Analysis of a Gossip Protocol in PRISM

Marta Kwiatkowska, Gethin Norman and David Parker

Oxford University Computing Laboratory, Wolfson Building, Parks Road, Oxford, OX1 3QD

ABSTRACT

Gossip protocols have been proposed as a robust and efficient method for disseminating information throughout dynamically changing networks. We present an analysis of a gossip protocol using probabilistic model checking and the tool PRISM. Since the behaviour of these protocols is both probabilistic and nondeterministic in nature, this provides a good example of the exhaustive, quantitative analysis that probabilistic model checking techniques can provide. In particular, we compute minimum and maximum values, representing the best- and worst-case performance of the protocol under any scheduling, and investigate both their relationship with the average values that would be obtained through simulation and the precise scheduling which achieve these values.

1. INTRODUCTION

Gossip protocols are a class of communication protocols which, inspired by the way that gossiping propagates messages in social networks, disseminate content through a network based on periodic exchanges of data with random members of the network. These techniques are designed to function robustly and efficiently on networks that are large, heterogeneous and dynamic in nature. They are hence becoming increasingly important due to the prevalence of, for example, mobile ad-hoc networks, wireless sensor networks and peer-to-peer technologies.

Gossip-based protocols require each node of the system to periodically exchange information with a number of its peers. The choice of which peers nodes communicate with is crucial to how information gets disseminated through the network. Theoretically, a node could randomly select a subset of all the available nodes in the network. In practice, however, this is not feasible since it would require each node to keep a complete network membership table which is expensive to store and maintain. In this paper, we study the *peer sampling* framework of [9] where each node instead maintains a relatively small local membership table providing a partial view of the network which is periodically updated using a gossiping procedure.

Probabilistic model checking is a formal verification technique for the analysis of stochastic systems. It is based on the construction of a probabilistic model from a precise, high-level description of a system's behaviour. A quantitative analysis of this model is then performed, by applying a combination of exhaustive search techniques and numerical solution methods. In this paper, we investigate how probabilistic model checking can be used to study gossip protocols. The majority of existing analyses of such systems are based on discrete-event simulation. In contrast, probabilistic model checking provides both an exhaustive search of all possible behaviours of the system, including best- and worst-case scenarios, and exact, rather than approximate, quantitative results. Of course, a trade-off inevitably exists. Simulation-based approaches are scalable to much larger and more complex models, at the expense of exhaustiveness and numerical accuracy. The intention of this work is not to show that model checking is 'better' than simulation-based approaches, but rather to highlight that model checking can be used in conjunction with simulation to provide additional insights into a system.

Gossip protocols exhibit both nondeterministic and probabilistic behaviour. Nondeterminism arises because we consider a distributed network in which the activities of individual nodes occur asynchronously. Other actions, such as the random selection of a node with whom to exchange information, are inherently probabilistic. We model the protocol as a Markov decision process (MDP) using the probabilistic model checker PRISM [8, 16]. We investigate the expected number of rounds of gossiping required for the nodes to form a connected network and how the expected path length between nodes evolves over the execution of the protocol. The presence of nondeterminism means that these measures can take a range of values. Hence, we compute minimum and maximum values, representing the best- and worst-case performance of the protocol under any scheduling of nodes. We investigate the relationship of these results with average values, as would be obtained through simulation. We also use PRISM to identify the precise situations under which the best- and worst-case behaviour arises.

Related Work. Simulation-based studies of the peer sampling service used in the paper can be found in [9]. A survey of how different formal verification techniques can be applied to the analysis of gossip protocols is presented in [3]. Probabilistic model checking and PRISM have been used by Fehnker and Gao [6] to study the influence of different modelling choices on message propagation in flooding and gossiping protocols, and by Werner and Schmitt [19] to analyse the performance of a secure authenticated query flooding protocol. PRISM has also been used to model and analyse a number of different wireless protocols, for example, the IEEE 802.11 backoff mechanism [14] and device discovery in Bluetooth [5]. See the PRISM case study repository [16] for more details and further examples.

2. PROBABILISTIC MODEL CHECKING AND PRISM

Probabilistic model checking is a formal verification technique for systems that exhibit stochastic behaviour. It involves the construction, from a precise description in a high-level specification language, of a probabilistic model describing the behaviour of the system to be analysed. Typically, this model takes the form of a state-transition system, augmented with probabilistic information. States of the model correspond to the possible configurations of the system; transitions represent the ways in which the system can evolve between these states and include information about the likelihood (or timing) with which they will occur.

A *discrete-time Markov chain* (DTMC) labels each transition in the model with a probability such that the sum over the outgoing transitions for each state equals one. These give the probability, from any state in the model, of moving to any other state in the next discrete time-step. *Markov decision processes* (MDPs) extend DTMCs by also modelling nondeterministic behaviour. More precisely, each state of an MDP is associated with a set of probability distributions over the states of the MDP. A transition between states of the model occurs in two steps: first, there is a nondeterministic choice between available distributions in the current state; second, the next state is selected at random according to the chosen distribution.

The behaviour of a DTMC is fully probabilistic. We can, in standard fashion [10], define a probability space over infinite paths through the model and thus quantify the likelihood of a particular event occurring. For an MDP, on the other hand, we can only reason about its probabilistic behaviour once the nondeterministic choices have been resolved. We refer to a particular resolution of nondeterminism as an *adversary* (sometimes also called a scheduler or policy). Under a given adversary, the execution of an MDP can be represented by a (potentially infinite state) DTMC. To reason about MDPs, we can compute the *minimum* or *maximum* probability of an event occurring, over all adversaries, i.e. over all possible resolutions of nondeterminism.

In probabilistic model checking, properties to be analysed of a system are typically expressed in temporal logics, such as PCTL [7]. For example, the following:

$$\mathbf{P}_{\max=?}[\ \mathbf{F}^{\leq T}\ \text{“}error\text{”}\ ]$$

represents the maximum probability of an error occurring within T time-steps. DTMCs and MDPs can also be augmented with *reward structures*, which label states and transitions with numerical values. These can be used to reason about a wide range of quantitative measures, such as “elapsed time”, “energy consumed” or “number of messages sent”. For example:

$$\mathbf{R}_{\{\text{“}rounds\text{”}\}\min=?}[\ \mathbf{F}\ \text{“}terminated\text{”}\ ]$$

represents the minimum expected number of rounds of an algorithm required before it terminates.

Another alternative to DTMCs is continuous-time Markov chains (CTMCs) which offer a dense model of time. In CTMCs, transitions are labelled with rates which represent parameters of negative exponential distributions and give the delay until the transition is enabled. For further details see, for example, [1].

PRISM [8, 16] is a probabilistic model checking tool developed at the Universities of Birmingham and Oxford. It provides support for several types of probabilistic models, including DTMCs, MDPs and CTMCs, and provides a simple, high-level modelling language for describing such models. The tool automatically calculates the results for temporal logic queries, such as those given above. The underlying computation in PRISM involves a combination of graph-theoretical algorithms and numerical solution methods. See, for example, [4, 2, 17] for further details. Notable features of PRISM include the uses a state-of-the-art *symbolic* approach, allowing compact representation and efficient manipulation of large, structured models, and a discrete-event simulation engine, generating approximate solutions through Monte Carlo methods and sampling.

3. MODELLING THE PROTOCOL

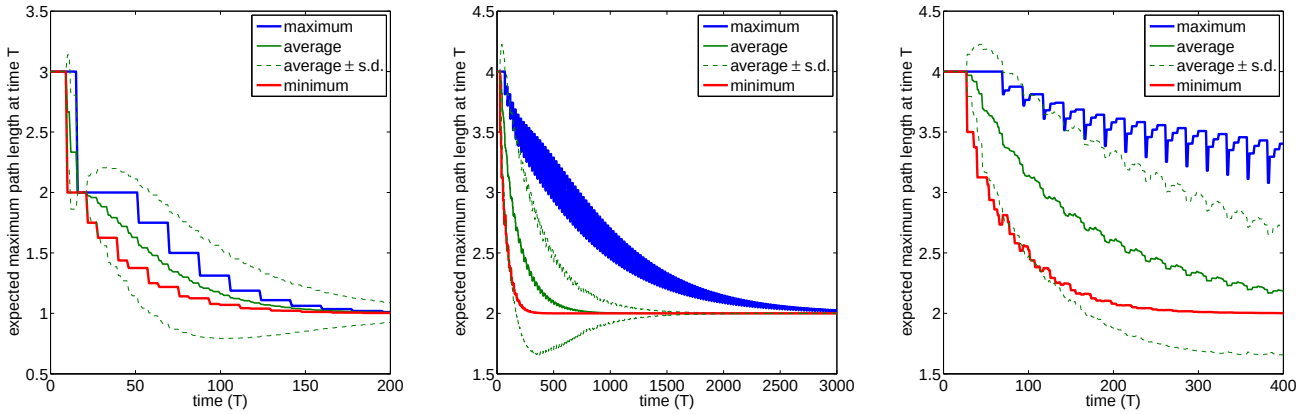
We base our gossip protocol model on the framework of [9]. We assume a network of N nodes, each with an address that is required for sending a message to it (as in, for example, a wireless network). Each node maintains a *partial view* of the network: a list of up to c ($< N$) *node descriptors*, each of which comprises a node's network address and an *age* that represents the freshness of the descriptor.

Periodically each node will execute a gossiping algorithm which exchanges the information contained in their views. This allows information about the topology of the network (and changes to it) to be propagated between nodes. The fundamental idea behind gossip-based protocols is that each node passes information to a small, random subset of the other nodes. This prevents overloading of the network with large numbers of superfluous messages.

The framework of [9] includes a number of design choices regarding the gossip protocol. For example *peer selection*, which is the choice of a node to exchange views with, can be done at random or by selecting the node in the view with the oldest age. We will assume a random choice. Several strategies also exist for *view propagation*, which defines *how* two nodes exchange their views, e.g. *push* or *pushpull*: one- or two-way exchange of views between the sending and receiving node. We use the former and assume that the sending node pushes the whole of its view to the receiving node. Finally, the receiving node requires a strategy for *view selection*, which combines the incoming and the existing view information. In our model, we in fact use *hop-counts* as a coarse (bounded) measure of the age of each node descriptor. A receiving node increments the the hop-count of all the incoming descriptors, merges these with the descriptors in its own view (keeping the entry with the youngest count in cases of duplication) and then keeps the c newest entries from the combined set.

With regards to the timing of the protocol, we assume that the exchange of data between nodes occurs periodically (with some fixed period) and that each node sends its data exactly once in each round of execution. Such a scheme can be achieved in practice through synchronisation of local clocks. Due to the distributed nature of the system, however, the order in which the nodes participate in each round is unknown (and may be different each time).

The gossip protocol therefore exhibits both probabilistic behaviour (random peer selection) and nondeterministic behaviour (scheduling of nodes within a round) and is naturally modelled as a Markov decision process (MDP). We constructed a model of the gossip protocol in PRISM, using



(a) N=3 (b) N=4 (c) N=4 (zoom)

Figure 1: Expected path length: Minimum, maximum and average (± standard deviation).

the tool's high-level description language.[1]

We opt to build a small, but detailed, model of the system. The model comprises several components: one for for each node in the network and one representing a scheduler who chooses (nondeterministically) the order in which nodes execute the protocol. The state of each node in the network includes its current view, information about which part of the protocol it is currently executing and a buffer to store incoming data from other nodes. The scheduler simply keeps track of which nodes have sent data in the current round. For simplicity, we assume that the sending and updating of views is an atomic step, i.e. throughout the process of one node sending its view to another, no other communication occurs in the network.

Because of the detailed nature of the model and the corresponding state space size, we consider only very small sizes of network (N=3, 4) and fix a local view size of $c = 2$. It is possible, though, that anomalies observed in these small models will also be exhibited by networks of a more realistic size. As regards the initial configuration of the model (i.e. the initial local views of each node), we assume that one node is "public" and that all other nodes know the address of this node (but it is not aware of the others). This configuration is suitably realistic and ensures that a connected network is possible.

For N=3, the model has 829 states and 946 transitions; for N=4, it has 74,034 states and 87,410 transitions. These MDPs are constructed by PRISM in 1.73 seconds and 95.0 seconds, respectively (on a 2GHz PC with 2GB RAM).

4. ANALYSIS AND RESULTS

We used PRISM to analyse the performance of the gossip protocol described in the previous section, illustrating the kind of analysis that can be performed with probabilistic model checking. We concentrate on how the topology of the network induced by the local views of the nodes varies over time, investigating first the maximum path length (longest route between nodes) and then the time for the network to become connected. For the former, we demonstrate an analysis of the best- and worst-case behaviour of the model, and how this relates to the average case. For the latter, we show how obtaining best and worst-case adversaries (schedulers) of the MDP can help to identify the scenarios in which the best and worst cases occur. The next two sections describe these two illustrative properties in some detail; following this, we list some other properties that could be analysed.

[1]The full PRISM model can be found at:
`http://www.prismmodelchecker.org/casestudies/gossip.php`

4.1 Best, worst and average case behaviour

We first study how the longest path length between nodes in the network varies over the execution of the protocol. This is done using PRISM properties of the form:

$$\mathbf{R}_{\{\text{“}path\_len\text{”}\}\min=?}[\,\mathbf{I}^{=T}\,]$$
$$\mathbf{R}_{\{\text{“}path\_len\text{”}\}\max=?}[\,\mathbf{I}^{=T}\,]$$

which represent the minimum/maximum expected value of "*path_len*" at time instant T. This assumes that we have added a reward structure called "*path_len*" to the PRISM model, which associates with each state of the MDP a value representing the longest path length between any two nodes at that point (for the case where the graph is not connected, we let "*path_len*" be N).

These properties give the *minimum* and *maximum* values over all possible resolutions of nondeterminism in the MDP which, in this model, means quantifying over all possible schedulings of the nodes. This allows us to determine the best- and worst-case behaviour of the system in any eventuality. Since the gossip protocol makes random choices, its execution under a particular scheduling is probabilistic. Hence our use of minimum/maximum *expected* values.

It is also possible, with a simple modification of the PRISM model, to compute the *average* value of the longest path length over time. This is done by replacing nondeterminism in the scheduler component of the PRISM model with uniform probabilistic choices, yielding a DTMC instead of an MDP. Although this is no longer an accurate model of the scheduling, it is interesting because the results computed from the DTMC model can be seen as the values that would be obtained through simulation by averaging the results obtained over a large number of simulation runs.

Furthermore, we can also calculate the standard deviation of the random variable corresponding to the longest path length; again this is information that could be obtained (approximately) through simulation. Computing this value

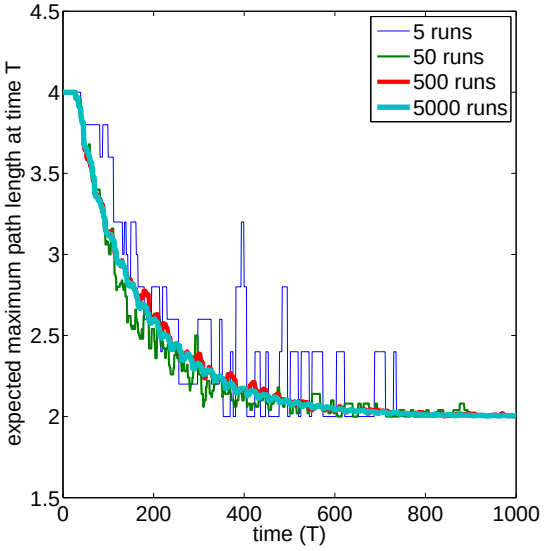


Figure 2: Simulation results (N=4).

(exactly) with PRISM is done by adding a second reward structure, which associates each state of the model with the square of the "*path_len*" value, and then using the equivalence $\sigma(X)^2 = \mathbf{E}(X^2) - \mathbf{E}(X)^2$.

Figures 1(a) and 1(b) show the full set of these results for N=3 and N=4 nodes, respectively. The thicker solid lines show the minimum and maximum expected longest path length after T time-steps, for a range of values of T. In between these, the thinner solid line shows the average (i.e. expected) value for the same time points. The dashed lines indicate the standard deviation.

The results demonstrate that there is a significant difference between the minimum and maximum values, i.e. between the best- and worst- case behaviour of the protocol. Both values eventually stabilise at 1, for N=3, and 2, for N=4 (in each case, this is the shortest possible longest path length since the local views are of size two).

Note that, despite the discontinuities seen in the graphs, these results are exact and have been computed for every time step. In fact, plots of this kind are typical for systems which operate in rounds, each one requiring multiple discrete time-steps. In the case of the maximum values for N=4 (Figure 1(b)) we see that, although the longest path length is converging towards two, there are many small jumps where it increases and then decreases again. This phenomenon can be observed more clearly in Figure 1(c), which shows the same plots for a smaller range of time values. This behaviour can be attributed to the fact that, within each round of the protocol, the adversary can schedule nodes in a malicious fashion such that the longest path length temporarily increases. Because of the design of the protocol, though, the expected longest path length decreases as the rounds progress. Figure 1(c) also demonstrates that, although they are not as pronounced, the same fluctuations occur for the other plots (average and minimum values).

It is also interesting to observe the relationship between the minimum and maximum values (obtained from the MDP) and the average and standard deviation values (obtained from the DTMC). For the case where N=3, we see that the average values and standard deviation give a slightly pessimistic view: in fact, the best and worst possible behaviour is within the bounds given by the standard deviation. For N=4, on the other hand, the worst-case (maximum values) are significantly higher. Also, for N=3, the average case falls roughly half-way between the minimum and maximum values, where as, for N=4, it is much closer to the best-case (minimum) behaviour.

Lastly, to illustrate the relationship between the above results and those obtained from discrete-event simulation, in Figure 2 we have included the average results over 5, 50, 500 and 5,000 simulation runs for the network of 4 nodes. The plots demonstrate that, as we increase the number of simulation runs, the average values converge to the (average) results for the DTMC model given in Figure 1(b).

4.2 Best and worst-case scheduling

One weakness with the property analysed in the previous section is the notion of time used. Each time-step (as measured on the X-axis in the plots) corresponds to a single transition in the model. Because the model comprises a set of processes running in parallel this does not give an accurate measure of elapsed time. Since the gossip protocol proceeds in rounds of fixed time interval, however, we can improve this by considering the number of rounds.

More precisely, we compute the (minimum and maximum) expected number of complete gossiping rounds required before the combined views of the nodes generate a connected network. This is a desirable configuration for the network to reach since, when the local views do not form a connected topology, the nodes have insufficient information to ensure that a message gets propagated to all other nodes in the network. The PRISM properties are:

$$\mathbf{R}_{\{\text{“}num\_rounds\text{”}\}\min=?}[\ \mathbf{F}\ \text{“}connected\text{”}\ ]$$
$$\mathbf{R}_{\{\text{“}num\_rounds\text{”}\}\max=?}[\ \mathbf{F}\ \text{“}connected\text{”}\ ]$$

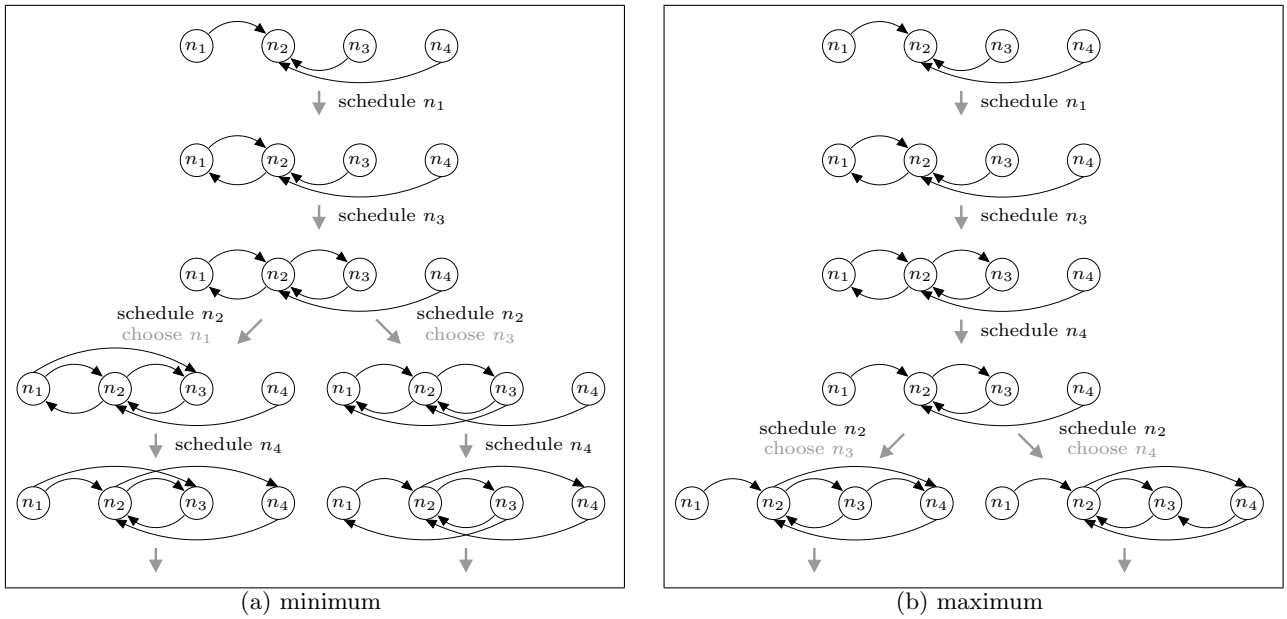
where "*num_rounds*" is a reward structure that assigns a reward of 1 to transitions marking the end of a round and "*connected*" labels states in which a path exists between any pair of nodes in the network.

As well as computing these measures, we use PRISM to generate actual adversaries that result in the minimum and maximum values.[2] Since the only nondeterminism in the model is due to scheduling of the nodes (i.e. the order in which they forward their views to their neighbours), we can extract from an adversary the corresponding scheduling.

We consider first the network of three nodes. The minimum and maximum number of complete rounds required for the local views to generate a connected network is 0 and 1 respectively. For comparison, in the DTMC model, where nondeterminism has been replaced by uniform random choice, the expected number of rounds is 0.667.

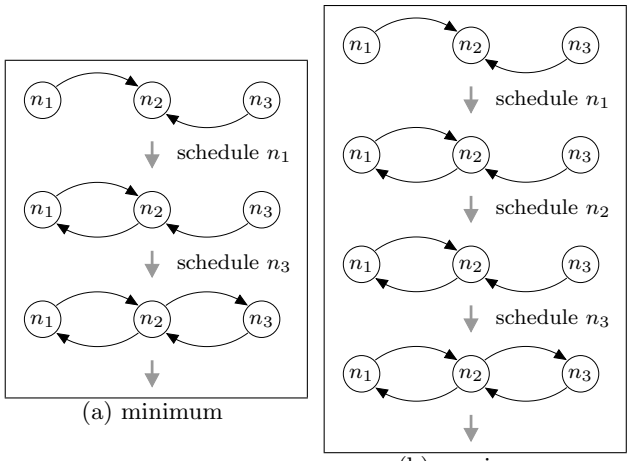
Figure 3 illustrates the sequences of node schedulings that result in these minimum and maximum values. Initially, all nodes can see node n_2, but no others. The best-case behaviour (minimum expected number of complete rounds) can be obtained by first scheduling node n_1 and then node n_3, after which n_2 has added both n_1 and n_3 to its view and the network is connected. Since n_2 has yet to be scheduled, the minimum expected number of complete rounds is 0. For worst-case behaviour (maximum value), we can schedule n_2

[2]We used a prototype extension of the tool that includes this functionality.



(a) minimum (b) maximum

Figure 4: Scheduling for the 4 node network (first gossiping round).



(a) minimum (b) maximum

Figure 3: Scheduling for the 3 node network.

before either n_1 or n_3 is scheduled. This adds no new information to the local views and means that a complete gossiping round is required before the network becomes connected.

For the network consisting of four nodes, we find that the minimum and maximum expected number of complete rounds before connectivity are 1.5 and 4.5 respectively. For comparison, the expected number of rounds for the DTMC model is 2.788 which, unlike in the case of the three node network, is closer to the minimum than the maximum.

This case is more complex than the three node network. Since more than one round may be required before the network becomes connected and the number of possible neighbours exceeds the size of the view, descriptors can be dropped from the views as more recent information becomes available. Furthermore, we must consider a node's choice of who to send data to. Figure 4 shows part of the scheduling (the first gossiping round) that can result in the minimum and maximum expected number of complete rounds. In both

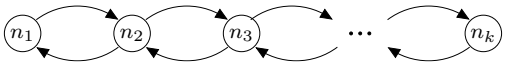


Figure 5: Chain of k nodes.

cases notice that, when node n_4 is scheduled the view of n_2 is updated, causing the removal of n_1 (the oldest descriptor). Not also that, when node n_2 is scheduled it makes a (random) selection between communicating with n_1 or n_3 since both are in its view at the time.

For the minimum case (Figure 4(a)) we see that, if n_2 chooses to gossip with n_3 (i.e. the right-hand branch), then the network is complete by the end of the first round. If it chooses n_1 (i.e. the left branch) this is not the case (there are no paths to n_1) and further rounds of the protocol are required. This leads to a (minimum) expected number of rounds of 1.5. For the maximum case (Figure 4(b)), the network is not connected under either choice and several further rounds are required.

The properties analysed in both this and the previous section demonstrate a considerable discrepancy between the minimum and maximum values. To give a simple intuitive explanation for this, consider a chain of nodes of length k (illustrated in Figure 5) in which n_1 is trying to pass a message to node n_k. Suppose that all nodes are scheduled in each round and that nodes send only messages to their right-hand neighbour (n_i only sends messages to n_{i+1}). Then, the scheduling n_1, n_2, ..., n_k would propagate the message to node n_k in a single round but the scheduling n_k, n_{k-1}, ..., n_1 would require $k-1$ rounds to achieve this.

As the number of nodes in the network increases, so does the amount of nondeterminism present in the model. Hence, the potential influence of the scheduling (the difference between the minimum and maximum values) is also likely to increase. As the comparison of the cases of three and four nodes suggests, however, it may also be the case that, for larger numbers of nodes, the average behaviour is closer to the best-case behaviour than the worst-case.

4.3 Other properties

The PRISM model we have constructed could also be used to analyse a variety of other properties. For example:

- the maximum probability that a connected network eventually becomes disconnected;
- the minimum probability node n_i can communicate with node n_j after k gossiping rounds or t time steps;
- the probability that node n_i can communicate with node n_j before it can communicate with node n_k;
- the maximum expected number of updates to the partial views before the network is connected.

Furthermore, the model could easily be adapted to study the effect on performance of a variety of other factors such as failures of network links and the dynamic addition/removal of additional nodes. It could also be modified to study possible ways of preventing the potential inefficiencies highlighted by our analysis. We could, for example, investigate the performance of a modified version of the gossip protocol in which the delays between each node's execution of the protocol is also randomised.

5. CONCLUSIONS

In this paper, we have shown that probabilistic model checking can be used to find interesting properties of gossiping protocols that would be difficult to discover using alternative analysis techniques such as simulation. Although using probabilistic model checking limits the size of the networks that can be analysed, these 'small' networks can still highlight interesting behaviour that may also occur in more realistic network configurations. The results we have obtained demonstrate that modelling unknown choices (the scheduling of the nodes in each gossiping round) with randomness causes a loss of information: the average case can be very different from the extreme (best/worst) cases.

Possible future work includes extending the approach to include timing characteristics by using probabilistic timed automata [13, 15]. These can be considered as an extension of MDPs that allows the modelling of real-time characteristics, in addition to probabilistic and nondeterministic behaviour. Another direction would be to employ techniques such as abstraction [11, 18] and symmetry reduction [12] to enable the analysis of larger network configurations.

Acknowledgments

The authors are supported in part by the EPSRC grants EP/D07956X and EP/D076625. This work was initiated at the "Two Decades of Probabilistic Verification - Reflections and Perspectives" meeting at the Lorentz Center, Leiden, organised by the NWO/DFG-funded VOSS II project.

6. REFERENCES

[1] C. Baier, B. Haverkort, H. Hermanns, and J.-P. Katoen. Model-checking algorithms for continuous-time Markov chains. *IEEE Trans. Software Engineering*, 29(6):524–541, 2003.

[2] C. Baier and M. Kwiatkowska. Model checking for a probabilistic branching time logic with fairness. *Distributed Computing*, 11(3):125–155, 1998.

[3] R. Bakhshi, F. Bonnet, W. Fokkink, and B. Haverkort. Formal analysis techniques for gossiping protocols. *ACM SIGOPS Operating Systems Review*, 41(5):28–36, 2007.

[4] A. Bianco and L. de Alfaro. Model checking of probabilistic and nondeterministic systems. In *Proc. FST&TCS'95*, volume 1026 of *LNCS*, pages 499–513. Springer, 1995.

[5] M. Duflot, M. Kwiatkowska, G. Norman, and D. Parker. A formal analysis of Bluetooth device discovery. *Int. Journal on Software Tools for Technology Transfer*, 8(6):621–632, 2006.

[6] A. Fehnker and P. Gao. Formal verification and simulation for performance analysis for probabilistic broadcast protocols. In *Proc. ADHOC-NOW'06*, volume 4104 of *LNCS*, pages 128–141. Springer, 2006.

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

[8] A. Hinton, M. Kwiatkowska, G. Norman, and D. Parker. PRISM: A tool for automatic verification of probabilistic systems. In *Proc. TACAS'06*, volume 3920 of *LNCS*, pages 441–444. Springer, 2006.

[9] M. Jelasity, S. Voulgaris, R. Guerraoui, A. Kermarrec, and M. van Steen. Gossip-based peer sampling. *ACM Trans. Computer Systems*, 25(3), 2007.

[10] J. Kemeny, J. Snell, and A. Knapp. *Denumerable Markov Chains*. Springer-Verlag, 2nd edition, 1976.

[11] M. Kwiatkowska, G. Norman, and D. Parker. Game-based abstraction for Markov decision processes. In *Proc. QEST'06*, pages 157–166. IEEE Press, 2006.

[12] M. Kwiatkowska, G. Norman, and D. Parker. Symmetry reduction for probabilistic model checking. In *Proc. CAV'06*, volume 4114 of *LNCS*, pages 234–248. Springer, 2006.

[13] M. Kwiatkowska, G. Norman, D. Parker, and J. Sproston. Performance analysis of probabilistic timed automata using digital clocks. *Formal Methods in System Design*, 29:33–78, 2006.

[14] M. Kwiatkowska, G. Norman, and J. Sproston. Probabilistic model checking of the IEEE 802.11 wireless local area net work protocol. In *Proc. PAPM/PROBMIV'02*, volume 2399 of *LNCS*, pages 169–187. Springer, 2002.

[15] M. Kwiatkowska, G. Norman, J. Sproston, and F. Wang. Symbolic model checking for probabilistic timed automata. *Information and Computation*, 205(7):1027–1077, 2007.

[16] PRISM web site. `www.prismmodelchecker.org`.

[17] J. Rutten, M. Kwiatkowska, G. Norman, and D. Parker. *Mathematical Techniques for Analyzing Concurrent and Probabilistic Systems, P. Panangaden and F. van Breugel (eds.)*, volume 23 of *CRM Monograph Series*. AMS, 2004.

[18] B. Wachter, L. Zhang, and H. Hermanns. Probabilistic model checking modulo theories. In *Proc. QEST'07*, pages 129–140. IEEE Press, 2007.

[19] F. Werner and P. Schmitt. Analysis of the authenticated query flooding protocol by probabilistic means. In *Proc. WONS'08*, pages 101–104, 2008.