A view-probability-matrix approach to the modelling of gossiping protocols

Thomas Krieger, Martin Riedl, Johann Schuster, Markus Siegle
University of the German Federal Armed Forces Munich
Department of Computer Science
{thomas.krieger, martin.riedl, johann.schuster, markus.siegle}@unibw.de

ABSTRACT

This paper addresses the quantitative analysis of gossiping protocols. In contrast to existing approaches which are entirely based on the simulation of the individual nodes' behaviours, we present a new approach based on summary stochastic models for the peer sampling service. Instead of an ordinary state- and transition-based model, a matrix-based approach is presented. Starting from a basic model with static node population and without ageing of neighbourhood information, refinements of the model are presented which enable the modelling of ageing and dynamic population. The paper also contains some experimental results for the different models introduced in the paper.

1. INTRODUCTION

The growing demand for scalable distributed systems has caused considerable research activities towards modelling, simulation and implementation of self-organising networks [2], in particular gossiping networks. The nodes in a gossiping network act autonomously and carry some local information, i.e., each node has its own so-called (partial) *view* of the network. The peer sampling service [3] is responsible for distributing and updating the views among the nodes. The nodes in the view of a certain node are called its *neighbours*. There are many different possibilities of how to organise the peer sampling mechanism: Using a pure push-mechanism, communication is initiated by the sending node, i.e., a sending node pushes its view to some of its neighbours. If a pure pull-mechanism is chosen, communication is initiated by the receiving node, i.e., a view is demanded from one of its neighbouring node. Combinations of pushing and pulling can also be realised. Furthermore, there are many different possibilities of how to merge two views in order to obtain a resulting new view.

We are interested not only in the *qualitative* properties of a gossiping network, but also in its *quantitative* properties, such as, for example, the probability that it becomes disconnected or the mean length of the shortest path between two of its nodes. Given the inherent randomness of the peer sampling service, e.g., in the choice of a communication partner, we do not develop a *deterministic* model, but a *stochastic* model of a gossiping network, in which, for example, the pushing of a view to a neighbouring node occurs with a certain probability.

The interaction of the nodes exchanging and merging their views is a challenge for modellers. Basically, gossiping models can be built with standard discrete-state tools such as PRISM [5] or CASPA [4, 1]. Following a straight-forward approach, every node could be modelled as a separate component having as many states as there are possible views for a node, and the sending and receiving of views could be realised by pairwise synchronisation of the nodes. This would result in a stochastic (Markovian) model with finite state space, but in practice the notorious state space explosion problem would limit the models to a few nodes and very small views. Using this approach with N nodes and view size C, each node has $N-1$ potential neighbours as, by definition, the node itself will never appear in its own view. As the view size is fixed to C and we assume no doubled entries, this yields $\binom{N-1}{C}$ combinations, i.e., valid views, resulting in a total number of $\binom{N-1}{C}^N$ states for the entire model.

This may be the reason why the behaviour of large scale gossiping networks has so far either been explored by real world measurements or by means of explicitly simulating the nodes' behaviour. Not much work on rigorous quantitative *analysis* has been published yet, that would help to investigate and understand the cause of certain behaviour of gossiping networks. Therefore, it is an important goal to develop gossiping models which carry enough information for thorough mathematical analysis, but on the other hand do not suffer too much from the state space explosion problem. The current paper presents such an approach, i.e., modelling gossiping networks by means of a matrix, representing probability distributions over the views of all nodes at a particular time instant. We propose an iterative calculation that mimics the gossiping policy which determines how the nodes behave and therefore how the network evolves.

This paper is organised as follows: Sec. 2 proposes a compact matrix representation for models with fixed node population and without ageing of view information, and introduces a round-based mechanism for the evolution of the state of this model. Sec. 2 also briefly discusses convergence properties of the basic model. An extension of the basic model, by adding age information to view entries, is presented in Sec. 3, where the update mechanism is modified accordingly. Sec. 4 presents a further extension to the model that allows for dynamic node population, a phenomenon that is very important in practice, since nodes may decide to join and leave the gossiping network over the course of time. Note, however, that those dynamic changes of the population are dealt with in a simulative way in the experiments. Experimental results for the different types of models, obtained with the help of a prototypical software tool, are presented in Sec. 5, and Sec. 6 concludes the paper.

2. MODEL WITHOUT AGE

2.1 Description of the model

This paper considers a gossiping network with overall N nodes. For the moment, it is assumed that N is fixed and that all nodes always participate in the gossiping protocol (for dynamic node population see Sec. 4).

A view V is a set of fixed size C, whose elements are (pairwise different) node identifiers n_i:

$$V = \{n_1, n_2, \ldots, n_C\} \quad \text{where} \quad i \neq j \rightarrow n_i \neq n_j\,.$$

We propose a model in which we store for each node the probability of having a certain view. These probabilities are stored in a matrix of size $N \times \binom{N}{C}$ (one row for each node). Each row of the matrix has at most $\binom{N-1}{C}$ non-zero entries, since the view of any node n will never contain n itself. All row sums of the matrix are equal to one.

Example:
We consider a fixed population of $N = 4$ nodes and view size $C = 2$. This results in the following matrix, where $p^{(i)}_{nm}$ is used as a shorthand notation for $p^{(i)}_{\{n,m\}}$, which is the probability that node i has nodes m and n in its view. The entries marked by "–" are invalid:

	(1,2)	(1,3)	(1,4)	(2,3)	(2,4)	(3,4)
1	–	–	–	$p^{(1)}_{23}$	$p^{(1)}_{24}$	$p^{(1)}_{34}$
2	–	$p^{(2)}_{13}$	$p^{(2)}_{14}$	–	–	$p^{(2)}_{34}$
3	$p^{(3)}_{12}$	–	$p^{(3)}_{14}$	–	$p^{(3)}_{24}$	–
4	$p^{(4)}_{12}$	$p^{(4)}_{13}$	–	$p^{(4)}_{23}$	–	–

End of Example.

The memory needed to store this matrix is in the order of $N\binom{N-1}{C}$ (only the non-zero entries need to be stored), which is of course much better than the straightforward approach mentioned in the introduction which needs to store $\binom{N-1}{C}^N$ states (plus the transitions between states).

From this matrix, most of the important metrics of the gossiping network, such as the probability of being connected, the mean length of the shortest path (measured in hop counts) between any two nodes, the average clustering coefficient, etc., can be computed (at least in principle). This is done by selecting one entry from each row and constructing a directed graph with N nodes and the arcs according to the selected views. The metrics of each such graph can be determined by standard algorithms, and since each graph is associated with a probability (computed as the product of the selected matrix entries), the overall averages can be computed. Note that this assumes that events associated with entries in different rows of the matrix are independent. (This is not really the case, but apart from trivial cases the dependencies are difficult to quantify, and for the limiting case, i.e., after many rounds, the independence assumption is justified.) This procedure, however, is practicable only for very small values of N, since it needs to consider all possible graph configurations.

We now describe how the dynamic evolution of the network is modelled by the matrix: First, a valid initial configuration is chosen, either at random (with the help of a pseudo-random-number generator) or deterministically. Then, the matrix is updated repeatedly as specified by the following update policy:

- A round-based scheme is considered, where the views of all nodes are updated simultaneously, i.e., the matrix update is in Jacobi- rather than Gauss-Seidel style.
- The scheme is based on the "push"-principle (as opposed to the "pull" or "push-pull" principles), which means that the sending of a view is initiated by the sender.
- A node s (sender) can send its view to another node r (receiver), provided that r is currently in the view of s.
- When a view is sent, in the sent view the receiver's id is replaced by the sender's id (since the receiver does not need to receive information about itself, but it wants to know the sender's id).
- During the "merge" of two views V_s (sender's view) and V_r (receiver's view), the new view of the receiver is determined by selecting a set of C nodes from $V_s \cup V_r$. Thereby the following rules are observed:
  - The merged view will always contain the entry s (sender's id), because this is considered "fresh" information (even though we do not consider age information at this stage).
  - The other $C - 1$ elements of the resulting view are determined by a random selection from $(V_s \cup V_r) \setminus \{s\}$.

Formally, this is expressed by an update equation; the new probability for the receiver to have view V is given by

$$p^{(r,new)}_V := c_r \cdot \sum_{s \neq r} \sum_{V_r} \sum_{V_s} f(r, V, V_s, V_r), \tag{1}$$

where c_r is a normalisation constant and the function f is defined as

$$f(r, V, V_s, V_r) = \begin{cases} p^{(s)}_{V_s} \cdot p^{(r)}_{V_r} \cdot \frac{1}{|merge(V_s, V_r)|} & \text{if} \quad \begin{array}{c}(r \in V_s \\ \wedge \\ V \in merge(V_s, V_r))\end{array} \\ 0 & \text{else.} \end{cases}$$

It remains to define the set $merge(V_s, V_r)$, which is the set of all views which may result (with equal probability) from the merging of V_s and V_r.

$$\begin{aligned} merge(V_s, V_r) \quad = \quad & \{V \mid \\ & s \in V \\ & \text{(sender is always in merged view)} \\ & \wedge |V| = C \\ & \text{(view size is fixed)} \\ & \wedge \forall k : ((k \in V \wedge k \neq s) \rightarrow k \in V_s \cup V_r) \\ & \text{(elements (except } s\text{) are all from operand views)} \\ \} & \end{aligned}$$

The normalisation factor is given by

$$c_r = \frac{1}{\sum_{s \neq r} \sum_{V_s \ni r} p^{(s)}_{V_s}}$$

i.e., the normalisation factor c_r when updating row r of the matrix is the sum of certain view probabilities of all possible senders s, where exactly those views of the senders are summed which contain the receiver id (because if a sender does not know the receiver he cannot push his view on him).

Example:
Returning to the previous example, the update equation for matrix entry $p_{23}^{(1,new)}$ is as follows:

$$\begin{aligned} p_{23}^{(1,new)} &:= c_1 \cdot \left(p_{23}^{(1)} \cdot \left(p_{13}^{(2)} + \frac{1}{2} \cdot p_{14}^{(2)} + p_{12}^{(3)} + \frac{1}{2} \cdot p_{14}^{(3)} \right) \right. \\ &+ p_{24}^{(1)} \cdot \left(\frac{1}{2} \cdot p_{13}^{(2)} + \frac{1}{2} \cdot p_{12}^{(3)} + \frac{1}{2} \cdot p_{14}^{(3)} \right) \\ &+ \left. p_{34}^{(1)} \cdot \left(\frac{1}{2} \cdot p_{13}^{(2)} + \frac{1}{2} \cdot p_{14}^{(2)} + \frac{1}{2} \cdot p_{12}^{(3)} \right) \right) , \end{aligned}$$

where c_1 is given by

$$c_1 = \left(p_{13}^{(2)} + p_{14}^{(2)} + p_{12}^{(3)} + p_{14}^{(3)} + p_{12}^{(4)} + p_{13}^{(4)} \right)^{-1} ,$$

provided that the fraction is well-defined. We explain some terms of this equation: If node 1 has old view $(2,3)$, and node 2 sends view $(1,3)$, which is changed to $(2,3)$, then the only choice for the resulting view is $(2,3)$. On the other hand, if node 1 has old view $(2,3)$, and node 2 sends view $(1,4)$, which is changed to $(2,4)$, then the union is $(2,3,4)$, from which views $(2,3)$ is selected with probability $\frac{1}{2}$ (the other choice would be a resulting view $(2,4)$, so the same term appears in the equation for $p_{24}^{(1,new)}$, also with weight factor $\frac{1}{2}$). The probabilities $p_{xy}^{(4)}$ do not appear in the equation for $p_{23}^{(1,new)}$, since if a view were sent from node 4 to node 1, the sender id (i.e., 4) would have to be in the new view.
End of Example.

2.2 Convergence properties

In this section, several effects of the model behaviour are described. Let $\mathbf{p}^r$ $(r = 1, \ldots, N)$ be the vector of all valid matrix entries of node r and let $\mathbf{p} := (\mathbf{p}^1, \ldots, \mathbf{p}^N)$. The case $N = 4$ and $C = 2$ leads for instance to

$$\mathbf{p}=\left(p_{23}^{(1)},p_{24}^{(1)},p_{34}^{(1)},p_{13}^{(2)},p_{14}^{(2)},p_{34}^{(2)},p_{12}^{(3)},p_{14}^{(3)},p_{24}^{(3)},p_{12}^{(4)},p_{13}^{(4)},p_{23}^{(4)}\right)^T .$$

Since every node has $\binom{N-1}{C}$ valid views, $\mathbf{p}$ is a vector of view probabilities if and only if

$$\mathbf{p} \in \left(\triangle_{\binom{N-1}{C}} \right)^N ,$$

where $\triangle_k$ ($k \in \mathbb{N}$) is the standard simplex in $\mathbb{R}^k$ (i.e., $\triangle_k := \{\mathbf{v} \in \mathbb{R}^k \mid \sum_{i=1}^k v_i = 1 \wedge v_1 \geq 0 \wedge \ldots \wedge v_k \geq 0\}$ with v_i the i-th component of $\mathbf{v}$). Let

$$T : \left(\triangle_{\binom{N-1}{C}} \right)^N \to \left(\triangle_{\binom{N-1}{C}} \right)^N$$

be the operator which transforms the vector $\mathbf{p}$ into the vector $\mathbf{p}^{(new)}$ via formula (1), i.e., $\mathbf{p}^{(new)} := T(\mathbf{p})$. The asymptotic behaviour of the system can now be investigated by evaluating $\lim_{n\to\infty} T^n(\mathbf{p}_0)$ (if it exists), where $\mathbf{p}_0$ is the initial configuration of the matrix. Experiments suggest the following conjecture: If $\mathbf{p}_0 > 0$, i.e., every component of $\mathbf{p}_0$ is greater than zero, then $\lim_{n\to\infty} T^n(\mathbf{p}_0)$ exists and is independent of $\mathbf{p}_0$. Furthermore, it is conjectured that for all $i = 1, \ldots, N\binom{N-1}{C}$

$$e_i^T \lim_{n\to\infty} T^n(\mathbf{p}_0) = \binom{N-1}{C}^{-1} , \qquad (2)$$

where e_i is the i-th unit vector in $\mathbb{R}^{N\binom{N-1}{C}}$, i.e., all valid views of a node are uniformly distributed in the limiting case. This result seems plausible: Since $\mathbf{p}_0 > 0$ the graph belonging to the initial matrix is connected in the sense that for every pair of nodes (r_1, r_2) there is a directed path from r_1 to r_2, and therefore no node is favoured over any other node on the long run.

An attempt to prove conjecture (2) is as follows. Let $\mathbf{p_{eq}^i} \in \triangle_{\binom{N-1}{C}}$ be the vector of uniformly distributed view probabilities for node i and $\mathbf{p_{eq}} := (\mathbf{p_{eq}^1}, \ldots, \mathbf{p_{eq}^N}) \in \left(\triangle_{\binom{N-1}{C}} \right)^N$ the corresponding vector of view probabilities. With the definitions

$$\begin{aligned} \tilde{\mathbf{p}} &:= \mathbf{p} - \mathbf{p_{eq}} \\ \tilde{T}(\tilde{\mathbf{p}}) &:= T(\tilde{\mathbf{p}} + \mathbf{p_{eq}}) - \mathbf{p_{eq}} \\ &= T(\mathbf{p}) - \mathbf{p_{eq}} \end{aligned}$$

equation (2) is transformed to

$$\lim_{n\to\infty} \tilde{T}^n(\tilde{\mathbf{p}}_\mathbf{0}) = \mathbf{0}, \qquad (3)$$

where $\mathbf{0}$ is the zero vector. The following argumentation is similar to the proof of the Banach fixed point theorem [6]: If there exists a $k \in (0,1)$ such that

$$\|\tilde{T}(\tilde{\mathbf{p}})\| \leq k \cdot \|\tilde{\mathbf{p}}\| , \qquad \forall\, \tilde{\mathbf{p}} \in \left(\triangle_{\binom{N-1}{C}} \right)^N - \mathbf{p_{eq}} ,$$

it follows that $\|\tilde{T}^n(\tilde{\mathbf{p}}_\mathbf{0})\| \leq k^n \cdot \|\tilde{\mathbf{p}}_\mathbf{0}\|$, where $\left(\triangle_{\binom{N-1}{C}} \right)^N - \mathbf{p_{eq}}$ is the translation of $\left(\triangle_{\binom{N-1}{C}} \right)^N$ by $-\mathbf{p_{eq}}$. As $k < 1$ and $\|\tilde{\mathbf{p}}_\mathbf{0}\|$ is bounded, $\lim_{n\to\infty} \|\tilde{T}^n(\tilde{\mathbf{p}}_\mathbf{0})\| = 0$ and therefore Equation (3) is proven. Using the example in subsection 2.1 and the euclidian norm, numerical optimization with Mathematica indicated that $k = 0.85$, so the conjecture is considered to be proven in this case. Unfortunately no general proof exists so far.

Conjecture (2) might also be true if the requirement $\mathbf{p}_0 > 0$ is replaced by the postulation that the graph belonging to the initial matrix is not disconnected.

The components of $T(\mathbf{p})$ may behave quite arbitrarily: It may well be the case that a component of $\mathbf{p}$ is close to 1 and that the same component of $T(\mathbf{p})$ is close to zero. The convergence rate depends strongly on the values of the initial matrix.

3. MODEL WITH AGE

In this section we extend the model of Sec. 2 by adding age information to view entries. This stems from the fact that real-life gossiping protocols use age information in their peer selection and view merging mechanisms. The age information, sometimes also referred to as "hop count", denotes the number of times a certain view entry has been communicated from one node to another. In this extended model, a view V is a set of fixed size C, whose elements are pairs (n_i, a_i) of (pairwise different) node identifiers and associated age information.

$$V = \{(n_1, a_1), (n_2, a_2), \ldots, (n_C, a_C)\}, \quad i \neq j \to n_i \neq n_j.$$

Again, we employ a model in form of a matrix, but it will have more columns than before, since the age information is included in the view. The size of the matrix is therefore

$$N \times \left(\binom{N}{C} \cdot (A^C - (A-1)^C) \right) ,$$

where $A = maxage$ is the maximal permitted age value: the formula reflects the fact, that we require that at any time at least one entry carries age information 1. Note that again, only the non-zero entries of the matrix need to be stored, which results in a required memory of $N \cdot \binom{N-1}{C} \cdot (A^C - (A-1)^C)$ entries.

The update policy for this model is based on the one described in Sec. 2, i.e., again a round-based scheme is considered, based on the "push"-policy, but several new rules need to be established in order to deal with the age information properly:

- During the "merge" of two views V_s (sender's view) and V_r (receiver's view), in case of "duplicates" the entry with the larger age is discarded (i.e., fresh information is preferred).
- The merged view will always contain the entry $(s, 1)$ (sender's id with age 1).
- Before merging takes place, the age of all entries of the sender's view is incremented by one. This is the only situation in which the age is increased in this model, but in Sec. 4 we will briefly comment on possible "local ageing".
- The merge procedure is probabilistic, where candidate views with small age sum are preferred.

Update equation:
The new probability for the receiver to have view V is given as before by

$$p_V^{(r,new)} = c'_r \cdot \sum_{s \neq r} \sum_{V_r} \sum_{V_s} f(r, V, V_s, V_r)$$

where c'_r is a normalisation constant (*not* equal to the constant in Sec. 2), but now the function f is defined differently as

$$f(r, V, V_s, V_r) = \begin{cases} p_{V_s}^{(s)} \cdot p_{V_r}^{(r)} \cdot mprob(V, V_s, V_r) & \text{if } \begin{array}{c}(r \in V_s \\ \wedge \\ V \in merge(V_s, V_r))\end{array} \\ 0 & \text{else.} \end{cases}$$

It remains to define the set $merge(V_s, V_r)$, which is the set of all views which may result (with non-zero probability) from the merging of V_s and V_r, and the probability $mprob(V, V_s, V_r)$, which is the probability that the merging of V_s and V_r results in the new view V. First we define the "age-incremented" view of the sender as

$$V_s^{incr} = \{(n_i, a_i) | (n_i, a_i - 1) \in V_s \wedge a_i \leq A\}.$$

Note that such an age-incremented view may be incomplete, i.e., contain less than C elements, since entries with maximum permissible age are dropped. Furthermore the age sum of a given view is defined as

$$agesum(V) = \sum_{(n_i, a_i) \in V} a_i.$$

This allows us to define the set of all possible merged views as

$$\begin{aligned} merge(V_s, V_r) \quad = \quad & \{V \mid \\ & (s, 1) \in V \\ & \text{(sender is always in merged view (with age 1))} \\ & \wedge |V| = C \\ & \text{(view size is fixed)} \\ & \wedge \forall k, i : ((k, i) \in V \wedge (k, i) \neq (s, 1)) \rightarrow (k, i) \in V_s^{incr} \cup V_r \\ & \text{(elements (except } (s, 1)) \text{ are all from operand views } V_s \text{ or } V_r) \\ & \wedge \forall i : (r, i) \notin V \\ & \text{(receiver itself is not in merged view)} \\ & \wedge \forall k, i : (k, i) \in V \rightarrow (\nexists j < i : (k, j) \in V_s^{incr} \cup V_r) \\ & \text{(elements with non-minimal age are excluded)} \\ \} & \end{aligned}$$

and the merge-probability as

$$mprob(V, V_s, V_r) = \frac{\frac{1}{agesum(V)}}{\sum_{W \in merge(V_s, V_r)} \frac{1}{agesum(W)}}$$

Example:
Consider again a fixed population of $N = 4$ nodes and a view size of $C = 2$. Furthermore, we assume a maximum age $A = 2$. This results in the matrix shown in Fig. 1, where the entries marked by "–" are invalid and $p_{n_a m_b}^{(i)}$ is used as a shorthand notation for $p_{\{(n,a),(m,b)\}}^{(i)}$. Note that in the matrix there are no columns for the age combination 22, since at least one element (the most recent sender) will always have age 1.
End of Example.

4. MODEL WITH DYNAMIC POPULATION

In real life, the set of nodes participating in a gossip-style communication is not fixed. New nodes may join at any time, and participating nodes may decide to leave. It is quite clear that many characteristics of a gossiping network's observed behaviour are due to the dynamic nature of the node population.

We decided to model the dynamics of a network as follows. There is an overall population of N nodes, but at a given time only a subset of these nodes may actively participate in the gossiping. Nodes which are *active (online)* may at any time decide to become *passive (inactive, offline)*, and passive nodes may in turn decide to become active again. The durations of nodes' active- and passive-times are modelled as random variables, and since our model follows a round-based scheme, where view information is updated in every round, it was decided to work with geometrically distributed random variables. This means that a node that is currently active will remain active in the next round with probability $1 - p$, and become passive with probability p. In turn, a node that is currently passive will remain passive in the next round with probability $1-q$, and become active with probability q. Note that under these assumptions, the mean number of active nodes (in equilibrium) is given by $N\frac{q}{p+q}$. It would not be difficult to model different discrete distributions for the active- and passive periods, but for the moment this is the only option considered.

Next, it is necessary to define an *activation policy* and a *deactivation policy*, which define what happens when a node becomes active or passive. We start with the latter, since it is the more straightforward of the two.

When a node becomes passive, it does no longer participate in the gossiping, so it does no longer push its view information on

nodes:	(1,2)			(1,3)			(1,4)			(2,3)			(2,4)			(3,4)		
age:	11	12	21	11	12	21	11	12	21	11	12	21	11	12	21	11	12	21
1	–	–	–	–	–	–	–	–	–	$p^{(1)}_{2_1 3_1}$	$p^{(1)}_{2_1 3_2}$	$p^{(1)}_{2_2 3_1}$	$p^{(1)}_{2_1 4_1}$	$p^{(1)}_{2_1 4_2}$	$p^{(1)}_{2_2 4_1}$	$p^{(1)}_{3_1 4_1}$	$p^{(1)}_{3_1 4_2}$	$p^{(1)}_{3_2 4_1}$
2	–	–	–	$p^{(2)}_{1_1 3_1}$	$p^{(2)}_{1_1 3_2}$	$p^{(2)}_{1_2 3_1}$	$p^{(2)}_{1_1 4_1}$	$p^{(2)}_{1_1 4_2}$	$p^{(2)}_{1_2 4_1}$	–	–	–	–	–	–	$p^{(2)}_{3_1 4_1}$	$p^{(2)}_{3_1 4_2}$	$p^{(2)}_{3_2 4_1}$
3	$p^{(3)}_{1_1 2_1}$	$p^{(3)}_{1_1 2_2}$	$p^{(3)}_{1_2 2_1}$	–	–	–	$p^{(3)}_{1_1 4_1}$	$p^{(3)}_{1_1 4_2}$	$p^{(3)}_{1_2 4_1}$	–	–	–	$p^{(3)}_{2_1 4_1}$	$p^{(3)}_{2_1 4_2}$	$p^{(3)}_{2_2 4_1}$	–	–	–
4	$p^{(4)}_{1_1 2_1}$	$p^{(4)}_{1_1 2_2}$	$p^{(4)}_{1_2 2_1}$	$p^{(4)}_{1_1 3_1}$	$p^{(4)}_{1_1 3_2}$	$p^{(4)}_{1_2 3_1}$	–	–	–	$p^{(4)}_{2_1 3_1}$	$p^{(4)}_{2_1 3_2}$	$p^{(4)}_{2_2 3_1}$	–	–	–	–	–	–

Figure 1: Probability matrix for the model with age

other nodes. In the model, each row of the matrix is augmented by a flag which indicates whether the corresponding node is active or passive. Only the view probabilities of active nodes are updated, no update is performed for passive nodes (from that point of view, the model with dynamic population is computationally cheaper than the model with fixed population, see Sec. 5). Upon deactivation of node n, not only is its flag set on passive, but also all values in row n are set to zero, which ensures that during the following update operations for an active node (say k) no view information is communicated from node n to node k.

Defining the activation policy is a bit more challenging, since a node that becomes active has to be assigned a valid initial view (or rather, valid view probabilities) through a bootstrapping process. There are many possibilities of how to realise this bootstrapping, so we had to make some design decisions here, which, however, could be easily modified for future studies. When node n becomes active, it chooses one of the currently active nodes (say k) at random as the source for its initial view information. However, it is not possible to simply copy line k of the matrix into line n, since some of the valid entries of line k are in positions which are invalid for node n (see Fig. 1). So we decided to copy only those entries of line k which are in valid column positions for node n, ignore the entries of line k which are in invalid column positions, and leave the remaining (valid or invalid) columns of row n at value zero. After the copy operation, the new values in row n are normalised to one, after which row n contains valid entries and node n may participate in the regular update process as of the next round.

The fact that a node has become passive is not known to the other nodes. On the contrary, passive nodes remain in the views of the active nodes (let us call them zombies), but this information dies out over time, since fresh information is favoured during the merging of views. Suppose that a zombie z is in the view of an active node n, and that its age is quite small, e.g. $a_z = 1$. When n pushes its view to another node, the age of the zombie z is incremented, and after several such steps the age will hit $A = maxage$ and the zombie will become forgotten. However, if the entry z survives the next merge, it will remain in the view of n with the same age, and this could happen repeatedly over several rounds. This could be the motivation for a more complicated ageing strategy, where the age of a view entry is not only incremented when the view is pushed to another node, but also when a view entry survives a merge locally. Such local ageing would of course have to happen at a slower rate than the regular ageing. Our current model, however, does not consider local ageing, but this seems to be all right, since we did not observe any tendency of zombies dominating the local views.

The model with dynamic population just described could be augmented in different directions. As an example, it would be easy to model an overall population which consists of different classes of nodes, some more active than others, by choosing the activation- and deactivation-parameters depending on a node's class.

5. EXPERIMENTAL RESULTS

For each considered model (without or with age information, the latter both with constant and dynamic population) we built a Python application and carried out experiments on different instantiations of the models.

5.1 Model Without Age

On the left-hand-side of Fig. 2 an instantiation of a model without age with $N = 6$ is shown. Every box represents one node, and within each box those views which have non-zero probability are depicted. The arc information, which is actually redundant, illustrates which nodes are known by a certain node. These probabilities are initial values which were set by hand. Within that model every node "knows" two nodes, meaning that it has a fixed view-size $C = 2$. Node 1 has the view $(2, 3)$ with probability 1.0, node 2 the view $(1, 3)$ with probability 1.0, node 3 the view $(1, 2)$ with probability 0.9 and view $(1, 4)$ with probability 0.1. Nodes 4,5 and 6 are only aware of each other. The network is nearly partitioned as there are two strongly related subnets (nodes $1, 2, 3$ and nodes $4, 5, 6$) in the network. The only interrelation exists at node 3 that knows 4 with probability 0.1. Using the program implementing the update algorithm provided in Sec. 2 for models without age, the probabilities converged to a uniform distribution with probability 0.1 as one can see on the right-hand-side of Fig. 2. From the figure we can also read that the probability of node 3 knowing node 4 is 0.4, which is sum of the probabilities of views of node 3 containing node 4.

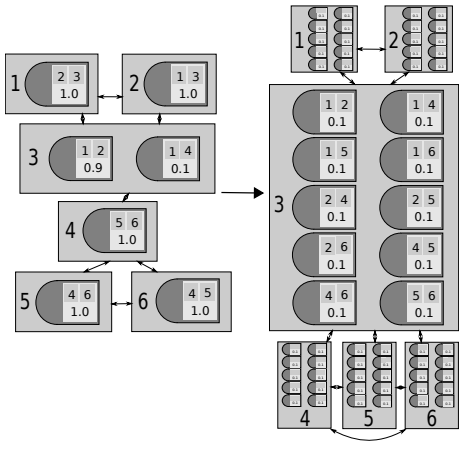

Figure 2: Model without age ($N = 6, C = 2$). Initial configuration (left), converged probabilities (right).

Fig. 3 shows the convergence behaviour for this model. In the figure, the view probabilities of all nodes are plotted over time,

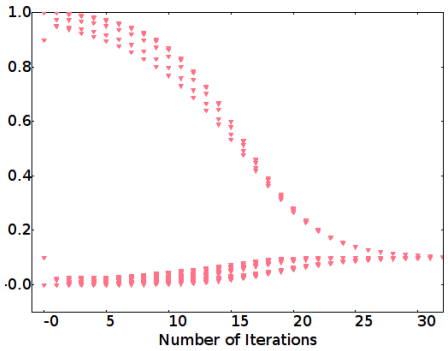

Figure 3: Convergence of view probabilities of all nodes for model from Fig. 2.

measured in rounds (synonymously called iterations). We see that the initial probability values are 0, 0.1, 0.9 and 1.0 and that after 30 iterations all probability values have converged to 0.1. All our other experiments with different gossiping network instantiations (number of nodes, view size, initial probability distribution) also showed that the view probabilities converge to a uniform distribution (unless the network is already disconnected at the beginning).

5.2 Model With Age

An instance of a model with age is described by a tuple $M_{age} = (N, C, A)$, where N is the fixed number of nodes, C is the fixed size of a view and A defines the maximum permitted age (starting at 1). We have implemented the model with age allowing us to process arbitrary network configurations specified by such tuple identifiers. Experiments have shown that the probabilities of the second model also converge to a limiting distribution. In Table 1 one can see an arbitrary initialized matrix of a (4,2,2) model as output by our program (our program prints the matrix in transposed form). After 30 update iterations the matrix is the one represented in Table 2. One can see the limit distribution for the age combinations of the view: for every view $\{(n_1, a_1), (n_2, a_2)\}$ with $a_1 = a_2 = 1$ the probability converged to 0.265986. For $a_1 = 1, a_2 = 2$ or $a_1 = 2, a_2 = 1$ the probability converged to 0.033674.

MATRIX				
View {(n_1,a_1),...,(n_i,a_i)}	Nodes: [1, 2, 3, 4]			
((1, 1), (2, 1))	--------	--------	0.023544	0.126685
((1, 1), (2, 2))	--------	--------	0.166562	0.024473
((1, 2), (2, 1))	--------	--------	0.171524	0.108916
((1, 1), (3, 1))	--------	0.091061	--------	0.135361
((1, 1), (3, 2))	--------	0.137136	--------	0.110231
((1, 2), (3, 1))	--------	0.158067	--------	0.159641
((1, 1), (4, 1))	--------	0.111335	0.113766	--------
((1, 1), (4, 2))	--------	0.064549	0.066746	--------
((1, 2), (4, 1))	--------	0.097139	0.100780	--------
((2, 1), (3, 1))	0.163572	--------	--------	0.125100
((2, 1), (3, 2))	0.092586	--------	--------	0.116538
((2, 2), (3, 1))	0.084521	--------	--------	0.093054
((2, 1), (4, 1))	0.010872	--------	0.140351	--------
((2, 1), (4, 2))	0.155088	--------	0.098462	--------
((2, 2), (4, 1))	0.137327	--------	0.118264	--------
((3, 1), (4, 1))	0.111677	0.122080	--------	--------
((3, 1), (4, 2))	0.154802	0.113918	--------	--------
((3, 2), (4, 1))	0.089556	0.104714	--------	--------

Table 1: (4,2,2) model with age: initialisation matrix.

In Fig. 4 each dot in the graphs represents an age-abstracted view probability. This means that for node k the probability to have view $\{n_1, \ldots, n_c\}$ is given by the sum

$$p^{(k)}_{\{n_1 \ldots n_c\}} = \sum_{a_1, a_2, \ldots, a_c} p^{(k)}_{\{(n_1,a_1),(n_2,a_2),\ldots(n_c,a_c)\}}$$

MATRIX				
View {(n_1,a_1),...,(n_i,a_i)}	Nodes: [1, 2, 3, 4]			
((1, 1), (2, 1))	--------	--------	0.265986	0.265986
((1, 1), (2, 2))	--------	--------	0.033674	0.033674
((1, 2), (2, 1))	--------	--------	0.033674	0.033674
((1, 1), (3, 1))	--------	0.265986	--------	0.265986
((1, 1), (3, 2))	--------	0.033674	--------	0.033674
((1, 2), (3, 1))	--------	0.033674	--------	0.033674
((1, 1), (4, 1))	--------	0.265986	0.265986	--------
((1, 1), (4, 2))	--------	0.033674	0.033674	--------
((1, 2), (4, 1))	--------	0.033674	0.033674	--------
((2, 1), (3, 1))	0.265986	--------	--------	0.265986
((2, 1), (3, 2))	0.033674	--------	--------	0.033674
((2, 2), (3, 1))	0.033674	--------	--------	0.033674
((2, 1), (4, 1))	0.265986	--------	0.265986	--------
((2, 1), (4, 2))	0.033674	--------	0.033674	--------
((2, 2), (4, 1))	0.033674	--------	0.033674	--------
((3, 1), (4, 1))	0.265986	0.265986	--------	--------
((3, 1), (4, 2))	0.033674	0.033674	--------	--------
((3, 2), (4, 1))	0.033674	0.033674	--------	--------

Table 2: (4,2,2) model with age: matrix after 30 iterations.

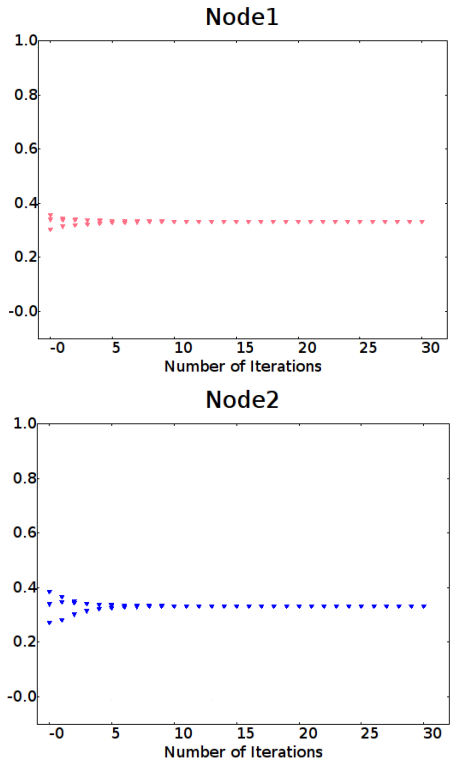

Figure 4: Experiment with a (4,2,2) network.

As an example, for a (4,2,2) model this means that the age-abstracted view consisting of nodes $n_1 = 1$ and $n_2 = 3$ is obtained by summing the detailed views $\{(1,1),(3,1)\}$, $\{(1,1),(3,2)\}$ and $\{(1,2),(3,1)\}$. If we look at Table 1 we can sum the values for these views, for example for node 4: We obtain

$$0.135361 + 0.110231 + 0.15964 = 0.405233$$

for the ordinate-value of a dot. If we take the corresponding values from Table 2 we obtain

$$0.265986 + 0.033674 + 0.033674 = 0.333334 \approx \frac{1}{3},$$

as expected, since each of the 3 possible age-abstracted views should be equally likely on the long run. We performed a number of experiments with different randomly generated initialisation matrices, but also with some matrices initialised by hand with "extreme"

values. All of them showed the same convergence behaviour, as expected, as there is no node that is in any way prioritized.

5.3 Model With Age and Dynamic Population

We also implemented the model with age and dynamic population, where nodes become active/inactive after a geometrically distributed number of rounds. The new view probabilities are calculated in a loop as depicted in Fig. 5. In each round and for each active node a pseudo-random-number generator produces a value uniformly distributed over the interval $[0, 1]$. If the value is between $[0, p]$ the node is made inactive. Likewise, for all inactive nodes the random generator is consulted, such that with probability q a node is made active. In the next step, for every node that becomes inactive the associated row in the matrix is set to zero and the node itself is marked inactive. For every node that becomes active a bootstrapping node is chosen (uniformly at random) from the nodes that remain active. The bootstrapping itself is done by copying all valid entries from the matrix row of the bootstrapping node to the row of the node to be activated and afterwards normalising that row. No update calculations need to be performed for inactive nodes. During the calculation of the new view probabilities of active nodes the contribution coming from inactive nodes is 0, as their row has been set to 0. As an example, consider an experiment with a $(5, 2, 2)$ network with dynamic population (see Fig. 6, where the probabilities for age-abstracted views are plotted for nodes 1 and 2). Initially, up to round 5, the probabilities converge to $\frac{1}{6}$, as expected for this model, since there are 6 possible view combinations for each node. In round 5 node 5 becomes inactive, leaving only 4 active nodes in the network. This means that the possible number of view combinations for each node is reduced to 3, resulting in a convergence towards the value $\frac{1}{3}$ for views not containing node 5, and to the value 0 for views containing node 5. A further disturbance of the view probabilities occurs in iterations 8 and 9, where node 3 becomes passive and after only a single iteration reenters the network. In round 9, node 3 bootstraps from node 2, and in the following rounds the probabilities converge again towards the value $\frac{1}{3}$. In round 16 node 2 becomes passive, which means that its view probabilities are all set to zero, as can be seen on the right-hand side of Fig. 6. As the network is now like a $(3, 2, 2)$ network, there remains only a single valid view for each node. Consequently, as one can see in the graph on the left of Fig. 6, the view probabilities converges to either 0 or 1.

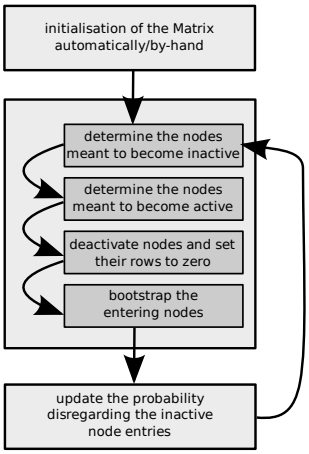

Figure 5: Update algorithm for the model with dynamic population.

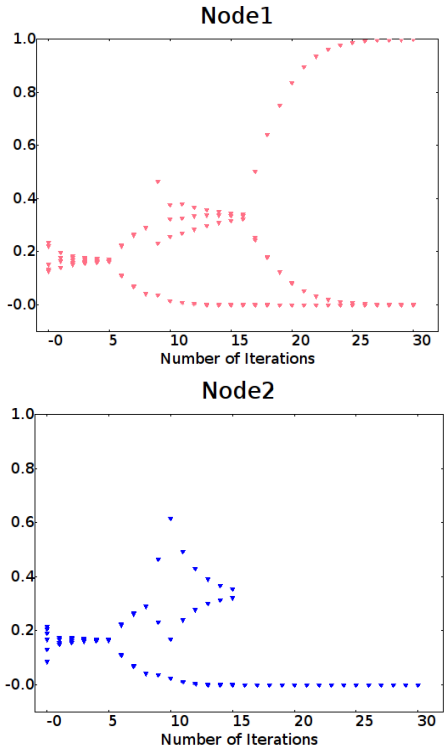

Figure 6: Experiment with a (5,2,2) network.

5.4 Performance Considerations

Timing measurements were taken for models with age (with both, fixed and dynamic population), varying the network parameter tuples (N, C, A) as shown in Table 3, where the entry n.e. stands for "not evaluated". As expected, the dynamic population model shows smaller user time consumption than the fixed population model, since no updates need to be performed for passive nodes.

model	population	
with age	fixed	dynamic
(4,2,1)	0.31s	n.e.
(4,2,2)	6.5s	n.e.
(4,2,3)	32.31s	21.23s
(5,2,1)	2.39s	n.e.
(5,2,2)	62.34s	n.e.
(5,2,3)	323.07s	228.3s
(5,3,3)	12522.24s	n.e.
(6,2,3)	1937.01s	1442.21s
(7,2,3)	8146.36s	n.e.
(8,2,3)	28287.33s	n.e.
(9,2,3)	80475,71s	n.e.

Table 3: Computation time (in seconds) for 30 iterations on a Pentium M (1700MHz / 1MB Cache / 1024MB).

The number of operations to be performed during each round of update calculations grows combinatorially with the parameters N, C and A. Therefore it is not practically feasible to perform calculations on large models. However, we measured the time which an update round takes and counted the number of operations (in

terms of multiplications of matrix entries) which are performed per round for a certain model (N, C, A). Table 4 shows the measured values as well as the time/operation ratios between a certain model and the next smaller one. Quite unintuitively, one can see that the ratios decrease for larger models. This may support the hope that the ratios converge to a certain limit, but for the moment this is still an open question. Note, that the runtime of such a Python imple-

Model (N, C, A)	user time UT	$\frac{UT_N}{UT_{N-1}}$	Ops_N	$\frac{Ops_N}{Ops_{N-1}}$
(4,2,3)	32.31s	-	2880	-
(5,2,3)	323.07s	10.094	17100	5.937
(6,2,3)	1937.01s	5.997	62400	3.649
(7,2,3)	8146.36s	4,205	173250	2.776

Table 4: User time, operations per iteration and ratios of consecutive steps (model with age).

mentation should not be compared to a highly optimized C or C++ program, which would be significantly faster. The message of our results is not in the absolute values but in the growth of those values for increasing model parameters.

6. CONCLUSION

This paper described the current progress of a new modelling approach for gossiping networks. Our first results rely only on numerical solutions of the update equations. Mathematical proofs of convergence or speed of convergence so far have not been found. It is still an open problem how the calculation of network properties can be done efficiently from the matrix representing the model. Checking all possible configurations of the network as indicated in Sec. 2 leads to the desired network properties but is prohibitively slow. As a future task we will concentrate on efficient computation methods to derive network properties of the given model.

Acknowledgement(s): The idea of modelling a gossiping network by a matrix containing the probabilities of certain nodes having a certain view was brought up by Frits Vaandrager in November 2007 during the Lorentz-Center workshop on "Two Decades of Probabilistic Verification – Reflections and Perspectives".

This work was supported in part by ITIS e.V. (Institut für Technik Intelligenter Systeme e.V. an der Universität der Bundeswehr München) and by Deutsche Forschungsgemeinschaft (DFG) under grants SI 710/2 (Validation of Stochastic Systems (VOSS)) and SI 710/3.

7. REFERENCES

[1] J. Bachmann, M. Riedl, J. Schuster, and M. Siegle:. An Efficient Symbolic Elimination Algorithm for the Stochastic Process Algebra tool CASPA. In *Proc. of 35th International Conference on Current Trends in Theory and Practice of Computer Science*, to appear in 2009.

[2] P. Th. Eugster, R. Guerraoui, A.-M. Kermarrec, and L. Massoulie. From Epidemics to Distributed Computing. *IEEE Computer*, 37(5):60–67, 2004.

[3] M. Jelasity, R. Guerraoui, AM. Kermarrec, and M. van Steen. The peer sampling service: experimental evaluation of unstructured gossip-based implementations. In *Middleware '04: Proceedings of the 5th ACM/IFIP/USENIX international conference on Middleware*, pages 79–98, New York, NY, USA, 2004. Springer-Verlag New York, Inc.

[4] M. Kuntz, M. Siegle, and E. Werner. Symbolic Performance and Dependability Evaluation with the Tool CASPA. In *Proc. of First European Performance Engineering Workshop (EPEW), FORTE'04 Workshop*, pages 293–307. Springer LNCS 3236, 2004.

[5] PRISM website. `http://www.prismmodelchecker.org/`, September 26th, 2008.

[6] K. Yosida. *Functional Analysis*. Springer, 1995.