Broadcast Abstraction in a Stochastic Calculus for Mobile Networks*

Lei Song and Jens Chr. Godskesen

IT University of Copenhagen, Denmark

Abstract. We introduce a continuous time stochastic broadcast calculus for mobile and wireless networks. The mobility between nodes in a network is modeled by a stochastic *mobility function* which allows to change part of a network topology depending on an exponentially distributed delay and a *network topology constraint*. We allow continuous time stochastic behavior of processes running at network nodes, e.g. in order to be able to model randomized protocols. The introduction of group broadcast and an operator to help avoid *flooding* allows us to define a novel notion of *broadcast abstraction*. Finally, we define a weak bisimulation congruence and apply our theory on a leader election protocol.

1 Introduction

Mobile and wireless networks have become an important part of our life, for instance they have been applied to areas like wireless local area networks, mobile ad-hoc networks, sensor networks, and cellular networks for mobile telephony. Broadcast calculi for this kind of networks have been studied considerably for the last five years, e.g. in [1–5]. A common characteristic for all those calculi is that they deal with mobility and connectivity between nodes abstractly, i.e. a node can move arbitrarily and cause arbitrary change of the network topology, and either a node is connected or disconnected to another node, so none of the calculi address the problem of unreliable links.

In a recent paper [6] we introduced the feature of letting a communication link between two nodes not just be in either 'connected' or 'disconnected' in that we allowed a decoration of connection links with a probability. The meaning being that messages broadcasted along a connection decorated with a probability ρ will be received by that probability. Intuitively this reflects that connection links in wireless networks may not always be reliable. We also enforced restricted mobility by means of a *probabilistic mobility function* saying that a given node with a certain probability may move and thereby change the probability of the connection to another node. The models we obtain are discrete and each network in our calculus in [6] gives rise to a probabilistic automata [7]. A major contribution of this paper is a generalization of the notion of a mobility function. In [6] a mobility function returns the change (the new probability) of just a single connection between two nodes, in this paper we let a mobility function be able to change a number of connections at the same time, i.e. we recognize that mobility of a single node may not just influence the connection to a single neighbor, instead a mobility step

* The research presented in this paper has been supported by MT-LAB, a VKR Center of Excellence for the Modeling of Information Technology.

may change a larger part of the network topology. Moreover, the new kind of mobility functions introduced in this paper makes use of *network topology constraints*. For instance we may specify that the probability for the node l being connected to m must be the same as k being connected to m, i.e. $\rho_{l \longmapsto m} = \rho_{k \longmapsto m}$. Intuitively this may represent that k and l are always within the same distance from m. Another example could be to require that the likelihood one of k and l receiving a broadcast message from m is sufficiently high, we may for instance specify $\rho_{l \longmapsto m} + \rho_{k \longmapsto m} \geq 0.9$, intuitively meaning that m is always sufficiently close to at least one of k and l. We demonstrate the usefulness of topology constraints in Section 5.

Another contribution of this paper is the introduction of stochastically timed behavior for models for mobile and wireless networks, our contribution follows the tradition of having rates for exponential probability distributions, known from say continuous time Markov processes, as part of our calculus. A major motivation for this contribution is that we would like to more realistically being able to model mobility of nodes as time dependent stochastic phenomenon, this is obtained by letting a stochastic mobility function return no longer a discrete probability as in [6] but a rate for an exponential probability distribution. Formally we will write $mf(C, C', \phi) = \lambda$ where C is the current (partial) network configuration, C' is the new configuration reached by a mobility step, ϕ is the network topology constraint the transition from C to C' depends on, and the transition occurs with a delay exponentially distributed by λ. Intuitively the rate signifies how fast the network topology will change, i.e. the higher rate the more likely it is that the topology will change fast. Another reason for introducing continuous time stochastic behavior is that many protocols for mobile and wireless networks make use of time dependent randomized back-off techniques. In order to be able to model such protocols we introduce, in the style of Interactive Markov Chains [8], a prefix construct λ for processes such that we may write e.g. $A = p + \lambda \cdot A$ meaning that A may behave as p or it may after some delay exponentially distributed by λ back off and iterate its behavior. This back-off style encoding is utilized in our model of a leader election protocol for mobile and wireless networks defined in Section 5. By the introduction of the continuous time stochastic behavior it turns out that the semantics of our calculus is a combination of discrete and continuous time probability, non-determinism, and concurrency and thus gives rise to a Markov Automaton (MA) [9]. In [10] a related stochastic restricted broadcast process theory is introduced to model and analyze mobile ad hoc network protocols. Their stochastic model is in PEPA style [11] where the duration of each action is exponentially distributed. After resolving non-determinism a continuous-time Markov chain is derived for each network. Differently our stochastic model is in Interactive Markov Chain [8] style where the rate is used to specify the delay rather than the duration of each action.

A third contribution is that we allow for two novel operators as part of our calculus. To the best of our knowledge these two operators have not before been considered in calculi for mobile and wireless systems. In many broadcast protocols it is quite common for a node to broadcast messages just to a limited number of nodes and hence not to all nodes in the network; to accommodate this feature we introduce a *group broadcast* prefix in our calculus denoted by $\langle x \triangleright L \rangle$ where x is the message to be broadcasted and L is the set of intended receivers of x. The other new operator is a kind of a low level

protocol that is often used in many wireless broadcast protocols, it is meant to deal with the problem of *flooding*. Flooding occurs when the same message is broadcasted over and over again in the execution of a protocol, but where it is sufficient to have received and dealt with the message just once. Flooding may e.g. occur in a protocol if a node is naively supposed to forward all requests for being part of a protocol, a node receiving similar requests for participating in the same execution of the protocol from multiple neighbors will then forward each of these requests to its neighbors although forwarding just one of these identical requests would ideally be sufficient. The operator is defined by introducing a memory M for each node, formally we write $\lfloor p \rfloor_l^M$ for a node with the processes p running a location l and with memory M. Intuitively the semantics is that whenever the node receives a broadcast message x it is first checked whether x belongs to M, if it does x is discarded and p will remain unchanged, otherwise x is added to M and p is updated accordingly. In this short version of our theory flooding avoidance input is the only broadcast input dealt with.

A fourth and major contribution is that we introduce a novel notion of *broadcast abstraction*. We abstract from the sender of a broadcast message since two broadcast messages should not be distinguished if they can deliver the same message to the same destinations with the same probability, despite that they may originate at different locations. Due to the introduction of group broadcast we can move even further such that one broadcast message can be simulated by several broadcast messages in a row. Intuitively, if a broadcast message α can deliver x to nodes at locations l and k with probability ρ_1 and ρ_2 respectively, and if we have two broadcast messages β_1 and β_2 such that β_1 can only deliver x to l with probability ρ_1 and β_2 can only deliver x to k with probability ρ_2, then β_1 and β_2 together can simulate α. In general we need to assume that the destinations of β_1 and β_2 are disjoint, since otherwise nodes at joint locations may receive x twice with positive probability which will never happen by performing α. The memory M plays a role here, if a node has received x, it will simply ignore it and stay unchanged whenever it receives x again, thus in this case the destinations of β_1 and β_2 may not necessarily be disjoint. For instance, if α can only deliver x to location l with probability ρ, and β_1 and β_2 can only deliver x to l with probability ρ_1 and ρ_2 respectively, then β_1 and β_2 in sequence can simulate α provided that $1 - (1 - \rho_1) \cdot (1 - \rho_2) = \rho$, where $(1 - \rho_1) \cdot (1 - \rho_2)$ equals the probability of x failing to reach l after both β_1 and β_2.

In summary, the main contribution of this paper is a continuous time stochastic broadcast calculus for wireless networks with a stochastic mobility function depending on topology constraints where group broadcast and flooding avoidance are integrated operators. As illustrated above the two operators facilitate abstraction of broadcast messages where several messages may be simulated by one. The paper is organized as follows: the syntax of the calculus is presented in the next section and in Section 3 we give a labeled transition system semantics of our calculus. In Section 4 a weak bisimulation is defined. We apply our calculus on a leader election protocol [12] in Section 5. Finally we end by a conclusion.

2 The Calculus

We presuppose a countable set $\mathscr{N}$ of names, ranged over by x, y, z and a countable set $\mathcal{L}$ of location names, ranged over by k, l, m, and n. Accordingly K, L, M, and N are used to range over finite subsets of $\mathcal{L}$. We also write l directly for a singleton set $\{l\}$. In addition, we also suppose a finite set of probabilities $\wp$ including 0 and 1 ranged over by $\rho, \rho', \rho_1 \ldots$. We define a *location connectivity set*, ranged over by $\mathbb{L}, \mathbb{K} \ldots$, as a finite set $\{(\rho, l) \mid l \in L, \rho \in \wp\}$. We use $l(\mathbb{L}) = \{l \mid (\rho, l) \in \mathbb{L}\}$ to denote all the locations in $\mathbb{L}$.

Let $\mathcal{P}$ denote the set of the processes which is ranged over by $p, q, r \ldots$, and defined by the following grammar:

$$p, q ::= 0 \mid Act \cdot p \mid p + q \mid [x = y]p, q \mid \nu x p \mid A \text{ where } Act ::= \lambda \mid \langle x \triangleright L^* \rangle \mid (x)$$

where 0 is the deadlock process. $Act \cdot p$ means that p is prefixed by Act and will behave as p after Act being performed. Specially, $\lambda \cdot p$ means that p is guarded by a delay which is exponentially distributed with rate $\lambda \in \mathbb{Q}^+$.[1] Let $\langle x \triangleright L^* \rangle$ and (x) denote (group) broadcast and reception respectively where L^* is either L or $\mathcal{L}$. We usually write $\langle x \triangleright \mathcal{L} \rangle$ as $\langle x \rangle$ for simplicity. If $L^* = L$, then $\langle x \triangleright L \rangle$ denotes a group broadcast which can deliver the message x only to nodes at locations in L. $p + q$ denotes nondeterministic choices between p and q. $[x = y]p, q$ is a conditional choice, it will evolve into p if $x = y$, otherwise it evolves into q. $\nu x p$ means that x is bounded in p. $A \in \mathcal{A}$ is a process constant where $\mathcal{A}$ is a set of process identifiers. By defining $A \stackrel{def}{=} p$, A will behave in the same way as p. The set of networks $\mathcal{N}$ is defined by:

$$E, F ::= 0 \mid \lfloor p \rfloor_l^M \mid \{\mathbb{L} \longmapsto l\} \mid \nu x E \mid E \parallel F$$

where node $\lfloor p \rfloor_l^M$ is a process p at location l with memory M which is used to keep track of all the messages having been received. The parameter M is often omitted if it is not important for the discussion. $\nu x E$ and $E \parallel F$ are restriction and parallel composition respectively which have the standard meaning; $\{\mathbb{L} \longmapsto l\}$ denotes connectivity information, i.e. if $(\rho, k) \in \mathbb{L}$, the node at location k is connected to l and can receive messages from l with probability ρ. Let $\mathcal{CN}$ be the set of *connectivity networks* which only contain connectivity information, that is, $C, C' ::= 0 \mid \{\mathbb{L} \longmapsto l\} \mid C \parallel C'$.

A network distribution is a function $\mathbb{E} : \mathcal{N} \to [0, 1]$ satisfying $|\mathbb{E}| = \sum_{E \in \mathcal{N}} \mathbb{E}(E) \leq 1$. Let $\mathcal{ND}$ denote the set of distributions over $\mathcal{N}$, ranged over by $\mathbb{E}, \mathbb{F}, \mathbb{G} \ldots$. The support of $\mathbb{E}$, $Supp(\mathbb{E}) = \{E \mid \mathbb{E}(E) > 0\}$, is the set of networks in $\mathbb{E}$ with positive probability. Sometimes we also write $\{(\rho_i : E_i) \mid \mathbb{E}(E_i) = \rho_i\}$ to denote $\mathbb{E}$. If $\mathbb{E}(E) = 1$, then $\mathbb{E}$ is the *Dirac* distribution δ_E. Given a real number a, $a \cdot \mathbb{E}$ is the distribution such that $(a \cdot \mathbb{E})(E) = a \cdot \mathbb{E}(E)$ for each $E \in Supp(\mathbb{E})$ if $a \cdot |\mathbb{E}| \leq 1$. Moreover $\mathbb{E} = \mathbb{E}_1 + \mathbb{E}_2$ whenever for each E, $\mathbb{E}(E) = \mathbb{E}_1(E) + \mathbb{E}_2(E)$. Parallel composition of network distributions $\mathbb{E} \parallel \mathbb{F}$ is defined as a distribution such that $(\mathbb{E} \parallel \mathbb{F})(E \parallel F) = \mathbb{E}(E) \cdot \mathbb{F}(F)$. Given an equivalence relation $\mathcal{R}$ on networks, $\mathbb{E}\, \mathcal{R}\, \mathbb{F}$ iff $\mathbb{E}(S) = \mathbb{F}(S)$ for each $S \in \mathcal{N}/\mathcal{R}$ where $\mathbb{E}(S) = \sum_{E \in S} \mathbb{E}(E)$.

A substitution $\{y/x\}$ can be applied to a process, network, or network distribution. When applied to a network distribution, it means applying the substitution to each network in the support of the distribution. The set of free and bound names in E, denoted by

[1] $\mathbb{Q}^+$ is the set of all the positive rational numbers.

Table 1. Structural congruence of processes and networks.

$p + 0 \equiv p$	$p + q \equiv q + p$	$\nu x\nu y p \equiv \nu y\nu x p$	$(p + q) + r \equiv p + (q + r)$
$E \parallel 0 \equiv E$	$\nu x\nu y E \equiv \nu y\nu x E$	$\{\emptyset \longmapsto l\} \equiv 0$	$\lfloor \nu x p \rfloor_l^M \equiv \nu x \lfloor p \rfloor_l^M, x \notin M$
$E \parallel F \equiv F \parallel E$	$(E \parallel F) \parallel G \equiv E \parallel (F \parallel G)$	$\nu x E \parallel F \equiv \nu x(E \parallel F), x \notin fn(F)$	
$\lfloor p \rfloor_l^M \equiv \lfloor q \rfloor_l^M, p \equiv q$	$\{\mathbb{L}_1 \longmapsto k\} \parallel \{\mathbb{L}_2 \longmapsto k\} \equiv \{\mathbb{L}_1 \cup \mathbb{L}_2 \longmapsto k\}, l(\mathbb{L}_1) \cap l(\mathbb{L}_2) = \emptyset$		

$fn(E)$ and $bn(E)$ respectively, are defined as expected except that $fn(\lfloor p \rfloor_l^M) = fn(p) \cup M$. Structural congruence of processes and networks, $\equiv$, is the least equivalence relation and congruence closed by α-conversion and the rules in Table 1, which can be extended to distributions as usual. Let $loc(E)$ denote the set of locations located in a network, i.e. $loc(0) = \emptyset$, $loc(\lfloor p \rfloor_l) = \{l\}$, $loc(\{\mathbb{L} \longmapsto l\}) = \emptyset$, $loc(\nu x E) = l(E)$, and $loc(E \parallel F) = l(E) \cup l(F)$. Differently, $l(E)$ is used to denote all the location names appearing in E including those in connectivity information. The definition of $l(E)$ coincides with $loc(E)$ except that $l(\{\mathbb{L} \longmapsto l\}) = l(\mathbb{L}) \cup \{l\}$.

We use $\rho_{k \longmapsto l}(E)$ to denote the connection probability from k to l in E. When the requested probability does not occur in E the result is $\theta_{k \longmapsto l}$ which denotes an *unknown probability*, i.e. $\rho_{k \longmapsto l}(E) = \rho$ if $E \equiv \{(\rho, k)\} \longmapsto l\} \parallel E'$ for some E', otherwise $\rho_{k \longmapsto l}(E) = \theta_{k \longmapsto l}$. We generalize network distributions to contain unknown probabilities. In the following let $\varrho_1, \varrho_2 ::= \rho \mid \theta_{k \longmapsto l} \mid (1 - \theta_{k \longmapsto l}) \mid \varrho_1 \cdot \varrho_2$ be the *generalized probability* which may contain unknown values. The set of *generalized network distribution*, $\mathcal{GND}$, is defined inductively as follows: i) $\mu \in \mathcal{GND}$ if $\mu \in \mathcal{ND}$; ii) $\mu \in \mathcal{GND}$ if there exists ϱ and $\mu_1, \mu_2 \in \mathcal{GND}$ such that $\mu = \varrho \cdot \mu_1 + (1 - \varrho) \cdot \mu_2$. Without causing any confusion, we also use $\mu, \mu', \mu_1, \ldots$ to range over $\mathcal{GND}$. For a generalized network distribution μ, we may substitute unknown probabilities in μ with known probabilities. In order to do so, we introduce the operator $\circ$ such that $\mu \circ \mathcal{D}_l(E)$ is a distribution equal to μ except that an unknown probability $\theta_{k \longmapsto l}$ in μ has been replaced with the probability ρ if $(\rho, k) \in \mathcal{D}_l(E)$. Formally, $(\mu \circ \mathcal{D}_l(E))(F) = (\mu(F)) \circ \mathcal{D}_l(E)$ for each $F \in Supp(\mu)$ where $\circ$ is overloaded to deal with generalized probabilities such that i) $\varrho \circ \mathcal{D}_l(E) = \rho$ if $\varrho = \rho$; ii) $\theta_{k \longmapsto l} \circ \mathcal{D}_l(E) = \rho$ and $(1 - \theta_{k \longmapsto l}) \circ \mathcal{D}_l(E) = 1 - \rho$ if $(k, \rho) \in \mathcal{D}_l(E)$; iii) $(\varrho_1 \cdot \varrho_2) \circ \mathcal{D}_l(E) = (\varrho_1 \circ \mathcal{D}_l(E)) \cdot (\varrho_2 \circ \mathcal{D}_l(E))$.

As mentioned in the introduction we make use of network topology constraints in order to restrict the mobility of nodes. We define the syntax of *topology constraints* Φ, ranged over by ϕ, as follows: $\phi ::= \rho_{k \longmapsto l} = \rho \mid \phi \wedge \phi \mid \phi \vee \phi$ where $\rho_{k \longmapsto l}$ refers to the variable connection probability from k to l, $\rho \in \wp$, and ϕ evaluates to true and false in the obvious way. The above syntax is simple but expressive. For example we can define constraints such as $\rho_{l \longmapsto k} \geq 0.8$ and $\rho_{l \longmapsto m} + \rho_{l \longmapsto n} = 1$ as follows where $\bowtie \in \{<, >, \leq, \geq\}$:

1. $\rho_{l \longmapsto k} \bowtie \rho = \bigvee_{\rho' \in \wp \wedge \rho' \bowtie \rho} \rho_{l \longmapsto k} = \rho'$,
2. $\rho_{l \longmapsto m} + \rho_{l \longmapsto n} \bowtie \rho = \bigvee_{\rho_1, \rho_2 \in \wp \wedge \rho_1 + \rho_2 \bowtie \rho} (\rho_{l \longmapsto m} = \rho_1 \wedge \rho_{l \longmapsto n} = \rho_2)$.

Given a topology constraint ϕ, define operator $E[\phi]$ to evaluate ϕ under a network E by $E[\phi_1 \bowtie \phi_2] = E[\phi_1] \bowtie E[\phi_2]$ with $\bowtie \in \{\wedge, \vee\}$, $E[\rho_{l \longmapsto k} = \rho] = \mathit{true}$ if $\rho_{l \longmapsto k}(E) = \rho$, otherwise $E[\rho_{l \longmapsto k} = \rho] = \mathit{false}$, and boolean operators are evaluated as usual.

Topology constraints together with connectivity networks is the source for defining continuous time stochastic mobility. A *stochastic mobility function* (SMF) $mf : \mathcal{CN} \times \mathcal{CN} \times \Phi \to \mathbb{Q}^+$ is a partial function where $mf(C, C', \phi)$ returns the mobility rate from C to C' given the topology constraint ϕ. We assume $mf(C, C, true) = 0$ if the connectivity network C is static, i.e. it cannot evolve into other networks. For simplicity we let $mf(C, C', \phi) = \perp$ denote that the mobility rule from C to C' under condition ϕ is undefined. An SMF is *valid* if for each C, C' such that $mf(C, C', \phi) \neq \perp$ for some ϕ, then $\rho_{k \mapsto\!\!\!\!\rightarrow l}(C) = \theta_{k \mapsto\!\!\!\!\rightarrow l}$ iff $\rho_{k \mapsto\!\!\!\!\rightarrow l}(C') = \theta_{k \mapsto\!\!\!\!\rightarrow l}$ for all k and l. Intuitively, the condition guarantees that when a mobility step from C to C' happens, it only changes the probability of connectivities in C, we can neither obtain information about connectivities not in C, nor lose connectivities in C. For instance let $C = \{\{(0.5, m), (0.9, n)\} \longmapsto l\}$ and $C' = \{\{(0.8, m)\} \longmapsto l\}$, a mobility rule from C to C' is not valid since the connectivity information of $\rho_{n \mapsto\!\!\!\!\rightarrow l}$ is lost in C', similarly a mobility rule from C' to C is not valid either. In the following we will only consider valid SMFs, and we assume that there is a given mf throughout the paper.

Since we have infinitely many connectivity networks, it is not reasonable to always define mobility rules for all of them. Instead we allow an mf to be defined for just finitely many pairs C and C' and topology constraints ϕ. We call those rules *explicit* mobility rules. A connection probability $\rho_{l \mapsto\!\!\!\!\rightarrow k}$ has an explicit mobility rule if there exists $mf(C, C', \phi) \neq \perp$ with $\rho_{l \mapsto\!\!\!\!\rightarrow k}(C) \neq \theta_{l \mapsto\!\!\!\!\rightarrow k}$. For any connection probability $\rho_{l \mapsto\!\!\!\!\rightarrow k}$ with no explicit mobility rule we assume it has the implicit mobility rule $mf(\{\{(0, l)\} \longmapsto k\}, \{\{(0, l)\} \longmapsto k\}, true) = 0$, that is l is not and will never be connected to k. The default implicit mobility can be changed without affecting our theory.

The structural congruence closed set of *well-formed* networks $\mathcal{N}_{mf}$ under a given SMF mf is inductively defined as follows:

1. $0 \in \mathcal{N}_{mf}$, $\lfloor p \rfloor_l^M \in \mathcal{N}_{mf}$, and $\nu x E \in \mathcal{N}_{mf}$ if $E \in \mathcal{N}_{mf}$,
2. $E \parallel F \in \mathcal{N}_{mf}$ if $E, F \in \mathcal{N}_{mf}$ with $loc(E) \cap loc(F) = \emptyset$ and there does not exist $l, k \in \mathcal{L}$ such that $\rho_{l \mapsto\!\!\!\!\rightarrow k}(E) \neq \theta_{l \mapsto\!\!\!\!\rightarrow k}$ and $\rho_{l \mapsto\!\!\!\!\rightarrow k}(F) \neq \theta_{l \mapsto\!\!\!\!\rightarrow k}$,
3. $C \in \mathcal{N}_{mf}$ if there exists C' and ϕ such that $mf(C, C', \phi) \neq \perp$.

Clause 1 is trivial. Clause 2 means that locations are unique and that connectivity information for a single connection can only appear once, while clause 3 (together with clause 2) requires that the connectivity network part of a network can be divided into subnetworks for each of which mobility must be defined by the given mf.

3 Labeled Transition System

We use $\mathcal{A}_p$ to denote the actions of processes, defined as follows:

$$\alpha_p ::= \nu \tilde{x} \langle x \triangleright L^* \rangle \mid (x) \mid \lambda,$$

where $\nu \tilde{x} \langle x \triangleright L^* \rangle$ denotes broadcasting the message x to nodes at locations in L^*. Whenever x is bounded $\tilde{x} = \{x\}$, otherwise $\tilde{x} = \emptyset$. The (x) means that the process can receive a (group) broadcast message. λ denotes a Markovian action with specified rate. The semantics of processes is given in Table 2 where all the rules are standard, and $\rightsquigarrow = (\rightarrow \cup \twoheadrightarrow)$ with $\twoheadrightarrow$ denoting Markovian transitions.

Table 2. Labeled Transition System of Processes

$$\frac{}{\lambda \cdot p \overset{\lambda}{\twoheadrightarrow} p}\ (\text{MAR}) \qquad \frac{}{\langle x \triangleright L^* \rangle \cdot p \xrightarrow{\langle x \triangleright L^* \rangle} p}\ (\text{PRE}) \qquad \frac{p \overset{\alpha_p}{\rightsquigarrow} p'}{p + q \overset{\alpha_p}{\rightsquigarrow} p'}\ (\text{SUM}) \qquad \frac{p \overset{\alpha_p}{\rightsquigarrow} p' \quad x = y}{[x = y]p, q \overset{\alpha_p}{\rightsquigarrow} p'}\ (\text{IF})$$

$$\frac{}{(x) \cdot p \xrightarrow{(y)} p\{y/x\}}\ (\text{INP}) \qquad \frac{p \overset{\alpha_p}{\rightsquigarrow} p' \quad x \notin fn(\alpha_p)}{\nu x p \overset{\alpha_p}{\rightsquigarrow} \nu x p'}\ (\text{RES}) \qquad \frac{q \overset{\alpha_p}{\rightsquigarrow} q' \quad x \neq y}{[x = y]p, q \overset{\alpha_p}{\rightsquigarrow} q'}\ (\text{ELSE})$$

$$\frac{q \equiv p \overset{\alpha_p}{\rightsquigarrow} p' \equiv q'}{q \overset{\alpha_p}{\rightsquigarrow} q'}\ (\text{STR}) \qquad \frac{p \overset{\alpha_p}{\rightsquigarrow} p' \quad A \overset{def}{=} p}{A \overset{\alpha_p}{\rightsquigarrow} p'}\ (\text{CON}) \qquad \frac{p \xrightarrow{\langle x \triangleright L^* \rangle} p' \quad y \notin fn(\nu x p)}{\nu x p \xrightarrow{\nu y \langle y \triangleright L^* \rangle} p'\{y/x\}}\ (\text{bOPEN})$$

We use $\mathcal{A}$ to denote the actions of networks defined as follows:

$$\alpha ::= \nu \tilde{x} \langle x \triangleright L^*, \mathbb{L} \rangle @ l \mid (x @ L^*, \mathbb{L}) \triangleleft l \mid \lambda \mid \phi : \lambda \mid \tau.$$

Different from process actions, for the actions of networks connectivity information is attached to any broadcast and reception action. Action $\nu \tilde{x} \langle x \triangleright L^*, \mathbb{L} \rangle @ l$ denotes that the node at location l can broadcast the message x to the node at location $k \in L^*$ with probability ρ if $(\rho, k) \in \mathbb{L}$. Accordingly $(x @ L^*, \mathbb{L}) \triangleleft l$ means that the node at location $k \in L^*$ can receive the message x from location l with probability ρ if $(\rho, k) \in \mathbb{L}$. The $\phi : \lambda$ is a novel action named *condition guarded Markovian action*. This action is used to model topology constrained mobility where mobility is triggered only when certain conditions are satisfied. The τ and λ are standard.

The semantics of networks is given in Table 3 with $\rightsquigarrow$ (as in Table 2) being the union of $\twoheadrightarrow$ and $\rightarrow$. For readability we also write δ_E directly as E. The behavior of a node is determined by the process in it, but the actions of a node may be enriched with connectivity information as well as the source and destination respectively if the action is either a broadcast or a reception action. Rule (nREC1) says that when a process in a node located at a location can perform a reception, then the node can also perform a reception action, similarly for (nBRD) which deals with broadcast actions. In (nBRD) we remove l from L^* since a node cannot receive messages broadcasted from itself. Note in (nBRD) and (nREC1) there is no connectivity information, so the corresponding connectivity sets in the labels are empty, and furthermore in (nREC1) the node at location l is able to receive a message from location k with unknown probability denoted by $\theta_{l \longmapsto k}$, this is the only rule where unknown probability is added. Two parallel networks E and F communicate by broadcast as shown by (nSYN) where one network can perform a broadcast action while the other one can perform a reception action, similarly in (nREC2) we let two networks in parallel can perform a reception action simultaneously. As shown in both (nSYN) and (nREC2), we require that the destinations of the broadcast and reception actions of the two participants coincide.

Rules (nBRD), (nREC1), (nREC2), and (nSYN) deal with group broadcast when $L^* = L$. Different from broadcast where the broadcast messages can be received by any node in any location, group broadcast has specified destinations, nodes at locations which are not in the set of the destinations will simply ignore the messages and stay

unchanged, this is taken care of by rule (nIGN). As explained in the introduction we introduce a low level protocol taking care of flooding assuming that a message can only be received by a node at most once. The parameter M at a node is used to keep track of the messages already received, so only if the coming message is not already in M, it will be dealt with, otherwise it will be simply ignored as explained in rules (nREC1) and (nIGN). On the other hand, if process p at location l cannot perform a reception, it will simply ignore all the coming messages, and stay unchanged as illustrated by (nIGN).

In (nSYN) E and F may obtain new connectivity information $\mathbb{L}$ and $\mathbb{K}$ from each other and update the unknown probabilities that might appear in distributions $\mathbb{E}$ and $\mathbb{F}$ via the operator $\circ$, similarly for (nREC2). In (nSYN) K is the union of the set of locations in F, $loc(F)$, and the set of locations in $\mathbb{K}$ which are not connected to l, $\mathcal{Z}(\mathbb{K}) = \{k \mid (0,k) \in \mathbb{K}\}$. We remove K from the resulting action where $\mathbb{L} \setminus K = \{(\rho,k) \in \mathbb{L} \mid k \notin K\}$. It makes sense to remove $\mathcal{Z}(\mathbb{K})$ from the destination set of the broadcast action since nodes at locations $\mathcal{Z}(\mathbb{K})$ will for sure not receive messages from l. Also we remove locations $loc(F)$ since all the nodes at locations $loc(F)$ in F after the transition will receive the broadcast message.

If an action is not broadcast or reception, networks can execute in parallel without synchronization, this gives the rule (nPAR). Network $\{\mathbb{K} \longmapsto l\}$ only contains connectivity information about l, it can reveal its connectivity information by performing a (group) reception which is shown by (nCONN); it can also, in order to synchronize on broadcast from locations not being l, perform a (group) reception whose source location is different from l with empty connectivity information as illustrated by the rule (nLOS). A broadcast with empty destination has no impact to the outside of the emitting network, therefore it should be seen as an internal action τ which is shown by (nLOC). Due to (nSYN) and (nREC2), (nLOC) can only happen at top level. Rule (nMOB) allows a connectivity network to evolve into another according to the mobility rule defined by the given *mf* carrying out a condition guarded Markovian action $\phi : \lambda$. By (nTRU) if ϕ is evaluated to true, then $\phi : \lambda$ will become a Markovian transition λ. Note in (nREC1) and (nIGN), we require that $l \neq k$ which means that a process at location l cannot receive messages broadcasted from the same location. The rules (nOPEN), (nRES), (nMAR), and (nSTR) are standard.

In our calculus we allow continuous delay, probabilistic choice, and non-deterministic choice, as result each network corresponds to a Markov Automaton [9] which is the integration of probabilistic automata [7] with interactive Markov chains [8]. As usual we assume networks to be free of divergence with probability 1, see e.g. [7], in order to avoid an unrealistic situation where infinitely many actions can happen in finite time. For instance network $E \stackrel{def}{=} \lfloor A \rfloor_l \parallel \lfloor \lambda \cdot 0 \rfloor_k$ with $A \stackrel{def}{=} \langle x \rangle \cdot A$ is not free of divergence, since E can perform broadcast from l for infinitely many times, and thus blocks the Markovian transition at k for ever.

4 Weak Bisimulation

In this section we provide a weak bisimulation congruence for our calculus. We say that a network E is *stable*, written $E\!\downarrow$, if $E \stackrel{\tau}{\not\mapsto}$ and $E \stackrel{\langle x \triangleright L^*, \mathbb{L} \rangle @ l}{\not\mapsto}$. Note that broadcasts are considered to be immediate and take no time, since they are non-blocking and will be

Table 3. Labeled Transition System of Networks

$$\frac{p \xrightarrow{(x)} p' \quad (l \in L^* \wedge x \notin M \wedge k \neq l)}{\lfloor p \rfloor_l^M \xrightarrow{(x@L^*,\emptyset)\triangleleft k} \{(\theta_{l\longmapsto k} : \lfloor p' \rfloor_l^{M\cup\{x\}}), (1-\theta_{l\longmapsto k} : \lfloor p \rfloor_l^M)\}} \text{ (nREC1)}$$

$$\frac{E \xrightarrow{\nu\tilde{y}\langle y \triangleright L^*,\mathbb{L}\rangle@l} \mathbb{E} \quad F \xrightarrow{(y@L^*,\mathbb{K})\triangleleft l} \mathbb{F} \quad \tilde{y} \cap fn(F) = \emptyset \quad K = loc(F) \cup \mathcal{Z}(\mathbb{K})}{E \parallel F \xrightarrow{\nu\tilde{y}\langle y \triangleright (L^*\setminus K),(\mathbb{L}\cup\mathbb{K})\setminus K\rangle@l} (\mathbb{E} \circ \mathcal{D}(F)) \parallel (\mathbb{F} \circ \mathcal{D}(E))} \text{ (nSYN)}$$

$$\frac{E \xrightarrow{\langle x \triangleright L^*,\mathbb{L}\rangle@l} \mathbb{E} \quad y \notin fn(\nu xE)}{\nu xE \xrightarrow{\nu y\langle y \triangleright L^*,\mathbb{L}\rangle@l} \mathbb{E}\{y/x\}} \text{ (nOPEN)} \qquad \frac{E \xrightarrow{(x@L^*,\mathbb{L})\triangleleft l} \mathbb{E} \quad F \xrightarrow{(x@L^*,\mathbb{K})\triangleleft l} \mathbb{F}}{E \parallel F \xrightarrow{(x@L^*,\mathbb{L}\cup\mathbb{K})\triangleleft l} (\mathbb{E} \circ \mathcal{D}(F)) \parallel (\mathbb{F} \circ \mathcal{D}(E))} \text{ (nREC2)}$$

$$\frac{p \overset{\lambda}{\twoheadrightarrow} p'}{\lfloor p \rfloor_l^M \overset{\lambda}{\twoheadrightarrow} \lfloor p' \rfloor_l^M} \text{ (nMAR)} \qquad \frac{p \xrightarrow{\nu\tilde{x}\langle x \triangleright L^*\rangle} p'}{\lfloor p \rfloor_l^M \xrightarrow{\nu\tilde{x}\langle x \triangleright (L^*\setminus l),\emptyset\rangle@l} \lfloor p' \rfloor_l^M} \text{ (nBRD)} \qquad \frac{E \xrightarrow{\phi:\lambda} \mathbb{E} \quad E[\phi] = true}{E \overset{\lambda}{\twoheadrightarrow} \mathbb{E}} \text{ (nTRU)}$$

$$\frac{F \equiv E \overset{\alpha}{\rightsquigarrow} \mathbb{E} \equiv \mathbb{F}}{F \overset{\alpha}{\rightsquigarrow} \mathbb{F}} \text{ (nSTR)} \qquad \frac{E \overset{\alpha}{\rightsquigarrow} \mathbb{E} \quad \alpha \in \{\lambda, \phi : \lambda\}}{E \parallel F \overset{\alpha}{\rightsquigarrow} \mathbb{E} \parallel F} \text{ (nPAR)} \qquad \frac{E \xrightarrow{\nu\tilde{y}\langle x \triangleright \emptyset,\mathbb{L}\rangle@l} \mathbb{E}}{E \xrightarrow{\tau} \mathbb{E}} \text{ (nLOC)}$$

$$\frac{E \overset{\alpha}{\rightsquigarrow} \mathbb{E} \quad x \notin fn(\alpha)}{\nu xE \overset{\alpha}{\rightsquigarrow} \nu x\mathbb{E}} \text{ (nRES)} \qquad \frac{mf(C, C', \phi) = \lambda}{C \xrightarrow{\phi:\lambda} C'} \text{ (nMOB)} \qquad \frac{k \neq l \wedge (l \notin L^* \vee x \in M \vee p \overset{(x)}{\nrightarrow})}{\lfloor p \rfloor_l^M \xrightarrow{(x@L^*,\emptyset)\triangleleft k} \lfloor p \rfloor_l^M} \text{ (nIGN)}$$

$$\frac{l \neq k}{\{\mathbb{K} \longmapsto k\} \xrightarrow{(x@L^*,\emptyset)\triangleleft l} \{\mathbb{K} \longmapsto k\}} \text{ (nLOS)} \qquad \frac{}{\{\mathbb{K} \longmapsto l\} \xrightarrow{(x@L^*,\mathbb{K})\triangleleft l} \{\mathbb{K} \longmapsto l\}} \text{ (nCONN)}$$

triggered immediately. Accordingly, a network distribution $\mathbb{E}$ is stable, written $\mathbb{E}\downarrow$, iff $E\downarrow$ for each $E \in Supp(\mathbb{E})$.

In order to evaluate the exit rate of a network we, similar with [8], define the function $\gamma : \mathcal{N}_{mf} \times 2^{\mathcal{N}_{mf}} \mapsto \mathbb{Q}^+$ which returns the exit rate from a given network to a set of networks via Markovian transitions. The formal definition is as follows where $\{\!|\}\!|$ denotes multiset: $\gamma(E, S) = \sum\{\!|\lambda \cdot \mathbb{E}(S) \mid E \overset{\lambda}{\twoheadrightarrow} \mathbb{E}\}\!|$. Due to a *race condition* [11, 8] among Markovian transitions they will compete in order to be executed first, this gives us the following natural transitions. Let $E \xrightarrow{\lambda} \mathbb{E}$ if $E\downarrow$ where $\lambda = \gamma(E, \mathcal{N}_{mf})$ and $\mathbb{E}(F) = \frac{\gamma(E,\{F\})}{\lambda}$ for all F. Refer to the following example for an illustration of race condition.

Example 1. Let $E = \lfloor \lambda_1 \cdot p + \lambda_2 \cdot q \rfloor_l$. It is easy to see that E has two Markovian transitions according to Table 2 and 3: $E \overset{\lambda_1}{\twoheadrightarrow} \lfloor p \rfloor_l$ and $E \overset{\lambda_2}{\twoheadrightarrow} \lfloor q \rfloor_l$. The exit rate of E is equal to $\lambda = \lambda_1 + \lambda_2$, and moreover the two Markovian transitions will compete to be executed first. Due to the race condition, the first transition will be executed with probability $\frac{\lambda_1}{\lambda}$, while the second one will be executed with probability $\frac{\lambda_2}{\lambda}$, i.e. $E \xrightarrow{\lambda} \{\frac{\lambda_1}{\lambda} : \lfloor p \rfloor_l, \frac{\lambda_2}{\lambda} : \lfloor q \rfloor_l\}$.

We use $E \overset{\alpha}{\Longrightarrow} \mathbb{E}$ to denote that a distribution $\mathbb{E}$ is reached through a sequence of steps which are internal except one being equal to α. Formally $\overset{\alpha}{\Longrightarrow}$ is the least relation such that, $E \overset{\alpha}{\Longrightarrow} \mathbb{E}$ iff

1. $\alpha = \tau$ and $\mathbb{E} = \delta_E$, or

2. there exists a step $E \xrightarrow{\beta} \mathbb{E}'$ such that $\mathbb{E} = \sum_{E' \in Supp(\mathbb{E}')} \mathbb{E}'(E') \cdot \mathbb{E}_{E'}$, where $E' \stackrel{\tau}{\Longrightarrow} \mathbb{E}_{E'}$ if $\beta = \alpha$, otherwise $E' \stackrel{\alpha}{\Longrightarrow} \mathbb{E}_{E'}$ and $\beta = \tau$.

As in [7] we also define the combined transition $\stackrel{\alpha}{\Longrightarrow}_c$ such that: $E \stackrel{\alpha}{\Longrightarrow}_c \mathbb{E}$ iff there exists $\{E \stackrel{\alpha}{\Longrightarrow} \mathbb{E}_i\}_{1 \leq i \leq n}$ and $\{w_i\}_{1 \leq i \leq n}$ with $\sum_{1 \leq i \leq n} w_i = 1$ and $\sum_{1 \leq i \leq n} w_i \cdot \mathbb{E}_i = \mathbb{E}$.

As an abstraction we disregard the emitter of a broadcast message and allow to equate $\nu\tilde{x}\langle x \rhd L^*, \mathbb{L}\rangle@l$ and $\nu\tilde{x}\langle x \rhd L^*, \mathbb{L}\rangle@k$ indicating that in a wireless broadcast setting the sender of a message is not important, that is only the message (and the probability by which it is received), since the receiver of a message may not precisely know whom is the actual emitter of the message. To further enforce what we in the Introduction called broadcast abstraction we will also allow that a broadcast can be simulated by several broadcast messages. In order to do so we define the combination of two broadcast actions such that

$$\nu\tilde{x}\langle x \rhd L_1, \mathbb{L}_1\rangle@l_1 \otimes \nu\tilde{x}\langle x \rhd L_2, \mathbb{L}_2\rangle@l_2 = \nu\tilde{x}\langle x \rhd L, \mathbb{L}\rangle@l$$

where $L = L_1 \cup L_2$, l is any location name, and $\mathbb{L} = \mathbb{M}_1 \cup \mathbb{M}_2$ with

$$\mathbb{M}_1 = \{(\rho, k) \in \mathbb{L}_1 \mid k \in L_1 \setminus L_2\} \cup \{(\rho, k) \in \mathbb{L}_2 \mid k \in L_2 \setminus L_1\},$$

$$\mathbb{M}_2 = \{(1 - (1 - \rho_1) \cdot (1 - \rho_2), k) \mid k \in L_1 \cap L_2 \wedge (\rho_1, k) \in \mathbb{L}_1 \wedge (\rho_2, k) \in \mathbb{L}_2\}.$$

Intuitively, the resulting combination of two actions has the same effects as the original two. There are three cases to consider. If a location k is only in L_1, then the probability for location k receiving the broadcast message x will not be changed by $\nu\tilde{x}\langle x \rhd L_2, \mathbb{L}_2\rangle@l_2$, similarly for locations only in L_2. For a location k appearing in both L_1 and L_2, the probability for k not receiving x is equal to $(1-\rho_1)\cdot(1-\rho_2)$ if $(\rho_1, k) \in \mathbb{L}_1$ and $(\rho_2, k) \in \mathbb{L}_2$, as a result the probability for a node at location k receiving x is equal to $1 - (1 - \rho_1) \cdot (1 - \rho_2)$. Obviously, $\otimes$ is associative and commutative. We extend the broadcast transitions in the following way: $E \xrightarrow{\langle x \rhd L^*, \mathbb{L}\rangle@l} \mathbb{E}$ iff $E \stackrel{\alpha_1}{\Longrightarrow} \stackrel{\alpha_2}{\Longrightarrow} \ldots \stackrel{\alpha_n}{\Longrightarrow} \mathbb{E}$ with $\langle x \rhd L^*, \mathbb{L}\rangle@l = (\bigotimes_{1 \leq i \leq n} \alpha_i)$.

According to Table 3 there might occur unknown probabilities during the evolution of networks. Intuitively, to compare two network distributions where unknown probabilities may occur, we consider all the possibilities for substitution of those unknown probabilities by concrete probabilities i.e. two networks are equivalent if they behave equivalently in all possible substitution contexts. In order to do so, we introduce operator $\bullet$ such that $E \bullet C$ denotes a network behaving like E but obtaining new information from C, that is, $E \bullet 0 = E$, $E \bullet (C \parallel C') = (E \bullet C) \bullet C'$, and $E \bullet (\{\{(\rho, k)\} \cup \mathbb{L} \longmapsto l\}) = E \bullet (\{\mathbb{L} \longmapsto l\})$ if $\rho_{k \longmapsto l}(E) \neq \theta_{k \longmapsto l}$, otherwise $E \bullet (\{\{(\rho, k)\} \cup \mathbb{L} \longmapsto l\}) = (E \parallel \{\{(\rho, k)\} \longmapsto l\}) \bullet (\{\mathbb{L} \longmapsto l\})$. Intuitively $\bullet$ is used to supply a network E with auxiliary connection probabilities, information about connections which probability are already known in E will simply be ignored.

In the definition of our bisimulation we make use of the following finite sets of connectivity networks: $\mathcal{CN}_L = \{C \in \mathcal{CN} \mid \forall l, k \in L. \rho_{k \longmapsto l}(C) \neq \theta_{k \longmapsto l}\}$. Intuitively, $\mathcal{CN}_L$ contains all the connectivity networks such that the probability of $\rho_{k \longmapsto l}$ is known for all $l, k \in L$. Below follows the definition of weak bisimulation of networks where we

use $C_{E,F,k}$ to range over $\mathcal{CN}_{(l(E)\cup l(F)\cup\{k\})}$, and we let α_k range over all actions including λ except the reception actions from locations l where $l \neq k$.

Definition 1. *An equivalence relation $\mathcal{R} \subseteq \mathcal{N}_{mf} \times \mathcal{N}_{mf}$ is a weak bisimulation iff $E\ \mathcal{R}\ F$ implies that for each k and $C_{E,F,k}$, whenever $E \bullet C_{E,F,k} \xrightarrow{\alpha_k} \mathbb{E}$, there exists $F \bullet C_{E,F,k} \stackrel{\alpha_k}{\Longrightarrow}_c \mathbb{F}$ such that $\mathbb{E}\ \mathcal{R}\ \mathbb{F}$. Let E and F be weak bisimilar, written as $E \approx_{mf} F$, if there exists a weak bisimulation $\mathcal{R}$ such that $E\ \mathcal{R}\ F$.*

The cases when α_k is τ or λ are standard. When $\alpha_k = (x@L, \mathbb{L}) \triangleleft k$, any received message must be matched by receiving the same message with the same probabilities from the same sender. Observe that the source of the message cannot appear in $loc(E)$ due to the semantics in Table 3, as a consequence one may prove that $E \approx_{mf} F$ implies $loc(E) = loc(F)$.

Example 2. Given a *mf* such that l and k can always connect to all locations except m with the same probability, and all locations can always connect to l and k with the same probability. Then $\lfloor (x) \cdot \langle x \rangle \rfloor_l \parallel \lfloor 0 \rfloor_k \parallel \lfloor 0 \rfloor_m \approx_{mf} \lfloor (x) \cdot \langle x \rangle \rfloor_k \parallel \lfloor 0 \rfloor_l \parallel \lfloor 0 \rfloor_m$ but since l and k can receive messages from the node at location m with different probabilities $\lfloor (x) \cdot \langle x \rangle \rfloor_l \parallel \lfloor 0 \rfloor_k \not\approx_{mf} \lfloor (x) \cdot \langle x \rangle \rfloor_k \parallel \lfloor 0 \rfloor_l$.

When a network is not stable, then all the Markovian transitions are blocked, and cannot affect the behavior of the network. This is related to the *maximal progress assumption* which is a quite common in time (discrete and continuous) process algebra [13, 14, 8].

Example 3. Consider two networks:$E = \lfloor \langle x \triangleright L \rangle \cdot p + \lambda \cdot q \rfloor_l$ and $F = \lfloor \langle x \triangleright L \rangle \cdot p \rfloor_l$, since E is not stable due to $E \xrightarrow{\langle x \triangleright L, \emptyset \rangle @ l}$, therefore the Markovian transition $E \xrightarrow{\lambda}$ can be omitted, obviously $E \approx_{mf} F$.

When $\alpha_k = \nu \tilde{x} \langle x \triangleright L^*, \mathbb{L} \rangle @ l$ any broadcast message x must be matched by a broadcast action containing the same x, and x must be received at the same locations with the same probability, but the emitter need not be the same.

Example 4. Given a *mf* where l is disconnected from k forever, then $\lfloor \langle x \triangleright l \rangle \rfloor_k \approx_{mf} \lfloor 0 \rfloor_k$. If $\rho_{l \mapsto k}$ is not always 0 then $\lfloor \langle x \triangleright l \rangle \rfloor_k \not\approx_{mf} \lfloor 0 \rfloor_k$, but if reception at the node at l has no effect then e.g. $\lfloor \langle x \triangleright l \rangle \rfloor_k \parallel \lfloor 0 \rfloor_l \approx_{mf} \lfloor 0 \rfloor_k \parallel \lfloor 0 \rfloor_l$.

Additionally when $\alpha_k = \nu \tilde{x} \langle x \triangleright L^*, \mathbb{L} \rangle @ l$, we also allow that a broadcast can be simulated by a series of broadcasts whose combination is equivalent to the original broadcast. This relies on the assumption that each message can only be received by a node at most once.

Example 5. Given a *mf* such that location l can receive messages from location k with probability either 1 or 0. Then $\lfloor \langle x \triangleright l \rangle \cdot \langle x \triangleright l \rangle \rfloor_k \parallel \lfloor p \rfloor_l^M \approx_{mf} \lfloor \langle x \triangleright l \rangle \rfloor_k \parallel \lfloor p \rfloor_l^M$ for any p. The reason is that after the process at location k receives the message x, it will remember it, and if it receives the same message for the second time, it will simply ignore it and stay unchanged.

In all cases in Definition 1 we use $C_{E,F,k}$ to eliminate all the possible unknown probabilities during the evolution of both E and F. Observe that unknown probabilities can only appear in derivatives on networks in case of broadcast and reception actions. The reason to include k is because k might be any location not appearing in either E or F, thus when E or F performs a reception from k, an unknown probability $\theta_{l\longmapsto k}$ with $l \in l(E) \cup l(F)$ may arise. Such an unknown probability may be eliminated by applying any $C_{E,F,k}$. When performing broadcasts the only possible unknown probability in a derivative from E and F is of the form $\theta_{m\longmapsto n}$ with $m, n \in l(E) \cup l(F)$, thus it can also be removed by applying any $C_{E,F,k}$.

Example 6. Suppose a *mf* such that $\rho_{m\longmapsto n}$ is always equal to 0.5 and two networks: $E = \{\{(0.5, m)\} \longmapsto n\}$ and $F = 0$. Without applying a $C_{E,F,k}$, we will conclude that $E \not\approx_{mf} F$ since $E \xrightarrow{(x@L^*,\{(0.5,m)\})\triangleleft n} \delta_E$ which cannot be simulated by F. This is against our intuition since we know that $\rho_{m\longmapsto n}$ is always equal to 0.5, thus F should be able to exploit this fact from the given *mf*. By applying any $C_{E,F,k}$ it is easy to check that $E \approx_{mf} F$.

The following theorem shows that the weak bisimulation is a congruence.

Theorem 1. *$\approx_{mf}$ is a congruence.*

The definition of our bisimulation depends on a given SMF *mf*, the more restricted the *mf* the more bisimilar networks we can obtain. For instance, if we consider the extreme case where all the nodes are disconnected from each other all the time, that is, they cannot influence each other's behaviors, we then have $\lfloor p \rfloor_l \approx_{mf} \lfloor q \rfloor_l$ for any p, q.

5 A Leader Election Protocol

We illustrate the application of our calculus by modeling an adaption of the leader election protocol in [12]. Before giving the model we first explain how this protocol works. It is assumed that each node has a unique ID i. A node may regularly initiate an election of a new leader; it will start the process of building a spanning tree by broadcasting a message *Election* to its neighbors and then wait for acknowledgement messages, *Ack*, from its children in the tree. An *Ack* message will contain the information about the node with the highest ID the child has found. When a node j receives an *Election* from another node i, it will set i as its parent and then propagate *Election* to its neighbors and then wait for the acknowledgements *Ack* from its children. In a state waiting for *Ack* messages a node keeps track of the highest ID received before it times out after a certain time limit. When timing out a node (not being the root of the spanning tree) reports the highest ID found to its parent via an *Ack* message and enters a state where it waits to be informed about the new leader found. When the initiator of the run of the protocol times out waiting for *Ack* messages it broadcasts the new leader, i.e. the node with the highest ID found, to its neighbors via the message *Leader*. Notice that due to node mobility a child may disconnect from its parent before it sends the acknowledgement, the time out in this case prevents the parent getting stuck waiting for the acknowledgement forever. Similarly for a node waiting for announcements of a new leader, it will

Model 1 The model of the leader election protocol

$$
\begin{array}{ll}
Node(i,l,m,p) & = \lambda_{init} \cdot \langle E\_i \rhd I \rangle \cdot Init(i,l,m,p) \\
 & + \sum_{x \neq i}(E\_x) \cdot \langle E\_i \rhd I \rangle \cdot waitAck(i,l,m,x) \\
Init(i,l,m,p) & = \sum_{x \neq i}(A\_x) \cdot ([x > m]Init(i,l,x,p), Init(i,l,m,p)) \\
 & + \lambda_{exp} \cdot \langle L\_m \rhd I \rangle \cdot Node(i,m,m,p) \\
waitAck(i,l,m,p) & = \sum_{x \neq i}(A\_x) \cdot ([x > m]waitAck(i,l,x,p), waitAck(i,l,m,p)) \\
 & + \lambda_{exp} \cdot \langle A\_m \rhd p \rangle \cdot waitLeader(i,l,m,p) \\
waitLeader(i,l,m,p) & = \sum_{x \neq i}(L\_x) \cdot Node(i,x,m,p) \\
 & + \lambda_{par} \cdot \langle L\_m \rhd I \rangle \cdot Node(i,m,m,p)
\end{array}
$$

either receive the announcement in time, or it will time out and announce the node with highest ID it has found so far as the new leader.

The state of a node is represented by $Node(i,l,m,p)$ where i is the ID, l is the ID of its leader, m is the maximum ID known in a protocol run, and p is the ID of its parent.

To model this protocol we define three types of messages (names) where I is a finite set of all the possible ID numbers: $\{E\_i \mid i \in I\}$ is the set of *Election* messages, $\{A\_m \mid m \in I\}$ is the set of *Ack* messages, and $\{L\_l \mid l \in I\}$ is the set of *Leader* messages which announces the elected leader. In [12] the messages in a given election are all assigned a unique index used to distinguish the protocol run from other runs. For simplicity we omit these details in the model of the protocol in this paper.

To make the model more compact we extend the match operator in the following way: $[x > m]p, q$ denotes that the process will evolve into p if $x > m$, otherwise it will evolve into q, this operator can be defined using the standard operators in a straightforward way. The operator $\sum_{x \neq i}(E\_x)$ means that the input only accepts *Election* messages not from i, and ignores all the other messages, the operator can easily be encoded by a sequence of conditional operators prefixed by (x). We introduce similar operators for accepting just one type of protocol messages. The model of the protocol is given in Model 1 where λ_{init} and λ_{exp} denote the rate of initializing a new run of the protocol and the rate of timeout from waiting for the acknowledgements from children respectively. If a node is not involved in any election, it will be at state *Node*. The node with ID i can initialize an election by broadcasting the message E_i to its neighbors, and evolve into *Init*. When the neighbor nodes receive the message E_i, they will join the election and evolve into *waitAck* after forwarding the *Election* message to their neighbors. While at *Init* or *waitAck*, a node will wait for the acknowledgements from its neighbors. In order not to get stuck and wait for the acknowledgements forever, we let each node stop waiting with rate λ_{exp}. When the node at *Init* stops waiting for the acknowledgements, it will announce m, the maximal ID found so far, as the new leader. Differently, when timing out nodes at *waitAck* will send an acknowledgement together with the parameter m to their parents, and then evolve into *waitLeader* waiting for the announcement of the new leader. It may happen that a node will timeout when waiting for the announcement from its parent while at *waitLeader*, in this case it will simply announce m as its leader and terminate the election. Each node at *waitLeader* will timeout with a certain delay by rate λ_{par}.

Next we will show how to define mobility rules for our example. For simplicity we assume that there are four locations in the network: l, k, m, and n where all the nodes

are stationary except the node at l. Suppose that nodes at location k and l are always disconnected, thus the move of node at l will not affect the value of $\rho_{k\longmapsto l}$ and $\rho_{l\longmapsto k}$. There are two possible positions Pos_1 and Pos_2 for the node at location l such that when in Pos_1 it will be closer to the node at location m than the node at location n i.e. $\rho_{m\longmapsto l} > \rho_{n\longmapsto l}$ while in Pos_2 we have $\rho_{m\longmapsto l} < \rho_{n\longmapsto l}$. When the node at location l is at Pos_1, it will move to Pos_2 with rate 2, while in Pos_2 it will move to Pos_1 with rate 5. Moreover no matter how the node at location l moves, we guarantee that $\rho_{m\longmapsto l}+\rho_{n\longmapsto l} = 1$ as long as $\rho_{m\longmapsto n} = 1$ and $\rho_{n\longmapsto m} = 1$. Since $\rho_{m\longmapsto l}$ and $\rho_{n\longmapsto l}$ may both change when l moves, their mobility rules should be defined together in our SMF. Suppose that $\rho_{m\longmapsto l} = 0.8$ and $\rho_{n\longmapsto l} = 0.2$ when the node at location l moves to Pos_1, and $\rho_{m\longmapsto l} = 0.3$ and $\rho_{n\longmapsto l} = 0.7$ when it is at Pos_2. By letting $mf(C_1, C_2, \phi) = 2$ and $mf(C_2, C_1, \phi) = 5$ we complete the definition of the mobility rules with $C_1 = \{\{(0.8, m), (0.2, n)\} \longmapsto l\}$, $C_2 = \{\{(0.3, m), (0.7, n)\} \longmapsto l\}$, and $\phi = (\rho_{m\longmapsto n} = 1 \wedge \rho_{n\longmapsto m} = 1)$. Note that more complicated rules can be defined, for instance when the condition ϕ does not hold i.e. m and n are not close enough, we can let the $\rho_{m\longmapsto l}$ and $\rho_{n\longmapsto l}$ evolve into other values such that $\rho_{m\longmapsto l} + \rho_{n\longmapsto l} \neq 1$. For simplicity we will omit the details.

It is not hard to see that in this example we use group broadcast often between nodes internally in the network, as a result we can abstract from the concrete execution of the model. Suppose we only care whether each node in a network has a leader or not, then the model can be simplified as Model 2 where the node which initializes the election always chooses itself as the new leader.

In Model 2, the acknowledgement messages $\langle A\_i \triangleright I\rangle$ can be abstracted totally, and we can establish that: $\|_{i\in I} \lfloor Node'(i)\rfloor_i \approx_{mf} \|_{i\in I} \lfloor Node(i, l, m, p)\rfloor_i$. Intuitively, this equivalence holds because all the group broadcasts will become internal. In Model 2 the group broadcasts dealing with acknowledgements used to find the node with the highest ID are abstracted away, since we do not care about the specific ID of the leader. Essentially in Model 2 the node which initializes the election simply commutes between two states depending on whether it has a valid leader or not, while the nodes participating in an election simply commutes between three states depending on whether they have a valid leader, are part of an election waiting for acknowledgements from children, or are part of an election waiting for the announcement of the leader.

6 Conclusion

In this paper we have introduced a novel continuous time stochastic broadcast calculus for mobile and wireless broadcasting networks, which is able to model stochastic phenomena in mobile networks, like e.g. random back off protocols. We also allow for simultaneous mobility of several nodes due to a stochastic mobility model, and the mobility of nodes may be limited due to network constraints. Also, in order to minimize the state space of our models we have introduced an operator to avoid flooding in networks, and we allow for group broadcast, these two operators facilitate a novel notion of abstraction of broadcast messages where several broadcast messages may be simulated by just one broadcast message or simply be abstracted and become an internal message. A weak bisimulation congruence $\approx_{mf}$ is defined and applied on the example of a leader election protocol for wireless networks.

Model 2 An simplified model of the leader election protocol

$$
\begin{array}{ll}
Node'(i) & = \lambda_{init} \cdot \langle E\_i \triangleright I \rangle \cdot Init'(i) + (E\_x) \cdot \langle E\_i \triangleright I \rangle \cdot waitAck'(i) \\
Init'(i) & = \lambda_{exp} \cdot \langle L\_i \triangleright I \rangle \cdot Node'(i) \\
waitAck'(i) & = \lambda_{exp} \cdot waitLeader'(i) \\
waitLeader'(i) & = (L\_x) \cdot Node'(i) + \lambda_{par} \cdot \langle L\_i \triangleright I \rangle \cdot Node'(i)
\end{array}
$$

References

1. Sebastian Nanz and Chris Hankin. A framework for security analysis of mobile wireless networks. *Theor. Comput. Sci.*, 367:203–227, November 2006.
2. Anu Singh, C. R. Ramakrishnan, and Scott A. Smolka. A process calculus for mobile ad hoc networks. In *Proceedings of the 10th international conference on Coordination models and languages*, COORDINATION'08, pages 296–314. Springer-Verlag, 2008.
3. Massimo Merro. An observational theory for mobile ad hoc networks. *Electron. Notes Theor. Comput. Sci.*, 173:275–293, April 2007.
4. Fatemeh Ghassemi, Wan Fokkink, and Ali Movaghar. Restricted broadcast process theory. In *Proceedings of the 2008 Sixth IEEE International Conference on Software Engineering and Formal Methods*, pages 345–354. IEEE Computer Society, 2008.
5. Jens Chr. Godskesen. A calculus for mobile ad hoc networks. In *Proceedings of the 9th international conference on Coordination models and languages*, COORDINATION'07, pages 132–150. Springer-Verlag, 2007.
6. Lei Song and Jens Chr. Godskesen. Probabilistic mobility models for mobile and wireless networks. In Cristian Calude and Vladimiro Sassone, editors, *Theoretical Computer Science*, volume 323 of *IFIP Advances in Information and Communication Technology*, pages 86–100. Springer Boston, 2010. 10.1007/978-3-642-15240-5_7.
7. Roberto Segala and Nancy Lynch. Probabilistic simulations for probabilistic processes. *Nordic J. of Computing*, 2:250–273, June 1995.
8. Holger Hermanns. *Interactive Markov chains: and the quest for quantified quality*. Springer-Verlag, Berlin, Heidelberg, 2002.
9. Christian Eisentraut, Holger Hermanns, and Lijun Zhang. On probabilistic automata in continuous time. In *Proceedings of the 2010 25th Annual IEEE Symposium on Logic in Computer Science*, LICS '10, pages 342–351. IEEE Computer Society, 2010.
10. Fatemeh Ghassemi, Mahmoud Talebi, Ali Movaghar, and Wan Fokkink. Stochastic restricted broadcast process theory. In *Proceedings of the 8th European Performance Engineering Workshop*, EPEW '11, pages 72–86, 2011.
11. Jane Hillston. *A compositional approach to performance modelling*. Cambridge University Press, New York, NY, USA, 1996.
12. Sudarshan Vasudevan, Jim Kurose, and Don Towsley. Design and analysis of a leader election algorithm for mobile ad hoc networks. In *Proceedings of the 12th IEEE International Conference on Network Protocols*, pages 350–360. IEEE Computer Society, 2004.
13. Xavier Nicollin and Joseph Sifakis. An overview and synthesis on timed process algebras. In *Proceedings of the 3rd International Workshop on Computer Aided Verification*, CAV '91, pages 376–398. Springer-Verlag, 1992.
14. Wang Yi. CCS + Time = An Interleaving Model for Real Time Systems. In *Proceedings of the 18th International Colloquium on Automata, Languages and Programming*, pages 217–228. Springer-Verlag, 1991.