Parameterized Verification of Ad Hoc Networks

Giorgio Delzanno[1], Arnaud Sangnier[1], and Gianluigi Zavattaro[2]

[1] University of Genova
[2] University of Bologna

Abstract. We study decision problems for parameterized verification of a formal model of Ad Hoc Networks with selective broadcast and spontaneous movement. The communication topology of a network is represented as a graph. Nodes represent states of individual processes. Adjacent nodes represent single-hop neighbors. Processes are finite state automata that communicate via selective broadcast messages. Reception of a broadcast is restricted to single-hop neighbors. For this model we consider verification problems that can be expressed as reachability of configurations with one node (resp. all nodes) in a certain state from an initial configuration with an arbitrary number of nodes and unknown topology. We draw a complete picture of the decidability boundaries of these problems according to different assumptions on communication graphs, namely static, mobile, and bounded path topology.

1 Introduction

In recent years there has been an increasing interest in the formal specification of protocols used in Ad Hoc Networks. Building on previous models like [7,19,21], in [22] Singh, Ramakrishnan and Smolka define the ω-calculus as a formal model of Ad Hoc Networks with selective broadcast and spontaneous movement. In the ω-calculus a configuration consists of a finite set of processes. Each process has a local state and an interface containing a finite set of group names. A group name represents a possible communication link with other processes in the network. From an abstract point of view, the structure underlying a configuration of the ω-calculus is a finite graph that defines the communication topology of a network. A node in the graph represents the current state of an individual process. There exists and edge between two nodes if the corresponding interfaces share a common group name. Adjacent nodes are called single-hop neighbors. Processes communicate through selective broadcast. Specifically, a broadcast message can be received only by the set of single-hop neighbors of the emitter.

When the number of nodes is fixed a priori, formal models of Ad Hoc Networks like those provided by the ω-calculus can be verified by using finite-state model checking [11] or constraint-based model checking [22]. Lifting the study of verification problems to the parameterized case in which networks have arbitrary size and possibly unknown topology is a challenging problem for this class of distributed systems.

In the present paper we study parameterized verification problems for an automata-based model of Ad Hoc Networks, we named AHN, inspired by the

P. Gastin and F. Laroussinie (Eds.): CONCUR 2010, LNCS 6269, pp. 313–327, 2010.

ω-calculus of [22]. Each node of a network is modeled as a finite-state automaton in which local transitions model either an internal action, a broadcast or a reception of a message taken from a finite alphabet. A protocol is defined then as the composition of a finite but arbitrary number of copies of the automaton running in parallel. As in the ω-calculus a configuration can naturally be viewed as a graph that defines the communication topology. We use group names and interfaces to select the set of neighbors of an emitter that are capable to receive a broadcast message. In our model we define verification problems parametric on the size (number of nodes) and shape (topology of the communication graph) of the initial configurations. Our investigations take into account different assumptions on the communication topology. Specifically, we consider configurations either with static or mobile topology and with static and bounded path topology. In the latter case we assume that there is an upper bound on the length of simple paths in the communication graphs underlying the initial configurations. For each of the above mentioned assumptions, we present a systematic analysis of the following decision problems: (COVER) reachability of a configuration with one node in a given state, (TARGET) reachability of a configuration with all nodes in a given state, (REPEAT-COVER) existence of a computation traversing infinitely often configurations with at least one node in a given state.

Our main negative result is that all three parameterized problems are undecidable for arbitrary static topology. The proofs are based on a simulation of a Turing complete formalism which is correct only for topologies of a given size and shape. As the topology is arbitrary, the simulation is preceded by a protocol able to explore the current topology and to start the simulation only if it is of the expected form.

Perhaps surprisingly, all three problems become decidable in the mobile case. This result is similar to what happens in channel systems where introducing lossiness simplifies the verification task [3]. For static bounded path topologies, TARGET and REPEAT-COVER turn out to be undecidable while COVER is still decidable. The latter result is similar to those presented in [23,15] for bounded depth processes with point-to-point communication. However, due to broadcast communication we need to resort to a different proof technique. Namely, even if we use the theory of *well structured transition systems* (WSTS) [1,2,12] as in [15,23], we need to consider a stronger ordering on configurations based on the induced subgraph ordering [5] instead of the subgraph embedding. To the best of our knowledge, this is the first case of application of the induced subgraph ordering in the context of WSTS.

Related Work. Formal models of networks in which all processes receive a broadcast message at once are presented and analyzed in [6,8,19]. In our setting this kind of broadcast is modeled by configurations whose underlying graph is a clique. Selective broadcast has been studied in several process calculi for ad hoc networks and wireless communication like those presented in [7,13,14,16,21], some of which turn out to be extensions of the pi-calculus [17]. A distinguished feature of the ω-calculus [21,22] is that mobility of processes is abstracted from

their communication actions, i.e., mobility is spontaneous and it does not involve any communication. In [22] the authors define a constraint-based analysis for configurations with a fixed number of nodes. The shape of topologies leading to bad configurations is constructed in a lazy manner during a symbolic exploration of the state space. The symbolic approach in [22] seems to improve verification results obtained with more standard model checking tools like Uppaal [11]. In the same paper the authors mention that, without name restriction, reachability of a configuration from an initial one is decidable. In the present paper we lift our reachability problems to the parameterized case in which the initial configuration has unknown size and shape. For networks of arbitrary size, in [20] Saksena et al. define a symbolic procedure based on graph-transformations to analyze routing protocol for Ad Hoc Networks. The symbolic representation is based on upward closed sets of graphs ordered by subgraph inclusion. The procedure is not guaranteed to terminate. In our paper we use a different ordering on graphs, namely induced subgraph, for ensuring termination of backward reachability on graphs with paths of bounded length.

Due to lack of space, omitted proofs can be found in [4].

2 A Formal Model for Ad Hoc Network Protocols

Following [22], a configuration of an Ad Hoc Network is modeled as a tuple of nodes $\langle n_1, \ldots, n_k \rangle$ with $k \geq 1$. A node n_i maintains information about the current state of an individual process and its current set of communication links. The behavior of a single node is described by a finite-state automaton, called *process*, in which transitions represent internal, broadcast, or reception actions. Concerning the representation of communication links, a convenient way to describe the network topology is based on the use of group names. Specifically, to each node we associate an interface that defines the set of group names to which the node belongs. Two nodes are connected if their interfaces share at least one common group name. The semantics of the transitions specifies how different nodes interact in a global configuration. We assume that nodes cannot dynamically be created or deleted.

Definition 1. *An Ad Hoc Network Protocol (shortly AHN) is a pair* $\langle P, \mathcal{G} \rangle$ *where* P *is a process definition and* $\mathcal{G}$ *is a denumerable set of group names. A process* $P = \langle Q, \Sigma, E, Q_0 \rangle$ *defines the behavior of a single node of the network. Here* Q *is a finite set of control states,* Σ *is a finite alphabet,* $E \subseteq Q \times (\{\tau\} \cup \{\mathbf{b}(a), \mathbf{r}(a) \mid a \in \Sigma\}) \times Q$ *is the transition relation, and* $Q_0 \subseteq Q$ *is a set of initial control states.*

The label τ represents an internal action of a process, the label $\mathbf{b}(a)$ represents the capability of broadcasting message a, and $\mathbf{r}(a)$ the capability of receiving message a.

Definition 2. *Assume* $P = \langle Q, \Sigma, E, Q_0 \rangle$. *A* node *$n$ is a pair* $\langle q, I \rangle$, *where* $q \in Q$ *is called the* state *and* $I \subseteq \mathcal{G}$ *is the called the* interface *of the node. A*

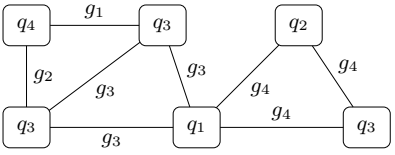

Fig. 1. Graph associated to a configuration

configuration γ is a tuple $\langle n_1, \ldots, n_k \rangle$ of nodes with size $k \geq 1$.
We use $\mathcal{C}$ to denote the set of configurations of any size.

A configuration γ defines a given network topology specified by the graph $G(\gamma)$. The vertices in $G(\gamma)$ are in bijection with the nodes of γ. The label of a vertex is the state of the corresponding node in γ. Furthermore, there exists an edge between two vertices in $G(\gamma)$ if and only if the intersection of the interfaces of the corresponding nodes in γ is not empty.

For instance, consider a configuration γ with nodes $n_1 = \langle q_4, \{g_1, g_2\}\rangle$, $n_2 = \langle q_3, \{g_1, g_3\}\rangle$, $n_3 = \langle q_3, \{g_2, g_3\}\rangle$, $n_4 = \langle q_1, \{g_3, g_4\}\rangle$, $n_5 = \langle q_2, \{g_4\}\rangle$, and $n_6 = \langle q_3, \{g_4\}\rangle$, the communication topology induced by γ is depicted in Figure 1.

We define functions σ and ι to extract the state and the interface of a node, i.e., $\sigma(\langle q, I\rangle) = q$ and $\iota(\langle q, I\rangle) = I$. We extend σ and ι to configurations in the natural way. For a configuration γ, we sometimes consider $\sigma(\gamma)$ as a set rather than a vector and use $q \in \sigma(\gamma)$ to denote that there exists a node n_i in γ such that $\sigma(n_i) = q$. The set of indexes of nodes adjacent to a node n_i in a configuration $\gamma = \langle n_1, \ldots, n_k\rangle$ (single-hop neighbors of n_i in $G(\gamma)$) is defined as $Shn(\gamma, i) = \{j \in [1..k] \mid \iota(n_i) \cap \iota(n_j) \neq \emptyset \text{ and } j \neq i\}$. For a broadcast message $a \in \Sigma$, we define the set of indexes $Rec(\gamma, a) = \{j \in [1..k] \mid (\sigma(n_j), \mathbf{r}(a), q) \in E \text{ for some } q \in Q\}$. The set of nodes in γ enabled by a broadcast a sent by node n_i is then defined as $Enabled(\gamma, i, a) = Shn(\gamma, i) \cap Rec(\gamma, a)$.

Operational Semantics. The semantics of an AHN $\langle P = \langle Q, \Sigma, E, Q_0\rangle, \mathcal{G}\rangle$ is given by its associated transition system $TS(P, \mathcal{G}) = \langle \mathcal{C}, \Rightarrow, \mathcal{C}_0\rangle$ (we recall that $\mathcal{C}$ is the set of configurations of all possible size), $\mathcal{C}_0$ is the set of initial configurations defined as $\mathcal{C}_0 = \{\gamma \in \mathcal{C} \mid \sigma(\gamma) \subseteq Q_0\}$ and $\Rightarrow$ is the transition relation in $\mathcal{C} \times \mathcal{C}$ defined as follows.

Definition 3. *For $\gamma = \langle n_1, \ldots, n_k\rangle$ and $\gamma' = \langle n'_1, \ldots, n'_k\rangle$, $\gamma \Rightarrow \gamma'$ iff one of the following conditions holds:*

Internal. *There exists $i \in [1..k]$ such that $(\sigma(n_i), \tau, \sigma(n'_i)) \in E$, $\iota(n_i) = \iota(n'_i)$, and $n'_j = n_j$ for all $j \in [1..k] \setminus \{i\}$.*

Broadcast. *There exists $a \in \Sigma$ and $i \in [1..k]$ such that $(\sigma(n_i), \mathbf{b}(a), \sigma(n'_i)) \in E$, $\iota(n_i) = \iota(n'_i)$, and the following conditions hold:*
- *For all $j \in Enabled(\gamma, i, a)$, $(\sigma(n_j), \mathbf{r}(a), \sigma(n'_j)) \in E$ and $\iota(n'_j) = \iota(n_j)$;*
- *For all $p \notin (Enabled(\gamma, i, a) \cup \{i\})$, $n'_p = n_p$.*

We denote by $\Rightarrow^*$ the reflexive and transitive closure of $\Rightarrow$. An execution is a sequence $\gamma_0\gamma_1 \ldots$ such that $\sigma(\gamma_0) \subseteq Q_0$ and $\gamma_i \Rightarrow \gamma_{i+1}$ for $i \geq 0$.

As an example, consider a process definition in which $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{a\}$, and E contains the rules

$$(q_1, \tau, q_2), \quad (q_2, \mathbf{b}(a), q_4), \quad (q_1, \mathbf{r}(a), q_2), \text{ and } (q_3, \mathbf{r}(a), q_2)\}$$

Starting from a connected component in which nodes have label q_1 or q_3, once an alarm is detected by a q_1 node (τ action), it is flooded (broadcast of message a) to all single-hop neighbors which, in turn, forward the alarm to their neighbors, and so on. After some steps, the alarm reaches all multi-hop neighbors yielding a configuration in which all nodes (in the connected component) have label q_4.

3 Decision Problems

In this section we consider decision problems related to verification of safety and liveness properties like those studied for Petri nets [9,10]. We remark that in our formulation the size and the shape of of the initial configurations is not fixed a priori. In the following definitions we assume an AHN $\langle P, \mathcal{G}\rangle$ with transition system $TS(P, \mathcal{G}) = \langle \mathcal{C}, \Rightarrow, \mathcal{C}_0\rangle$.

The first problem is *control state reachability* (COVER) defined as follows: given a control state q of P, do there exist $\gamma \in \mathcal{C}_0$ and $\gamma' \in \mathcal{C}$ such that $\gamma \Rightarrow^* \gamma'$ and $q \in \sigma(\gamma')$?
We recall that a configuration γ is initial if $\sigma(\gamma) \subseteq Q_0$. Notice that being initial does not enforce any particular constraint on the topology. Thus, assume that the state q represents an error state for a node of the network. If we can solve COVER, then we can decide if there exists a topology of the network and a sufficient number of processes from which we can generate a configuration in which the error is exposed.

The second problem is *target reachability* (TARGET) which we define as follows: given a subset of control states F of P, do there exist $\gamma \in \mathcal{C}_0$ and $\gamma' \in \mathcal{C}$ such that $\gamma \Rightarrow^* \gamma'$ and $\sigma(\gamma') \subseteq F$?
Assume that the subset F represents blocking states for nodes of the network. If we can solve TARGET, then we can decide if there exists a topology of the network and a sufficient number of processes from which we can reach a configuration in which processes can no longer move.

Finally we will also study the *repeated control state reachability problem* (REPEAT-COVER): given a control state q of P, does there exist an infinite execution $\gamma_0 \Rightarrow \gamma_1 \Rightarrow \ldots$ such that the set $\{i \in \mathbb{N} \mid q \in \sigma(\gamma_i)\}$ is infinite?
This problem is a classical extension of the COVER problem that can be used, for instance, to verify whether a protocol is able to react to the occurrence of errors by reaching a state from which errors do not occur any longer. Assume that q represents the error state. If we can solve REPEAT-COVER, then we can decide if there exists a topology of the network and a sufficient number of processes that can generate a computation including infinitely many error states.

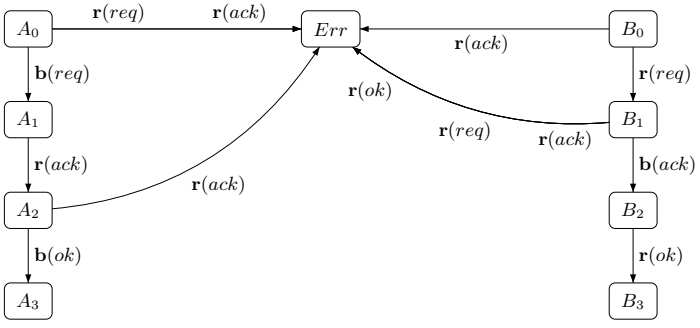

Fig. 2. The RAO (Req/Ack/Ok) protocol

4 Static Topology

In this section, we will prove that COVER, TARGET and REPEAT-COVER are all undecidable problems. We first recall that in our decision problems there are no assumptions on the number of nodes and on the communication topology of the initial configurations. Furthermore, the model does not admit dynamic reconfigurations of the topology. Broadcast communication can be used however to ensure that a specific protocol succeeds only if the network topology has a certain form. To be more precise, consider the protocol specified by the process Req/Ack/Ok (RAO) of Figure 2 where A_0 and B_0 are the initial states. The following property then holds.

Proposition 1. *Let $\mathcal{G}$ be a denumerable set of group names and γ an initial configuration of the AHN $\langle RAO, \mathcal{G}\rangle$. If γ' is a configuration such that $\gamma \Rightarrow^* \gamma'$ and such that $B_3 \in \sigma(\gamma')$, then the graph $G(\gamma')$ has the following properties:*

- *each node n labeled with B_3 is adjacent to a unique node labeled with A_3 (we denote this node by $f(n)$)[1];*
- *for each node n labeled with B_3, all the nodes adjacent to n or $f(n)$ are labeled with Err (except of course n and $f(n)$).*

Proof. Assume n is a node of γ' in state B_3. Since n has received a message *ok* to reach B_3, it is necessarily adjacent to a node in state A_3. No other node adjacent to n can be in state A_3. Indeed, if n receives two *req* messages before sending an *ack*, then n moves to state Err. Furthermore, if n sends an *ack*, then all adjacent nodes that are in states A_0 (ready to send a *req*) move to state Err. Rule $(A_0, \mathbf{r}(req), Err)$ ensures that, in $G(\gamma')$, no node labeled A_i is adjacent to a node labeled A_3. Rules $(B_0, \mathbf{r}(ack), Err)$ and $(B_1, \mathbf{r}(ack), Err)$ ensure that, when n has label B_3, its single-hop neighbors cannot have label B_i. Rule $(B_1, \mathbf{r}(ok), Err)$ ensures that a node different from n but adjacent to $f(n)$ must have state different from B_i. Indeed, if such a node is in state B_1, then the broadcast *ok* sent by $f(n)$ sends it to Err, and if such a node moves to B_2 sending *ack* then it sends node $f(n)$ to Err before it can reach A_3. □

[1] Two nodes are adjacent iff there is an edge between them.

Using an extension of the RAO protocol, we can define an AHN which simulates the execution of a deterministic two-counter Minsky machine and reduce the halting problem to COVER. A deterministic Minsky machine manipulates two integer variables c_1 and c_2, which are called counters, and it is composed of a finite set of instructions. Each of the instruction is either of the form (1) $L : c_i := c_i + 1$; `goto` L' or (2) L : `if` $c_i = 0$ `then goto` L' `else` $c_i := c_i - 1$; `goto` L'' where $i \in \{1, 2\}$ and L, L', L'' are labels preceding each instruction. Furthermore there is a special label L_F from which nothing can be done. The halting problem consists then in deciding whether or not the execution that starts from L_0 with counters equal to 0 reaches L_F.

The intuition behind the reduction is as follows. In the first phase we use a variant of the RAO protocol to select a control node and two list of nodes, with state representing value 0 or 1, used to simulate the content of the counters. The length of each list must be sufficient to represent the maximum value stored in each counter during the simulation. All other nodes in the vicinity of the control state and of the two lists are sent to an error state during the execution of the first phase. In the second phase the control node starts the simulation of the instructions. It operates by querying and changing the state of the nodes in the two lists according to the type of instructions to be executed. In this phase all nodes in the same list behave in the same way. Requests are propagated back and forth a list by using broadcast sent by a node to its (unique) single-hop successor/predecessor node. The protocols that define the two phases are fairly complicated; the corresponding automata are described in detail in [4]. From the undecidability of the halting problem for two-counter Minsky machines [18], we obtain the following result.

Theorem 1. COVER *is an undecidable problem.*

Furthermore, we have the following corollary.

Corollary 1. TARGET *and* REPEAT-COVER *are undecidable problems.*

Proof. Assume $P = \langle Q, \Sigma, E, Q_0 \rangle$ and let $\mathcal{G}$ be a denumerable set of group names and $q \in Q$. To reduce COVER to REPEAT-COVER we simply add a loop of the form (q, τ, q) to E. To reduce COVER to TARGET, we build the process $P' = \langle Q', \Sigma', E', Q'_0 \rangle$ defined as follows:

- $Q' = Q \uplus \{r_0, r_1, r_F\}$ (with $Q \cap \{r_0, r_1, r_F\} = \emptyset$);
- $\Sigma' = \Sigma \uplus \{F_1, F_2\}$ (with $\Sigma \cap \{F_1, F_2\} = \emptyset$);
- $E' = E \uplus \{(q, \mathbf{b}(F_1), r_F), (r_0, \mathbf{r}(F_1), r_1), (r_1, \mathbf{b}(F_2), r_F)\} \cup \{(q', \mathbf{r}(F_2), r_F) \mid q' \in Q\}$;
- $Q'_0 = Q_0 \uplus \{r_0\}$.

Let $TS(P, \mathcal{G}) = \langle \mathcal{C}, \Rightarrow, \mathcal{C}_0 \rangle$ and $TS(P', \mathcal{G}) = \langle \mathcal{C}', \Rightarrow', \mathcal{C}'_0 \rangle$. It is then easy to see that there exist $\gamma_2 \in \mathcal{C}'_0$ and $\gamma'_2 \in \mathcal{C}'$ such that $\gamma_2 \Rightarrow'^* \gamma'_2$ and $\sigma(\gamma'_2) \subseteq \{r_F\}$ if and only if there exists $\gamma_1 \in \mathcal{C}_0$ and $\gamma'_1 \in \mathcal{C}$ such that $\gamma_1 \Rightarrow^* \gamma'_1$ and $q \in \sigma(\gamma'_1)$. In fact, in $TS(P', \mathcal{G})$ after being in the state q a node can broadcast the message F_1 which launches a protocol whose goal is to send all the other nodes in the state r_F. □

5 Mobile Topology

In this section we consider a variant on the semantics of AHN obtained by adding spontaneous movement of nodes as in [22]. Node mobility is modeled by non-deterministic updates of their interfaces. Formally, let $\langle P, \mathcal{G}\rangle$ be a AHN with $TS(P,\mathcal{G}) = \langle \mathcal{C}, \Rightarrow, \mathcal{C}_0\rangle$. The semantics of $\langle P, \mathcal{G}\rangle$ with mobility is given by the transition system $TS_M(P,\mathcal{G}) = \langle \mathcal{C}, \Rightarrow_M, \mathcal{C}_0\rangle$ where the transition $\Rightarrow_M$ is defined as follows.

Definition 4 (Transition Relation with Mobility). *For $\gamma, \gamma' \in \mathcal{C}$ with $\gamma = \langle n_1, \ldots, n_k\rangle$ and $\gamma' = \langle n'_1, \ldots, n'_k\rangle$, we have $\gamma \Rightarrow_M \gamma'$ iff one the following conditions holds:*

- *$\gamma \Rightarrow \gamma'$ (**no movement**);*
- *there exists $i \in [1..k]$ such that, $\sigma(n'_i) = \sigma(n_i)$ (state does not change), $\iota(n'_i) \subseteq \mathcal{G}$ (interface changes in an arbitrary way), and for all $j \in [1..k] \setminus \{i\}$, $n'_j = n_j$ (all other nodes remain unchanged) (**movement**).*

We prove next that COVER, REPEAT-COVER and TARGET are decidable for AHN with mobility. Intuitively, this follows from the observation that the topology of the network changes in an unpredictable and uncontrollable manner. Hence, every broadcast message sent by a node is received by a non-deterministically chosen set of nodes, namely those in the transmission range of the emitter at the time the message is sent. Formally, we reduce COVER, TARGET and REPEAT-COVER respectively to the *marking coverability*, *marking reachability* and *repeated marking coverability* problems for Petri nets, which are known to be decidable [9,10].

A *Petri net* (see e.g. [9]) is a tuple $N = (S, T, m_0)$, where S and T are finite sets of *places* and *transitions*, respectively. A finite multiset over the set S of places is called a *marking*, and m_0 is the initial marking. Given a marking m and a place p, we say that the place p contains $m(p)$ *tokens* in the marking m if there are $m(p)$ occurrences of p in the multiset m. A transition is a pair of markings written in the form $m' \mapsto m''$. The marking m of a Petri net can be modified by means of transitions firing: a transition $m' \mapsto m''$ can fire if $m(p) \geq m'(p)$ for every place $p \in S$; upon transition firing the new marking of the net becomes $n = (m \setminus m') \uplus m''$ where $\setminus$ and $\uplus$ are the difference and union operators for multisets, respectively. This is written as $m \to n$. We use $\to^*$ [resp. $\to^+$] to denote the reflexive and transitive closure [resp. the transitive closure] of $\to$. We say that m' is *reachable from* m if $m \to^* m'$. The *coverability* problem for marking m consists of checking whether $m_0 \to^* m'$ with $m'(p) \geq m(p)$ for every place $p \in S$. The *reachability* problem for marking m consists of checking whether $m_0 \to^* m$. Finally, the *repeated coverability problem* for marking m consists of checking wether there exists an infinite execution $m_0 \to^+ m_1 \to^+ m_2 \to^+ \ldots$ such that for all $i \in \mathbb{N}$, $m_i(p) \geq m(p)$ for every place $p \in S$. The coverability, reachability and repeated coverability problems are decidable for Petri nets [9,10].

We now show how to build a Petri net which simulates the behavior of an AHN with mobility. Figure 3 gives an example of a Petri net associated to a process. In the Petri net, each control state q has a corresponding place q, and

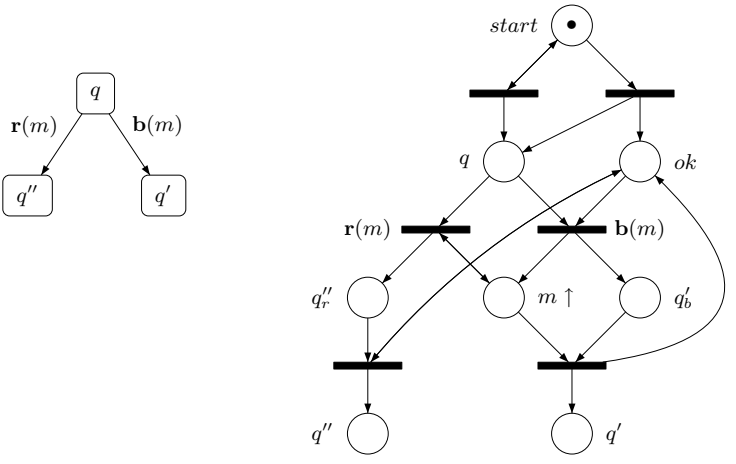

Fig. 3. A Petri net which simulates an AHN with mobility

each node $\langle q, I\rangle$ of the network is represented by a token in the place q. The interfaces of nodes (thus also the network topology) are abstracted away in the Petri net. In the first phase, the net non-deterministically puts tokens in the places corresponding to the initial control states of the process. Then it produces a token in the place *ok* and the simulation begins. The broadcast communication is modeled by a *broadcast protocol* whose effect is to deliver the emitted message to a non-deterministically chosen set of potential receivers. More precisely, the broadcast protocol can be started by a token in a place q such that $(q, \mathbf{b}(m), q')$; then the token is moved to a transient place q'_b and a token is produced in the place $m\uparrow$. During the execution of the protocol, every token in a place r such that $(r, \mathbf{r}(m), r')$ can receive the message moving in a transient place r'_r. The protocol ends when the token in the transient place q'_b moves to the place q'. The tokens in the transient places r'_r can move to the corresponding places r' only when no broadcast protocol is running (when a broadcast protocol is running, there is no token in the place *ok*). This broadcast protocol does not faithfully reproduce the broadcast as formalized in the AHN model: in fact, in the Petri net there is no guarantee that the tokens in the transient places r'_r move to the corresponding places r' at the end of the execution of the protocol. A token that remains in those transient places (thus losing the possibility to interact with the other tokens in the Petri net) corresponds to a node in the AHN model that disconnects, due to mobility, from the other nodes in the system. Testing whether there is an execution in the AHN with mobility which ends in a configuration where one of the nodes is in the control state q can be done by testing whether the marking $\{q, ok\}$ can be covered in the associated Petri net. Hence:

Theorem 2. *There exists a reduction from the* COVER *problem for AHN with mobility to the marking coverability problem for Petri nets.*

Using the same construction, we also obtain:

Theorem 3. *There exists a reduction from the* REPEAT-COVER *problem for AHN with mobility to the marking repeated coverability problem for Petri nets.*

In order to reduce TARGET to marking reachability we need to extend the Petri net associated to an AHN, adding a final stage in the computation, dedicated to the elimination of tokens from the places corresponding to the final states in F. Intuitively we add a transition of the form $\{ok\} \mapsto \{end\}$ and for each $q \in F$ we add a transition $\{end, q\} \mapsto \{end\}$ and we then test if the marking where all the places are empty except the place *end* is reachable.

Theorem 4. *There exists a reduction from the* TARGET *problem for AHN with mobility to the marking reachability problem for Petri nets.*

From these three last theorems and from the fact that the marking coverability, marking repeated coverability and marking reachability problems are decidable for Petri nets, we finally deduce:

Corollary 2. COVER*,* REPEAT-COVER *and* TARGET *are decidable for AHN with mobility.*

6 Static Bounded Path Topology

Let us go back to the AHN model with static topology. The possibility for a message to pass through an unbounded number of new nodes is a key feature in the proof of Theorem 1 (undecidability of COVER for static topology). For this reason, it seems natural to study COVER, TARGET and REPEAT-COVER for a restricted class of configurations in which, for a fixed K, a message can pass through at most K-different nodes. Formally, given an AHN $\langle P, \mathcal{G} \rangle$ with $TS(P, \mathcal{G}) = \langle \mathcal{C}, \Rightarrow, \mathcal{C}_0 \rangle$ our class of restricted configurations is defined as follows:

Definition 5. *Given an integer $K \geq 1$, a configuration γ is a* K-bounded path configuration *if the longest simple path in the associated graph $G(\gamma)$ has length at most K.*

We denote by $\mathcal{C}^K$ the set of K-bounded path configurations. The semantics of the AHN $\langle P, \mathcal{G} \rangle$ resticted to K-bounded path configurations is given by the transition system $TS_K(P, \mathcal{G}) = \langle \mathcal{C}^K, \Rightarrow_K, \mathcal{C}_0^K \rangle$ where the transition relation $\Rightarrow_K$ is the restriction of $\Rightarrow$ to $\mathcal{C}^K \times \mathcal{C}^K$ and $\mathcal{C}_0^K = \mathcal{C}_0 \cap \mathcal{C}^K$. For fixed K, the class of K-bounded path configurations contains an infinite set of graphs. To give some examples, stars with a center node and any number of satellite nodes have always 3-bounded paths, whereas cliques of arbitrary size may have paths of unbounded length.

6.1 Decidability of COVER

In order to study COVER restricted to bounded path configurations, we first introduce some definitions and prove auxiliary properties. First of all, we give the definition of the *induced subgraph* relation.

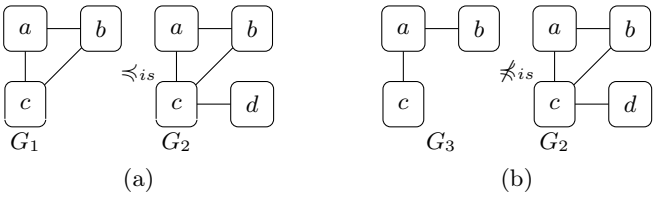

Fig. 4. Examples of the induce subgraph relation

Definition 6. *For configurations γ_1 and γ_2, we define $\gamma_1 \preccurlyeq_{is} \gamma_2$ if there exists a label preserving injection h from nodes of $G_1 = G(\gamma_1)$ to nodes of $G_2 = G(\gamma_2)$ such that (n, n') is an edge in G_1* if and only if *$(h(n), h(n'))$ is an edge in G_2, i.e., G_1 is* isomorphic *to an induced subgraph of G_2.*

Notice that the induced subgraph relation is stronger than the *subgraph relation*. The subgraph ordering requires only a homomorphic embedding of G_1 into G_2. To illustrate, in Fig. 4 (a) G_1 is isomorphic to an induced subgraph of G_2, hence $G_1 \preccurlyeq_{is} G_2$. In (b) G_3 is obtained from G_1 by removing the edge from node a to node c. The induced graph of G_2 with nodes a, b, c is no more isomorphic to G_3, hence $G_3 \not\preccurlyeq_{is} G_2$. Notice, however, that G_3 is still a subgraph of G_2. The following lemma then holds.

Lemma 1. *Given $K \geq 1$, $(\mathcal{C}^K, \preccurlyeq_{is})$ is a well-quasi ordering (shortly wqo), i.e., for every infinite sequence of K-bounded path configurations $\gamma_1\gamma_2 \ldots$ there exist $i < j$ s.t. $\gamma_i \preccurlyeq_{is} \gamma_j$.*

Proof. It immediately follows from Ding's Theorem (Theorem 2.2 in [5]).

Given a subset $S \subseteq \mathcal{C}^K$ we define $S \uparrow = \{\gamma' \in \mathcal{C}^K \mid \gamma \in S \text{ and } \gamma \preccurlyeq_{is} \gamma'\}$, i.e., $S \uparrow$ is the set of configurations generated by those in S via $\preccurlyeq_{is}$. A set $S \subseteq \mathcal{C}^K$ is an *upward closed set* w.r.t. to $(\mathcal{C}^K, \preccurlyeq_{is})$ if $S \uparrow = S$. Since $(\mathcal{C}^K, \preccurlyeq_{is})$ is a wqo, we obtain that every set of configurations that is upward closed w.r.t. $(\mathcal{C}^K, \preccurlyeq_{is})$ has a finite basis, i.e., it can be finitely represented by a finite number of K-bounded path configurations. We can exploit this property to define a decision procedure for COVER. For this purpose, we apply the methodology proposed in [1]. The first property we need to prove is that the transition relation induced by our model is compatible with $\preccurlyeq_{is}$.

Lemma 2 (Monotonicity). *For every $\gamma_1, \gamma_2, \gamma_1' \in \mathcal{C}^K$ such that $\gamma_1 \Rightarrow_K \gamma_2$ and $\gamma_1 \preccurlyeq_{is} \gamma_1'$, there exists $\gamma_2' \in \mathcal{C}^K$ such that $\gamma_1' \Rightarrow_K \gamma_2'$ and $\gamma_2 \preccurlyeq_{is} \gamma_2'$.*

Proof. For lack of space, we focus here on the application of a broadcast rule with label $\mathbf{b}(a)$. Assume that the rule is applied to a node n adjacent in $G(\gamma_1)$ to nodes $N = \{n_1, \ldots, n_k\}$. Assume that the subset N' of N contains nodes that are enabled by message a. By applying the operational semantics, the state of n and the states of nodes in N' are updated simultaneously Assume now that $G(\gamma_1)$ is isomorphic to an induced subgraph of $G(\gamma_1')$ via the injection h. Then,

$h(n)$ is adjacent to the set of nodes $h(N)$ (there cannot be more connections since $h(G(\gamma_1))$ is an induced subgraph of $G(\gamma_1')$). Thus, the same rule is enabled in $h(n)$ and in $h(N')$ and yields the same effect on the labels. Thus, we obtain γ_2' such that $G(\gamma_2) \preccurlyeq_{is} G(\gamma_2')$. □

Monotonicity ensures that if S is an upward closed set of configurations (w.r.t. $(\mathcal{C}^K, \preccurlyeq_{is})$), then the set of predecessors of S accroding to $\Rightarrow_K$, defined as $pre_K(S) = \{\gamma \mid \gamma \Rightarrow_K \gamma' \text{ and } \gamma' \in S\}$, is still upward closed. We now show that we can effectively compute a finite representation of $S \cup pre_K(S)$.

Lemma 3. *Given a finite basis B of an upward closed set $S \subseteq \mathcal{C}^K$, there exists an algorithm to compute a finite basis B' of $S \cup pre_K(S)$ s.t. $S \cup pre_K(S) = B' \uparrow$.*

Proof. We focus on the the backward application of a broadcast rule $(q, \mathbf{b}(a), q')$ to a configuration in the upward closure of B. The computation of the set $B \uparrow \cup pre_K(B \uparrow)$, where $pre_K(B \uparrow) = \{\gamma \mid \gamma \Rightarrow_K \gamma' \text{ and } \gamma' \in B \uparrow\}$ is done according to the following steps.
Initially, we set $B' := B$.
Then, for each $\gamma \in B$:

1. For each vertex n labeled with q' in the graph $G(\gamma)$, let N be the set of nodes adjacent to q', we apply then the following rule:
   - If there exists a node in N with state r such that $(r, \mathbf{r}(a), r')$ is a rule in the model, then we add no predecessor to B' (because every node $n' \in S$ in state r **must** react to the broadcast);
   - otherwise, for any subset $N' = \{n_1, \ldots, n_k\}$ of nodes in N such that n_i has state r_i' and $(r_i, \mathbf{r}(a), r_i')$ is a rule in the model, we build a predecessor configuration γ' in which the label of n is updated to q and the label of n_i is updated to r_i for $i \in \{1, \ldots, k\}$ and if there is no γ'' in B' such that $\gamma'' \preccurlyeq_{is} \gamma'$, we add γ' to B' (Note that we have to select all possible subset N' of N because we must consider the cases in which nodes connected to n are already in the target state of some reception rule).
2. Let Γ' be the set of configurations γ' in $\mathcal{C}^K$ obtained by adding a node n in state q' to γ such that in $G(\gamma')$, n is adjacent to at least one node (i.e. in Γ' we have all the configurations obtained by added a connected node to γ and which are still K-bounded path configurations). We then apply the precedent rule 1. to each configuration in Γ' considering the added node n labeled with q'. □

We can now state the main theorem of this section.

Theorem 5. *For $K \geq 1$,* COVER *is decidable for AHN restricted to K-bounded path configurations.*

Proof. From Lemmas 1, 2, and 3 it follows that the transition system induced by any AHN is well structured with respect to $(\mathcal{C}^K, \preccurlyeq_{is})$. The theorem then follows from the general properties of well structured transition systems [1,2,12]. The decision algorithm is based on a symbolic backward exploration of the state space that, starting from a graph with a single node denoting the target state, saturates the set of symbolic predecessors computed using the operator described

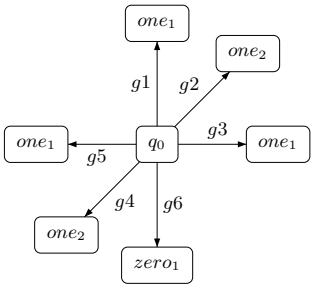

Fig. 5. Configuration $\langle q_0, c_1 = 3, c_2 = 2\rangle$ for $c_1 \in [0, 4]$ and $c_2 \in [0, 2]$

in Lemma 3. Termination is ensured by the wqo property of the induced subgraph relation over K-bounded path configurations. □

6.2 Undecidability of TARGET and REPEAT-COVER

In order to show that TARGET is undecidable for K-bounded path configurations, we show how to model a Minsky machine in such a way that the machine terminates if and only if the corresponding AHN has a computation (restricted to K-bounded path configurations) that reaches a configuration in which all nodes are in some specific final state. For this purpose, we design a protocol that succeeds only on star topologies in which the center node represents the current control state and the satellite nodes the units of the two counters. Such units are initially in the $zero_i$ state (with $i \in \{1, 2\}$). The number of satellite nodes needed to guess the maximal values reached by the counters during the computation is non-deterministically chosen in a preliminary part of the simulation. Only runs that initially guess a sufficient number of satellite nodes can successfully terminate the simulation. A satellite node moves from the $zero_i$ to the one_i state when the i-th counter is incremented, and a single node moves from the one_i back to the $zero_i$ state when the counter is decremented. For instance, the star in Figure 5 represents a configuration with control state q_0 and counters $c_1 = 3$ (with maximal value equals to 4), and $c_2 = 2$ (with maximal value equals to 2).

The simulation of instructions with zero-test is a more difficult task. The problem is that it is not possible to check the absence of neighbors with state one_i. Nevertheless, it is possible to ensure that no node is in the state one_i after a test for zero is executed. It is sufficient to use broadcast communication to move all the satellite nodes in the one_i state to a special *sink* state. If the simulation terminates exposing the final control state and no node is in the *sink* state (i.e. a configuration is reached in which all the nodes are in the final control state, in the $zero_i$, or the one_i state), we can conclude that the simulated computation is correct, thus also the corresponding Minsky machine terminates.

Note that the number of satellite nodes is not fixed a priori. However the graph have bounded path (the construction works for paths of length 3), so we can conclude what follows:

Theorem 6. TARGET *is undecidable for AHN restricted to K-bounded path configurations (with $K \geq 3$).*

As a corollary we now prove the undecidability of REPEAT-COVER. We need to slightly modify the way we model Minsky machines. The idea is to repeatedly simulate the computation of the Minsky machine, in such a way that the final control state can be exposed infinitely often if and only if the simulated Minsky machine terminates.

Every simulation phase simulates only a finite number of steps of the Minsky machine, and if the final control state is reached then a new simulation phase is started. This is achieved by including in the initial star topology also satellite nodes in the $free$ state, and ensuring that every simulated action moves one of those nodes to the *done* state. In this way, a simulation cannot perform more steps than the number of $free$ nodes in the initial star topology. If the final control state is reached, a new simulation is started by moving all the nodes from the *done* to the $free$ state, all the nodes from the one_i to the $zero_i$ state, and by restarting from the initial control state. Notice that nodes reaching the *sink* state (due to a wrong execution of a test for zero action) are no longer used in the computation. For this reason, as every time a wrong test for zero is executed some node moves in the *sink* state, we are sure that only finitely many wrong actions can occur. Hence, if the final control state is exposed infinitely often, we have that only finitely many simulation phases could be wrong, while infinitely many are correct. As all simulation phases reach the final control state (necessary to start the subsequent phase), we have that the corresponding Minsky machine terminates. Hence, we have the following Corollary of Theorem 6:

Corollary 3. REPEAT-COVER *is undecidable for AHN restricted to K-bounded path configurations (with $K \geq 3$).*

7 Conclusions

In this paper we have studied different types of verification problems for a formal model of Ad Hoc Networks in which communication is achieved via a selective type of broadcast. Perhaps surprisingly, a model with static topology turns out to be more difficult to analyze with respect to a model with spontaneous node movement. A similar dichotomy appears in verification of perfect and lossy channel systems. Studying the expressiveness of other variations on the semantics to model for instance conflicts, noise and lossiness, is an interesting research direction for future works.

Acknowledgments. The authors would like to thank Prof. Barbara Koenig for having pointed out to us the connection with bounded-depth processes, and Prof. Guoli Ding for useful insights into the theory of graph orderings.

References

1. Abdulla, P.A., Čerāns, C., Jonsson, B., Tsay, Y.-K.: General decidability theorems for infinite-state systems. In: LICS 1996, pp. 313–321 (1996)
2. Abdulla, P.A., Čerāns, C., Jonsson, B., Tsay, Y.-K.: Algorithmic analysis of programs with well quasi-ordered domains. Inf. Comput. 160(1-2), 109–127 (2000)

3. Abdulla, P.A., Jonsson, B.: Verifying programs with unreliable channels. Inf. Comput. 127(2), 91–101 (1996)
4. Delzanno, G., Sangnier, A., Zavattaro, G.: Parameterized verification of Ad Hoc Networks (Extended version). DISI-TR-10-01 (June 2010), http://www.disi.unige.it/index.php?research/techrep
5. Ding, G.: Subgraphs and well quasi ordering. J. of Graph Theory 16(5), 489–502 (1992)
6. Emerson, E.A., Namjoshi, K.S.: On model checking for non-deterministic infinite-state systems. In: LICS 1998, pp. 70–80 (1998)
7. Ene, C., Muntean, T.: A broadcast based calculus for Communicating Systems. In: IPDPS 2001, p. 149 (2001)
8. Esparza, J., Finkel, A., Mayr, R.: On the verification of Broadcast Protocols. In: LICS 1999, pp. 352–359 (1999)
9. Esparza, J., Nielsen, M.: Decidability issues for Petri nets - a survey. Bulletin of the EATCS 52, 245–262 (1994)
10. Esparza, J.: Some applications of Petri Nets to the analysis of parameterised systems. Talk at WISP 2003 (2003)
11. Fehnker, A., van Hoesel, L., Mader, A.: Modelling and verification of the LMAC protocol for wireless sensor networks. In: Davies, J., Gibbons, J. (eds.) IFM 2007. LNCS, vol. 4591, pp. 253–272. Springer, Heidelberg (2007)
12. Finkel, A., Schnoebelen, P.: Well-structured transition systems everywhere! TCS 256(1-2), 63–92 (2001)
13. Godskesen, J.C.: A calculus for Mobile Ad Hoc Networks. In: Murphy, A.L., Vitek, J. (eds.) COORDINATION 2007. LNCS, vol. 4467, pp. 132–150. Springer, Heidelberg (2007)
14. Merro, M.: An observational theory for Mobile Ad Hoc Networks. Inf. Comput. 207(2), 194–208 (2009)
15. Meyer, R.: On boundedness in depth in the pi-calculus. In: IFIP TCS 2008, pp. 477–489 (2008)
16. Mezzetti, N., Sangiorgi, D.: Towards a calculus for wireless systems. ENTCS 158, 331–353 (2006)
17. Milner, R.: Communicating and mobile systems: the pi-calculus. Cambridge Univ. Press, Cambridge (1999)
18. Minsky, M.: Computation: finite and infinite machines. Prentice Hall, Englewood Cliffs (1967)
19. Prasad, K.V.S.: A Calculus of Broadcasting Systems. Sci. of Comp. Prog. 25(2-3), 285–327 (1995)
20. Saksena, M., Wibling, O., Jonsson, B.: Graph grammar modeling and verification of Ad Hoc Routing Protocols. In: Ramakrishnan, C.R., Rehof, J. (eds.) TACAS 2008. LNCS, vol. 4963, pp. 18–32. Springer, Heidelberg (2008)
21. Singh, A., Ramakrishnan, C.R., Smolka, S.A.: A process calculus for Mobile Ad Hoc Networks. In: Lea, D., Zavattaro, G. (eds.) COORDINATION 2008. LNCS, vol. 5052, pp. 296–314. Springer, Heidelberg (2008)
22. Singh, A., Ramakrishnan, C.R., Smolka, S.A.: Query-Based model checking of Ad Hoc Network Protocols. In: Bravetti, M., Zavattaro, G. (eds.) CONCUR 2009. LNCS, vol. 5710, pp. 603–661. Springer, Heidelberg (2009)
23. Wies, T., Zufferey, D., Henzinger, T.A.: Forward analysis of depth-bounded processes. In: Ong, L. (ed.) FOSSACS 2010. LNCS, vol. 6014, pp. 94–108. Springer, Heidelberg (2010)