail completely when it is at 0 or 1. Due to this, one cannot devise a sequential sampling algorithm that stops sampling once the confidence interval falls below a certain value (user defined). Wilson interval does not have these limitations and allows us more flexibility and efficiency in our model checker.

3 PERFORMANCE

It is not difficult to appreciate that an online approach is almost certainly more efficient than offline in terms of time efficiency since it only runs as long as it needs to and does not read and write to the hard disk.

One offline model checker similar to MIRACH is MC2 (PLTLc) by Donaldson and Gilbert (2008a). Both model checkers are written in Java and supports PLTL. Therefore, we will use MC2 (PLTLc) to illustrate the differences between online and offline checkers.

To draw comparisons between the two model checkers, we need a sample model that can be run on both of the checkers. Our model of choice is a SBML model by Levchenko et al. (2000) as it was also used as an example in Donaldson and Gilbert’s (2008a) paper.

From Table 1, we see that MIRACH outperforms MC2 (PLTLc) and the time saved increases with sample size. When comparing the runtime for just 1000 samples, the time saved by using MIRACH is already 400%. The sample size needed depends on the problem at hand but in most situations, thousands of samples are insufficient especially with the growing trend of using model checkers as part of parameter estimation routine (Batt et al., 2010; Donaldson et al., 2008b).

Our group has also combined MIRACH with parameter estimation (Koh et al., 2010) to investigate cell fate determination of gustatory neurons in Caenorhabditis elegans (Saito et al., 2006).

In that work, we had to run 20 million samples.

Another performance measure is the minimum memory requirement. Precise memory requirements depend on several factors such as the model used and the properties to be checked. The memory requirement of online checking is likely to be higher than offline checking because the offline method does not carry out checking and simulation concurrently. As described in Section 2, in the checking step, MIRACH needs to store the values of involved species in memory (RAM) when a LTL cannot be decided (neither TRUE nor FALSE) at that time point. However, even in an extreme case, where there are 100 species involved and that property cannot be decided for 100,000 time points, the additional memory (RAM) needed is still <80 MB (100 × 10,000 × 8 bytes). Note that this memory space used will be freed once that particular simulation ends and will not increase with the number of simulation runs.

4 CONCLUSION

In this article, we have presented an efficient model checker, MIRACH 1.0, for validating the ever-growing biological pathway simulation models—both in complexity and quantity. Major contributions include the implementation of the more efficient on-the-fly approach that saves significant amounts of computation time with minimal memory increase, the ability to accept quantitative models directly in the popular SBML and CSML formats, and the first model checker to be integrated with the HFPNe (Nagasaki et al., 2010) simulation engine, an expressive and powerful Petri net framework for defining biological pathway models.

ACKNOWLEDGEMENTS

We are thankful to Sharene Lin for her help rendered in the manuscript preparation and proofreading of this manuscript.

Funding: Singapore National Research Foundation grant NRF-G-CRP-2997-04-082(d) (to L.W. and C.H.K., in parts); National University of Singapore NGS scholarship (to C.H.K., in parts).

Conflict of Interest: none declared.

REFERENCES


735