Analysis of Dynamic Process Networks

Kedar S. Namjoshi[1]* and Richard J. Trefler[2]**

[1] Bell Laboratories, Alcatel-Lucent kedar@research.bell-labs.com
[2] University of Waterloo trefler@cs.uwaterloo.ca

Abstract. We formulate a method to compute global invariants of dynamic process networks. In these networks, inter-process connectivity may be altered by an adversary at any point in time. Dynamic networks serve as models for ad-hoc and sensor-network protocols. The analysis combines elements of compositional reasoning, symmetry reduction, and abstraction. Together, they allow a small "cutoff" network to represent arbitrarily large networks. A compositional invariant computed on the small network generalizes to a parametric invariant of the shape "for all networks and all processes: property p holds of each process and its local neighborhood." We illustrate this method by showing how to compute useful invariants for a simple dining philosophers protocol, and the latest version of the ad-hoc routing protocol AODV (version 2).

1 Introduction

For communication protocols, model checking is typically applied to small network instances to detect errors. Full correctness requires analyzing networks of arbitrary size. This is, in general, an undecidable problem. Our work considers the question of analyzing the behavior of a *dynamic* process network. In a dynamic network, an adversary can modify the structure of the network at any time by adding or dropping nodes and connections. Dynamic process networks are practically relevant, as they may be used to model ad-hoc and sensor-network protocols, which usually operate under adversarial conditions.

The analysis question is mathematically challenging, as verification of a dynamic network requires showing correctness for an unbounded family of networks. Consider, say, a dynamic ring network. Starting with a two-node ring, dynamic changes result in rings of arbitrary size. It is interesting that showing parametric correctness (i.e., for all fixed rings) does not suffice. For instance,

* Supported, in part, by DARPA under agreement number FA8750-12-C-0166. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government.

** Supported in part by Natural Sciences and Engineering Research Council of Canada Discovery and Collaborative Research and Development Grants.

consider a valid state of a dining philosophers' protocol where eating philosophers are separated by a non-eating philosopher. Removing the middle node and collapsing the ring creates an invalid state with adjacent eating philosophers. Protocols on dynamic networks typically incorporate recovery actions to be taken on adjacency changes so as to avoid such erroneous outcomes.

In this work, we propose a new approach to this challenging verification question. Our analysis method combines elements of compositional reasoning, symmetry reduction and abstraction. To motivate these ingredients, consider that in a dynamic network, global invariants must be maintained in the face of adversarial, unpredictable changes. Hence, the coupling between process states is likely to be weak, which is just the situation where compositional analysis is most effective. In order to obtain a uniform invariant which applies to all nodes, it is necessary to abstract from differences between node neighborhoods (such as the number of neighbors). This is achieved by neighborhood abstraction, which induces local symmetries between nodes. Local symmetries suffice to ensure that compositional invariants are isomorphic across all nodes in a network family.

In a nutshell, the method works as follows: First, one picks a small representative instance R, and sets up a collection of local (i.e., neighborhood) symmetries between nodes from arbitrary instances and nodes of R. Next, a compositional invariant is calculated for R. Local symmetries guarantee that the compositional invariant for R generalizes appropriately to all instances. Dynamic changes are handled by showing that they do not violate the compositional invariant. For a property P that is preserved by the symmetries, it follows that P holds of all instances if it holds for R. The result is a universally quantified, inductive invariant, of the form "for all networks and all processes: property P holds of each process and its local neighborhood."

The method is only partly automated. The specification of symmetries is done manually, while the compositional calculation on R is automatic. As the symmetries concern only the neighborhood of a node, in our experience, it is not too difficult to determine a good set of symmetries. We illustrate this through two protocols: the first is a Dining Philosophers protocol, modified to operate over dynamic graphs; the second is a model of the AODV protocol used for routing in ad-hoc networks (we analyze the latest version, AODVv2 [2]).

Known methods for analyzing dynamic networks operate with sets of graphs of arbitrary size. Termination is ensured either through heuristics which recognize graph patterns [27] or through results from the theory of well quasi ordering (WQO) [13]. Our method is rather different, simpler to implement and, we believe, closer to intuition. Many interesting parametric verification questions are undecidable on general graphs and ad-hoc networks [14]. The AODV protocol has gone through several versions. Earlier analyses targeted AODV [1] and DYMO [27]. We analyze the latest version (AODV version 2), which is quite different from the original. In the course of constructing a formal model, we noticed a correctness issue, which has been acknowledged by the authors. Our proofs apply to the corrected protocol.

2 Dynamic Networks and Compositional Reasoning

We define the computational model of a dynamic process network, focusing on dynamic changes. We then define the rules to validate a proposed compositional invariant. Abstraction and symmetry properties are treated in Section 3.

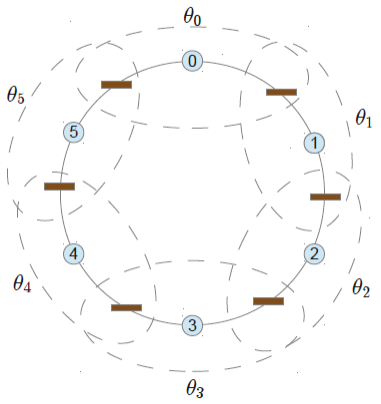

Fig. 1. A ring network. Nodes are represented by circles, edges by boxes, and links by lines connecting them. The neighborhood of a node is shown as a dashed ellipse.

We begin with an informal overview. A process network is defined by a process template and a way of instantiating the template on a communication graph. Every node of the graph is associated with a process. The transitions of a process depend on its internal state and the state on its adjacent edges. Hence, processes communicate by reading and modifying state on shared edges. In a dynamic setting, at any point in time, an adversary can modify the network by adding or deleting a node-edge link, or by adding or removing isolated nodes and edges. A process has the capability to react instantaneously and atomically to a change in the neighborhood of its node. In the following, we define these concepts precisely.

2.1 Networks, Assignment, and States

A *network*, G, is defined as the tuple (N, E, L), where N is a set of *nodes*, E is a set of *edges*, and $L \subseteq (N \times E) \cup (E \times N)$ is a set of node-edge *links*. (Note that an edge is not just a pair of nodes, but an element in its own right.) The network in Figure 1 shows a ring system. Each element has a color. Edge e is an *output edge* for node m if (m, e) is a link, and an input edge if (e, m) is a link. In either case, we say that e is adjacent to m. The *neighborhood* of a node

m, written $nbd(G, m)$, is formed by m together with its adjacent edges. Node m *points to* node n if an output edge of m is adjacent to n.

An *assignment* to a network specifies a domain of values for each edge, and a process for each node. A *process* assigned to a node n with k adjacent edges has the structure (S, S^0, T), where S is a set of states defined as the Cartesian product $S_I \times E_1 \times \ldots E_k$, where S_I is the internal state set, and $E_1, \ldots, E_k$ are the domains of the adjacent edges, S^0 is a set of initial states, and T is a transition relation, a subset of $S \times S$. We refer to several types of states:

- A *global state* is a vector formed by specifying an internal state for the process on each node and by a value from the domain of each edge
- A *local state* for node n with adjacent edges $e_1, \ldots, e_K$ is a vector formed by specifying an internal state for the process assigned to n, and by a value from the domain of each of the edges $e_1, \ldots, e_K$
- An *internal state* for node n is an internal state of the process assigned to n
- A *joint state* for nodes m, n is written as a pair $[a, b]$ where a is a local state for m, and b is a local state for n, and a and b have identical values for every edge e that is adjacent to both m and n. Joint states are used to formulate interference between processes.

For a global state s, we write $s[n]$ for the internal state of node n, $s[nbd(G, n)]$ for the local state of node n, and $s[e]$ for the value on edge e.

2.2 Semantics: static and dynamic

A network can transition from a global state s to global state t by the action of a single process, say at node n. This action may change at most the local state of node n, so T_n relates $s[nbd(G, n)]$ to $t[nbd(G, n)]$. Moreover, $t[e] = s[e]$ for every edge that is not adjacent to n, and $t[m] = s[m]$ for every node $m \neq n$. The set of *initial global states* is denoted I. For a family of networks, we assume that there is an initial global state for every network in the family. The projection of states in I on m, the set of *initial local states* for m, is denoted by I_m, which must be a subset of the initial states of that process. The global transition graph is induced by interleaving transitions from individual processes, starting from an initial state.

Adversarial actions change the underlying network graph. We consider the following actions:

- addition or removal of a link
- addition or removal of an isolated node (i.e., one without adjacent edges)
- addition or removal of an isolated edge (i.e., one without adjacent nodes)

To allow a process to react to a link addition or removal, we specify transitions $link(n, e)$ (link addition) and $unlink(n, e)$ (link removal) for node n and edge e. These transitions operate on the local state of node n after the change.

To account for graph (and hence, state vector) changes, we introduce the concept of a *configuration*, which is a pair (G, s), where G is the network graph

and s is a global state of G. In a static transition, the graph stays constant, so the corresponding change is from a configuration (G, s) to (G, t). For a dynamic change, the graph also changes as does the global state. This is detailed below for two representative changes, the other cases are similar.

- *Addition of a link* (n, e) *in a graph* $G = (N, E, L)$ *and state* s. The new configuration is (G', s'). The new network G' is given by $N' = N$, $E' = E$, and $L' = L \cup \{(n, e)\}$. The new state s' is the successor of s by the $link(n, e)$ transition at node n, taken in the configuration (G', s).
- *Addition of an isolated node* n *in a graph* $G = (N, E, L)$ *and state* s. The new configuration is (G', s'). The new network G' has node set $N' = N \cup \{n\}$, edge set $E' = E$ and links $L' = L$. The new state s' is such that $s'[x] = s[x]$ for each element (node or edge) of G, and $s'[n]$ is an initial state of the process assigned to n.

2.3 Inductive and Compositional Invariants

An *invariant* for a dynamic network is a set of configurations which includes all the configurations reachable from a set of initial configurations. This set is *inductive* if it is closed under all transitions. A *compositional invariant* is an inductive invariant of a special shape and satisfying a special set of constraints. It is formed as the conjunction of a number of local invariants, one for each process in a network. Thus, we use (G, n) to identify the process at node n in network G, and write $\theta(G, n)$ as the set of local states for (G, n) which form its local invariant. These sets (viewed equivalently as predicates) must satisfy the following conditions. For all (G, n):

- (Initiality) All initial states of the process at (G, n) are in $\theta(G, n)$
- (Step) $\theta(G, n)$ is closed under transitions of the process at (G, n)
- (Non-interference) If node m points to node n in G, for any joint state $[b, a]$ of (m, n) where $a \in \theta(G, n)$ and $b \in \theta(G, m)$: if $T_{(G,m)}$ transforms $[b, a]$ to $[b', a']$, then $a' \in \theta(G, n)$.

Remark 1. Notational conventions are inspired by [15]. Sets are represented as predicates. The notation $[\varphi]$ indicates that the formula φ is valid.

In addition, there are conditions to be met for dynamic changes. Let G be the original network and let G' be the network obtained by adding a (fresh) link between node n and edge e. Two constraints arise from link addition (similar constraints arise for other dynamic actions):

- θ is preserved by the action $link(n, e)$ at n. I.e., for any valuation a for $nbd(G, n)$ and any value v for edge e: if $a \in \theta(G, n)$ and $link(n, e)$ transforms (a, v) to (a', v'), then $(a', v') \in \theta(G', n)$.
- Interference due to the *link* action must preserve θ for any node m that is pointed to by n. I.e., for any joint state $[b, a]$ of (m, n) where $a \in \theta(G, n)$ and $b \in \theta(G, m)$, and any valuation v for edge e: if $link(n, e)$ transforms $([b, a], v)$ to $([b', a'], v')$, then $b' \in \theta(G', m)$.

Theorem 1 *If the compositional constraints hold, the assertion* $\xi = (\forall G, n : \theta(G, n))$ *is an inductive invariant of the dynamic network.*

Consider θ as a vector (or a map) from network-node pairs (G, n) to sets of local states. The set of vectors forms a complete lattice, with vectors ordered point-wise by the subset relation. The constraints form a set of simultaneous implications of the form $[F_{(G,n)}(\theta) \Rightarrow \theta(G, n)]$, where F is monotonic in θ. Hence, there is a least solution by the Knaster-Tarski theorem. The least solution to the first three constraints alone is the strongest "non-dynamic" compositional invariant, denoted Σ^*. The least solution to all constraints is the strongest dynamic compositional invariant, which we denote by Δ^*. Note that Δ^* is weaker than Σ^* as it must satisfy the non-dynamic constraints.

3 Symmetry Reduction

We show how to define and use localized neighborhood symmetries to reduce the computation of a compositional invariant to a small set of representative processes. This technique applies equally well to a single network or to a family of networks.

3.1 Fixed Networks

Fix a network G and a process assignment. The local symmetries of the network are defined as a relation B with entries of the form (m, β, n), where β is itself a relation between the local state spaces of processes m and n. Intuitively, β relates similar neighborhood states. The relation B should satisfy some structural properties:

- (Identity) For every m, there is an entry (m, β, m) in B. For every such entry, β is an equivalence relation
- (Symmetry) If $(m, \beta, n) \in B$ then $(n, \beta^{-1}, m) \in B$
- (Transitivity) if (m, β, n) and (n, γ, k) are in B, then $(n, \beta; \gamma, k)$ is in B

It follows that the *orbit relation* $m \sim n$, which holds if (m, β, n) is in B for some β, is an equivalence.

Definition 1 *Local Symmetry*

A relation B satisfying the structural conditions forms a local symmetry if, for every (m, β, n) in B, any step or interference transition of m can be simulated by a step or interference transition of n. More precisely, the following forward-simulation properties hold.

- (initial match) For every initial state x of the process at m, there is an initial state y of the process at n such that $(x, y) \in \beta$.

– (local simulation) If $(x, y) \in \beta$ and there is a transition (x, x') in T_m, then either there is y' such that (y, y') is in T_n and $(x', y') \in \beta$, or there is a neighbor j of n and a state b of j which is reachable by j-steps alone such that there is a joint (j, n) transition $([b, y], [b', y'])$ due to T_j and $(x', y') \in \beta$.
– (interference simulation) If $(x, y) \in \beta$, and i is a neighbor of m, and there is a joint (i, m) transition $([a, x], [a', x'])$ due to T_i, then either there is a neighbor j of n for which (i, γ, j) is in B and for every b such that $(a, b) \in \gamma$, there is a joint (j, n) transition $([b, y], [b', y'])$ due to T_j, such that $(x', y') \in \beta$, or there is a transition (y, y') in T_n such that $(x', y') \in \beta$.

Remark 2. We refer to this notion as a symmetry as it is the semantic form of a structural local symmetry definition formulated in [23]. The earlier formulation is in terms of network structure, as a *groupoid*, a set of tuples of the form (m, β, n), where β is an isomorphism on the graph neighborhoods of nodes m and n. While that is more obviously a structural symmetry, it is limiting since it does not allow defining symmetries up to an abstraction.

Definition 2 *For a relation R and a set Y, let $\langle R \rangle Y$ be the set of elements which have an R-successor in Y. I.e., $\langle R \rangle Y = \{x \mid xRy \text{ for some } y \text{ in } Y\}$.*

Theorem 2 ***(Symmetry Theorem)*** *Let B be a local symmetry on G. Let θ^* be the strongest compositional invariant on G. Then, for every m, n such that (m, β, n) is in B, it is the case that $[\theta^*_m \Rightarrow \langle \beta \rangle \theta^*_n]$.*

Proof: From the chaotic iteration theorem [9], every fair schedule of updates computes the least fixpoint. We use a schedule where the initialization is done first; then the schedule alternates a single transition step for all processes, and a single interference step for all processes. This proof applies to a non-dynamic network.

The proof is by induction on fixpoint stages. Assume that the statement is true for every state in θ^k. Now consider m, n such that (m, β, n) is in B and let x' be in θ^{k+1}_m but not in θ^k_m.
[Basis] If x' is an initial state of m, the claim holds by the initial match condition.
[Induction: Step] Suppose that x' is a successor of a state x in θ^k_m. By the inductive hypothesis, there is a state y in θ^*_n such that $x \beta y$. In the first case of local simulation, there is a transition (y, y') in T_n such that $x' \beta y'$. In that case, y' is also in θ^*_n by its closure under step transitions. In the other case, there is a neighbor j of n and a reachable state b of j such that there is a joint transition $([b, y], [b', y'])$ of T_j and $(x', y') \in \beta$. As b is reachable through step transitions alone, it must be in θ^*_j, so that $[b, y]$ is a joint state which satisfies θ^*_j and θ^*_n. By closure of θ^* under interference, the state y' is in θ^*_n.
[Induction: Interference] Now suppose that x' is obtained through interference. I.e., there is a transition T_i, for some neighbor i of m, from a joint state $[a, x]$ to joint state $[a', x']$, where $a \in \theta^k_i$ and x is in θ^k_m. By the inductive hypothesis, there is a state y in θ^*_n such that $x \beta y$. The first case of interference simulation ensures that there is a neighbor j of n such that (i, γ, j) is in B, and for all b such that $(a, b) \in \gamma$, there is a transition $([b, y], [b', y'])$ due to T_j where $(x', y') \in \beta$.

By the induction hypothesis, as $a \in \theta_i^k$, there is some $b \in \theta_j^*$ which is related to a by γ. For that b, the joint state $[b, y]$ satisfies θ_j^* and θ_n^* so, by closure under interference, y' is in θ_n^*. The other case of interference simulation ensures that there is a transition (y, y') in T_n such that $(x', y') \in \beta$; in that case, y' is in θ_n^* by closure under step transitions. **EndProof.**

Note that the claim holds also if single-step conditions are replaced with stuttering (i.e., where a single step is matched by a possibly empty path). We will use this relaxation for the examples.

Corollary 1 *Let P be a property of local states. If (m, β, n) is in symmetry B, and P is invariant under β, then $[\theta_m^* \Rightarrow P]$ if, and only if, $[\theta_n^* \Rightarrow P]$.*

Equivalence reduction. From the structural properties of B, the relation $\sim$ is an equivalence relation. Hence, in order to check a property that is invariant under all the β-relations in B, by Corollary 1, it suffices to compute the θ components for a representative of each equivalence class of $\sim$. Therefore, for a fixed network, the use of local symmetries can substantially reduce the number of fixpoint computations. (We show in an earlier work [23] that for a ring network with K nodes, it suffices to compute only one component of the fixpoint instead of all K.) Moreover, it suggests that, for a parameterized or dynamic network, symmetries that span members of a network family can be used to reduce the problem of checking a property for all instances to checking it for a small, fixed-size set of representative instances. We consider this next.

3.2 Parameterized and Dynamic Network Families

We apply the local symmetry definitions to a family of networks by redefining the symmetry relation to relate two nodes in (possibly) different networks. I.e., the relation consists of triples $((G, m), \beta, (H, n))$. We immediately obtain the analogues of Theorem 2 and Corollary 1 for a parametric network family.

Theorem 3 *Let B be a local symmetry on a parametric network family. Let θ^* be the strongest compositional invariant. Then, for every (G, m) and (H, n) such that $((G, m), \beta, (H, n))$ is in B, it is the case that $[\theta^*(G, m) \Rightarrow \langle\beta\rangle\theta^*(H, n)]$.*

Corollary 2 *Let P be a property of local states. If $((G, m), \beta, (H, n))$ is in symmetry B, and P is invariant under β, then $[\theta^*(G, m) \Rightarrow P]$ if, and only if, $[\theta^*(H, n) \Rightarrow P]$.*

An example of this reduction is as follows. Consider a family of ring networks $\{R_i\}$, where the process at each node is the same regardless of ring size. Then the relation β connects (R_k, m) with (R_l, n) precisely if the local state of m in the ring R_k is identical to the local state of n in ring R_l. For any property P which depends only on local states, the two nodes will satisfy P in the same way, by Corollary 2. Moreover, as any two nodes are connected by a local symmetry, all nodes in the family fall into a single equivalence class. Hence, it suffices

to compute a compositional invariant for a 2-node ring instance and check the property P for that instance in order to deduce that it holds for the entire family.

We would like to extend this form of reasoning to dynamic networks. In order to do so, we make the following assumption:

– (React) Any reaction to a dynamic change preserves the *non-dynamic* invariant. Formally, let Σ^* be the strongest *non-dynamic* compositional invariant. Consider a node (G, m) and a dynamic change at m or at one of its neighbors which changes the graph to G'. If, before the reaction, (G, m) and its neighbors have local states in Σ^*, then the local state of m after the reaction is in $\Sigma^*(G', m)$.

We believe that this is a reasonable assumption. A protocol designer must place a node in a "safe" local state after a dynamic change. It is reasonable to imagine that this safe state is one that is known to be a locally reachable state and, therefore, belongs to Σ^*. As we show in the following section, our example protocols satisfy this assumption.

Theorem 4 *Under the React assumption, the strongest compositional invariant for the non-dynamic and the dynamic systems are identical.*

Proof Sketch: Recall that Σ^* denotes the strongest non-dynamic compositional invariant, and Δ^* the strongest dynamic compositional invariant. As the initial configurations cover all graphs, $\Sigma^*(G, m)$ and $\Delta^*(G, m)$ are non-empty for all nodes (G, m).

The constraints defining validity of a compositional invariant for the dynamic case are an extension of those for the non-dynamic case. Hence, Δ^* is also a non-dynamic compositional invariant. Therefore, Σ^* is below Δ^* (point-wise).

We use React to show the other direction. The proof is by induction on the fixpoint stages. The claim is that, at stage k, Δ^k is below Σ^* for all components. This is true initially as Δ^0 is the set of initial states. Assume that it is true at stage k. The step and interference updates are common to both and, as the update function is monotonic, the hypothesis continues to hold at the next stage. In the dynamic setting, Δ^{k+1} is also updated as the result of reactions to network changes at node (G, m) or at one of its neighbors. By the induction hypotheses, the originating states are in Δ^k, and therefore in Σ^*. By the React assumption, the resulting local state for (G', m) is also in $\Sigma^*(G', m)$. Hence, the hypothesis continues to hold at stage $k + 1$. A newly added node begins at an initial state, which is already covered in Σ^*; a deleted node has no effect. **EndProof.**

This theorem lets us reduce the dynamic case to the non-dynamic, parameterized case. I.e., we can apply symmetry reductions as illustrated by the ring network example. We show in the following section how this applies to the two protocols we consider in this work.

4 Applications

In this section, we show how two dynamic protocols may be analyzed using the following procedure, based on the theory laid out in the previous sections:

1. Define a symmetry B for the entire network family, with finitely many equivalence classes
2. Find a representative network instance, R, whose nodes cover all of the equivalence classes
3. Compute the strongest non-dynamic compositional invariant for R.
4. Check that the React assumption is satisfied for all nodes in the family. In order to do so, it suffices to show that every reaction which results in a state of (G', m) is related to a state in the invariant of the representative instance.

Let property P be invariant under the symmetries in B. By Corollary 2, if P holds of $\Sigma^*(R, k)$ for all nodes k in R, it holds of all nodes in the entire family.

The first two steps are not automated, but one may consider some guidelines. If the network family has graphs with nodes of arbitrary degree, the symmetry relation must abstract from the degree of the node. A localized abstraction of the state space may be needed if the space is large or unbounded. The symmetry and React conditions could be checked with an SMT solver or a theorem prover. The fixpoint computation of Σ^* for the representative network can be done automatically if the local state space is finite.

4.1 Dynamic Dining Philosophers Protocol

A non-dynamic version which operates on arbitrary networks was analyzed in [24]; here, we consider modifications to respond to dynamic network changes.

The basic protocol. The protocol has a number of similar processes operating on an arbitrary network. Every edge on the network models a shared "fork". The edge between nodes i and j is called f_{ij} or, equivalently, f_{ji}. Its value is one of $\{i, j, \perp\}$. Node i is said to *own* the fork f_{ij} if $f_{ij} = i$; node j owns this fork if $f_{ij} = j$; and the fork is available if $f_{ij} = \perp$. The process at node i goes through the internal states T (thinking), H (hungry), E (eating), and R (release). Let $nbr(i, j, G)$ be a predicate true for nodes i, j in network G if the two nodes share an edge. The transitions for the process at node i are as follows.

- A transition from T to H is always enabled.
- In state H, the process acquires forks, but may also choose to release them
  - (acquire fork) if $nbr(i, j, G)$ and $f_{ij} = \perp$, set $f_{ij} := i$,
  - (release fork) if $nbr(i, j, G)$ and $f_{ij} = i$, set $f_{ij} := \perp$, and
  - (to-eat) if $(\forall j : nbr(i, j, G) : f_{ij} = i)$, then change state to E.
- A transition from E to R is always enabled.
- In state R, the process releases its owned forks.
  - (release fork) if $nbr(i, j, G)$ and $f_{ij} = i$, set $f_{ij} := \perp$
  - (to-think) if $(\forall j : nbr(i, j, G) : f_{ij} \neq i)$, change state to T

The initial state of the system is one where all processes are in state T and all forks are available (i.e., have value $\perp$). The desired safety property is that there is no reachable global state where neighboring processes are in the state E.

Responding to Dynamic Network Changes. The protocol is safe for a fixed network. It is, however, unsafe under dynamic changes. For instance, the addition of a fresh link changes the *nbr* relation, and can cause two processes both of which are in their eating (E) state to become adjacent, violating safety. To avoid this possibility, the *link* transition is defined so that if a process is eating, it moves back to hungry (H), without releasing its forks; otherwise, the state is unchanged. The removal of a link cannot violate safety; in fact, it may make a hungry process eligible to eat, if all remaining edges are owned by that process. Hence, the *unlink* transition keeps the local state unchanged. The initial state of a newly added node is thinking (T).

Symmetry Reduction We show (informally) how to carry out the steps described previously. Define the symmetry B with entries $((G, m), \beta, (H, n))$ for all G, H and nodes m in G and n in H, where β is defined as follows. Local states x of (G, m) and y of (H, n) are related by β if they have the same internal state (i.e., one of T, H, E, R), and node m owns all of its neighboring forks if, and only if, node n owns all of its neighboring forks. This meets the structural conditions on B, and there is a single equivalence class for the orbit relation (as every node is related to every other node).

We now sketch the check for simulation, which holds under stuttering. For instance, state x of (G, m) may be one where node m is hungry and has acquired 2 out of 4 neighboring forks, while state y of (H, n) is a state where node n is hungry and has acquired 6 out of 7 neighboring forks. An interference transition of m which results in it being granted one additional fork is matched by y (i.e., by a stuttering step), while an interference transition of n which results in it being granted the last remaining fork is matched by a sequence of two interference transitions from x.

The representative system consists of the smallest instance, a two-node ring. Its strongest compositional invariant can be calculated automatically. This asserts (cf. [24]) that, for each node, if the node is in state E, it owns all neighboring forks. By Theorem 3, this assertion holds for *all* nodes in the family.

Finally, we have to check the React assumption which allows a reduction from dynamic to parametric analysis. As outlined earlier in this section, we consider dynamic transitions where the origin state is related to the invariant of the representative system. For a link removal, the local state does not change. The interesting case is where the local state is E. Since the origin state of node m is related to the representative invariant, node m must own all forks. Removing one fork does not change this property, and the resulting state is also in the representative invariant. For link addition, the protocol moves to a non-E internal state, which trivially belongs to the representative invariant.

It follows by Theorem 4 that the same invariant holds for the dynamic system. It follows that neighboring nodes cannot be in state E, as that would imply (from the invariant) the existence of a shared edge which is owned by both nodes, a contradiction.

4.2 Analyzing The AODV Protocol

AODV is used to establish routes in a mobile ad-hoc network. The first version of this protocol [1] was analyzed in [6] with theorem proving, model checking and (manual) abstraction, in [10], using predicate abstraction on the global state (although this version omits sequence numbering, which is used to handle dynamic changes) and recently in [19], using process algebra. The property of interest is whether the protocol can be put into a state where there is a routing loop. We analyzed the latest version of AODV, called AODVv2 [2]. Enough has changed that these earlier proofs are no longer applicable to the new version. In the course of our modeling, we discovered an error in the AODVv2 protocol (in the handling of broken routes) and suggested a fix. The authors have acknowledged the error, and accepted the fix. The following analysis is for the corrected protocol.

The AODV protocol is used in a network where nodes are mobile, so that connectivity between two nodes may change at any time. The protocol is used to establish a route from a source node S to a target node T. The source floods the network with RREQs (route requests). When an RREQ reaches T, it responds with an RREP (route reply) message, which makes its way back to S through the request tree created during the RREQ flood. Unlike the earlier version, AODVv2 maintains two entries for each route: one pointing back to S, the other pointing back to T. Each entry has a next-hop field, a hop-count (more generally, a route cost), and a sequence number. The intuition is that higher sequence numbers indicate newer routes, while lower hop-counts indicate shorter routes. Thus, there is a natural way to compare routes. We say that a route x is "better than" a route y if $(seq_x, -hop_x)$ is lexicographically strictly greater than $(seq_y, -hop_y)$. I.e., if $seq_x > seq_y$ or if $seq_x = seq_y$ and $hop_x < hop_y$. In this situation, we also say that route y is "worse" than route x. We write this relationship as $y \prec x$.

The methodology developed so far considers a node as the focal point. One could as well consider a pair of adjacent nodes as the focal point, so that the invariant assertions have the form $\theta(G, (m, e, n))$ where m and n are adjacent in G with a single shared edge e. The concepts, compositional constraints, and theorems carry over in a straightforward way. For instance, a local state of (m, e, n) is interfered with by transitions at nodes which are adjacent to m and n.

As sequence numbers and hop-count are both unbounded, we need to compute compositional invariants under an abstraction. The natural abstraction, given the intuition, is to relate routes using the better-than relation, and to keep track of whether m is the next-hop of n. We define a relation β between $(G, (m, e, n))$ and $(H, (k, f, l))$ as follows. Local states x and y are related in β if, and only if, the internal states are the same, and corresponding pairs of route entries (e.g., those for (m, n) and (k, l); and for (m, e) and (k, f)) are related in the same way using the better-than relation, and next-hop relations are also comparable. For a small, 3 process network, the compositional invariant shows that $r_m \succeq r_e$, and if m is the next-hop for n, then $r_m \succ r_n$. (The notation r_i represents the route entry for node or edge i.)

We then check that the symmetry simulation conditions hold, limiting attention to states which are related by β to states in the compositional invariant for

the representative. The React conditions also hold, as the only reaction of the protocol to a deleted edge is to mark a route which follows that edge as being broken, and to send RERR (error) messages to adjacent nodes. As the reactions do not modify the actual route entries, they preserve the compositional invariant. From the symmetry theorem, the invariant for the representative network extends to the entire dynamic system.

5 Related Work and Conclusions

Compositional analysis has a long history and a large literature (cf. [11]); we refer here only to the most directly related work. The fixpoint definition used in this paper is a fully compositional (i.e., assume-guarantee) form of the Owicki-Gries method [25]. Parameterized verification is undecidable in general [5], but decidability results are known (cf. [18, 16]), and several semi-decision procedures have been developed. Typically, these approximate the infinite family of networks with finitary objects, for instance, the finite basis for upward-closed sets from well-quasi-orderings [3] and finite-state transducers [20]. These objects are, however, complex: mathematically and in terms of practical manipulation. A number of recent papers [26, 22–24, 4] have shown that the substantially simpler methods of localized analysis can be used to show parametric correctness for a number of non-trivial protocols.

The new contribution of this work is to provide a localized method to show the correctness of protocols which operate on dynamic networks. This requires two key steps. First, it is necessary to define new compositional constraints which correspond to actions taken by a protocol in response to dynamic changes. Second, as the set of possible networks is infinite, it is necessary to collapse the reasoning on to a representative network through the use of localized symmetries, induced by abstractions. The result of compositional analysis on the representative generalizes to a quantified inductive invariant which holds for the entire dynamic network family.

There are several other approaches to the analysis of dynamic and ad-hoc networks. The work in [7] shows that Hoare triples for restricted logics are decidable. Work in [14, 12] applies well-quasi-ordering (wqo) theory to ad-hoc networks, while the algorithm of [13] relies on symbolic forward exploration. Reasoning with these approaches is in terms of global states. The methods given here are localized in nature, which ensures simple representation and calculations, carried out with a small number of abstract processes. Also, as argued in the introduction, the dynamic nature of the changes contributes to the effectiveness of compositional reasoning.

The single abstract process view is also found in the network grammar method [28] and the environment abstraction method [8] for analysis of parametric protocols. In [21], the techniques in [17] are extended to dynamically changing systems represented with graph grammars. Despite the high-level similarities to these methods, our approach differs in being grounded in compositional analysis and in its ability to analyze dynamic changes.

In this work, we put forward a straightforward analysis framework for dynamic network protocols, and show that it suffices to construct correctness proofs for two non-trivial protocols, over an infinite set of possible networks. This strengthens the conjecture which originally inspired this work: that dynamic network protocols must be loosely coupled, and hence especially amenable to compositional analysis. The simplicity and naturalness of the symmetry relations for the two protocols lead us to believe that there is much scope for heuristic methods which automatically determine an appropriate symmetry relation.

References

1. Ad Hoc On-Demand Distance Vector (AODV) Routing. Internet Draft, IETF Mobile Ad hoc Networks Working Group.
2. Dynamic MANET On-demand (AODVv2) Routing. Internet Draft, IETF Mobile Ad hoc Networks Working Group, see `http://datatracker.ietf.org/doc/draft-ietf-manet-aodvv2/`.
3. P. A. Abdulla, K. Cerans, B. Jonsson, and Y.-K. Tsay. General decidability theorems for infinite-state systems. In *LICS*, pages 313–321. IEEE Computer Society, 1996.
4. P. A. Abdulla, F. Haziza, and L. Holík. All for the price of few. In *VMCAI*, volume 7737 of *Lecture Notes in Computer Science*, pages 476–495. Springer, 2013.
5. K. R. Apt and D. Kozen. Limits for automatic verification of finite-state concurrent systems. *Inf. Process. Lett.*, 22(6):307–309, 1986.
6. K. Bhargavan, D. Obradovic, and C. A. Gunter. Formal verification of standards for distance vector routing protocols. *J. ACM*, 49(4):538–576, 2002.
7. A. Bouajjani, Y. Jurski, and M. Sighireanu. A generic framework for reasoning about dynamic networks of infinite-state processes. In *TACAS*, volume 4424 of *Lecture Notes in Computer Science*, pages 690–705. Springer, 2007.
8. E. M. Clarke, M. Talupur, and H. Veith. Environment abstraction for parameterized verification. In *VMCAI*, volume 3855 of *LNCS*, pages 126–141, 2006.
9. P. Cousot and R. Cousot. Automatic synthesis of optimal invariant assertions: mathematical foundations. In *ACM Symposium on Artificial Intelligence & Programming Languages*, Rochester, NY, ACM SIGPLAN Not. 12(8):1–12, Aug. 1977.
10. S. Das and D. L. Dill. Counter-example based predicate discovery in predicate abstraction. In *FMCAD*, volume 2517 of *Lecture Notes in Computer Science*, pages 19–32, 2002.
11. W.-P. de Roever, F. de Boer, U. Hannemann, J. Hooman, Y. Lakhnech, M. Poel, and J. Zwiers. *Concurrency Verification: Introduction to Compositional and Noncompositional Proof Methods*. Cambridge University Press, 2001.
12. G. Delzanno, A. Sangnier, R. Traverso, and G. Zavattaro. On the complexity of parameterized reachability in reconfigurable broadcast networks. In *FSTTCS*, volume 18 of *LIPIcs*, pages 289–300. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.
13. G. Delzanno, A. Sangnier, and G. Zavattaro. Parameterized verification of safety properties in ad hoc network protocols. In *PACO*, volume 60 of *EPTCS*, pages 56–65, 2011.
14. G. Delzanno, A. Sangnier, and G. Zavattaro. Verification of ad hoc networks with node and communication failures. In *FMOODS/FORTE*, volume 7273 of *LNCS*, pages 235–250. Springer, 2012.

15. E. Dijkstra and C. Scholten. *Predicate Calculus and Program Semantics*. Springer Verlag, 1990.
16. E. Emerson and K. Namjoshi. Reasoning about rings. In *ACM Symposium on Principles of Programming Languages*, 1995.
17. E. A. Emerson, R. Trefler, and T. Wahl. Reducing model checking of the few to the one. In *ICFEM*, pages 94–113, 2006.
18. S. German and A. Sistla. Reasoning about systems with many processes. *Journal of the ACM*, 1992.
19. P. Höfner, R. J. van Glabbeek, W. L. Tan, M. Portmann, A. McIver, and A. Fehnker. A rigorous analysis of aodv and its variants. In *MSWiM*, pages 203–212. ACM, 2012.
20. Y. Kesten, O. Maler, M. Marcus, A. Pnueli, and E. Shahar. Symbolic model checking with rich ssertional languages. In *CAV*, volume 1254 of *LNCS*, pages 424–435, 1997.
21. Z. Langari and R. Trefler. Symmetry for the analysis of dynamic systems. In *NASA Formal Methods 2011*, pages 252–266, 2011.
22. K. S. Namjoshi. Symmetry and completeness in the analysis of parameterized systems. In *VMCAI*, volume 4349 of *LNCS*, pages 299–313, 2007.
23. K. S. Namjoshi and R. J. Trefler. Local symmetry and compositional verification. In *VMCAI*, volume 7148 of *LNCS*, pages 348–362, 2012.
24. K. S. Namjoshi and R. J. Trefler. Uncovering symmetries in irregular process networks. In *VMCAI*, LNCS 7737, pages 496–514, 2013.
25. S. S. Owicki and D. Gries. Verifying properties of parallel programs: An axiomatic approach. *Commun. ACM*, 19(5):279–285, 1976.
26. A. Pnueli, S. Ruah, and L. D. Zuck. Automatic deductive verification with invisible invariants. In *TACAS*, volume 2031 of *LNCS*, pages 82–97, 2001.
27. M. Saksena, O. Wibling, and B. Jonsson. Graph grammar modelling and verification of ad hoc routing protocols. LNCS, pages 18–32, 2008.
28. Z. Shtadler and O. Grumberg. Network grammars, communication behaviors and automatic verification. In *Automatic Verification Methods for Finite State Systems (1989)*, volume 407 of *LNCS*, pages 151–165, 1990.