# On the Complexity of Parameterized Reachability in Reconfigurable Broadcast Networks

**Giorgio Delzanno[1], Arnaud Sangnier[2], Riccardo Traverso[1], and Gianluigi Zavattaro[4]**

1 **DIBRIS, University of Genova, Italy**
2 **LIAFA, Univ Paris Diderot, Sorbonne Paris Cité, CNRS, France**
4 **University of Bologna, INRIA - FOCUS Research Team, Italy**

## Abstract

We investigate the impact of dynamic topology reconfiguration on the complexity of verification problems for models of protocols with broadcast communication. We first consider reachability of a configuration with a given set of control states and show that parameterized verification is decidable with polynomial time complexity. We then move to richer queries and show how the complexity changes when considering properties with negation or cardinality constraints.

**1998 ACM Subject Classification** F.3 Logics and Meanings of Programs, F.1.1 Models of Computation

**Keywords and phrases** Broadcast Communication, Parameterized Verification, Complexity

## 1 Introduction

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 [4, 14, 16, 19, 20], this type of communication can naturally be specified in models of computation in which a network configuration is represented as a graph and in which individual nodes run an instance of a common protocol. A protocol typically specifies a sequence of control states in which a node can send a message (emitter role), wait for a message (receiver role), or perform an update of its internal state.

Already at this level of abstraction, verification of protocols with broadcast communication turns out to be a very difficult task. A formal account of this problem is given in [3, 4], where *parameterized control state reachability* is proved to be undecidable in an automata-based protocol model in which configurations are arbitrary graphs. The parameterized control state reachability problem consists in verifying the existence of an initial network configuration (with unknown size and topology) that may evolve into a configuration in which at least one node is in a given control state. If such a control state represents a protocol error, then this problem naturally expresses (the complement of) a safety verification task in a setting in which nodes have no information a priori about the size and connection topology of the underlying network.

In presence of non-deterministic reconfigurations of the network topology during an execution, parameterized control state reachability becomes decidable [3]. Reconfiguration models spontaneous node movement, i.e. each node can dynamically connect (resp. disconnect) to (resp. from) any other node in the network. Furthermore, it also models the dynamic addition (resp. removal) of nodes by means of connection to the network of a previously disconnected idle node (resp. the definitive disconnection of a previously connected node). The decidability proof in [3] does not give exact complexity bounds of the problem; it


Submitted to Fsttcs.
Editors: ...; pp. 1–12
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

simply gives a reduction to Petri net coverability, an ExpSpace-complete problem [13, 21]. The precise complexity of parameterized reachability was left as an open problem in [3].

In this paper we present a comprehensive analysis of the complexity of reachability problems for reconfigurable broadcast networks. We start by generalizing the problem by considering reachability queries defined over assertions that: (i) check the presence or absence of control states in a given configuration generated by some initial configuration, and (ii) cardinality queries that define lower and upper bounds for the number of occurrences of control states in a reachable configuration. In any case the problems require, at least in principle, the exploration of an infinite-state space. Indeed they are formulated for arbitrary initial configurations, and upper bounds to the number of processes per control state are not mandatory in case (ii). We then move to the analysis of the complexity of the considered problems by showing that reachability queries for constraints that only check for the presence of a control state can be checked in polynomial time. When considering both constraints for checking presence and absence of control states the problem turns out to be NP-complete. Finally, we show that the problem becomes PSpace-complete for cardinality queries.

**Related Work.** As mentioned in the introduction no precise complexity bounds were given for the parameterized control state reachability problem proved decidable in [3] via a reduction to Petri net marking coverability. In the present paper we attack this problem for different types of reachability queries. The interreducibility of control state reachability in models with dynamic reconfiguration, spontaneous mobility, and node-, message-, or link-failures has been formally studied in [5]. Based on the results in [5], the PTime-algorithm presented in the current paper can be applied not only to reconfigurable networks but also to a variety of protocols models with failures.

Symbolic backward exploration procedures for network protocols specified in graph rewriting have been presented in [11] (termination guaranteed for ring topologies) and [18] (approximations without termination guarantees). Decidability issues for broadcast communication in unstructured concurrent systems (or, equivalently, in fully connected networks) have been studied, e.g., in [8], whereas verification of unreliable communicating FIFO systems has been studied, e.g., in [1].

To our knowledge, exact algorithms (and relative complexity) for parameterized verification has not been studied in previous work on graph-based models of synchronous or asynchronous broadcast communication like [17, 19, 20, 16, 7, 9, 10, 14, 18, 11].

**Notes.** Sketches of the proofs are included in the body of the paper; detailed proofs are given in [6].

## 2 A Model for Reconfigurable Broadcast Networks

### 2.1 Syntax and semantics

Our model for reconfigurable broadcast networks is defined in two steps. We first define graphs used to denote network configurations and then define protocols running on each node. The label of a node denotes its current control state. Finally, we give a transition system for describing the interaction of a vicinity during the execution of the same protocol on each node.

▶ **Definition 2.1.** A $Q$-graph is a labeled *undirected graph* $\gamma = \langle V, E, L \rangle$, where $V$ is a finite set of *nodes*, $E \subseteq V \times V \setminus \{\langle v, v \rangle \mid v \in V\}$ is a finite set of *edges*, and $L$ is a labeling function from $V$ to a set of labels $Q$.

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

▶ **Definition 2.2.** A process is a tuple $\mathcal{P} = \langle Q, \Sigma, R, Q_0\rangle$, where $Q$ is a finite set of control states, $\Sigma$ is a finite alphabet, $R \subseteq Q \times (\{!!a, ??a \mid a \in \Sigma\}) \times Q$ is the transition relation, and $Q_0 \subseteq Q$ is a set of initial control states.

The label $!!a$ [resp. $??a$] represents the capability of broadcasting [resp. receiving] a message $a \in \Sigma$. For $q \in Q$ and $a \in \Sigma$, we define the set $R_a(q) = \{q' \in Q \mid \langle q, ??a, q'\rangle \in R\}$ which contains the states that can be reached from the state $q$ when receiving the message $a$. We assume that $R_a(q)$ is non empty for every $a$ and $q$, i.e. nodes always react to broadcast messages. Local transitions (denoted by the special label $\tau$) can be derived by using a special message $m_\tau$ such that $\langle q, ??m_\tau, q'\rangle \in R$ implies $q' = q$ for every $q, q' \in Q$ (i.e. receivers do not modify their local states).

Given a process $\mathcal{P} = \langle Q, \Sigma, R, Q_0\rangle$, in the corresponding Reconfigurable Broadcast Network (RBN) a configuration is a $Q$-graph and an initial configuration is a $Q_0$-graph. We use $\Gamma$ [resp. $\Gamma_0$] to denote the set of configurations [resp. initial configurations] associated to $\mathcal{P}$. Note that even if $Q_0$ is finite, there are infinitely many possible initial configurations (the number of $Q_0$-graphs). 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, which means that a broadcasted message is received by the nodes which are adjacent to the sender. Nondeterminism in reception is modeled by means of graph reconfigurations. We next formalize this intuition.

Given a process $\mathcal{P} = \langle Q, \Sigma, R, Q_0\rangle$, a reconfigurable broadcast network is defined by the transition system $RBN(\mathcal{P}) = \langle \Gamma, \rightarrow, \Gamma_0\rangle$ where the transition relation $\rightarrow \subseteq \Gamma \times \Gamma$ is such that: for $\gamma, \gamma' \in \Gamma$ with $\gamma = \langle V, E, L\rangle$, we have $\gamma \rightarrow \gamma'$ iff $\gamma' = \langle V, E', L'\rangle$ and one of the following conditions holds:

**Broadcast** $E' = E$ and $\exists v \in V$ s.t. $\langle L(v), !!a, L'(v)\rangle \in R$ and $L'(u) \in R_a(L(u))$ for every $u \sim v$, and $L(w) = L'(w)$ for any other node $w$.

**Graph reconfiguration** $E' \subseteq V \times V \setminus \{\langle v, v\rangle \mid v \in V\}$ and $L = L'$.

We use $\rightarrow^*$ to denote the reflexive and transitive closure of $\rightarrow$. RBN is an adequate formalism to abstractly represent broadcast communication with features like spontaneous mobility, node-, message- and link-failures.

## 2.2 Parameterized Reachability Problems

Given a process $\mathcal{P} = \langle Q, \Sigma, R, Q_0\rangle$, a *cardinality constraint* $\varphi$ over $\mathcal{P}$ is a formula which defines lower and upper bounds for the number of occurrences of each control state in a configuration. The formulae are defined by the following grammar, where $a \in \mathbb{N}$, $q \in Q$, and $b \in (\mathbb{N} \setminus \{0\}) \cup \{+\infty\}$:

$$\varphi ::= a \leq \#q < b \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi$$

We denote by CC the class of cardinality constraints, by CC[$\geq 1$] the class in which negation is forbidden and atomic proposition have only the form $\#q \geq 1$ (there exists at least one occurrence of $q$), and finally by CC[$\geq 1, = 0$] the class of cardinality constraints as in CC[$\geq 1$] but where atoms can also be of the form $\#q = 0$. Given a configuration $\gamma = \langle V, E, L\rangle$

of $\mathcal{P}$ and $q \in Q$, we denote by $\#\gamma(q)$ the number of vertices in $\gamma$ labeled by $q$, that is $\#\gamma(q) = |\{v \in V \mid L(v) = q\}|$. The satisfaction relation $\models$ for atomic formulas is defined as follows $\gamma \models a \leq \#q < b$ iff $a \leq \#\gamma(q) < b$. It is defined in the natural way for compound formulas.

We are now ready to state the cardinality reachability problem (CRP):

**Input:** A process $\mathcal{P}$ with $RBN(\mathcal{P}) = \langle \Gamma, \rightarrow, \Gamma_0 \rangle$ and a cardinality constraint $\varphi$.

**Output:** Yes, if $\exists \gamma_0 \in \Gamma_0$ and $\gamma_1 \in \Gamma$ s.t. $\gamma_0 \rightarrow^* \gamma_1$ and $\gamma_1 \models \varphi$; no, otherwise.

If the answer to this problem is yes, we will write $\mathcal{P} \models \Diamond\varphi$. Note that when dealing with the complexity of this problem we will suppose that the size of the input is the size of the process defined by the product of the number of states times the number of edges added to the size of the formula in which the integer values are encoded in unary.

We use the term *parameterized* to remark that the initial configuration is not fixed a priori. In fact, the only constraint that we put on the initial configuration is that the nodes have labels taken from $Q_0$ without any information on their number or connection links. As a special case we can define the control state reachability problems studied in [3] as the CRP for the simple constraint $\#q \geq 1$ (i.e. is there a reachable configuration in which the state $q$ is exposed?). Similarly, we can define the target reachability problem studied in [3] as an instance of CRP in which control states that must not occur in a target configuration are constrained by formulas like $\#q = 0$.

According to our semantics, the number of nodes stays constant in each execution starting from the same initial configuration. As a consequence, when fixing the initial configuration $\gamma_0$, we obtain finitely many possible reachable configurations. Thus, checking if there exists $\gamma_1$ reachable from a given $\gamma_0$ s.t. $\gamma_1 \models \varphi$ for a constraint $\varphi$ is a decidable problem. On the other hand, checking the parameterized version of the reachability problem is in general much more difficult. E.g. consider constraints of the form $\#q \geq 1$: CRP is undecidable for a semantics without non-deterministic graph reconfigurations [3]. In [3] it is also proved that CRP for the same class of constraints is decidable. However, the proposed decidability proof is based on a reduction to the problem of coverability in Petri nets which is known to be ExpSpace-complete [21, 13]. Since no lower-bound was provided, the precise complexity of CRP with simple constraints was left as an open problem that we close in this paper by showing that it is PTime-complete.

## 3 CRP restricted to constraints in CC[$\geq 1$]

In this section, we study CRP restricted to CC[$\geq 1$]. These constraints characterize configurations in which a given set of control states is present but they cannot express neither the absence of states nor the number of their occurrences. We first give a lower bound for this problem.

▶ **Proposition 3.1.** *CRP restricted to CC[$\geq 1$] is* PTime*-hard.*

*Sketch of proof.* The idea for the proof is based on a LogSpace-reduction from the Circuit Value Problem (CVP), which is know to be PTime-complete [12]. The protocol $\mathcal{P}$ built from the CVP instance has an initial state for each of the input variables which broadcasts its truth assignment, and another one for each gate of the input circuit. In the sub-protocol associated to individual gates, a process waits for messages representing inputs and then broadcasts messages representing outputs in such a way that CVP is satisfied iff $\mathcal{P} \models \Diamond\#ok \geq 1$, where $ok$ is a state reached only when the last gate produces the expected output. ◀

We now show that CRP restricted to CC[$\geq$ 1] is in PTIME. We first observe that, in order to decide if control state $q$ can be reached, we can focus our attention on initial configurations in which the topology is fully connected (i.e. graphs in which all pairs of nodes are connected). Indeed, graph reconfigurations can be applied to non-deterministically transform a topology into any other one.

Another key observation is that if the control state $q$ is reached once from the initial configuration $\gamma_0$, then it can be reached an arbitrary number of times by considering larger and larger initial configurations $\gamma_0'$. More specifically, the initial configuration $\gamma_0'$ is obtained by replicating several times the initial graph $\gamma_0$. The replicated parts are then connected in all possible ways (to obtain a fully connected topology). We can then use dynamic reconfiguration in order to mimic parallel executions of the original system and reach a configuration with several repeated occurrences of state $q$.

For what concerns constraints in CC[$\geq$ 1] this property of CRP avoids the need of counting the occurrences of states. We just have to remember which states can be generated by repeatedly applying process rules. As a consequence, in order to define a decision

**Algorithm 1** Computing the set of control states reachable in a RBN

```
Input: P = ⟨Q, Σ, R, Q_0⟩ a process
Output: S ⊆ Q the set of reachable control states in RBN(P)
  S := Q_0
  oldS := ∅
  while S ≠ oldS do
    oldS := S
    for all ⟨q_1, !!a, q_2⟩ ∈ R such that q_1 ∈ oldS do
      S := S ∪ {q_2} ∪ {q' ∈ Q | ⟨q, ??a, q'⟩ ∈ R ∧ q ∈ oldS}
    end for
  end while
```

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; (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.

We now formalize the previous observations. Let $\mathcal{P} = \langle Q, \Sigma, R, Q_0 \rangle$ be a process with $RBN(\mathcal{P}) = \langle \Gamma, \rightarrow, \Gamma_0 \rangle$ and let $\texttt{Reach}(\mathcal{P})$ be the set of reachable control states equals to $\{q \in Q \mid \exists \gamma \in \Gamma_0 . \exists \gamma' \in \Gamma.\text{ s.t. } \gamma \rightarrow^* \gamma' \text{ and } q \in L(\gamma')\}$. We will now prove that Algorithm 1 computes $\texttt{Reach}(\mathcal{P})$. Let $S$ be the result of the Algorithm 1 (note that this algorithm necessarily terminates because the **while**-loop is performed at most $|Q|$ times). We have then the following lemma.

▶ **Lemma 3.2.** *The two following properties hold:*
***(i)*** *There exist two configurations $\gamma_0 \in \Gamma_0$ and $\gamma \in \Gamma$ such that $\gamma_0 \rightarrow^* \gamma$ and $L(\gamma) = S$.*
***(ii)*** *$S = \texttt{Reach}(\mathcal{P})$.*

**Proof.** We first prove (i). We denote by $S_0, S_1, \ldots, S_n$ the content of $S$ after each iteration of the loop of the Algorithm 1. We recall that an undirected graph $\gamma = \langle V, E, L \rangle$ is complete if $\langle v, v' \rangle \in E$ for all $v, v' \in V$. We will now consider the following statement: for all $j \in \{0, \ldots, n\}$, for all $k \in \mathbb{N}$, there exists a complete graph $\gamma_{j,k} = \langle V, E, L \rangle$ in $\Gamma$ verifying the two following points:

1. $L(\gamma_{j,k}) = S_j$ and for each $q \in S_j$, the set $\{v \in V \mid L(v) = q\}$ has more than $k$ elements (i.e. for each element $q$ of $S_j$ there are more than $k$ nodes in $\gamma_{j,k}$ labeled with $q$),

2. there exits $\gamma_0 \in \Gamma_0$ such that $\gamma_0 \rightarrow^* \gamma_{j,k}$.

To prove this statement we reason by induction on $j$. First, for $j = 0$, the property is true, because for each $k \in \mathbb{N}$, the graph $\gamma_{0,k}$ corresponds to the complete graphs where each of the initial control states appears at least $k$ times. We now assume that the property is true for all naturals smaller than $j$ (with $j < n$) and we will show it is true for $j + 1$. We define $C_a$ as the set $\{\langle\langle q_1, !!a, q_2\rangle, \langle q, ??a, q'\rangle\rangle \in R \times R \mid q_1, q \in S_j\}$ and $M$ its cardinality. Let $k \in \mathbb{N}$ and let $N = k + 2 * k * M$. We consider the graph $\gamma_{j,N}$ where each control state present in $S_j$ appears at least $N$ times (such a graph exists by the induction hypothesis). From $\gamma_{j,N}$, we build the graph $\gamma_{j+1,k}$ obtained by repeating $k$ times the following operations:

- for each pair $\langle\langle q_1, !!a, q_2\rangle, \langle q, ??a, q'\rangle\rangle \in C_a$, select a node labeled by $q_1$ and one labeled by $q$ and update their label respectively to $q_2$ and $q'$ (this simulates a broadcast from the node labeled by $q_1$ received by the node labeled $q$ in the configuration in which all the other nodes have been disconnected thanks to the reconfiguration and reconnected after). Note that the two selected nodes can communicate because the graph is complete.

By applying these rules it is then clear that $\gamma_{j,N} \rightarrow^* \gamma_{j+1,k}$ and also that $\gamma_{j+1,k}$ verifies the property 1 of the statement. Since by induction hypothesis, we have that there exists $\gamma_0 \in \Gamma_0$ such that $\gamma_0 \rightarrow^* \gamma_{j,N}$, we also deduce that $\gamma_0 \rightarrow^* \gamma_{j+1,k}$, hence the property 2 of the statement also holds. From this we deduce that (i) is true.

To prove (ii), from (i) we have that $S \subseteq \texttt{Reach}(\mathcal{P})$ and we now prove that $\texttt{Reach}(\mathcal{P}) \subseteq S$. Let $q \in \texttt{Reach}(\mathcal{P})$. We show that $q \in S$ by induction on the minimal length of an execution path $\gamma_0 \rightarrow^* \gamma$ such that $\gamma_0 \in \Gamma_0$ and $q \in L(\gamma)$. If the length is 0 then $q \in Q_0$ hence also $q \in S$. Otherwise, let $\gamma' \rightarrow \gamma$ be the last transition of the execution. We have that there exists $q_1 \in L(\gamma')$ such that $\langle q_1, !!a, q\rangle \in R$ [or $q_1, q_2 \in L(\gamma')$ such that $\langle q_1, !!a, q_3\rangle, \langle q_2, ??a, q\rangle \in R$]. By induction hypothesis we have that $q_1 \in S$ [or $q_1, q_2 \in S$]. By construction, we can conclude that also $q \in S$. ◀

Since constraints in CC[$\geq 1$] check only the presence of states and do not contain negation, given a configuration $\gamma$ and a constraint $\varphi$ in CC[$\geq 1$] such that $\gamma \models \varphi$, we also have that $\gamma' \models \varphi$ for every $\gamma'$ such that $L(\gamma) \subseteq L(\gamma')$. Moreover, given a process $\mathcal{P}$, by definition of $\texttt{Reach}(\mathcal{P})$ we have that $L(\gamma) \subseteq \texttt{Reach}(\mathcal{P})$ for every reachable configuration $\gamma$, and by Lemma 3.2 there exists a reachable configuration $\gamma_f$ such that $L(\gamma_f) = \texttt{Reach}(\mathcal{P})$. Hence, to check $\mathcal{P} \models \Diamond\varphi$ it is sufficient to verify whether $\gamma_f \models \varphi$ for such a configuration $\gamma_f$. This can be done algorithmically as follows: once the set $\texttt{Reach}(\mathcal{P})$ is computed, check if the boolean formula obtained from $\varphi$ by replacing each atomic constraint of the form $\#q \geq 1$ by *true* if $q \in \texttt{Reach}(\mathcal{P})$ and by *false* otherwise is valid. This allows us to state the following theorem.

▶ **Theorem 3.3.** *CRP restricted to CC[$\geq 1$] is* PTIME*-complete.*

**Proof.** The lower bound is given by Proposition 3.1. To obtain the upper bound, it suffices to remark that the Algorithm 1 is in PTIME since it requires at most $|Q|$ iterations each one requiring at most $|R|^2$ look-ups (of active broadcast/receive transitions) for computing new states to be included, and also that evaluating the validity of a boolean formula can be done in polynomial time. ◀

## 4 CRP restricted to constraints in CC[$\geq 1, = 0$]

We consider now decidability and complexity of CRP for constraints in CC[$\geq 1, = 0$]. This kind of queries can be used to specify that a given control state is not present in a configuration (using atomic constraints of the form $\#q = 0$).

▶ **Proposition 4.1.** *CRP for constraints in CC[$\geq 1, = 0$] is* NP*-hard.*

*Sketch of proof.* The proof is based on a reduction of the boolean satisfiability problem (SAT), which is known to be NP-complete. The encoding of the SAT instance for a boolean formula $\Phi$ with variables in $V$ is based on a protocol with only local transitions from a single initial state into states that encode truth assignments in $\{v, \overline{v} \mid v \in V\}$. A CC[$\geq 1, = 0$] query is then built in order to guarantee that there are no contradicting assignments to variables. The query also ensures that the selected assignments satisfy the formula $\Phi$, where positive literals $v$ are replaced by $\#v \geq 1$ and negative literals $\neg v$ are replaced by $\#v = 0$. ◀

We will now give an algorithm in NP to solve CRP for constraints in CC[$\geq 1, = 0$]. As for Algorithm 1, this new algorithm works on sets of control states. The algorithm works in two main phases. In a first phase it generates an increasing sequence of sets of control states that can be reached in the considered process definition. At each step the algorithm adds the control states obtained from the application of the process rules to the current set of labels. Unlike the Algorithm 1, this new algorithm does not merge different branches, i.e. application of distinct rules may lead to different sequences of sets of control states. In a second phase the algorithm only removes control states applying again process rules in order to reach a set of control states that satisfies the given constraint.

**Algorithm 2** Solving CRP for constraints in CC[$\geq 1, = 0$]

```
Input:  P = ⟨Q, Σ, R, Q_0⟩ a process and φ a constraint over P in CC[≥ 1, = 0]
Output: Does P ⊨ ◊φ ?
  guess S_0, ..., S_m, T_1, ..., T_n ⊆ Q with m, n ≤ |Q|
  if S_0 ⊈ Q_0 then return false
  for all i ∈ {0, ..., m − 1} do
    if S_{i+1} ∉ postAdd(P, S_i) then return false
  end for
  T_0 = S_m
  for all i ∈ {0, ..., n − 1} do
    if T_{i+1} ∉ postDel(P, T_i) then return false
  end for
  If T_n satisfies φ then return true else return false
```

For a process $\mathcal{P} = \langle Q, \Sigma, R, Q_0 \rangle$ and a set $S \subseteq Q$, we define the operator $\texttt{postAdd}(\mathcal{P}, S) \subseteq 2^Q$ as follows: $S' \in \texttt{postAdd}(\mathcal{P}, S)$ if and only if the two following conditions are satisfied: (i) $S \subseteq S'$ and (ii) for all $q' \in S' \setminus S$, there exists a rule $\langle q, !!a, q' \rangle \in R$ such that $q \in S$ ($q'$ is produced by a broadcast) or there exist rules $\langle p, !!a, p' \rangle$ and $\langle q, ??a, q' \rangle \in R$ such that $q, p \in S$ and $p' \in S'$ ($q'$ is produced by a reception). In other words, all the states in $S' \in \texttt{postAdd}(\mathcal{P}, S)$ are either in $S$ or states obtained from the application of broadcast/reception rules to labels in $S$. Similarly, we define the operator $\texttt{postDel}(\mathcal{P}, S) \subseteq 2^Q$ as follows: $S' \in \texttt{postDel}(\mathcal{P}, S)$ if and only if $S' \subseteq S$ and one of the following conditions hold: either $S \setminus S' = \emptyset$ or $[S \setminus S' = \{q\}$ and there exists a rule $\langle q, !!a, q' \rangle \in R$ such that $q' \in S']$ or $[S \setminus S' = \{q\}$ and there exist two rules $\langle p, !!a, p' \rangle, \langle q, ??a, q' \rangle \in R$ such that $p, p', q' \in S'$ ($q$ is consumed by a broadcast)$]$ or $[S \setminus S' = \{p, q\}$ and there exist two rules $\langle p, !!a, p' \rangle, \langle q, ??a, q' \rangle \in R$ such that $p', q' \in S'$ ($p$ and $q$ are consumed by a broadcast)$]$.

Finally, we say that a set $S \subseteq Q$ satisfies an atom $\#q = 0$ if $q \notin S$ and it satisfies an atom $\#q \geq 1$ if $q \in S$; satisfiability for composite boolean formulae of CC[$\geq 1, = 0$] is then defined in the natural way. We have then the following Lemma.

▶ **Lemma 4.2.** *There is an execution of Algorithm 2 which answers YES on input $\mathcal{P}$ and $\varphi$ iff $\mathcal{P} \models \Diamond\varphi$.*

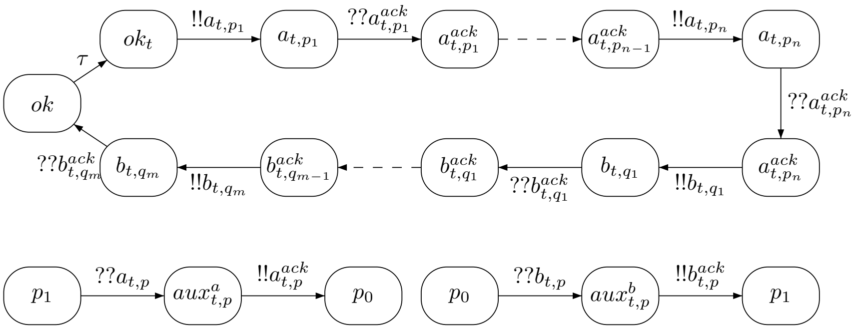


**Figure 1** Simulation of a transition $t$ with ${}^\bullet t = \{p_1, \dots, p_n\}$ and $t^\bullet = \{q_1, \dots, q_m\}$.

It is then clear that each check performed by the Algorithm 2 (i.e. $S_0 \subseteq Q_0$ and $S_{i+1} \in \texttt{postAdd}(\mathcal{P}, S_i)$ and $T_{i+1} \in \texttt{postAdd}(\mathcal{P}, T_i)$ and $T_n$ satisfies $\varphi$) can be performed in polynomial time in the size of the process $\mathcal{P}$ and of the formula $\varphi$ and since $m$ and $n$ are smaller than the number of control states in $\mathcal{P}$, we deduce the following theorem (the lower bound being given by Proposition 4.1).

▶ **Theorem 4.3.** *CRP for constraints in CC[$\geq 1, = 0$] is* NP*-complete.*

## 5 Complexity of CRP in Full CC

In this section we will show that CRP for the entire class of cardinality constraints CC is PSPACE-complete. First we prove the lower bound.

▶ **Proposition 5.1.** *CRP is* PSPACE*-hard.*

**Proof.** We use a reduction from reachability in 1-safe Petri nets. A Petri net $N$ is a tuple $N = \langle P, T, \vec{m_0} \rangle$, where $P$ is a finite set of places, $T$ is a finite set of transitions $t$, such that ${}^\bullet t$ and $t^\bullet$ are multisets of places (pre- and post-conditions of $t$), and $\vec{m_0}$ is a multiset of places that indicates how many tokens are located in each place in the initial net marking. Given a marking $\vec{m}$, the firing of a transition $t$ such that ${}^\bullet t \subseteq \vec{m}$ leads to a new marking $\vec{m'}$ obtained as $\vec{m'} = \vec{m} \setminus^\bullet t \cup t^\bullet$. A Petri net $P$ is 1-safe if in every reachable marking every place has at most one token. Reachability of a specific marking $\vec{m_1}$ from the initial marking $\vec{m_0}$ is decidable for Petri nets, and PSPACE-complete for 1-safe nets [2].

Given a 1-safe net $N = \langle P, T, \vec{m_0} \rangle$ and a marking $\vec{m_1}$, we encode the reachability problem as a CRP problem for the process $\mathcal{P}$ and cardinality constraint $\varphi$ defined next. For each place $p \in P$, we introduce control states $p_1$ and $p_0$ to denote the presence or absence of the token in $p$, respectively. Furthermore, we introduce a special control state *ok*. The control state is used to control the net simulation. Transitions of the controller are depicted in the upper part of Fig. 1. The first rule of the controller selects the current transition to simulate. The simulation of the transition $t$ with ${}^\bullet t = \{p_1, \dots, p_n\}$ and $t^\bullet = \{q_1, \dots, q_m\}$ is defined via two sequences of messages (we denote ${}^\bullet t$ and $t^\bullet$ as sets instead of multisets because we are considering a 1-safe net and it is hence not possible that a transition consumes or produces more than one token for each place). The first one is used to remove the token from $p_1, \dots, p_n$, whereas the second one is used to put the token in $q_1, \dots, q_m$. To guarantee that every involved place reacts to the protocol —i.e. messages are not lost— the controller waits for an acknowledgement from each of them. Transitions of places are depicted in the lower part of Fig. 1. It is not restrictive to assume that there is only one token in the initial marking $\vec{m_0}$ (otherwise we add an auxiliary initial place and a transition that generates $\vec{m_0}$ by consuming the initial token). Let $p^0$ be such a place. We define the initial states $Q_0$ of the process $\mathcal{P}$ as $\{p_1^0, ok\} \cup \{p_0 \mid p \in P \setminus \{p^0\}\}$, in order to initially admit control states representing the controller, the presence of the initial token, and the absence of tokens in other places. The reduction does not work if there are several copies of controller nodes and/or place representations (i.e. $p_1, p_0, \ldots$) interacting during a simulation (interferences between distinct nodes representing controllers/places may lead to incorrect results). However we can ensure that the reduction is accurate by checking the number of occurrences of states exposed in the final configuration: it is sufficient to check that only one controller and only one node per place in the net are present. Besides making this check, the cardinality constraint $\varphi$ should also verify that the represented net marking coincides with $\vec{m_1}$. Namely, we define $\varphi$ as follows:

$$\varphi = \bigwedge_{p \in \vec{m_1}, t \in T} \big(\#p_1 = 1 \wedge \#p_0 = 0 \wedge \#aux^a_{t,p} = 0 \wedge \#aux^b_{t,p} = 0\big) \wedge$$
$$\bigwedge_{q \notin \vec{m_1}, t \in T} \big(\#q_1 = 0 \wedge \#q_0 = 1 \wedge \#aux^a_{t,q} = 0 \wedge \#aux^b_{t,q} = 0\big) \wedge \#ok = 1 \wedge$$
$$\bigwedge_{t \in T} (\#ok_t = 0) \wedge \bigwedge_{t \in T, q \in P} \big(\#a_{t,q} = 0 \wedge \#b_{t,q} = 0 \wedge \#a^{ack}_{t,q} = 0 \wedge \#b^{ack}_{t,q} = 0\big)$$

Since the number of nodes stays constant during an execution, the post-condition specified by $\varphi$ is propagated back to the initial configuration. Therefore, if the protocol satisfies CRP for $\varphi$, then in the initial configuration there must be one single controller node with state $ok$, and for each place $p$ one single node with either state $p_1$ or state $p_0$. Under this assumption, it is easy to check that a run of the protocol corresponds precisely to a firing sequence in the 1-safe net. Thus an execution run satisfies $\varphi$ if and only if the corresponding firing sequence reaches the marking $\vec{m_1}$. ◀

We now show that there exists an algorithm to solve CRP in PSpace. The main idea is to use a symbolic representation of configurations in which the behavior of a network is observed exactly for a fixed number of nodes only. For all the other nodes, we only maintain the control state they are labeled with and not their precise number.

Without loss of generality, we consider for simplicity only processes with $Q_0 = \{q_0\}$, as multiple initial states can be encoded through local transitions from $q_0$. Given a process $\mathcal{P} = \langle Q, \Sigma, R, \{q_0\}\rangle$ and a cardinality constraint $\varphi$ over $\mathcal{P}$ we denote by $\mathrm{val}(\varphi) \in \mathbb{N}$ the largest natural constant that appears in $\varphi$. We then denote by $\mathrm{psize}(\varphi)$ the natural $|Q| * \mathrm{val}(\varphi)$. Intuitively $\mathrm{psize}(\varphi)$ is the number of witness nodes we keep track of: we reserve $\mathrm{val}(\varphi)$ processes to each control state that may appear in $\varphi$.

A symbolic configuration for $\mathcal{P}$ and $\varphi$ is then a pair $\theta = \langle v, S\rangle$ where $v \in Q^{\mathrm{psize}(\varphi)}$ is a vector of $\mathrm{psize}(\varphi)$ elements of $Q$ and $S \subseteq Q$. For $q \in Q$, we then write $\#v(q)$ to indicate the number of occurrences of $q$ in the vector $v$. Note that by definition $0 \leq \#v(q) \leq \mathrm{psize}(\varphi)$ for every $q \in Q$ and that $\sum_{q \in Q} \#v(q) = \mathrm{psize}(\varphi)$. This allows us to describe the set of configurations $[\![\theta]\!] \subseteq \Gamma$ characterized by a symbolic configuration $\theta = \langle v, S\rangle$ as follows: we have $\gamma \in [\![\theta]\!]$ if and only $\#\gamma(q) > \#v(q)$ for every $q \in S$ and $\#\gamma(q) = \#v(q)$ for every $q \in Q \setminus S$. Hence a symbolic configuration $\theta = \langle v, S\rangle$ represents all the configurations such that the number of occurrences of a control state $q$ is greater than the number of occurrences of $q$ in $v$ if $q \in S$, or equal when $q \notin S$. We will say that a symbolic configuration $\theta$ satisfies the cardinality constraint $\varphi$, written $\theta \models \varphi$, iff $\gamma \models \varphi$ for all $\gamma \in [\![\theta]\!]$. We use $\Theta$ to represent the set of symbolic configurations.

We make the following non restrictive assumptions: there is no constraint on the unique initial state $q_0$ in the cardinality constraints, the only outgoing transitions from the state $q_0$

are local transitions (labelled with $\tau$), in the symbolic configurations $\langle v, S\rangle$ we always have $q_0 \in S$, and the initial configuration $\theta_0$ is $\langle (q_0,\ldots,q_0), \{q_0\}\rangle$. The most important assumption is the first one about the absence of constraints on $q_0$: it is needed to guarantee the correctness of our symbolic procedure. For instance, consider a process $\mathcal{P} = \langle \{q_0\}, \Sigma, R, \{q_0\}\rangle$ and a cardinality constraint $\varphi$ of the form $1 \leq \#q_0 < 2$. We have then $\mathrm{psize}(\varphi) = 2$ and the symbolic configurations are of the form $\langle (q_0, q_0), S\rangle$. It is then obvious that all the symbolic configurations do not satisfy $\varphi$ while the initial concrete configuration with only one node does. The above assumptions are not restrictive because given a process $\mathcal{P} = \langle Q, \Sigma, R, \{q_0\}\rangle$ and a cardinality constraint $\varphi$, we can define a new process $\mathcal{P}' = \langle Q', \Sigma, R', \{q_{init}\}\rangle$ where $Q' = Q \cup \{q_{init}\}$ and $R' = R \cup \{\langle q_{init}, \tau, q_0\rangle\}$, i.e. $q_{init}$ is a new initial state from which the process is enabled to go to $q_0$ thanks to a local transition. As there is no constraint in $\varphi$ about $q_{init}$ it is immediate to prove the following Lemma:

▶ **Lemma 5.2.** *$\mathcal{P} \models \Diamond\varphi$ if and only if $\mathcal{P}' \models \Diamond\varphi$.*

We now define a relation on the symbolic configurations to represent the effect that process rules have on symbolic configurations. Let $\mathcal{P} = \langle Q, \Sigma, R, \{q_0\}\rangle$ be a process, $\varphi$ a cardinality constraint and $\theta$ the associated set of symbolic configurations. For each rule $r \in R$ of form $\langle q, !!a, q'\rangle$, we define the symbolic transition relation $\rightsquigarrow_r \subseteq \Theta \times \Theta$ as follows, we have $\langle v, S\rangle \rightsquigarrow_r \langle v', S'\rangle$ if and only if at least one of the two following conditions holds:

1. *(broadcast from a state in v)* there exists $i \in \{1, \ldots, \mathrm{psize}(\varphi)\}$ such that $v[i] = q$ and $v'[i] = q'$ (i.e. the sending process switches state according to $r$) and:
   - for all $j \in \{1, \ldots, \mathrm{psize}(\varphi)\} \setminus \{i\}$ we have either $v[j] = v'[j]$ or there exists $\langle q_r, ??a, q_r'\rangle \in R$ such that $v[j] = q_r$ and $v'[j] = q_r'$ (i.e. other processes in the pool may or may not react to the broadcast);
   - for each $q_s \in Q \setminus \{q_0\}$:
     - if $q_s \in S' \setminus S$ then there exists $q_s' \in S$ and $\langle q_s', ??a, q_s\rangle \in R$,
     - if $q_s \in S \setminus S'$ then there exists $q_s' \in S'$ and $\langle q_s, ??a, q_s'\rangle \in R$.
2. *(broadcast from a state in S)* we have $q \in S$ and $q' \in S'$ (note that we could have that $q \in S'$ or $q \notin S'$), and the following conditions hold:
   - for all $j \in \{1, \ldots, \mathrm{psize}(\varphi)\}$ we have either $v[j] = v'[j]$ or there exists $\langle q_r, ??a, q_r'\rangle \in R$ such that $v[j] = q_r \wedge v'[j] = q_r'$;
   - for each $q_s \in Q \setminus \{q, q'\}$, we have:
     - if $q_s \in S' \setminus S$ then there exists $\langle q_s', ??a, q_s\rangle \in R$ with $q_s' \in S$,
     - if $q_s \in S \setminus S'$ then there exists $\langle q_s, ??a, q_s'\rangle \in R$ with $q_s' \in S'$.

We denote by $\rightsquigarrow \subseteq \Theta \times \Theta$ the relation such that $\theta \rightsquigarrow \theta'$ if and only if there exists a rule $r \in R$ such that $\theta \rightsquigarrow_r \theta'$, and $\rightsquigarrow^*$ represents its reflexive and transitive closure. The intuition behind this construction is that we do not perform any abstraction on the states present in the vector $v$ but only on the states present in $S$, this because the states present in $v$ are used as witnesses to satisfy the cardinality constraint $\varphi$.

As an example, for $\mathrm{psize}(\varphi) = 5$, let $\langle (q_1, q_2, q_0, q_0, q_0), \{q_0, q_1, q_2\}\rangle$ be a symbolic configuration, and $\langle q_1, !!a, q_1'\rangle$ and $\langle q_2, ??a, q_2'\rangle$ be two transition rules. With a broadcast from a process in the vector we may reach, among others, $\langle (q_1', q_2, q_0, q_0, q_0), \{q_0, q_1, q_2\}\rangle$, $\langle (q_1', q_2', q_0, q_0, q_0), \{q_0, q_1, q_2, q_2'\}\rangle$, or $\langle (q_1', q_2', q_0, q_0, q_0), \{q_0, q_1, q_2'\}\rangle$, whereas a broadcast from a process in the set may lead to $\langle (q_1, q_2, q_0, q_0, q_0), \{q_0, q_1, q_2, q_1'\}\rangle$, $\langle (q_1, q_2, q_0, q_0, q_0), \{q_0, q_1, q_1', q_2'\}\rangle$, $\langle (q_1, q_2', q_0, q_0, q_0), \{q_0, q_1, q_2, q_1', q_2'\}\rangle$, or $\langle (q_1, q_2', q_0, q_0, q_0), \{q_0, q_1', q_2'\}\rangle$.

We will now prove that the symbolic configurations are well-suited to solve CRP. First, we show that if a symbolic configuration which satisfies $\varphi$ is reachable from the initial symbolic configuration, then there is a concrete configuration reachable from an initial configuration in $\gamma_0$ which also satisfies $\varphi$. This ensures a sound reasoning on symbolic configurations.

▶ **Lemma 5.3.** *If there exists $\theta \in \Theta$ such that $\theta_0 \rightsquigarrow^* \theta$ and $\theta \models \varphi$, then $\mathcal{P} \models \Diamond\varphi$.*

*Sketch of proof.* For a symbolic $\theta = \langle v, S\rangle$ in $\Theta$ and $N \in \mathbb{N}$, we denote by $[\![\theta]\!]_N = \{\gamma \in [\![\theta]\!] \mid \forall q \in S. \#\gamma(q) > (N + \#v(q))\}$, i.e. the set of configurations which belong to $[\![\theta]\!]$ in which for each $q \in S$, there are at least $N$ vertices (in addition to those already in the vector $v$). Note that with this definition $[\![\theta]\!]_0 = [\![\theta]\!]$. We then can prove the following property: given $\theta \in \Theta$ such that $\theta_0 \rightsquigarrow^* \theta$, there exists $N \in \mathbb{N}$ such that for all $\gamma \in [\![\theta]\!]_N$, there exists an initial configuration $\gamma_0 \in \Gamma_0$ such that $\gamma_0 \rightarrow^* \gamma$. To show that this property is true, we reason by induction on the length of the execution choosing the $N$ adequately at each step of the induction. Then if there exists $\theta \in \Theta$ such that $\theta_0 \rightsquigarrow^* \theta$ and $\theta \models \varphi$, then there exists $\gamma_0 \in \theta_0$ and $\gamma \in [\![\theta]\!]$ such that $\gamma_0 \rightarrow^* \gamma$, and by the definition of $\models$ for symbolic configuration we deduce also that $\gamma \models \varphi$. Hence $\mathcal{P} \models \Diamond\varphi$. ◀

We will now show that a reasoning on symbolic configurations leads to completeness, in other words that if there is a reachable configuration that satisfies the cardinality constraint $\varphi$, then there is a reachable symbolic configuration that satisfies $\varphi$.

▶ **Lemma 5.4.** *If $\mathcal{P} \models \Diamond\varphi$, then there exists $\theta \in \Theta$ such that $\theta_0 \rightsquigarrow^* \theta$ and $\theta \models \varphi$.*

*Sketch of proof.* In order to prove this Lemma, we need to introduce some auxiliary notations. Given a configuration $\gamma \in \Gamma$, we define $\uparrow_{q_0} \gamma$ as the set $\{\gamma' \in \Gamma \mid \forall q \in Q \setminus \{q_0\}. \#\gamma'(q) = \#\gamma(q)\}$. The above definition is needed because we could reach a configuration $\gamma$ which does not have enough processes to be represented by a symbolic configuration, but we can complete it by adding new vertices labelled by the initial state $q_0$ in order to solve the problem. We can then prove the following property by induction on the length of the concrete execution: for $\gamma_0 \in \Gamma_0$ and $\gamma \in \Gamma$ such that $\gamma_0 \rightarrow^* \gamma$, for all $\theta \in \Theta$ verifying $\uparrow_{q_0} \gamma \cap [\![\theta]\!] \neq \emptyset$, we have $\theta_0 \rightsquigarrow^* \theta$. Basically, this property stipulates that given a reachable configuration $\gamma$, each symbolic configuration $\theta$ whose semantics $[\![\theta]\!]$ contains $\gamma$ (modulo processes in state $q_0$) is also reachable.

The next step consists in proving that if $\gamma \in \Gamma$ is a configuration satisfying $\gamma \models \varphi$ then there exists $\theta \in \Theta$ such that $\uparrow_{q_0} \gamma \cap [\![\theta]\!] \neq \emptyset$ and $\theta \models \varphi$. This can be proved providing an algorithm that builds $\theta = \langle v, S\rangle$ such that, for each $q \in Q$, either the processes in state $q$ can be exactly represented within $v$ only when $\#\gamma(q) \leq \mathrm{val}(\varphi)$, or $\#v(q) = \mathrm{val}(\varphi)$ and $q \in S$ when $\#\gamma(q) > \mathrm{val}(\varphi)$ (i.e. $v$ is not large enough, recall that, apart for the states $q_0$ used to fill the "holes" in $v$, we reserve only up to $\mathrm{val}(\varphi)$ processes per state in $v$). Consider, e.g., a process with states $Q = \{q_0, q_1, q_2\}$, the formula $\varphi = 0 \leq \#q_1 < 3 \wedge 1 \leq \#q_2 < +\infty$ and the configuration with five processes $\gamma = \langle q_1, q_2, q_2, q_2, q_2\rangle$ such that $\gamma \models \varphi$. The symbolic configuration $\theta$ obtained is then $\langle (q_1, q_2, q_2, q_2, q_0, q_0, q_0, q_0, q_0), \{q_0, q_2\}\rangle$.

Since $\mathcal{P} \models \Diamond\varphi$, there exists an initial configuration $\gamma_0 \in \Gamma_0$ and a configuration $\gamma \in \Gamma$ such that $\gamma_0 \rightarrow^* \gamma$ and $\gamma \models \varphi$. By the second property we know there exists $\theta \in \Theta$ such that $\uparrow_{q_0} \gamma \cap [\![\theta]\!] \neq \emptyset$ and $\theta \models \varphi$, and the first property allows us to say that $\theta_0 \rightsquigarrow^* \theta$. ◀

We will now explain why CRP is in PSpace. The main idea is that we can reason on the graph of symbolic configurations. Note that by definition, since $\Theta = Q^{\mathrm{psize}(\varphi)} \times 2^Q$, the total number of symbolic configurations is $|\Theta| = |Q|^{\mathrm{psize}(\varphi)} * 2^{|Q|}$. Furthermore, checking whether a symbolic configuration satisfies a cardinality constraint can be done in PTime and checking whether two symbolic configurations belong to the symbolic transition relation $\rightsquigarrow$ can also be done in PTime. The PSpace algorithm (which is in reality an NPSpace algorithm) at each step guesses a new symbolic configuration, checks whether it is reachable from the previous guessed one and verifies whether it satisfies $\varphi$. When it encounters a symbolic configuration that satisfies $\varphi$, it returns that $\mathcal{P} \models \Diamond\varphi$. Note that this algorithm needs only to store the number of configurations it has seen until now, and when this number reaches $|Q|^{\text{psize}(\varphi)} * 2^{|Q|}$, it means that the algorithm have seen all the symbolic configurations. Hence to store this number and the current and next symbolic configurations, the algorithm needs polynomial space (a number smaller than $|Q|^{\text{psize}(\varphi)} * 2^{|Q|}$ can be stored into a counter which requires at most $\text{psize}(\varphi) * \log(|Q|) + |Q| \log(2)$ space). Finally, lemmas 5.3 and 5.4 ensure us that such an algorithm is sound and complete and from Proposition 5.1 we have also a lower bound for CRP. Hence we deduce the main result of this paper.

▶ **Theorem 5.5.** *CRP is* PSPACE*-complete.*

## References


1 Abdulla, P. A., Jonsson, B.: Verifying programs with unreliable channels. Inf. Comput. 127(2): 91–101 (1996)
2 Cheng, A., Esparza, J., Palsberg, J.: Complexity Results for 1-Safe Nets. Theor. Comput. Sci. 147(1&2): 117-136 (1995)
3 Delzanno, G., Sangnier, A., Zavattaro, G.: Parameterized verification of Ad Hoc Networks. CONCUR'10: 313–327
4 Delzanno, G., Sangnier, A., Zavattaro, G.: On the Power of Cliques in the Parameterized verification of Ad Hoc Networks. FOSSACS'11: 441–455
5 Delzanno, G., Sangnier, A., Zavattaro, G.: Verification of Ad Hoc Networks with Node and Communication Failures. FORTE'12: 235-250
6 Delzanno, G., Sangnier, A., Traverso, R., Zavattaro, G.: On the Complexity of Parameterized Reachability in Reconfigurable Broadcast Networks. Research Report hal-00740518, HAL, CNRS, France, 2012
7 Ene, C., Muntean, T.: A broadcast based calculus for Communicating Systems. IPDPS'01: 149
8 Esparza, J., Finkel, A., Mayr, R.: On the verification of Broadcast Protocols. LICS'99: 352–359
9 Godskesen, J.C.: A calculus for Mobile Ad Hoc Networks. COORDINATION'07: 132–150
10 S. Joshi, B. König: Applying the Graph Minor Theorem to the Verification of Graph Transformation Systems. CAV'08: 214-226
11 Ladner, R. E.: The circuit value problem is logspace complete for P. SIGACT News: 18–20 (1977)
12 Lipton R.J.: The Reachability Problem Requires Exponential Space. Department of Computer Science. Research Report. Yale University. (1976)
13 Merro, M.: An observational theory for Mobile Ad Hoc Networks. Inf. Comput. 207(2): 194–208 (2009)
14 Nanz, S., Hankin, C.: A Framework for security analysis of mobile wireless networks. TCS, 367(1–2):203-227 (2006)
15 Prasad, K.V.S.: A Calculus of Broadcasting Systems. SCP, 25(2–3): 285–327 (1995)
16 Saksena, M., Wibling, O., Jonsson, B.: Graph grammar modeling and verification of Ad Hoc Routing Protocols. TACAS'08: 18–32
17 Singh, A., Ramakrishnan, C. R., Smolka, S. A.: A process calculus for Mobile Ad Hoc Networks. COORDINATION'08: 296–314
18 Singh, A., Ramakrishnan, C. R., Smolka, S. A.: Query-Based model checking of Ad Hoc Network Protocols. CONCUR '09: 603–619
19 Rackoff C.: The Covering and Boundedness Problems for Vector Addition Systems. TCS, 6:223-231 (1978)