Verification of Unreliable Ad Hoc Networks

Giorgio Delzanno[1], **Arnaud Sangnier**[2], **Gianluigi Zavattaro**[3]

[1]University of Genova

[2]Liafa-University Paris 7

[3]University of Bologna

ABSTRACT. We investigate parametric verification of safety properties for a formal model of ad hoc network in presence of node and communication failures. We start by considering three possible node failures: intermittence, restart, and crash. Then we move to three cases of communication failures: nondeterministic message loss, message loss due to conflicting emissions, and detectable conflicts. Interestingly, we prove that the considered decision problem is decidable for node intermittence and message loss (either nondeterministic or due to conflicts) while it turns out to be undecidable for node restart/crash, and conflict detection.

1 Introduction

Selective broadcast communication is often used in networks in which individual nodes have no precise information about the underlying connection topology (e.g. ad hoc wireless networks). As shown in [12, 9, 10, 16, 17, 3], this type of communication can naturally be specified in models in which a network configuration is represented as a graph and in which individual nodes run an instance of a given protocol specification. A protocol typically specifies a sequence of control states in which a node can either send a message (emitter role), wait for a message (receiver role), or perform an update of its internal state. Selective broadcast communication can be represented here as a simultaneous update of the state of the emitter node and of the states of its neighbors.

Already at this level of abstraction, verification of ad hoc networks protocols turns out to be a very difficult task. A formal account of this problem is given in [2, 3], where the *control state reachability problem* is proved to be undecidable for selective broadcast communication. The control state reachability problem consists in searching for an initial network configuration (with unknown size and topology) that may evolve into a configuration in which a node (or a subset of nodes) is in a given control state (representing a protocol error). This problem seems to naturally express verification tasks in a setting in which nodes have no information a priori about the size and connection topology of the underlying network. The analysis in [2, 3] works under the assumption that the underlying network and communication model are both reliable. This is a quite strong assumption since ad hoc networks have several sources of unreliability: from node failures to conflicts and interferences among different transmissions.

In this paper we study the impact of node and communication failures on the verification task, formally specified as control state reachability as in [2, 3], for ad hoc network protocols. We start our analysis by introducing node failure in a model of selective broadcast. For this purpose, we consider an intermittent semantics in which a node can be turned on/off at any time. As a first result, we show that control state reachability becomes decidable under the intermittent semantics. Decidability seems strictly related to the assumption that nodes cannot directly control (take decision that depend

on) intermittence. We then consider two restricted types of node failure, i.e., node crash (a node can only be turned off) and node restart (when it turns on, it restarts executing in a special restart state). In both cases we give nodes a way to control intermittence. The verification task becomes indeed undecidable in both cases.

We move then to consider different types of communication failure. We first consider a semantics in which a broadcast is not guaranteed to reach all neighbors of the emitter nodes (message loss). Control state reachability is again decidable in this case. We then introduce a semantics for selective broadcast specifically designed to capture possible conflicts during a transmission. Basically, a transmission of a broadcast message is split into two different phases: a starting and an ending phase. During the starting phase, receivers connected to the emitter move to a transient state. While being in the transient state, a reception from another node generates a conflict. In the ending phase an emitter always move to the next state whereas connected receivers move to their next state only when no conflicts have been detected. Time-out can be modeled here by allowing receivers to abandon a transmission at any time. In our model we also allow several emitters to simultaneously start a transmission. Once again, decidability of control state reachability depends on whether or not a receiver controls the detection of conflicts during a transmission. More precisely, decidability holds only when receivers ignore corrupted messages by remaining in their original state.

The positive results for the verification task in the different variants of the semantics are obtained by means of a polynomial time abstract reachability algorithm that explores sets of states that can be generated by saturating the application of enabled rules. The algorithm is based on the property that in models with uncontrollable failures we can decide control state reachability by focusing on a special class of topologies built on top of clique graphs.

Related Work. Perfect synchronous semantics for broadcast communication have been proposed, e.g., in [13, 16, 17, 4]. Semantics that take into consideration interferences and conflicts during a transmission have been proposed, e.g., in [7, 9, 10, 11]. In all these approaches a broadcast communication is split into several phases to model scenarios in which different transmission periods of different emitters overlaps. In this paper we also consider a semantics for explicitly representing conflicts. Differently from other models, our semantics allows multiple nodes to start a communication in the same instant.

In [2, 3] we have studied decision problems for verification of models of ad hoc networks with selective broadcast communication with perfect semantics and no conflicts. In this paper we lift our studies to unreliable networks and communication models and consider semantics for broadcast communication with conflicts. Parameterized verification problems like control state reachability have not been studied in related work on formal analysis of ad hoc networks we are aware of [13, 16, 17, 12, 4, 6, 7, 9, 10, 11].

We would also like to mention that decidability issues for broadcast communication in unstructured concurrent systems have been studied, e.g., in [5], whereas verification of unreliable communicating FIFO systems (with message loss) have been studied, e.g., in [1].

2 Ad Hoc Network Protocols

DEFINITION 1. *A Q-graph is a labeled undirected graph $\gamma = \langle V, E, L\rangle$, where V is a finite set of* nodes*, $E \subseteq V \times V$ is a finite set of* edges*, and L is a labeling function from V to a finite set of labels Q.*

With abuse of notation, we use $L(\gamma)$ to represent all the labels present in L. The nodes belonging to an edge are called the *endpoints* of the edge. For an edge $\langle u, v\rangle$ in E, we often use the notation $u \sim_\gamma v$ and say that the vertices u and v are adjacent to one another in the labeled undirected graph γ. We omit γ, and simply write $u \sim v$, when it is made clear by the context.

In our model of ad hoc networks a configuration is a Q-graph and we assume that each node of the graph is a process that runs a common predefined protocol defined by a communicating automaton with a finite set Q of control states. Communication is achieved via selective broadcast. The effect of a broadcast is in fact local to the vicinity of the sender. The initial configuration is any graph in which all the nodes are in an initial control state. Even if Q is finite, there are infinitely many possible initial configurations. We next formalize the above intuition.

DEFINITION 2. *A process is a tuple $\mathcal{P} = \langle Q, \Sigma, R, Q_0\rangle$, where Q is a finite set of control states, Σ is a finite alphabet, $R \subseteq Q \times (\{\tau\} \cup \{!!a, ??a \mid a \in \Sigma\}) \times Q$ is the transition relation, and $Q_0 \subseteq Q$ is a set of initial control states.*

The label τ represents the capability of performing an internal action, and the label $!!a$ ($??a$) represents the capability of broadcasting (receiving) a message $a \in \Sigma$. For $u \in V$, we define the set $R_a(u) = \{q \in Q \mid \langle L(u), ??a, q\rangle \in R\}$ that contains states that can be reached from the state $L(u)$ upon reception of message a. The network semantics is define then as follows.

DEFINITION 3. *An AHN associated to process $\mathcal{P}$ is defined by the transition system $AHN(\mathcal{P}) = \langle \mathcal{C}, \Rightarrow, \mathcal{C}_0\rangle$, where $\mathcal{C}$ is the set of Q-graphs (network configurations), $\mathcal{C}_0$ is the set of Q_0-graphs (initial configurations), and $\Rightarrow \subseteq \mathcal{C} \times \mathcal{C}$ is the transition relation defined next. For $\gamma = \langle V, E, L\rangle$ and $\gamma' = \langle V, E, L'\rangle$, $\gamma \Rightarrow \gamma'$ holds iff one of the following conditions holds:*
Local: *$\exists v \in V$ s.t. $(L(v), \tau, L'(v)) \in R$, and $L(u) = L'(u)$ for all u in $V \setminus \{v\}$;*
Broadcast: *$\exists v \in V$ s.t. $(L(v), !!a, L'(v)) \in R$ and for every $u \in V \setminus \{v\}$*

- *if $\langle v, u\rangle \in E$ and $R_a(u) \neq \emptyset$ (reception of a in u is enabled), then $L'(u) \in R_a(u)$.*
- *$L(u) = L'(u)$, otherwise.*

DEFINITION 4. *An execution is a sequence $\gamma_0\gamma_1\ldots$ such that γ_0 is an initial configuration, and $\gamma_i \Rightarrow \gamma_{i+1}$ for $i \geq 0$. We use $\Rightarrow^*$ to denote the reflexive and transitive closure of $\Rightarrow$.*

Observe that a broadcast message a sent by v is delivered only to the subset of neighbors interested in it. Such a neighbor u updates its state with a new state taken from $R_a(u)$. All the other nodes (including neighbors not interested in a) simply ignore the message. Also notice that the topology is static, i.e., the set of nodes and edges remain unchanged during a run.

An example of process definition and network semantics is given in appendix.

2.1 Safety Analysis: the Control State Reachability Problem

Following [2, 3] we consider decision problems related to verification of safety properties. We remark that in our formulation the size and topology of the initial configurations is not fixed a priori. The problem that we consider is *control state reachability* (COVER) defined as follows:

DEFINITION 5. *The* COVER *decision problem is defined as follows.*
Input: a process $\mathcal{P}$ with transition system $AHN(\mathcal{P}) = \langle \mathcal{C}, \Rightarrow, \mathcal{C}_0 \rangle$ and a control state q.
Output: yes, if there exists $\gamma \in \mathcal{C}_0$ and $\gamma' \in \mathcal{C}$ such that $\gamma \Rightarrow^* \gamma'$ and $q \in L(\gamma')$; no, otherwise.

Assume that state q represents an error state for a node of the network. COVER amounts at checking whether there exists an initial configuration (among the infinitely many possible ones) from which a configuration containing a node in the error state is reachable. The following property is proved in [2].

THEOREM 6. COVER *is undecidable for ad hoc network protocols.*

3 Node Failure: Intermittency, Crash, and Restart

In this section we introduce several notions of node failures and study decidability and complexity issues for the COVER problem.

3.1 Intermittent Nodes

We start our analysis from a semantic variant of AHN that models intermittent nodes. We modify the network semantics by using flag 1 to denote an active node, and 0 for a deactivated one.

DEFINITION 7. *Given a process definition $\mathcal{P} = \langle Q, \Sigma, R, Q_0 \rangle$, an i-configuration is a connected graph $\gamma = \langle V, E, L_i \rangle$ in which $L_i : V \to Q \times \{0,1\}$. We use $\mathcal{C}_i$ to denote the set of i-configurations associated to a process definition $\mathcal{P}$.*

The intermittent semantics of an ad hoc network with process $\mathcal{P} = \langle Q, \Sigma, R, Q_0 \rangle$ is given by its associated transition system $AHN_i(\mathcal{P}) = \langle \mathcal{C}_i, \dashrightarrow, \mathcal{C}_{i0} \rangle$ where $\mathcal{C}_{i0} = \{\gamma \in \mathcal{C}_i \mid L_i(\gamma) \subseteq Q_0 \times \{0,1\}\}$ is the set of initial configurations. The transition relation $\dashrightarrow \subseteq \mathcal{C}_i \times \mathcal{C}_i$ is defined as follows.

DEFINITION 8. *Given a configuration $\gamma = \langle V, E, L_i \rangle$, $\gamma \dashrightarrow \gamma'$ iff $\gamma' = \langle V, E, L_i' \rangle$ and one of the following conditions holds:*

Local: *$\exists v \in V$ s.t. $L_i(v) = \langle q, 1 \rangle$, $L_i'(v) = \langle q', 1 \rangle$, $(q, \tau, q') \in R$, and $L_i(u) = L_i'(u)$, for all u in $V \setminus \{v\}$;*

Broadcast: *$\exists v \in V$ s.t. $L_i(v) = \langle q, 1 \rangle$, $(q, !!a, q') \in R$, $L_i'(v) = \langle q', 1 \rangle$, and for every u in $V \setminus \{v\}$:*
- *if $\langle v, u \rangle \in E$, $L_i(v) = \langle u, 1 \rangle$, and $R_a(u) \neq \emptyset$, then $L_i'(v) = \langle u', 1 \rangle$ with $u' \in R_a(u)$.*
- *$L(u) = L'(u)$, otherwise.*

Intermittence: *$\exists v \in V$ s.t. $L_i(v) = \langle q, b \rangle$, $L_i'(v) = \langle q, 1-b \rangle$, and $L_i(u) = L_i'(u)$ for all u in $V \setminus \{v\}$.*

Notice that the transition relation is defined as in the previous section with only two differences: the transition already present in the previous definition now applies only to active nodes (i.e. those with associated value 1); additional transitions allows one node to move from the active to the passive state, and vice versa. We do not make any assumption about when such transition could occur, thus we apply them to every state of the node that, in other words, can always nondeterministically perform one of these transitions.

We denote by $\dashrightarrow^*$ the reflexive and transitive closure of $\dashrightarrow$. An example is shown in appendix.

$\mathbf{R} := Q_0$, $\mathbf{N} := \emptyset$;
repeat
$\mathbf{R} := \mathbf{R} \cup \mathbf{N}$; $\mathbf{N} := \{q_2 | \langle q_1, \tau, q_2 \rangle \in R,\ q_1 \in \mathbf{R}\}$;
For every rule $\langle q_1, !!a, q_2 \rangle \in R$ s.t. $q_1 \in \mathbf{R}$,
$\mathbf{N} := \mathbf{N} \cup \{q_2\} \cup \{t \mid \langle s, ??a, t \rangle \in R,\ s \in \mathbf{R}\}$;
until $\mathbf{N} \subseteq \mathbf{R}$

Figure 1: SFR: Symbolic forward reachability for solving COVER with intermittent nodes.

In the setting of intermittence we need to slightly modify the definition of COVER as labels now also contain the values 0 and 1.

DEFINITION 9. *The* COVER *decision problem for intermittent nodes is defined as follows.*
Input: a process $\mathcal{P}$ with transition system $AHN_i(\mathcal{P}) = \langle \mathcal{C}_i, \dashrightarrow, \mathcal{C}_{i0} \rangle$ and a control state q.
Output: yes, if there exists $\gamma \in \mathcal{C}_{i0}$ and $\gamma' \in \mathcal{C}_i$ such that $\gamma \dashrightarrow^* \gamma'$ and $\langle q, b \rangle \in L(\gamma')$ for some $b \in \{0,1\}$; no, otherwise.

We now move to the proof of the first result of our paper, namely the decidability of COVER for ad hoc network protocols with intermittent nodes. Let $\mathcal{P} = \langle Q, \Sigma, R, Q_0 \rangle$ and let $q \in Q$. The algorithm is based on the construction of a symbolic execution path that contains all states that can be generated by applying rules in R. The construction is based on the following key points. We first observe that, in order to decide if control state q can be reached, we can focus our attention on initial graphs with clique topology of arbitrary size (i.e. graphs in which all pairs of nodes are connected). Indeed, non-deterministic intermittence allows us to reconfigure a clique (by turning off a subset of its nodes). Another key observation is that by adding nodes to an initial clique, we can always ensure that if a clique K can reach K', then there exists a larger clique C that contains K and that can reach a clique C' that contains all the cliques generated along the path from K to K'. This property is based on the possibility of using intermittent steps to deactivate and reactivate a subset of nodes of a clique in order to simulate the same steps executed in the derivation from K to K'. This property of COVER avoids the need of counting the number of occurrences of states of a certain type. We just have to remenber which states can be generated by repeatedly applying process rules.

By exploiting the above mentioned observations, when defining the decision procedure for checking control state reachability we can take the following assumptions: (i) forget about the topology underlying the initial configuration; (ii) forget about the number of occurrences of control states in a configuration (if it is reached once, it can be reached an arbitrary number of times by considering larger initial configurations as explained before); (iii) consider a single symbolic path in which at each step we apply all possible rules whose preconditions can be satisfied in the current set and then collect the resulting set of computed states.

More formally, the decision procedure is defined then by the algorithm SFR working on sets of states defined in Fig. 1. Since Q is finite, the algorithm SFR always terminates in at most $|Q|$ iterations (those needed to saturate set $\mathbf{R}$ of visited states). The following theorem then holds.

LEMMA 10. *Given $\mathcal{P} = \langle Q, \Sigma, R, Q_0 \rangle$ with $AHN_i = \langle \mathcal{C}_i, \dashrightarrow, \mathcal{C}_{i0} \rangle$, and $q \in Q$, let $\boldsymbol{R}_f$ denote the result of the algorithm SFR when applied to process $\mathcal{P}$. We have that there exists $\gamma \in \mathcal{C}_0$ and $\gamma' = \langle V, E, L \rangle$ in $\mathcal{C}$ s.t. $\gamma \dashrightarrow^* \gamma'$ and $\langle q, 1 \rangle \in L(\gamma')$ iff $q \in \boldsymbol{R}_f$.*

PROOF. We first show that if there exists $\gamma \in \mathcal{C}_0$ and γ' in $\mathcal{C}$ s.t. $\gamma \dashrightarrow^* \gamma'$, then there exists a

clique graph K that contains γ as a subgraph from which we can reach a graph K' that contains γ' as a subgraph. We build K by adding all necessary missing edges to γ. We then show by induction that we can mimick the derivation from γ to γ'. The interesting case is the analysis of a single step from γ_1 reachable from γ to its successor γ_2. We assume that there exists K_1 that contains γ_1 as a subgraph reachable from clique K and consider two cases. If the transition from γ_1 to γ_2 is an internal step, then we immediately obtain the same effect on K_1 and reach clique K_2 that contains γ_2 as a subgraph. If the transition is a broadcast of a msg m with sender node n and active receivers $n_1, \ldots, n_k$, we can first deactivate (using a sequence of intermittence steps) all active nodes $m_1, \ldots, m_p$ connected to n in K_1 that are distinct from $n_1, \ldots, n_k$. We then apply the rule, and, by another sequence of intermittent steps, reactivate $m_1, \ldots, m_p$. Clearly, the resulting clique K_2 contains γ_2 as a subgraph. This property show that with intermittent nodes it is sufficient and necessary to consider COVER over clique graphs, only.

We now prove that if K' is reachable from K, then there exists a clique C that contains K and a clique C' that is reachable from C and that contains all cliques generated during the sequence from K to K' (and thus the union of their labels). We call this property clique-monotonicity. The proof is by induction on the length of a derivation. When focusing on a single step, C is obtained, e.g., by taking two copies of K, deactivating all nodes of one of the copies, applying the transition from K to K' to the other copy, and then reactivating the nodes in the first copy. The same idea can be iterated along the derivation.

We are ready now to prove the correcteness of SFR. Assume that γ' is reachable from γ and q occurs in γ'. Then, there exists a derivation from a clique C to a clique C' that satisfies the clique-monotonicity property and such that q occurs in C'. Since the set of labels in a sequence of cliques that satisfies monotonicity can only augment, the corresponding set of labels must be contained in the set $\mathbf{R}_f$ computed by the algorithm SFR which applies all possible rules at each iteration, and thus $q \in \mathbf{R}_f$.

Now, assume that $q \in \mathbf{R}_f$, then there exists a sequence of sets of states $R_0 R_1 R_n$ computed by the algorithm SFR such that $R_0 = Q_0$ and $q \in R_n = \mathbf{R}_f$. We now observe that if from R_0 we can compute R_1 by applying a subset S of rules, then we can build an initial clique C from which we can derive a clique C' containing the same states as R_1 and obtained with a sequence of applications of the same set of rules S. Again we use node intermittence to activate and deactivate nodes in order to apply the rules in S starting from a large enough clique C. We can then extend this argument by reasoning by induction on n and prove that there exists a clique C' that contains state q that is reachable from an initial clique C. ∎

We conclude this section by studying the complexity of the problem.

THEOREM 11. COVER *with intermittent nodes is* PTIME*-complete.*

PROOF. The algorithm SFR is in PTIME since it requires at most $|Q|$ iterations each one requiring at most $|R|^2$ look-ups (receive rules) for computing new states to be included in $\mathbf{N}$.
We now show that COVER is PTIME-hard by exhibiting a logspace reduction from the Circuit Value Problem (CVP). CVP is defined as follows: given an acyclic Boolean circuit with k inputs and m Boolean gates (and, or, not) and a single ouput z, and given a truth assignment to the inputs, is the value of the output equal to true? This problem is known to be PTIME-complete [8].

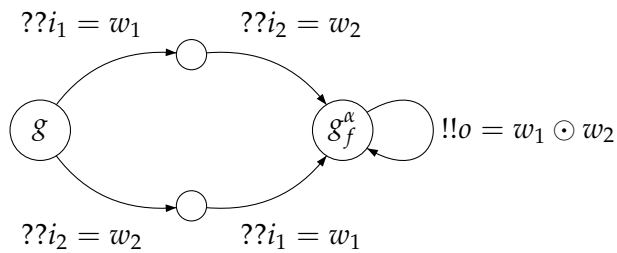
Assume a CVP C with input/ouput names taken from a set VN. Gate g is represented by its signature $g(\odot, i_1, i_2, o)$ for $i_1, i_2, o \in NV$ and $\odot \in \{\vee, \wedge\}$ or by $g(\neg, i, o)$ for $i, o \in VN$.

The encoding is based on two types of node. The process $\mathcal{P}_C$ associated to C has initial states: q_0 (init nodes), and g (gate node) for each gate g in C.

A node in state q_0 broadcasts (an arbitrary number of) messages that model the initial assignments to input variables. Since the assignment is fixed, broadcasting these messages several times (or receiving them from different initial nodes) does not harm the correctness of the encoding. When receiving an evaluation for their inputs (from an initial node or another gate node), a gate node evaluates the corresponding Boolean function and then repeatedly broadcast the value of the corresponding output. Since C is acyclic, once computed, the output value remains always the same (i.e. recomputing it does not harm). Finally, reception of a value v for output z sends an q_0 node into state ok. Reachability of a output value v reduces then to COVER of state ok.

Formally, the process rules are defined as follows. For $i : 1, \ldots, n$, we have rules $\langle q_0, !!x_i = v_i, q_0\rangle$ and $\langle q_0, ??z = v, ok\rangle$. They model the assignment of value v_i to input x_i and reception of output value v. Notice that the input value is fixed a priori.

For gate $g(\odot, i_1, i_2, o)$ and for each assignment $\alpha = \langle w_1, w_2\rangle$ of values to $\langle i_1, i_2\rangle$ (a constant number for each gate), we associate the following subprotocol:



We use a similar encoding for a *not* gate.

Consider now the resulting process definition $\mathcal{P}_C = \langle Q, \Sigma, R, \{init\}\rangle$ with corresponding transition system $AHN_i = \langle \mathcal{C}_i, \dashrightarrow, \mathcal{C}_{i0}\rangle$. We have that there exists $\gamma \in \mathcal{C}_0$ and $\gamma' = \langle V, E, L\rangle$ in $\mathcal{C}$ s.t. $\gamma \dashrightarrow^* \gamma'$ and $\langle ok, b\rangle \in L(\gamma')$ iff v is the value for z in C.

The above described reduction requires logarithmic space in the size of C (we need variables i, j, k to represent input and output variables of a gate, each one requiring $log|C|$ bits; for each gate we produce a constant number of rules). Together with the PTIME algorithm in Fig. 1, the above reduction show PTIME-completeness of COVER. ∎

3.2 Node Crash and Restart

We now consider two variants of the semantics with intermittence. In the first one, modeling node crash, nodes can only be turned off. In the second one, modeling node restart, nodes can also be turned on and they restart from a given state.

Given a process definition $\mathcal{P}$, its transition system with node crash denoted with $AHN_{cr}(\mathcal{P})$, is defined as the transition system $AHN_i(\mathcal{P})$ where the **Intermittence** transitions are replaced by the following **Crash** transitions:

Crash: $\exists v \in V$ s.t. $L_i(v) = \langle q, 1\rangle$, $L'_i(v) = \langle q, 0\rangle$, and $L_i(u) = L'_i(u)\ \forall u \in V \setminus \{v\}$.

Notice that there are only transitions turning off nodes.

The variant with restart requires the indication of the restart state. So a process definition $\mathcal{P} = \langle Q, \Sigma, R, Q_0, q_r \rangle$ now includes also $q_r \in Q$ and we assume that $Q_0 \neq \{q_r\}$. The transition system with node restart for $\mathcal{P}$, denoted with $AHN_r(\mathcal{P})$, is defined as the transition system $AHN_i(\langle Q, \Sigma, R, Q_0 \rangle)$ where the **Intermittence** transitions are replaced by the following **Restart** transitions:

Restart: $\exists v \in V$ s.t. $L_i(v) = \langle q, 1 \rangle$ and $L'_i(v) = \langle q, 0 \rangle$, or $L_i(v) = \langle q, 0 \rangle$ and $L'_i(v) = \langle q_r, 1 \rangle$, and $L_i(u) = L'_i(u)\ \forall u \in V \setminus \{v\}$.

Notice that in this case, besides the transitions turning off nodes, there are also transitions that turn on one node by changing its internal state to the restart state q_r.

The following theorem then holds.

THEOREM 12. COVER *is undecidable for ad hoc network protocols with node crash or restart.*

PROOF. The proof is by reduction from the undecidability of COVER for ad hoc network protocols (Theorem 6). Consider a process definition $\mathcal{P}$. It is trivial to see that a computation leading to a configuration that exposes the control state q in $AHN(\mathcal{P})$ has a corresponding computation in $AHN_{cr}(\mathcal{P})$ (in which no **Crash** transition is performed).

Consider now a computation in $AHN_{cr}(\mathcal{P})$ leading to a configuration that exposes the control state q. It is not restrictive to assume that the state q is exposed by a node that did not crash during the computation. Consider now a computation in $AHN(\mathcal{P})$ that performs the same **Local** and **Broadcast** transitions (but not the **Crash** transitions). It is easy to see that the nodes that did not crash during the computation in $AHN_{cr}(\mathcal{P})$ are in the same state also in the computation in $AHN(\mathcal{P})$. Hence also the latter computation leads to a configuration exposing the control state q.

The undecidability results can be easily lifted to the model with node restart. It is possible to translate a protocol with node crash into an equivalent protocol with node restart: it is sufficient to assume that the restart state q_r is deadlocked, i.e. it has no outgoing transition. ∎

We let as an open problem the decidability status of COVER with node restart when $Q_0 = \{q_r\}$ but we conjecture it also should be undecidable.

4 Communication Failures: Loss and Conflict

We now move to communication failures.

4.1 Message Loss

The first type of failures corresponds to nondeterministic message loss: when a message is broadcast, some of the receivers could not receive it.

Protocol definitions $\mathcal{P}$ is as usual. The transition system denoted with $AHN_l(\mathcal{P})$ is defined as $AHN(\mathcal{P})$ where the **Broadcast** transitions are replaced by the following **Message loss** transitions.

Message loss: $\exists v \in V$ s.t. $(L(v), !!a, L'(v)) \in R$ and for every $u \in V \setminus \{v\}$

- if $\langle v, u \rangle \in E$ and $R_a(u) \neq \emptyset$ (reception of a in u is enabled), then $L'(u) \in R_a(u)$ or $L'(u) = L(u)$.
- $L(u) = L'(u)$, otherwise.

Notice that upon the execution of a broadcast, some of the potential receivers could remain in their internal state. We now prove, by relating message loss with node intermittence, that COVER is decidable also for ad hoc network protocols with message loss. The proof is in appendix.

THEOREM 13. COVER *is in* PTIME*-complete for ad hoc network protocols with message loss.*

4.2 Conflict

The second type of failure we consider corresponds to transmission conflicts. Here we consider conflicts due to the contemporaneous emission of messages: if a node has (at least two) neighbors that contemporaneously emits signal, then such a node is unable to correctly receive the emitted messages. The modeling of this phenomenon requires a significant modification of the formal semantics. First of all we need to introduce the notion of individual state.

Individual State. The internal state of a node is characterized by the current state according to the protocol behavior, and by two additional decorations indicating whether the node is currently emitting or receiving a message. Formally, we define:

$$\mathcal{S} = \{[q, x, y] \mid q \in Q, x \in \{\bot\} \cup \Sigma, y \in \{\bot, \texttt{rcv}, \texttt{cnfl}\}\}$$

The field denoted with x represents whether the node is in transmission state ($\bot$ means no transmission, while $a \in \Sigma$ denotes transmission of message a). The field y represents whether the node is not receiving ($\bot$), it is currently receiving correctly a message (`rcv`), or the reception has been damaged due to a conflict (`cnfl`). The initial states are defined as follows:

$$\mathcal{S}_0 = \{[q, \bot, \bot] \mid q \in Q_0\}$$

Notice that nodes in their initial state are neither receiving nor emitting.

Also the network semantics require significant modifications.

Network Semantics. The semantics of a protocol $\mathcal{P} = \langle Q, \Sigma, R, Q_0\rangle$ is given by its associated transition system $AHN_{co}(\mathcal{P}) = \langle \mathcal{C}, \rightrightarrows, \mathcal{C}_0\rangle$ where $\mathcal{C}$ is the set of undirect graphs labeled on $\mathcal{S}$, where labels represent the individual state of the network nodes. The set of initial configurations $\mathcal{C}_0$ is defined as the set of graphs labeled on $\mathcal{S}_0$.

In the formal definition of the transition relation $\rightrightarrows \subseteq \mathcal{C} \times \mathcal{C}$ we make use of the function $emitter(\gamma, m)$, that returns the set of neighbors of m that are in transmission state in the configuration γ. Formally, given $\gamma = \langle V, E, L\rangle$ then

$$emitter(\gamma, m) = \{n \mid m \sim n, \quad L(n) = [q, a, y] \text{ for some } q \in Q, a \in \Sigma, y \in \{\bot, \texttt{rcv}, \texttt{cnfl}\}\}$$

Given a configuration $\gamma = \langle V, E, L\rangle$, $\gamma \rightrightarrows \gamma'$ iff $\gamma' = \langle V, E, L'\rangle$ and one of the following conditions holds:

Local/Time-out: $\exists n \in V$ s.t. $L(n) = [q, \bot, y]$, $y \in \{\bot, \texttt{cnfl}, \texttt{rcv}\}$, $(q, \tau, q') \in R$, $L'(n) = [q', \bot, \bot]$, and $L(m) = L'(m)$ for any other node m in V.

Start broadcast: $\exists n_1, \ldots, n_l \in V$ s.t. $\cup_{j\in\{1\ldots l\}} emitter(\gamma, n_j) = \emptyset$, $L(n_i) = [q_i, \bot, \bot]$, $(q_i, !!a_i, q_i') \in R$, $L'(n_i) = [q_i', a_i, \bot]$ $\forall i \in \{1 \ldots l\}$ and the following conditions hold:

- $\forall m \in V \setminus \{n_1, \ldots, n_l\}$ s.t. $m \sim n_i$, for some $i \in \{1 \ldots l\}$, and $L(m) = [r, \bot, y]$ with $y \in \{\texttt{rcv}, \bot\}$ then
  - if $y = \texttt{rcv}$ then $L'(m) = [r, \bot, \texttt{cnfl}]$;
  - if $y = \bot$ and $m \not\sim n_j \; \forall j \in \{1 \ldots l\} \setminus \{i\}$ then $L'(m) = [r, \bot, \texttt{rcv}]$;
  - if $y = \bot$ and $m \sim n_j$ for some $j \in \{1 \ldots l\} \setminus \{i\}$ then $L'(m) = [r, \bot, \texttt{cnfl}]$;
- $L(m) = L'(m)$ otherwise.

End broadcast: $\exists\, n \in V$ s.t. $L(n) = [q, a, \bot]$, $L'(n) = [q, \bot, \bot]$ and the following conditions hold:

- $\forall\, m \sim n$ s.t. $L(m) = [r, \bot, y]$, with $y \in \{\texttt{rcv}, \texttt{cnfl}\}$, and $\mathit{emitter}(\gamma, m) = \{n\}$ one of the following conditions hold:
  - if $y = \texttt{rcv}$ and $\exists\, r'$ s.t. $(r, ??a, r') \in R$ then $L'(m) = [r', \bot, \bot]$;
  - if $y = \texttt{rcv}$ and $\nexists\, r'$ s.t. $(r, ??a, r') \in R$ or $y = \texttt{cnfl}$ then $L'(m) = [r, \bot, \bot]$;
- $L(m) = L'(m)$ otherwise.

The local rule models internal and time-out steps (to abandon a transmission). In the start rule we select a set of emitters and check that no other emitters are in their vicinity. The selected emitters simultaneously start transmitting. Receiving nodes connected to a single emitter move to the `rcv` state, and to the `cnfl` state in case of connection with more than one emitter (e.g. a selected node and an emitter that started transmitting in a previous step). In the ending rule an emitter moves to its next state. A receiver connected to such a node moves to the next state only if it is still in the `rcv` state (no conflicts occurred in between the start and end phase).

As an example, consider the protocol $(S, !!m, T), (R, ??m, Q)$, and the execution in Fig. 4. In the initial configurations we have three receivers in state S (a, b, c from left to right), and three receivers in state R (d, e, f from left to right). Nodes a and b can simultaneously start transmitting m, since no other node is currently transmitting in their vicinity. Node d simultaneously moves to a conflict state (it is connected to both emitters), while node e moves to a reception state. When c starts transmitting m (again there are no other emitters in its vicinity), node e is forced to enter a conflict state, whereas node f goes to a reception state. When a stops transmitting, d goes back to the original state (a conflict occurred). If now c stops transmitting, f receives the message and moves to its next state Q (no conflicts occurred). Finally when b stops transmitting, e goes back to the original state (a conflict occurred). Other possible executions are obtained, e.g., by selecting only one of the nodes a, b for starting a transmission (the other node has to remain silent since it is connected to an active emitter) and by nondeterministically allowing receiver nodes to abandon a transmission.

We now prove that COVER is decidable in this model of conflicts.

THEOREM 14. COVER *is in* PTIME *for ad hoc network protocols with conflicts.*

PROOF. The proof is by reduction to COVER for ad hoc network protocols with node intermittence. Consider a process definition $\mathcal{P}$. It is easy to see that a computation leading to a configuration that exposes the control state q in $AHN_{co}(\mathcal{P})$ has a corresponding computation in $AHN_i(\mathcal{P})$: the **Local** transitions are faithfully reproduced, the **Start broadcast** transitions are not mimicked, and the **End broadcast** transitions are simulated via a protocol that firsts turn off the nodes that do not receive the message, then execute the broadcast, and then turn on the same nodes.

It is more complex to show that a computation in $AHN_i(\mathcal{P})$ that leads to a configuration that exposes the control state q can be reproduced in $AHN_{co}(\mathcal{P})$. We first assume, without loss of generality, that in the process definition there is at least one state with an outgoing broadcast transition

which is reachable from an initial state $q_0 \in Q_0$ doing only internal steps. If this is not the case, there is no communication in the system and the analysis of COVER can be trivially done by checking whether the *error* state q is reachable from an initial state in the automaton defining the process behavior doing only internal steps.

Consider now the computation in $AHN_i(\mathcal{P})$ that leads to a configuration that exposes the control state q. Let γ_0 be the initial configuration in the considered computation, and let $loss(n)$ be the number of messages that the node n loses during the computation as it is turned off. We now show the existence of an initial configuration in $AHN_{co}(\mathcal{P})$ able to reproduce such computation. This initial configuration contains γ_0 plus a set of additional nodes used to generate conflicts. Namely, we connect to each node n of the initial configuration $loss(n)$ additional nodes: each of these nodes is connected only with its corresponding node n. Each node n simulates the behavior of the corresponding node in the computation in $AHN_i(\mathcal{P})$. The additional nodes are initially in the state q_0. The simulation of the transitions in the computation in $AHN_i(\mathcal{P})$ is as follows. First of all, we consider local transitions for the additional nodes in state q_0 leading them in a state ready to perform a broadcast. Then the transitions are simulated as follows. **Local** transitions are faithfully reproduced. **Intermittence** transitions are not mimicked. To simulate **Broadcast** transitions performed by one node, say m, we proceed as follows: we partition the potential receivers in two groups, (i) those that actually receive the message and (ii) those that do not receive it as they are turned off. For each of the nodes corresponding to the group (ii) we take one of its connected additional nodes, and let him perform a **Start broadcast** transition. Then the node corresponding to m performs the broadcast (it executes both the **Start** and the **End broadcast** transitions). Finally, the additional nodes that performed the **Start broadcast** transitions perform the corresponding **End broadcast**. It is easy to see that the nodes in the clique corresponding to (i) receives the broadcast messages, while those corresponding to (ii) do not receives it, due to the conflict generated by the additional neighbor. ∎

5 Conflict detection

We now define a variant of the semantics in order to capture the notion of conflict detection. In fact, even if when a node receives overlapping signal emissions it is unable to reconstruct the emitted messages, it can usually infers that (at least) two neighbors have contemporaneously emitted their messages. This can be considered in our model of ad hoc networks simply by adding *conflict detection* transitions to the automata used to represent the protocols. Such transitions can be executed by nodes at the end of a receive phase during which more than one neighbor has performed a broadcast. Formally, we apply minor modifications to the definition of the *Individual Behavior* and of the *Network Semantics*.

Individual Behavior. The new definition of $\mathcal{P}$ is as usual with the unique difference that we can have transitions of the form (q, ρ, q') in R, representing conflict detection.

Network Semantics. The semantics of the transition system $AHN_{cd}(\mathcal{P})$ is defined as $AHN_{co}(\mathcal{P})$ where the **End broadcast** transitions are replaced by the following **End broadcast II** transitions.

End broadcast II: $\exists n \in V$ s.t. $L(n) = [q, a, \bot]$, $L'(n) = [q, \bot, \bot]$ and the following conditions hold:

- $\forall m \sim n$ s.t. $L(m) = [r, \bot, y]$, with $y \in \{\texttt{rcv}, \texttt{cnfl}\}$, and $emitter(\gamma, m) = \{n\}$ one of the following conditions hold:

  - if $y = \texttt{rcv}$ and $\exists r'$ s.t. $(r, ??a, r') \in R$ then $L'(m) = [r', \perp, \perp]$;
  - if $y = \texttt{cnfl}$ and $\exists r'$ s.t. $(r, \rho, r') \in R$ then $L'(m) = [r', \perp, \perp]$;
  - if $y = \texttt{rcv}$ and $\nexists r'$ s.t. $(r, ??a, r') \in R$, or $y = \texttt{cnfl}$ and $\nexists r'$ s.t. $(r, \rho, r') \in R$, then $L'(m) = [r, \perp, \perp]$;
- $L(m) = L'(m)$ otherwise.

As an example, consider the protocol $(S, !!m, T), (R, ??m, Q), (R, \rho, ERR)$, and the execution in Fig. 5. It consists of the same steps as those in Fig. 4 up to ending phases of broadcast messages. Receiver that detect a conflict move here to the special ERR states.

The addition of conflict detection transitions makes COVER undecidable.

THEOREM 15. COVER *is undecidable for ad hoc network protocols with conflict detection.*

PROOF. The proof is by reduction from the undecidability of COVER for ad hoc network protocol with node restart. Consider a process $\mathcal{P} = \langle Q, \Sigma, R, Q_0, q_r \rangle$ for ad hoc networks with node restart, with restart state q_r. Consider now the process definition $\mathcal{P}' = \langle Q \cup \{q_i\}, \Sigma, R, Q_0 \rangle$, for ad hoc networks with conflict detection, defined as $\mathcal{P}$ with the following additional transitions: from each node $q \in Q$ we consider a transition labeled with ρ leading to the additional intermediary state q_i, from which there is only one outgoing transition labeled with τ leading to the restart state q_r.

We first show that given a computation in $AHN_r(\mathcal{P})$ leading to a configuration that exposes the control state q, there exists a corresponding computation in $AHN_{cd}(\mathcal{P}')$. As in Theorem 14 we make the nonrestrictive assumption that in the process definition $\mathcal{P}$ there is at least one state with an outgoing broadcast transition which is reachable from an initial state $q_0 \in Q_0$ doing only internal steps. Let γ be the initial configuration of the considered computation in $AHN_r(\mathcal{P})$. For each node n in γ we denote with $restart(n)$ the number of restarts performed by n in the computation. We now show the existence of an initial configuration γ' that simulates in $AHN_{cd}(\mathcal{P}')$ the considered computation. The configuration γ' is as γ with the difference that each node n has exactly $restart(n) \times 2$ additional neighbor that are used to generate conflicts. These additional nodes are connected only to the corresponding node n. The simulation of the computation proceeds as follows. At the beginning the additional nodes in state q_0 perform the local transitions leading them in a state ready to perform a broadcast. Then the simulation starts. **Local** transitions are reproduced faithfully. A transition that turns off the node n is simulated via the following protocol: two of the additional nodes connected to n perform a **Start broadcast** transition and then execute the **End broadcast**. Due to the emission conflict, the node n moves to the internal state q_i. A transition that tuns on the node n is reproduced by an internal transition from the state q_i of n to the restart state q_r. Finally, **Broadcast** transitions are mimicked by performing in sequence a **Start** and an **End broadcast II** transition.

We now show that a computation in $AHN_{cd}(\mathcal{P}')$ leading to a configuration that exposes the control state q has a corresponding computation in $AHN_r(\mathcal{P})$. In the simulated computation the **Local** transitions are reproduced faithfully, the **Start broadcast** transitions are not mimicked, while **End broadcast II** transitions are simulated by the following protocol. Assume that the node that completes its signal emission in the **End broadcast II** transition is n, and let a be the emitted message. The neighbors of n able to receive a can be partitioned in three groups: (i) those that correctly receive message a, (ii) those that performs a conflict detection transition during the execution of the **End broadcast II** transition, and (iii) those that do not change its internal state because they are still under the effect of another signal emission. The simulation of the transition in $AHN_r(\mathcal{P})$ proceeds as follows. The nodes in (ii) and (iii) that are not currently crashed perform a **Crash** transition, then the

Broadcast transition is executed. Notice that at the end of this protocol the nodes in (ii) are in the intermediary state q_i in the computation in $AHN_{cd}(\mathcal{P}')$, while they are crashed in the corresponding computation in $AHN_r(\mathcal{P})$. The **Local** transitions that move the nodes form the state q_i to q_r are reproduced in $AHN_r(\mathcal{P})$ by **Restart** transitions. ∎

6 Conclusion

In this paper we have studied decidability issues for the control state reachability problem in ad hoc networks with different sources of node and communication failures. Decidability results are obtained in models with uncontrollable failures. Verification problems for models in which nodes are conscious of failures occurred in the system (e.g. detection of transmission conflicts) can be reduced to problems for perfect ad hoc networks, thus control state reachability turns out to be undecidable in those models. We have not considered node mobility: in [2] we have proved that if we assume that nodes can (nondeterministically) move then control state reachability is already decidable in models without failures. The decidability result for mobility is based on a reduction to Petri nets. The polynomial time algorithm we define for intermittence can however be applied also to node mobility. This improves the complexity bounds of the positive results in [2]. Investigating other types of failure or refined semantics for conflict detection (e.g. with duration time of transmissions) is part of our future research.

Bibliography

[1] Abdulla, P. A., Jonsson, B.: Verifying programs with unreliable channels. Inf. Comput. 127(2): 91–101 (1996)

[2] Delzanno, G., Sangnier, A., Zavattaro, G.: Parameterized verification of Ad Hoc Networks. CONCUR '10: 313–327

[3] Delzanno, G., Sangnier, A., Zavattaro, G.: On the Power of Cliques in the Parameterized verification of Ad Hoc Networks. FOSSACS '11: 441–455

[4] Ene, C., Muntean, T.: A broadcast based calculus for Communicating Systems. IPDPS '01: 149

[5] Esparza, J., Finkel, A., Mayr, R.: On the verification of Broadcast Protocols. LICS '99: 352–359

[6] Fehnker, A., van Hoesel, L., Mader, A.: Modelling and verification of the LMAC protocol for wireless sensor networks. IFM '07: 253–272

[7] Godskesen, J.C.: A calculus for Mobile Ad Hoc Networks. Coordination '07: 132–150

[8] Ladner, R. E.: The circuit value problem us logspace complete for P. SIGACT News: 18–20, 1977

[9] Merro, M.: An observational theory for Mobile Ad Hoc Networks. Inf. Comput. 207(2): 194–208 (2009)

[10] Merro, M., Ballardin, F., Sibilio, E. A Timed Calculus for Wireless Systems FSEN '09: 228-243, 2010.

[11] Lanese, I., Sangiorgi, D.: An operational semantics for a calculus for wireless systems. TCS, 411(19): 1928-1948 (2010)

[12] Nanz, S., Hankin, C.: A Framework for security analysis of mobile wireless networks. TCS, 367(1–2):203-227 (2006)

[13] Prasad, K.V.S.: A Calculus of Broadcasting Systems. SCP, 25(2–3): 285–327 (1995).

[14] Rackoff C.: The Covering and Boundedness Problems for Vector Addition Systems. TCS, 6:223-231 (1978)
[15] Saksena, M., Wibling, O., Jonsson, B.: Graph grammar modeling and verification of Ad Hoc Routing Protocols. TACAS '08: 18–32
[16] Singh, A., Ramakrishnan, C. R., Smolka, S. A.: A process calculus for Mobile Ad Hoc Networks. COORDINATION '08: 296–314
[17] Singh, A., Ramakrishnan, C. R., Smolka, S. A.: Query-Based model checking of Ad Hoc Network Protocols. CONCUR '09: 603–619

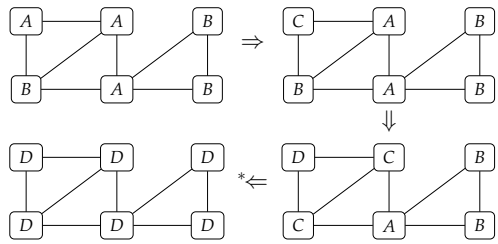


Figure 2: Example of execution

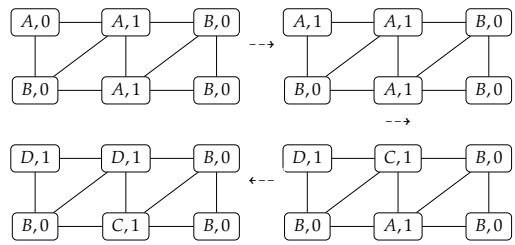


Figure 3: Example of intermittent execution

7 Example of AHN

As an example, consider a protocol consisting of the following rules: (A, τ, C), $(C, !!m, D)$, $(B, ??m, C)$, and $(A, ??m, C)$. As shown in Fig. 2 in appendix, starting from a configuration with only A and B nodes, an A node first moves to C and then send m to his/her neighbors. In turn, they forward the message m to their neighbors, and so on.

8 Example of AHN with intermittent nodes

As an example, consider the following protocol: $(A, !!m, D)$, $(B, ??m, C)$, and $(A, ??m, C)$. As shown in Fig. 3 in appendix, the top-left node is initially turned off. It then turns on, sends a message, and only active neighbors react, and so on.

9 Sketch of proof of Theorem 13

The proof is by reduction to COVER for ad hoc network protocols with node intermittence. Consider a process definition $\mathcal{P}$. It is easy to see that a computation leading to a configuration that exposes the control state q in $AHN_l(\mathcal{P})$ has a corresponding computation in $AHN_i(\mathcal{P})$: it is sufficient to mimic **Broadcast** transitions by executing before the broadcast a sequence of **Turn off** transitions that switch off the nodes that do not receive the message, and by performing after the broadcast the **Turn on** transitions ton the same nodes.

Consider now a computation in $AHN_i(\mathcal{P})$ leading to a configuration that exposes the control state q. This computation can be mimicked in $AHN_l(\mathcal{P})$ simply by assuming that the nodes that are turned off during a specific phase of the computation in $AHN_i(\mathcal{P})$, lose the messages that are broadcast in that phase in the corresponding computation in $AHN_l(\mathcal{P})$.

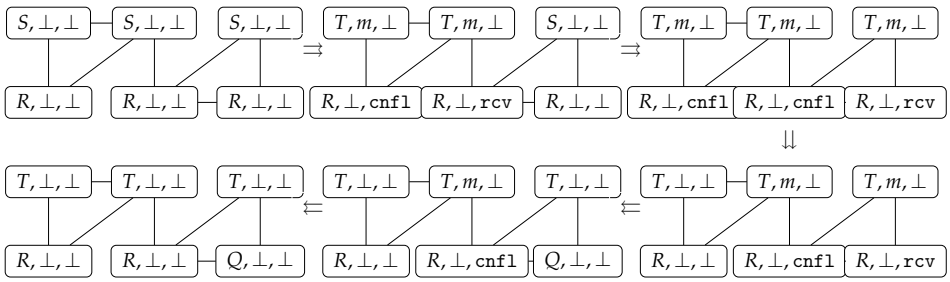


Figure 4: Example of execution with conflict/receive transient states.

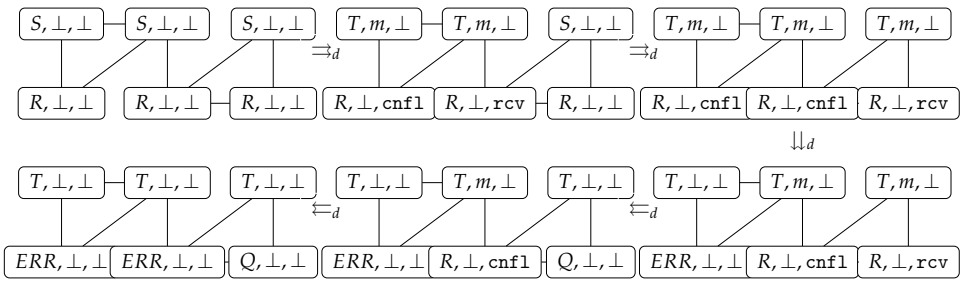


Figure 5: Example of execution with conflict detection (indicated as $\rightrightarrows_d$).