Equational Reasoning on Ad Hoc Networks

Fatemeh Ghassemi[1], Wan Fokkink[2], and Ali Movaghar[1]

[1] Sharif University of Technology, Tehran, Iran,
[2] Vrije Universiteit, Amsterdam, The Netherlands
fghassemi@mehr.sharif.edu, wanf@cs.vu.nl, movaghar@sharif.edu

Abstract. We provide an equational theory for *Restricted Broadcast Process Theory* to reason about ad hoc networks. We exploit an extended algebra called *Computed Network Theory* to axiomatize restricted broadcast. It allows one to define an ad hoc network with respect to the underlying topologies. We give a sound and complete axiomatization for the recursion-free part of the term algebra *CNT*, modulo what we call rooted branching computed network bisimilarity.

1 Introduction

In Mobile Ad hoc Networks (MANETs), nodes communicate directly with each other using wireless transceivers (possibly along multihop paths) without the need for a fixed infrastructure. The primitive means of communication in MANETs is local broadcast; only nodes located in the range of a transmitter receive data. Thus nodes participate in a broadcast according to the underlying topology of nodes. On the other hand, nodes move arbitrarily, and the topology of the network changes dynamically. Local broadcast and topology changes are the main modeling challenges in MANETs.

We introduced *Restricted Broadcast Process Theory* (*RBPT*) in [6], to specify and verify ad hoc networks, taking into account mobility. *RBPT* specifies an ad hoc network by composing nodes using a restricted (local) broadcast operator, and it specifies a node by specifying a protocol deployed at a node using *RBPT* node notation. We modeled topology changes implicitly in the semantics, and thus verified a network with respect to different topology changes. An advantage of *RBPT* compared to similar algebras is that the specification of an ad hoc network does not include any specification about changes of underlying topologies. The behavior of an ad hoc network is equivalent to all its behaviors with respect to the possible topologies.

In this paper we provide an equational system to reason about *RBPT* terms. To provide equations for *RBPT* terms, we need to consider not only their observational behaviors, but also the set of topologies for which such behaviors are observed. To this aim, we first extend *RBPT* with new terms, called *Computed Network Theory* (*CNT*) because the extended terms contain a specification of a set of topologies and their observed behavior is computed with respect to those topologies. Network restrictions on the underlying topology are expressed explicitly in the syntax. The operational semantics of *CNT* is given by constrained labeled transition systems, in which the transitions are subscripted by a set of network restrictions. Our axiomatization borrows from the process algebra *ACP* [3] auxiliary (left merge and communication merge) operators to axiomatize the interleaving behavior of parallel composition.

F. Arbab and M. Sirjani (Eds.): FSEN 2009, LNCS 5961, pp. 113–128, 2010.

We consider an axiomatization of *CNT* modulo what we call *rooted branching computed network bisimilarity*. We prove that the axiomatization is sound, and complete for the recursion-free part of *CNT*. The application of our equational system is illustrated with a small running example.

Related works. Related calculi to ours are CBS#, CWS, CMAN, CMN, and the ω-calculus [11,10,8,9,12]. A complete comparison between ad hoc network algebras can be found in [6]. They are compared in terms of their specification and modeling concepts. In all related approaches, the only equations between networks were defined by using structural congruence. None of these papers provides a complete axiomatization for their algebra of ad hoc networks.

2 Restricted Broadcast Process Theory

Before going through the formal syntax definitions, we define some notations applied in these definitions. Let V denote a countably infinite set of variables ranged over by x, y, z, and D a finite set of data values ranged over by u. Let w range over $V \cup D$. We use $\widehat{w}$ to denote a finite sequence $w_1, \ldots, w_k$ for some $k \in \mathbb{N}$, $|\widehat{w}|$ its arity k, and $\{\widehat{w}/\widehat{x}\}$ for simultaneous substitutions $\{w_1/x_1\}, \ldots, \{w_k/x_k\}$. Let M denote a set of message types communicated over a network and ranged over by m, while $par : M \to \mathbb{N}$ defines the number of parameters encapsulated in a message m. For each message type m, there is a finite set $domain_m : \mathbb{P}(D^{par(m)})$ that defines the set of possible value assignments to the message parameters of m. Let Loc denote a finite set of logical addresses, ranged over by ℓ which models the hardware addresses of nodes at which protocols run. We will also use ℓ to denote a parameter of type Loc. Moreover, $A, B, C, \cdots$ denote concrete addresses. An unknown address is presented by $?$. The set of addresses extended with the unknown address is denoted as $Loc_?$, which by abuse of notation is also ranged over by ℓ.

Restricted Broadcast Process Theory (*RBPT*) [6] provides a two-level syntax to define a set of processes deployed at a node, also called protocols, and a set of ad hoc networks composed of singleton nodes:

$$P ::= 0 \mid \alpha.P \mid P + P \mid [w_1 = w_2]P, P \mid A(\widehat{w}) \,, A(\widehat{x}) \stackrel{def}{=} P$$
$$N ::= 0 \mid [\![P]\!]_\ell \mid N \parallel N \mid (\nu\ell)N$$

A protocol can be a deadlock, modeled by 0. $\alpha.P$ is a process that performs action α and then behaves as process P. The action α can be a send action $m(\widehat{u})!$ or a receive action $m(\widehat{x})?$. The process $P_1 + P_2$ behaves non-deterministically as process P_1 or P_2. The guarded command $[w_1 = w_2]P_1, P_2$ defines process behavior based on $w_1 = w_2$; if it evaluates to true, the protocol behaves as P_1, and otherwise as P_2. We write $A(\widehat{w})$ to denote a process invocation defined via a definition $A(\widehat{x}) \stackrel{def}{=} P$, with $|\widehat{x}| = |\widehat{w}|$, where $\widehat{x}$ consists of all names that appear free in P.

As a running example, $P(x) \stackrel{def}{=} req(x)!.P(x)$ denotes a process that broadcasts a message $req(x)$ ($par(req) = 1$ and $domain_{req} = \{0, 1\}$) recursively, and $Q \stackrel{def}{=} req(x)?.rep(x)!.Q$ a process that receives the message req and replies by sending

$rep(x)$ ($par(rep) = 1$ and $domain_{rep} = \{0, 1\}$) recursively. An ad hoc network can be composed of several nodes using the parallel composition operator, where each node is provided with a unique address ($\ell \neq ?$) and deploys a protocol, and nodes communicate via restricted broadcast. For instance, the network process $[\![P(0)]\!]_A \parallel [\![Q]\!]_B$ specifies an ad hoc network composed of two nodes with logical addresses A and B deploying processes $P(0)$ and Q respectively. Some address of a network can be hidden from an external observer using the restriction operator. For example, in $(\nu A)[\![P(0)]\!]_A \parallel [\![Q]\!]_B$ the activities of node A are hidden from the external observer, and only activities performed by B can be observed.

In the following section the syntax of ad hoc networks is extended with new terms, to obtain the class of what we call computed network terms. As the semantics of *RBPT* is subsumed by the one of *CNT*, we postpone an exposition on the formal semantics of *RBPT* until Section 4.

3 Computed Network Theory

As mentioned before, to give the axioms of the equational theory *RBPT*, we use an extension of *RBPT*, called *Computed Network Theory* (*CNT*). This process theory exploits network restrictions, which define a set of topologies; the behavior of process terms is computed with regard to such network restrictions.

We assume a binary relation $>$ on $Loc \times Loc_?$, which imposes connection relations between addresses. A relation $A > ?$ denotes that a node with logical address A should be in the range of there unknown address, while $A > B$ denotes a node with address A is connected to a node with address B. The relation $>$ need not be symmetric and transitive. By default each node is connected to itself: $\ell > \ell$. A *network restriction* is a set of relations $\ell > \ell'$. The network restriction $C[B/A]$ is obtained from the network restriction C by substituting B for A, and $C[B/g]$ is obtained from the network restriction C by simultaneous substitution of B for $\ell \in g$ where $g \subseteq Loc$.

A *topology* is a function $\gamma : Loc \to I\!P\, Loc$, where $\gamma(\ell)$ denotes the set of nodes connected to ℓ. This function models unidirectional connectivity between nodes. Each network restriction C is representative of the set of topologies that satisfy the relations in C. In particular, the empty network restriction $\{\}$ denotes all possible topologies.

CNT extends *RBPT* with new terms called computed networks, having the structure as $C\eta.\mathcal{N}$, to denote a network whose behavior, with respect to the set of topologies defined by network restriction C, is performing the action η and then behaving as $\mathcal{N}$. The parallel composition and restriction are defined over computed networks the same as *RBPT* terms. Besides *CNT* extends *RBPT* with new operators; choice ($+$), *left execution* ($\between$) and *sync* ($|$):

$$\mathcal{N} ::= 0 \mid [\![P]\!]_\ell \mid C\eta.\mathcal{N} \mid \mathcal{N} + \mathcal{N} \mid \mathcal{N} \parallel \mathcal{N} \mid \mathcal{N} \between \mathcal{N} \mid \mathcal{N}|\mathcal{N} \mid (\nu\ell)\mathcal{N}$$

where η can be $m(\widehat{u})!\{\ell\}$ or $m(\widehat{u})?$, and C is a network restriction. The operator $+$ defines a non-deterministic choice between *CNT* terms, and parallel composition defines computed networks communicating via restricted broadcast. The restriction operator $(\nu\ell)$ hides a node with address ℓ from an external observer as before. In left execution

⫦, the left operand must perform the initial action. In the sync operator |, both operands perform a synchronized initial action.

Bound addresses can be α-converted, meaning that $(\nu\ell)\mathcal{N}$ equals $(\nu\ell')\mathcal{N}[\ell'/\ell]$ if $\mathcal{N}$ does not contain ℓ' as a free address. We define functions $fl(\mathcal{N})$ and $bl(\mathcal{N})$ to denote sets of free and bound addresses in a computed network term $\mathcal{N}$, respectively. Parameters of receive actions like $Cm(\widehat{x})?.\mathcal{N}$ are bound names in $\mathcal{N}$ while parameters of send actions like $Cm(\widehat{x})!\{\ell\}.\mathcal{N}$ are free names in $\mathcal{N}$. A computed network term is called closed if its set of free names is empty.

4 Operational Semantics of *CNT*

The operational semantics of *CNT* is given at two levels (similar to the syntax), in terms of the operational semantics of protocols and of computed network processes.

Given a protocol, the operational rules of Table 1 induce a labeled transition system, in which the transitions are of the form $P \xrightarrow{\alpha} P'$ with $\alpha \in \{m(\widehat{u})?, m(\widehat{u})!\}$. They are standard operational rules for basic process algebras. (For explanations about the protocol operational rules, the reader is referred to [6].)

Table 1. Semantics of protocols

$$\frac{}{m(\widehat{x})?.P \xrightarrow{m(\widehat{u})?} P\{\widehat{u}/\widehat{x}\}} : Pre_1 \qquad \frac{}{m(\widehat{u})!.P \xrightarrow{m(\widehat{u})!} P} : Pre_2$$

$$\frac{P\{\widehat{u}/\widehat{x}\} \xrightarrow{\alpha} P'}{A(\widehat{u}) \xrightarrow{\alpha} P'} : Inv,\ A(x) \stackrel{def}{=} P \qquad \frac{P_1 \xrightarrow{\alpha} P_1'}{P_1 + P_2 \xrightarrow{\alpha} P_1'} : Choice$$

$$\frac{P_1 \xrightarrow{\alpha} P_1'}{[u = u]P_1, P_2 \xrightarrow{\alpha} P_1'} : Then \qquad \frac{P_2 \xrightarrow{\alpha} P_2'}{[u_1 = u_2]P_1, P_2 \xrightarrow{\alpha} P_2'} : Else,\ u_1 \neq u_2$$

Generally the behavior of a computed network is defined in terms of a set of topologies; a transition, in which a set of nodes participate in a communication, is possible for all topologies in which the receiving nodes are connected to the sending node. Therefore in the operational semantics it is defined for each state which transitions are possible for which sets of topologies (out of all possible topologies). Network restrictions are used to define the set of underlying topologies for each transition.

Given a computed network, the operational rules in Table 2 induce a constrained labeled transition system of transitions $\mathcal{N} \xrightarrow{\eta}_C \mathcal{N}'$, where C is a network restriction defining a set of topologies under which this transition is possible, and η can be a send or receive. The operational rules of computed networks are shown in Table 2. The symmetric counterparts of rules $Choice'$, Bro, $Sync_2$ and Par have been omitted. In this table $hide(C, \ell)$ denotes $\{\ell_1 > \ell_2 | \ell_1 > \ell_2 \in C[?/\ell] \wedge \ell_1 \neq ?\}$. Moreover, $\eta[\ell'/\ell]$ denotes η with all occurrences of ℓ replaced by ℓ'.

$Inter_1$ denotes that a single node can perform the send actions of a protocol at this node under any valid topology, and its network address is appended to this action. $Inter_2$ denotes a single node performing a receive action, under the restriction that

Table 2. Semantics of *CNT* terms

$$\frac{P \xrightarrow{m(\widehat{u})!} P'}{[\![P]\!]_\ell \xrightarrow{m(\widehat{u})!\{\ell\}}_{\{\}} [\![P']\!]_\ell} : Inter_1 \qquad \frac{P \xrightarrow{m(\widehat{u})?} P'}{[\![P]\!]_\ell \xrightarrow{m(\widehat{u})?}_{\{\ell>?\}} [\![P']\!]_\ell} : Inter_2$$

$$\frac{}{C\eta.\mathcal{N} \xrightarrow{\eta}_C \mathcal{N}} : Pre' \qquad \frac{\mathcal{N}_1 \xrightarrow{\eta}_C \mathcal{N}_1'}{\mathcal{N}_1 + \mathcal{N}_2 \xrightarrow{\eta}_C \mathcal{N}_1'} : Choice'$$

$$\frac{\mathcal{N} \xrightarrow{\eta}_C \mathcal{N}'}{\mathcal{N} \xrightarrow{\eta}_{C'} \mathcal{N}'} : \ Exe, \quad C \subseteq C' \qquad \frac{\mathcal{N}_1 \xrightarrow{m(\widehat{u})?}_{C_1} \mathcal{N}_1' \ \ \mathcal{N}_2 \xrightarrow{m(\widehat{u})?}_{C_2} \mathcal{N}_2'}{\mathcal{N}_1 \parallel \mathcal{N}_2 \xrightarrow{m(\widehat{u})?}_{C_1 \cup C_2} \mathcal{N}_1' \parallel \mathcal{N}_2'} : Recv$$

$$\frac{\mathcal{N}_1 \xrightarrow{m(\widehat{u})!\{\ell\}}_{C_1} \mathcal{N}_1' \ \ \mathcal{N}_2 \xrightarrow{m(\widehat{u})?}_{C_2} \mathcal{N}_2'}{\mathcal{N}_1 \parallel \mathcal{N}_2 \xrightarrow{m(\widehat{u})!\{\ell\}}_{C_1 \cup C_2[\ell/?]} \mathcal{N}_1' \parallel \mathcal{N}_2'} : Bro \qquad \frac{\mathcal{N}_1 \xrightarrow{\eta}_C \mathcal{N}_1'}{\mathcal{N}_1 \parallel \mathcal{N}_2 \xrightarrow{\eta}_C \mathcal{N}_1' \parallel \mathcal{N}_2} : Par$$

$$\frac{\mathcal{N}_1 \xrightarrow{m(\widehat{u})?}_{C_1} \mathcal{N}_1' \ \ \mathcal{N}_2 \xrightarrow{m(\widehat{u})?}_{C_2} \mathcal{N}_2'}{\mathcal{N}_1 \mid \mathcal{N}_2 \xrightarrow{m(\widehat{u})?}_{C_1 \cup C_2} \mathcal{N}_1' \parallel \mathcal{N}_2'} : Sync_1 \qquad \frac{\mathcal{N}_1 \xrightarrow{\eta}_C \mathcal{N}_1'}{\mathcal{N}_1 \between \mathcal{N}_2 \xrightarrow{\eta}_C \mathcal{N}_1' \parallel \mathcal{N}_2} : \ LExe$$

$$\frac{\mathcal{N}_1 \xrightarrow{m(\widehat{u})!\{\ell\}}_{C_1} \mathcal{N}_1' \ \ \mathcal{N}_2 \xrightarrow{m(\widehat{u})?}_{C_2} \mathcal{N}_2'}{\mathcal{N}_1 \mid \mathcal{N}_2 \xrightarrow{m(\widehat{u})!\{\ell\}}_{C_1 \cup C_2[\ell/?]} \mathcal{N}_1' \parallel \mathcal{N}_2'} : Sync_2 \qquad \frac{\mathcal{N} \xrightarrow{\eta}_C \mathcal{N}'}{(\nu\ell)\mathcal{N} \xrightarrow{\eta[?/\ell]}_{hide(C,\ell)} (\nu\ell)\mathcal{N}'} : Rest$$

the node must be connected to some sender (denoted by ?) is added to the network restriction. Pre' indicates execution of a prefix action. $Choice'$ defines that a computed network can behave non-deterministically. Exe indicates that if a transition is possible for C, then it is also possible for any more restrictive C'. $Recv$ allows to group together nodes that are ready to receive the same message. Bro indicates the actual synchronization in local broadcast among a transmitter and receivers. This transition is valid for all topologies in which the transmitter is connected (not essentially bidirectly) to the receivers, which is captured by $C_1 \cup C_2[\ell/?]$. The communication results in a transition labeled with $m(\widehat{u})!\{\ell\}$, so the message $m(\widehat{u})!$ remains visible to be received by other computed networks.

We consider a possible transition of the running example introduced in Section 2. This transition, given below, results from applications of $Inter_1$, $Inter_2$ and Bro:

$$[\![P(0)]\!]_A \parallel [\![Q]\!]_B \xrightarrow{req(0)!\{A\}}_{\{B>A\}} [\![P(0)]\!]_A \parallel [\![rep(0)!.Q]\!]_B$$

In this transition, node A broadcasts a message $req(0)$ and node B receives it, so that the parameter x is substituted by 0. This transition is possible for topologies in which B is connected to A, i.e. the accompanying network restriction is $\{B > A\}$.

As the sync operator defines synchronization between two computed networks, its behavior is defined by $Sync_1$ and $Sync_2$ indicating synchronization on a receive action (sent by the context) or a communication. $LExe$ defines that in a term composed by the left execution, the left computed network performs the initial action, and then the resulting term proceeds as in parallel composition. Par defines locality for a computed

network; an event in a computed network may result from this same event in a subnetwork.

Another possible transition of $[\![P(0)]\!]_A \parallel [\![Q]\!]_B$, resulting from an application of $Inter_1$ and Par, is:

$$[\![P(0)]\!]_A \parallel [\![Q]\!]_B \xrightarrow{req(0)!\{A\}}_{\{\}} [\![P(0)]\!]_A \parallel [\![Q]\!]_B$$

In this transition, node A sends but B does not participate in communication. This transition is possible for all possible topologies (so B may be connected to A, but it has lost the message), denoted by $\{\}$.

$Rest$ makes sure that restrictions over invisible addresses are removed and the address of a sender with hidden address is concealed from the external observer by converting its address to ?. By using network restrictions, we can easily define the set of topologies over visible nodes under which such a transition is possible (by removing restrictions imposed on hidden nodes).

In the running example, if we hide node A, then the possible transitions when A broadcasts (resulting from $Inter_{1,2}$, $Rest$, Bro or Par) are:

$$(\nu A)[\![P(0)]\!]_A \parallel [\![Q]\!]_B \xrightarrow{req(0)!\{?\}}_{\{B>?\}} (\nu A)[\![P(0)]\!]_A \parallel [\![rep(0)!.Q]\!]_B$$

$$(\nu A)[\![P(0)]\!]_A \parallel [\![Q]\!]_B \xrightarrow{req(0)!\{?\}}_{\{\}} (\nu A)[\![P(0)]\!]_A \parallel [\![Q]\!]_B.$$

Here the observer cannot see who has performed the send action.

5 Computed Network Bisimulation

We define the notion of computed network bisimilarity between nodes in a constrained labeled transition system, based on the notion of branching bisimilarity [13]. Our observer is distributed over locations of nodes with visible addresses equipped with a sensor to sense signals (and decrypt the signals in wireless communications). If the strength of a signal at a node is of a predefined threshold, it concludes that the node has performed a send action. If it cannot conclude the sender of the message, it will consider it as a send action with an unknown sender. To define our observational equivalence relation, we introduce the following notations:

- $\Rightarrow$ denotes the reflexive and transitive closure of receive actions which preserve topologies:
  - $\mathcal{N} \Rightarrow \mathcal{N}$;
  - if $\mathcal{N} \xrightarrow{m(\widehat{u})?}_{\{\}} \mathcal{N}'$ and $\mathcal{N}' \Rightarrow \mathcal{N}''$, then $\mathcal{N} \Rightarrow \mathcal{N}''$.
- $\xrightarrow{\overline{\eta}}_C$ denotes that either $\xrightarrow{\eta}_C$, or η is of the form $m(\widehat{u})!\{?\}$ and $\xrightarrow{\eta[\ell/?]}_{C[\ell/?]}$.

Definition 1. *A binary relation $\mathcal{R}$ on computed network terms is a branching computed network simulation, if $\mathcal{N}_1\mathcal{R}\mathcal{N}_2$ implies whenever $\mathcal{N}_1 \xrightarrow{\eta}_C \mathcal{N}_1'$:*

- *η is of the form $m(\widehat{u})?$, and $\mathcal{N}_1'\mathcal{R}\mathcal{N}_2$;*
- *or there are $\mathcal{N}_2'$ and $\mathcal{N}_2''$ such that $\mathcal{N}_2 \Rightarrow \mathcal{N}_2'' \xrightarrow{\overline{\eta}}_C \mathcal{N}_2'$, where $\mathcal{N}_1\mathcal{R}\mathcal{N}_2''$ and $\mathcal{N}_1'\mathcal{R}\mathcal{N}_2'$.*

$\mathcal{R}$ is a branching computed network bisimulation if $\mathcal{R}$ and $\mathcal{R}^{-1}$ are branching computed network simulations. Computed networks $\mathcal{N}_1$ and $\mathcal{N}_2$ are branching computed network bisimilar, written $\mathcal{N}_1 \simeq_b \mathcal{N}_2$, if $\mathcal{N}_1 \mathcal{R} \mathcal{N}_2$ for some branching computed network bisimulation relation $\mathcal{R}$.

Computed network bisimilarity is not a congruence with respect to the choice operator. To obtain a congruence, we need to add a root condition.

Definition 2. *Two computed networks $\mathcal{N}_1$ and $\mathcal{N}_2$ are* rooted branching computed network bisimilar*, written $\mathcal{N}_1 \simeq_{rb} \mathcal{N}_2$,*

- *if $\mathcal{N}_1 \xrightarrow{\eta}_C \mathcal{N}_1'$, then there is an $\mathcal{N}_2'$ such that $\mathcal{N}_2 \xrightarrow{\overline{\eta}}_C \mathcal{N}_2'$, and $\mathcal{N}_1' \simeq_b \mathcal{N}_2'$;*
- *if $\mathcal{N}_2 \xrightarrow{\eta}_C \mathcal{N}_2'$, then there is an $\mathcal{N}_1'$ such that $\mathcal{N}_1 \xrightarrow{\overline{\eta}}_C \mathcal{N}_1'$, and $\mathcal{N}_1' \simeq_b \mathcal{N}_2'$.*

We proved that branching computed network bisimilarity and rooted branching computed network bisimilarity are equivalence relations over computed networks. Moreover, the latter constitutes a congruence with respect to *CNT*. See Appendix A and B.

6 *CNT* Axiomatization

Our axiomatization for *CNT* terms is given in Table 3. P_{1-8} axiomatize protocols deployed at a node. In P_4, summation $\sum$ is used to denote a choice over a finite set, in this case $domain_m$; summation over an empty set denotes 0. *Dead* explains that hiding an address in a deadlock computed network has no affect. *Con* expresses that when a same behavior happens under two different sets of topologies, and if one set is included in another set, then from the point view of an external observer, the behavior occurs for the superset of topologies. Obs expresses when a send from a hidden action has no effect and can be equated to any send from a visible action. Cho_{1-4} define idempotency, commutativity, associativity and unit element for the choice operator. The parallel composition of two computed network is defined in an interleaving semantics the same as in the process algebra *ACP* [3] by the axiom *Br*; in a network composed of two computed networks $\mathcal{N}_1$ and $\mathcal{N}_2$, each network may perform a local action, or they may have communication via local broadcast. LEx_{1-3} define axioms for left execution; in left execution, the left operand performs an action (LEx_1), choice operator can be distributed over left execution (LEx_2), and when the left operand cannot do any action, then left execution results into a deadlock (LEx_3). S_1 and S_2 define commutativity and distributivity of choice over the sync operator, respectively. S_3 defines that when an argument in a sync composition is a deadlock, then the result of sync composition is a deadlock. $Sync_{1-5}$ define synchronization between two computed network terms. Generally speaking, two terms can be synchronized if they send/receive the same message with the same parameter values. T_1 and T_2 express when a receive action can be removed. Res_1 defines scope extrusion of the restriction operator. $Res_{2,4}$ define that the order and number of repeats of the restriction operator have no effect on the behavior of computed network terms. Res_3 defines distribution of restriction over the choice operator. Res_{5-7} express the effect of the restriction operator: network restrictions over hidden addresses are removed. In Res_5, restriction has no effect on send actions from visible addresses, except for removing restrictions over hidden addresses. In Res_6, the address of a hidden sender is converted to ?.

Table 3. Axiomatization of *CNT* terms

$[\![0]\!]_\ell = 0$	P_1	$[\![m(\widehat{u})!.P]\!]_\ell = \{\}m(\widehat{u})!\{\ell\}.[\![P]\!]_\ell$	P_2
$[\![m(\widehat{u})?.P]\!]_\ell = \{\ell >?\}m(u)?.[\![P]\!]_\ell$	P_3	$[\![m(\widehat{y})?.P]\!]_\ell = \sum_{\widehat{u}\in domain_m}[\![m(\widehat{u})?.P[\widehat{u}/\widehat{y}]]\!]_\ell$	P_4
$[\![P_1 + P_2]\!]_\ell = [\![P_1]\!]_\ell + [\![P_2]\!]_\ell$	P_5	$[\![A(\widehat{u})]\!]_\ell = [\![P[\widehat{u}/\widehat{x}]]\!]_\ell, A(\widehat{x}) \stackrel{def}{=} P$	P_6
$[\![[u = u]P_1, P_2]\!]_\ell = [\![P_1]\!]_\ell$	P_7	$[\![[u_1 = u_2]P_1, P_2]\!]_\ell = [\![P_2]\!]_\ell \quad (u_1 \neq u_2)$	P_8

$0 = (\nu\ell)0$	*Dead*
$C_1\eta.\mathcal{N} + C_2\eta.\mathcal{N} = C_1\eta.\mathcal{N} \quad (C_1 \subseteq C_2)$	*Con*
$Cm(\widehat{u})!\{?\}.\mathcal{N} + C[\ell/?]m(\widehat{u})!\{\ell\}.\mathcal{N} = C[\ell/?]m(\widehat{u})!\{\ell\}.\mathcal{N}$	*Obs*

$\mathcal{N} + \mathcal{N} = \mathcal{N}$	Cho_1	$\mathcal{N}_1 + (\mathcal{N}_2 + \mathcal{N}_3) = (\mathcal{N}_1 + \mathcal{N}_2) + \mathcal{N}_3$	Cho_3
$\mathcal{N}_1 + \mathcal{N}_2 = \mathcal{N}_2 + \mathcal{N}_1$	Cho_2	$\mathcal{N} + 0 = \mathcal{N}$	Cho_4

$\mathcal{N}_1 \parallel \mathcal{N}_2 = \mathcal{N}_1 \between \mathcal{N}_2 + \mathcal{N}_2 \between \mathcal{N}_1 + \mathcal{N}_1 \mid \mathcal{N}_2$	Br
$C\eta.\mathcal{N}_1 \between \mathcal{N}_2 = C\eta.(\mathcal{N}_1 \parallel \mathcal{N}_2)$	LEx_1
$(\mathcal{N}_1 + \mathcal{N}_2) \between \mathcal{N} = \mathcal{N}_1 \between \mathcal{N} + \mathcal{N}_2 \between \mathcal{N}$	LEx_2
$0 \between \mathcal{N} = 0$	LEx_3
$\mathcal{N}_1 \mid \mathcal{N}_2 = \mathcal{N}_2 \mid \mathcal{N}_1$	S_1
$(\mathcal{N}_1 + \mathcal{N}_2) \mid \mathcal{N} = \mathcal{N}_1 \mid \mathcal{N} + \mathcal{N}_2 \mid \mathcal{N}$	S_2
$0 \mid \mathcal{N} = 0$	S_3

$C_1m(\widehat{u})!\{\ell\}.\mathcal{N}_1 \mid C_2m(\widehat{u})?.\mathcal{N}_2 = C_1 \cup C_2[\ell/?]m(\widehat{u})!\{\ell\}.(\mathcal{N}_1 \parallel \mathcal{N}_2)$	$Sync_1$
$C_1m(\widehat{u}_1)!\{\ell\}.\mathcal{N}_1 \mid C_2n(\widehat{u}_2)?.\mathcal{N}_2 = 0 \quad (m \neq n \vee \widehat{u}_1 \neq \widehat{u}_2)$	$Sync_2$
$C_1m(\widehat{u})?.\mathcal{N}_1 \mid C_2m(\widehat{u})?.\mathcal{N}_2 = C_1 \cup C_2m(\widehat{u})?.(\mathcal{N}_1 \parallel \mathcal{N}_2)$	$Sync_3$
$C_1m(\widehat{u}_1)?.\mathcal{N}_1 \mid C_2n(\widehat{u}_2)?.\mathcal{N}_2 = 0 \quad (m \neq n \vee \widehat{u}_1 \neq \widehat{u}_2)$	$Sync_4$
$C_1m(\widehat{u}_1)!.\mathcal{N}_1\{\ell_1\} \mid C_2n(\widehat{u}_2)!\{\ell_2\}.\mathcal{N}_2 = 0$	$Sync_5$

$C\eta.(C'm(\widehat{u})?.\mathcal{N} + \mathcal{N}) = C\eta.\mathcal{N}$	T_1
$C\eta.(\{\}m(\widehat{u})?.(\mathcal{N}_1 + \mathcal{N}_2) + \mathcal{N}_2) = C\eta.(\mathcal{N}_1 + \mathcal{N}_2)$	T_2

$(\nu\ell)\mathcal{N}_1 \parallel \mathcal{N}_2 = (\nu\ell)(\mathcal{N}_1 \parallel \mathcal{N}_2) \quad (\ell \notin fl(\mathcal{N}_2))$	Res_1	$(\nu\ell_1)(\nu\ell_2)\mathcal{N} = (\nu\ell_2)(\nu\ell_1)\mathcal{N}$		Res_2
$(\nu\ell)(\mathcal{N}_1 + \mathcal{N}_2) = (\nu\ell)\mathcal{N}_1 + (\nu\ell)\mathcal{N}_2$	Res_3	$(\nu\ell)\mathcal{N} = \mathcal{N}$	$(\ell \notin fl(\mathcal{N}))$	Res_4

$(\nu\ell)Cm(\widehat{u})!\{\ell'\}.\mathcal{N} = hide(C, \ell)m(\widehat{u})!\{\ell'\}.(\nu\ell)\mathcal{N} \quad (\ell \neq \ell')$	Res_5
$(\nu\ell)\,Cm(\widehat{u})!\{\ell\}.\mathcal{N} = hide(C, \ell)m(\widehat{u})!\{?\}.(\nu\ell)\mathcal{N}$	Res_6
$(\nu\ell)\,Cm(\widehat{u})?.\mathcal{N} = hide(C, \ell)m(\widehat{u})?.(\nu\ell)\mathcal{N}$	Res_7

Theorem 1. *CNT is a sound axiomatization of the term algebra $\mathbb{IP}(CNT)/\simeq_{rb}$, i.e. for all closed computed network terms $\mathcal{N}_1$ and $\mathcal{N}_2$, if $\mathcal{N}_1 = \mathcal{N}_2$ then $\mathcal{N}_1 \simeq_{rb} \mathcal{N}_2$.*

Theorem 2. *CNT is a complete axiomatization for the recursion-free part of the term algebra $\mathbb{IP}(CNT)/\simeq_{rb}$, i.e. for all closed, recursion-free computed network terms $\mathcal{N}_1$ and $\mathcal{N}_2$, $\mathcal{N}_1 \simeq_{rb} \mathcal{N}_2$ implies $\mathcal{N}_1 = \mathcal{N}_2$.*

We prove this theorem in [7] using a restricted graph model which is isomorphic to the term algebra $\mathbb{IP}(CNT)/\simeq_{rb}$, following the approach of [4,13,1]. The basic idea in the completeness proof is to establish a graph rewriting system on restricted graphs, which is confluent and strongly normalizing (up to restricted graph isomorphism), and for

which every rewrite step preserves rooted branching graph bisimilarity. Then we prove that a rewrite step can be mapped to a proof step in *CNT*. By finding an identity relation between functions relating graphs and *CNT* terms, completeness can be concluded. The identity relation can be easily proved for basic terms; a basic term only consists of prefix and choice operators. Axioms in Table 3 allow us to bring all recursion-free closed *CNT* terms in basic terms.

We apply the axioms in Table 3 to the running example.

$$\llbracket P(0)\rrbracket_A = \{\}req(0)!\{A\}.\llbracket P(0)\rrbracket_A$$

$$\llbracket Q\rrbracket_B = \textstyle\sum_{i=0,1}\{B >?\}req(i)?.\llbracket rep(i)!.Q\rrbracket_B$$

$$\begin{aligned}
&\llbracket P(0)\rrbracket_A \parallel \llbracket Q\rrbracket_B = \llbracket P(0)\rrbracket_A \between \llbracket Q\rrbracket_B + \llbracket Q\rrbracket_B \between \llbracket P(0)\rrbracket_A + \llbracket P(0)\rrbracket_A \mid \llbracket Q\rrbracket_B\\
&= \{\}req(0)!\{A\}.\llbracket P(0)\rrbracket_A \between \llbracket Q\rrbracket_B + \textstyle\sum_{i=0,1}\{B >?\}req(i)?.\llbracket rep(i)!.Q\rrbracket_B \between \llbracket P(0)\rrbracket_A\\
&\quad + \{\}req(0)!\{A\}.\llbracket P(0)\rrbracket_A \mid \textstyle\sum_{i=0,1}\{B >?\}req(i)?.\llbracket rep(i)!.Q\rrbracket_B\\
&= \{\}req(0)!\{A\}.\llbracket P(0)\rrbracket_A \parallel \llbracket Q\rrbracket_B + \textstyle\sum_{i=0,1}\{B >?\}req(i)?.\llbracket P(0)\rrbracket_A \parallel \llbracket rep(i)!.Q\rrbracket_B\\
&\quad + \{B > A\}req(0)!\{A\}.\llbracket P(0)\rrbracket_A \parallel \llbracket rep(0)!.Q\rrbracket_B
\end{aligned}$$

indicating that the behavior can be: A can broadcast a message but B does not participate, or B can receive a message sent by its context, or A can broadcast a message and B receives it for a set of topologies in which B is connected to A.

Now let C be a hidden node with a behavior like A:

$$\begin{aligned}
&\llbracket P(0)\rrbracket_A \parallel \llbracket Q\rrbracket_B \parallel (\nu C)\llbracket P(0)\rrbracket_C = (\nu C)(\llbracket P(0)\rrbracket_A \parallel \llbracket Q\rrbracket_B \parallel \llbracket P(0)\rrbracket_C)\\
&= (\nu C)(\{\}req(0)!\{A\}.\llbracket P(0)\rrbracket_A \parallel \llbracket Q\rrbracket_B \parallel \llbracket P(0)\rrbracket_C\\
&\qquad + \{\}req(0)!\{C\}.\llbracket P(0)\rrbracket_A \parallel \llbracket Q\rrbracket_B \parallel \llbracket P(0)\rrbracket_C\\
&\qquad + \textstyle\sum_{i=0,1}\{B >?\}req(i)?.\llbracket P(0)\rrbracket_A \parallel \llbracket rep(i)!.Q\rrbracket_B \parallel \llbracket P(0)\rrbracket_C\\
&\qquad + \{B > A\}req(0)!\{A\}.\llbracket P(0)\rrbracket_A \parallel \llbracket rep(0)!.Q\rrbracket_B \parallel \llbracket P(0)\rrbracket_C\\
&\qquad + \{B > C\}req(0)!\{C\}.\llbracket P(0)\rrbracket_A \parallel \llbracket rep(0)!.Q\rrbracket_B \parallel \llbracket P(0)\rrbracket_C)\\
&= \{\}req(0)!\{A\}.(\nu C)(\llbracket P(0)\rrbracket_A \parallel \llbracket Q\rrbracket_B \parallel \llbracket P(0)\rrbracket_C)\\
&\quad + \{\}req(0)!\{?\}.(\nu C)(\llbracket P(0)\rrbracket_A \parallel \llbracket Q\rrbracket_B \parallel \llbracket P(0)\rrbracket_C)\\
&\quad + \textstyle\sum_{i=0,1}\{B >?\}req(i)?.(\nu C)(\llbracket P(0)\rrbracket_A \parallel \llbracket rep(i)!.Q\rrbracket_B \parallel \llbracket P(0)\rrbracket_C)\\
&\quad + \{B > A\}req(0)!\{A\}.(\nu C)(\llbracket P(0)\rrbracket_A \parallel \llbracket rep(0)!.Q\rrbracket_B \parallel \llbracket P(0)\rrbracket_C)\\
&\quad + \{B >?\}req(0)!\{?\}.(\nu C)(\llbracket P(0)\rrbracket_A \parallel \llbracket rep(0)!.Q\rrbracket_B \parallel \llbracket P(0)\rrbracket_C)\\
&= \{\}req(0)!\{A\}.(\nu C)(\llbracket P(0)\rrbracket_A \parallel \llbracket Q\rrbracket_B \parallel \llbracket P(0)\rrbracket_C)\\
&\quad + \textstyle\sum_{i=0,1}\{B >?\}req(i)?.(\nu C)(\llbracket P(0)\rrbracket_A \parallel \llbracket rep(i)!.Q\rrbracket_B \parallel \llbracket P(0)\rrbracket_C)\\
&\quad + \{B > A\}req(0)!\{A\}.(\nu C)(\llbracket P(0)\rrbracket_A \parallel \llbracket rep(0)!.Q\rrbracket_B \parallel \llbracket P(0)\rrbracket_C)
\end{aligned}$$

We can derive $\llbracket P(0)\rrbracket_A \parallel \llbracket Q\rrbracket_B = \llbracket P(0)\rrbracket_A \parallel \llbracket Q\rrbracket_B \parallel (\nu C)\llbracket P(0)\rrbracket_C$, as the following equality holds:

$$\llbracket P(0)\rrbracket_A \parallel \llbracket rep(i)!.Q\rrbracket_B \parallel (\nu C)\llbracket P(0)\rrbracket_C = \llbracket P(0)\rrbracket_A \parallel \llbracket rep(i)!.Q\rrbracket_B.$$

Now consider a protocol called $R(x)$, which can send the request x or receive a request. When it receives a request y, it either replies by sending the request y, or ignores it and waits until it receives that request again. The definition of this protocol is

$$R(x) \stackrel{def}{=} req(x)!.R(x) + req(x)!.S(x) + req(y)?.Z(x,y)$$
$$S(x) \stackrel{def}{=} req(x)!.S(x) + rep(x)!.R(x)$$
$$Z(x,y) \stackrel{def}{=} req(y)?.Z(x,y) + rep(y)!.R(x) + req(x)!.Z(x,y)$$

The behavior of a network consisting of a hidden node B, with protocol $R(0)$ deployed, is:

$$[\![Z(0,i)]\!]_B = \{B>?\}req(i)?.[\![Z(0,i)]\!]_B + \{\}rep(i)!\{B\}.[\![R(0)]\!]_B + \{\}req(0)!\{B\}.[\![Z(0,i)]\!]_B$$

$$\begin{aligned}(\nu B)[\![R(0)]\!]_B &= (\nu B)(\{\}req(0)!\{B\}.[\![R(0)]\!]_B + \{\}req(0)!\{B\}.[\![S(0)]\!]_B \\ &\quad + \textstyle\sum_{i=0,1}\{B>?\}req(i)?.[\![Z(0,i)]\!]_B) \\ &= \{\}req(0)!\{?\}.(\nu B)[\![R(0)]\!]_B + \{\}req(0)!\{?\}.[\![S(0)]\!]_B \\ &\quad + \textstyle\sum_{i=0,1}\{\}req(i)?.(\nu B)[\![Z(0,i)]\!]_B\end{aligned}$$

We can derive $(\nu A,B)[\![P(0)]\!]_A \parallel [\![Q]\!]_B = (\nu B)[\![R(0)]\!]_B$, as the following equalities hold:

$$(\nu A,B)([\![P(0)]\!]_A \parallel [\![rep(0)!.Q]\!]_B) = (\nu A,B)([\![S(0)]\!]_B)$$
$$\{\}req(i)?.(\nu A,B)([\![P(0)]\!]_A \parallel [\![rep(i)!.Q]\!]_B) = \{\}req(i)?.(\nu B)[\![Z(0,i)]\!]_B.$$

For instance, $\{\}req(i)?.(\nu A,B)([\![P(0)]\!]_A \| [\![rep(i)!.Q]\!]_B) = \{\}req(i)?.(\nu B)[\![Z(0,i)]\!]_B$ holds as:

$$\begin{aligned}&\{\}req(i)?.(\nu B)[\![Z(0,i)]\!]_B = \\ &\qquad \{\}req(i)?.(\{\}req(i)?.(\nu B)[\![Z(0,i)]\!]_B + \{\}rep(i)!\{?\}.(\nu B)[\![R(0)]\!]_B \\ &\qquad +\{\}req(0)!\{?\}.(\nu B)[\![Z(0,i)]\!]_B) = \\ &\qquad \{\}req(i)?.((\nu B)[\![Z(0,i)]\!]_B + \{\}rep(i)!\{?\}.(\nu B)[\![R(0)]\!]_B \\ &\qquad +\{\}req(0)!\{?\}.(\nu B)[\![Z(0,i)]\!]_B)\end{aligned}$$

$$\begin{aligned}&\{\}req(i)?.(\nu A,B)([\![P(0)]\!]_A \parallel [\![rep(i)!.Q]\!]_B) \\ &\qquad = \{\}req(i)?.(\{\}rep(i)!\{?\}.(\nu A,B)([\![P(0)]\!]_A \parallel [\![Q]\!]_B) \\ &\qquad +\{\}req(0)!\{?\}.(\nu A,B)([\![P(0)]\!]_A \parallel [\![rep(i)!.Q]\!]_B))\end{aligned}$$

Thus the distributed protocol deployed at nodes A and B is equal to the protocol deployed at node B alone. In other words, two hidden networks are equal if their communication capabilities are equal (proving when two recursive specifications are equal is out of scope of this paper).

7 Conclusion

We have extended Restricted Broadcast Process Theory with new operators to obtain Computed Network Theory, in which the behaviors are computed with respect to a set of topologies defined by a network restriction. Next we provided a sound and complete axiomatization of the recursion-free part of the term algebra of computed network theory, modulo the new notion of rooted branching computed network bisimilarity.

To deal with recursion, we are going to extend the axiomatization with the *Recursive Definition Principle*, the *Recursive Specification Principle*, and the *Cluster Fair Abstraction Rule* (see e.g. [5]). Applying our equational system to real-world case studies will be our next step.

References

1. Baeten, J.C.M., Bergstra, J.A., Reniers, M.A.: Discrete time process algebra with silent step. In: Proof, language, and interaction: essays in honour of Robin Milner, pp. 535–569. MIT Press, Cambridge (2000)
2. Basten, T.: Branching bisimilarity is an equivalence indeed! Inf. Process. Lett. 58(3), 141–147 (1996)
3. Bergstra, J.A., Klop, J.W.: Process algebra for synchronous communication. Information and Control 60(1-3), 109–137 (1984)
4. Bergstra, J.A., Klop, J.W.: Algebra of communicating processes with abstraction. Theoretical Computer Science 37, 21–77 (1985)
5. Fokkink, W.J.: Introduction to Process Algebra. Springer, Heidelberg (2000)
6. Ghassemi, F., Fokkink, W.J., Movaghar, A.: Restricted broadcast process theory. In: Cerone, A., Gruner, S. (eds.) Proc. 6th Conference on Software Engineering and Formal Methods (SEFM 2008), pp. 345–354. IEEE, Los Alamitos (2008)
7. Ghassemi, F., Fokkink, W.J., Movaghar, A.: Equational reasoning on ad hoc networks. Technical report, Sharif University of Technology (2009), `http://mehr.sharif.edu/~fghassemi/Technical%20Report.pdf`
8. Godskesen, J.C.: A calculus for mobile ad hoc networks. In: Murphy, A.L., Vitek, J. (eds.) COORDINATION 2007. LNCS, vol. 4467, pp. 132–150. Springer, Heidelberg (2007)
9. Merro, M.: An observational theory for mobile ad hoc networks. In: Proc. 23rd Conference on the Mathematical Foundations of Programming Semantics (MFPS XXIII). Electronic Notes in Theoretical Computer Science, vol. 173, pp. 275–293. Elsevier, Amsterdam (2007)
10. Mezzetti, N., Sangiorgi, D.: Towards a calculus for wireless systems. In: Proc. 22nd Annual Conference on Mathematical Foundations of Programming Semantics (MFPS XXII). Electronic Notes in Theoretical Computer Science, vol. 158, pp. 331–353. Elsevier, Amsterdam (2006)
11. Nanz, S., Hankin, C.: A framework for security analysis of mobile wireless networks. Theoretical Computer Science 367(1), 203–227 (2006)
12. Singh, A., Ramakrishnan, C.R., Smolka, S.A.: A process calculus for mobile ad hoc networks. In: Lea, D., Zavattaro, G. (eds.) COORDINATION 2008. LNCS, vol. 5052, pp. 296–314. Springer, Heidelberg (2008)
13. van Glabbeek, R.J., Weijland, W.P.: Branching time and abstraction in bisimulation semantics. Journal of the ACM 43(3), 555–600 (1996)

A Branching Computed Network Bisimilarity Is an Equivalence

To prove that branching computed network bisimilarity is an equivalence, we exploit semi-branching computed network bisimilarity, following [2]. In the next definition, $\mathcal{N} \xrightarrow{(\eta)}_C \mathcal{N}'$ denotes either $\mathcal{N} \xrightarrow{\eta}_C \mathcal{N}'$, or $\eta = m(\widehat{u})?$ and $\mathcal{N} = \mathcal{N}'$.

Definition 3. *A binary relation $\mathcal{R}$ on computed network terms is a semi-branching computed network simulation, if $\mathcal{N}_1 \mathcal{R} \mathcal{N}_2$ implies whenever $\mathcal{N}_1 \xrightarrow{\eta}_C \mathcal{N}_1'$:*

- *there are $\mathcal{N}_2'$ and $\mathcal{N}_2''$ such that $\mathcal{N}_2 \Rightarrow \mathcal{N}_2'' \xrightarrow{(\overline{\eta})}_C \mathcal{N}_2'$, where $\mathcal{N}_1 \mathcal{R} \mathcal{N}_2''$ and $\mathcal{N}_1' \mathcal{R} \mathcal{N}_2'$.*

$\mathcal{R}$ is a semi-branching computed network bisimulation if $\mathcal{R}$ and $\mathcal{R}^{-1}$ are semi-branching computed network simulations. Computed networks $\mathcal{N}_1$ and $\mathcal{N}_2$ are semi-branching computed network bisimilar if $\mathcal{N}_1 \mathcal{R} \mathcal{N}_2$, for some semi-branching computed network bisimulation relation $\mathcal{R}$.

Lemma 1. *Let $\mathcal{N}_1$ and $\mathcal{N}_2$ be computed network terms, and $\mathcal{R}$ a semi-branching computed network bisimulation such that $\mathcal{N}_1 \mathcal{R} \mathcal{N}_2$.*

- *If $\mathcal{N}_1 \Rightarrow \mathcal{N}_1'$ then $\exists \mathcal{N}_2' \cdot \mathcal{N}_2 \Rightarrow \mathcal{N}_2' \wedge \mathcal{N}_1' \mathcal{R} \mathcal{N}_2'$*
- *If $\mathcal{N}_2 \Rightarrow \mathcal{N}_2'$ then $\exists \mathcal{N}_1' \cdot \mathcal{N}_1 \Rightarrow \mathcal{N}_1' \wedge \mathcal{N}_1' \mathcal{R} \mathcal{N}_2'$*

Proof. We only give the proof of the first property. The proof is by induction on the number of $\Rightarrow$ steps from $\mathcal{N}_1$ to $\mathcal{N}_1'$:

- Base: Assume that the number of steps equals zero. Then $\mathcal{N}_1$ and $\mathcal{N}_1'$ must be equal. Since $\mathcal{N}_1 \mathcal{R} \mathcal{N}_2$ and $\mathcal{N}_2 \Rightarrow \mathcal{N}_2$, the property is satisfied.
- Induction step: Assume $\mathcal{N}_1 \Rightarrow \mathcal{N}_1'$ in n steps, for some $n \geq 1$. Then there is an $\mathcal{N}_1''$ such that $\mathcal{N}_1 \Rightarrow \mathcal{N}_1''$ in $n-1 \Rightarrow$ steps, and $\mathcal{N}_1'' \xrightarrow{m(\widehat{u})?}_{\{\}} \mathcal{N}_1'$. By the induction hypothesis, there exists an $\mathcal{N}_2''$ such that $\mathcal{N}_2 \Rightarrow \mathcal{N}_2''$ and $\mathcal{N}_1'' \mathcal{R} \mathcal{N}_2''$. Since $\mathcal{N}_1'' \xrightarrow{m(\widehat{u})?}_{\{\}} \mathcal{N}_1'$ and $\mathcal{R}$ is a semi-branching computed network bisimulation, there are two cases to consider:
  - there is an $\mathcal{N}_2'$ such that $\mathcal{N}_2'' \Rightarrow \mathcal{N}_2'$, $\mathcal{N}_1'' \mathcal{R} \mathcal{N}_2'$, and $\mathcal{N}_1' \mathcal{R} \mathcal{N}_2'$. So $\mathcal{N}_2 \Rightarrow \mathcal{N}_2'$ such that $\mathcal{N}_1' \mathcal{R} \mathcal{N}_2'$.
  - or there are $\mathcal{N}_2'''$ and $\mathcal{N}_2'$ such that $\mathcal{N}_2'' \Rightarrow \mathcal{N}_2''' \xrightarrow{m(\widehat{u})?}_{\{\}} \mathcal{N}_2'$, where $\mathcal{N}_1'' \mathcal{R} \mathcal{N}_2'''$ and $\mathcal{N}_1' \mathcal{R} \mathcal{N}_2'$. By definition, $\mathcal{N}_2''' \xrightarrow{m(\widehat{u})?}_{\{\}} \mathcal{N}_2'$ yields $\mathcal{N}_2''' \Rightarrow \mathcal{N}_2'$. Consequently $\mathcal{N}_2 \Rightarrow \mathcal{N}_2'$ such that $\mathcal{N}_1' \mathcal{R} \mathcal{N}_2'$. □

Proposition 2. *The relation composition of two semi-branching computed network bisimulations is again a semi-branching computed network bisimulation.*

Proof. Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be semi-branching computed network bisimulations with $\mathcal{N}_1 \mathcal{R}_1 \mathcal{N}_2$ and $\mathcal{N}_2 \mathcal{R}_2 \mathcal{N}_3$. Let $\mathcal{N}_1 \xrightarrow{\eta}_C \mathcal{N}_1'$. It must be shown that

$$\exists \mathcal{N}_3', \mathcal{N}_3'' : \mathcal{N}_3 \Rightarrow \mathcal{N}_3'' \xrightarrow{(\overline{\eta})} \mathcal{N}_3' \wedge \mathcal{N}_1 \mathcal{R}_1 \circ \mathcal{R}_2 \mathcal{N}_3'' \wedge \mathcal{N}_1' \mathcal{R}_1 \circ \mathcal{R}_2 \mathcal{N}_3'$$

Since $\mathcal{N}_1 \mathcal{R}_1 \mathcal{N}_2$, there exist $\mathcal{N}_2'$, $\mathcal{N}_2''$ such that $\mathcal{N}_2 \Rightarrow \mathcal{N}_2'' \xrightarrow{(\overline{\eta})}_C \mathcal{N}_2'$, $\mathcal{N}_1 \mathcal{R}_1 \mathcal{N}_2''$ and $\mathcal{N}_1' \mathcal{R}_1 \mathcal{N}_2'$. Since $\mathcal{N}_2 \mathcal{R}_2 \mathcal{N}_3$ and $\mathcal{N}_2 \Rightarrow \mathcal{N}_2''$, Lemma 1 yields that there is a $\mathcal{N}_3''$ such that $\mathcal{N}_3 \Rightarrow \mathcal{N}_3''$ and $\mathcal{N}_2'' \mathcal{R}_2 \mathcal{N}_3''$. Two cases can be distinguished:

- $\eta \in \{m(\widehat{u})?\}$ and $\mathcal{N}_2'' = \mathcal{N}_2'$. It follows immediately that $\mathcal{N}_3 \Rightarrow \mathcal{N}_3'' \xrightarrow{(\overline{\eta})}_C \mathcal{N}_3''$, $\mathcal{N}_1 \mathcal{R}_1 \circ \mathcal{R}_2 \mathcal{N}_3''$ and $\mathcal{N}_1' \mathcal{R}_1 \circ \mathcal{R}_2 \mathcal{N}_3''$.

– Assume $\mathcal{N}_2'' \xrightarrow{\overline{\eta}}_C \mathcal{N}_2'$. Since $\mathcal{N}_2'' \mathcal{R}_2 \mathcal{N}_3''$ and $\mathcal{R}_2$ is a semi-branching computed network bisimulation, there are $\mathcal{N}_3'''$ and $\mathcal{N}_3'$ such that $\mathcal{N}_3'' \Rightarrow \mathcal{N}_3''' \xrightarrow{(\overline{\eta})}_C \mathcal{N}_3'$, $\mathcal{N}_2'' \mathcal{R}_2 \mathcal{N}_3'''$ and $\mathcal{N}_2' \mathcal{R}_2 \mathcal{N}_3'$. Since $\mathcal{N}_3 \Rightarrow \mathcal{N}_3''$, we have $\mathcal{N}_3 \Rightarrow \mathcal{N}_3''' \xrightarrow{(\overline{\eta})}_C \mathcal{N}_3'$. Furthermore, $\mathcal{N}_1 \mathcal{R}_1 \mathcal{N}_2'' \mathcal{R}_2 \mathcal{N}_3'''$ and $\mathcal{N}_1' \mathcal{R}_1 \mathcal{N}_2' \mathcal{R}_2 \mathcal{N}_3'$. □

Corollary 3. *Semi-branching computed network bisimilarity is an equivalence relation.*

Proposition 4. *Each largest semi-branching computed network bisimulation is a branching computed network bisimulation.*

Proof. Suppose $\mathcal{R}$ is the largest semi-branching computed network bisimulation for some given constrained labeled transition systems. Let $\mathcal{N}_1 \mathcal{R} \mathcal{N}_2$, $\mathcal{N}_2 \Rightarrow \mathcal{N}_2'$, $\mathcal{N}_1 \mathcal{R} \mathcal{N}_2'$ and $\mathcal{N}_1' \mathcal{R} \mathcal{N}_2'$. We show that $\mathcal{R}' = \mathcal{R} \cup \{(\mathcal{N}_1', \mathcal{N}_2)\}$ is a semi-branching computed network bisimulation.

1. If $\mathcal{N}_1' \xrightarrow{\eta}_C \mathcal{N}_1''$, then it follows from $(\mathcal{N}_1', \mathcal{N}_2') \in \mathcal{R}$ that there are $\mathcal{N}_2'''$ and $\mathcal{N}_2''$ such that $\mathcal{N}_2' \Rightarrow \mathcal{N}_2''' \xrightarrow{(\overline{\eta})}_C \mathcal{N}_2''$ with $(\mathcal{N}_1', \mathcal{N}_2''')$, $(\mathcal{N}_1'', \mathcal{N}_2'') \in \mathcal{R}$. And $\mathcal{N}_2 \Rightarrow \mathcal{N}_2'$ yields $\mathcal{N}_2 \Rightarrow \mathcal{N}_2''' \xrightarrow{(\overline{\eta})}_C \mathcal{N}_2''$.
2. If $\mathcal{N}_2 \xrightarrow{\eta}_C \mathcal{N}_2''$, then it follows from $(\mathcal{N}_1, \mathcal{N}_2) \in \mathcal{R}$ that there are $\mathcal{N}_1'''$ and $\mathcal{N}_1''$ such that $\mathcal{N}_1 \Rightarrow \mathcal{N}_1''' \xrightarrow{(\overline{\eta})}_C \mathcal{N}_1''$ with $(\mathcal{N}_1''', \mathcal{N}_2)$, $(\mathcal{N}_1'', \mathcal{N}_2'') \in \mathcal{R}$. Since $(\mathcal{N}_1, \mathcal{N}_2') \in \mathcal{R}$ and $\mathcal{N}_1 \Rightarrow \mathcal{N}_1'''$, by Lemma 1, there is an $\mathcal{N}_2'''$ such that $\mathcal{N}_2' \Rightarrow \mathcal{N}_2'''$ and $(\mathcal{N}_1''', \mathcal{N}_2''') \in \mathcal{R}$. Since $\mathcal{N}_1''' \xrightarrow{(\overline{\eta})}_C \mathcal{N}_1''$, there are $\mathcal{N}_2^{**}$ and $\mathcal{N}_2^{*}$ such that $\mathcal{N}_2''' \Rightarrow \mathcal{N}_2^{**} \xrightarrow{(\overline{\eta})}_C \mathcal{N}_2^{*}$ with $(\mathcal{N}_1''', \mathcal{N}_2^{**})$, $(\mathcal{N}_1'', \mathcal{N}_2^{*}) \in \mathcal{R}$. Since $\mathcal{N}_2' \Rightarrow \mathcal{N}_2'''$ and $\mathcal{N}_2''' \Rightarrow \mathcal{N}_2^{**}$, we have $\mathcal{N}_2' \Rightarrow \mathcal{N}_2^{**}$. By assumption, $(\mathcal{N}_1', \mathcal{N}_2') \in \mathcal{R}$, so by Lemma 1 there is an $\mathcal{N}_1^{**}$ such that $\mathcal{N}_1' \Rightarrow \mathcal{N}_1^{**}$ and $(\mathcal{N}_1^{**}, \mathcal{N}_2^{**}) \in \mathcal{R}$. Since $\mathcal{N}_2^{**} \xrightarrow{(\overline{\eta})}_C \mathcal{N}_2^{*}$, there are $\mathcal{N}_1^{***}$ and $\mathcal{N}_1^{*}$ such that $\mathcal{N}_1^{**} \Rightarrow \mathcal{N}_1^{***} \xrightarrow{(\overline{\eta})}_C \mathcal{N}_1^{*}$ with $(\mathcal{N}_1^{***}, \mathcal{N}_2^{**})$, $(\mathcal{N}_1^{*}, \mathcal{N}_2^{*}) \in \mathcal{R}$. And $\mathcal{N}_1' \Rightarrow \mathcal{N}_1^{**}$ yields $\mathcal{N}_1' \Rightarrow \mathcal{N}_1^{***} \xrightarrow{(\overline{\eta})}_C \mathcal{N}_1^{*}$.

$$\begin{array}{c}(\mathcal{N}_1^{***}, \mathcal{N}_2^{**}) \in \mathcal{R} \wedge (\mathcal{N}_2^{**}, \mathcal{N}_1''') \in \mathcal{R}^{-1} \wedge (\mathcal{N}_1''', \mathcal{N}_2) \in \mathcal{R} \\ \Rightarrow (\mathcal{N}_1^{***}, \mathcal{N}_2) \in \mathcal{R} \circ \mathcal{R}^{-1} \circ \mathcal{R} \\ (\mathcal{N}_1^{*}, \mathcal{N}_2^{*}) \in \mathcal{R} \wedge (\mathcal{N}_2^{*}, \mathcal{N}_1'') \in \mathcal{R}^{-1} \wedge (\mathcal{N}_1'', \mathcal{N}_2'') \in \mathcal{R} \\ \Rightarrow (\mathcal{N}_1^{*}, \mathcal{N}_2'') \in \mathcal{R} \circ \mathcal{R}^{-1} \circ \mathcal{R}\end{array}$$

By Proposition 2 $\mathcal{R} \circ \mathcal{R}^{-1} \circ \mathcal{R}$ is a semi-branching computed network bisimulation. Since $\mathcal{R}$ is the largest semi-branching computed network bisimulation, and clearly $\mathcal{R} \subseteq \mathcal{R} \circ \mathcal{R}^{-1} \circ \mathcal{R}$, we have $\mathcal{R} = \mathcal{R} \circ \mathcal{R}^{-1} \circ \mathcal{R}$. Concluding, $\mathcal{N}_1' \Rightarrow \mathcal{N}_1^{***} \xrightarrow{(\overline{\eta})}_C \mathcal{N}_1^{*}$ with $(\mathcal{N}_1^{***}, \mathcal{N}_2)$, $(\mathcal{N}_1^{*}, \mathcal{N}_2'') \in \mathcal{R}$.

So $\mathcal{R}'$ is a semi-branching computed network bisimulation. Since $\mathcal{R}$ is the largest semi-branching computed network bisimulation, $\mathcal{R}' = \mathcal{R}$.

We will now prove that $\mathcal{R}$ is a branching computed network bisimulation. Let $\mathcal{N}_1 \mathcal{R} \mathcal{N}_2$, and $\mathcal{N}_1 \xrightarrow{\eta}_C \mathcal{N}_1'$. We only consider the case when η is of the form $m(\widehat{u})?$, because for other cases, the transfer condition of Definition 1 and Definition 3 are the same. So there are $\mathcal{N}_2''$ and $\mathcal{N}_2'$ such that $\mathcal{N}_2 \Rightarrow \mathcal{N}_2'' \xrightarrow{(m(\widehat{u})?)}_C \mathcal{N}_2'$ with $\mathcal{N}_1 \mathcal{R} \mathcal{N}_2''$ and $\mathcal{N}_1' \mathcal{R} \mathcal{N}_2'$. Two cases can be distinguished:

1. $\mathcal{N}_2''' = \mathcal{N}_2'$: Since $\mathcal{N}_1 \mathcal{R} \mathcal{N}_2$, $\mathcal{N}_1 \mathcal{R} \mathcal{N}_2'$, and $\mathcal{N}_1' \mathcal{R} \mathcal{N}_2'$, we proved above that $\mathcal{N}_1' \mathcal{R} \mathcal{N}_2$. This agrees with the first case of Definition 1.
2. $\mathcal{N}_2''' \neq \mathcal{N}_2'$: This agrees with the second case of Definition 1.

Consequently $\mathcal{R}$ is a branching computed network bisimulation. □

Since any branching computed network bisimulation is a semi-branching computed network bisimulation, this yields the following corollary.

Corollary 5. *Two computed network terms are related by a branching computed network bisimulation if and only if they are related by a semi-branching computed network bisimulation.*

Corollary 6. *Branching computed network bisimilarity is an equivalence relation.*

Corollary 7. *Rooted branching computed network bisimilarity is an equivalence relation.*

B Rooted Branching Computed Network Bisimilarity Is a Congruence

Theorem 8. *Rooted branching computed network bisimilarity is a congruence with respect to the protocol and computed network operators.*

Proof. We need to prove that:

- $[\![P_1]\!]_\ell \simeq_{rb} [\![P_2]\!]_\ell$ implies $[\![\alpha.P_1]\!]_\ell \simeq_{rb} [\![\alpha.P_2]\!]_\ell$
- $[\![P_1]\!]_\ell \simeq_{rb} [\![P_2]\!]_\ell$ and $[\![P_1']\!]_\ell \simeq_{rb} [\![P_2']\!]_\ell$ implies $[\![P_1 + P_1']\!]_\ell \simeq_{rb} [\![P_2 + P_2']\!]_\ell$
- $[\![P_1]\!]_\ell \simeq_{rb} [\![P_2]\!]_\ell$ and $[\![P_1']\!]_\ell \simeq_{rb} [\![P_2']\!]_\ell$ implies $[\![[u_1 = u_2]P_1, P_1']\!]_\ell \simeq_{rb} [\![[u_1 = u_2]P_2, P_2']\!]_\ell$
- $\mathcal{N}_1 \simeq_{rb} \mathcal{N}_2$ implies $C\eta.\mathcal{N}_1 \simeq_{rb} C\eta.\mathcal{N}_2$
- $\mathcal{N}_1 \simeq_{rb} \mathcal{N}_2$ and $\mathcal{N}_1' \simeq_{rb} \mathcal{N}_2'$ implies $\mathcal{N}_1 + \mathcal{N}_1' \simeq_{rb} \mathcal{N}_2 + \mathcal{N}_2'$
- $\mathcal{N}_1 \simeq_{rb} \mathcal{N}_2$ implies $(\nu\ell)\mathcal{N}_1 \simeq_{rb} (\nu\ell)\mathcal{N}_2$
- $\mathcal{N}_1 \simeq_{rb} \mathcal{N}_2$ and $\mathcal{N}_1' \simeq_{rb} \mathcal{N}_2'$ implies $\mathcal{N}_1 \parallel \mathcal{N}_1' \simeq_{rb} \mathcal{N}_2 \parallel \mathcal{N}_2'$
- $\mathcal{N}_1 \simeq_{rb} \mathcal{N}_2$ and $\mathcal{N}_1' \simeq_{rb} \mathcal{N}_2'$ implies $\mathcal{N}_1 \between \mathcal{N}_1' \simeq_{rb} \mathcal{N}_2 \between \mathcal{N}_2'$
- $\mathcal{N}_1 \simeq_{rb} \mathcal{N}_2$ and $\mathcal{N}_1' \simeq_{rb} \mathcal{N}_2'$ implies $\mathcal{N}_1 \mid \mathcal{N}_1' \simeq_{rb} \mathcal{N}_2 \mid \mathcal{N}_2'$

Clearly, if $\mathcal{N}_1 \simeq_{rb} \mathcal{N}_2$ then $\mathcal{N}_1 \simeq_b \mathcal{N}_2$. Consequently the first five cases are straightforward. We prove the sixth case. To this aim we prove that if $\mathcal{N}_1 \simeq_b \mathcal{N}_2$ then $(\nu\ell)\mathcal{N}_1 \simeq_b (\nu\ell)\mathcal{N}_2$. Let $\mathcal{N}_1 \simeq_b \mathcal{N}_2$ be witnessed by the branching computed network bisimulation relation $\mathcal{R}$. We define $\mathcal{R}' = \{((\nu\ell)\mathcal{N}_1', (\nu\ell)\mathcal{N}_2') | (\mathcal{N}_1', \mathcal{N}_2') \in \mathcal{R}\}$. We prove that $\mathcal{R}'$ is a branching computed network bisimulation relation. Suppose $(\nu\ell)\mathcal{N}_1' \xrightarrow{\eta'}_{C'} (\nu\ell)\mathcal{N}_1''$ resulted from the application of *Rest* on $\mathcal{N}_1' \xrightarrow{\eta}_C \mathcal{N}_1''$. Since $(\mathcal{N}_1', \mathcal{N}_2') \in \mathcal{R}$, there are two cases; in the first case η is a receive action and $(\mathcal{N}_1'', \mathcal{N}_2') \in \mathcal{R}$, consequently $((\nu\ell)\mathcal{N}_1'', (\nu\ell)\mathcal{N}_2') \in \mathcal{R}'$. In second case there are $\mathcal{N}_2'''$ and $\mathcal{N}_2''$ such that $\mathcal{N}_2' \Rightarrow \mathcal{N}_2''' \xrightarrow{\overline{\eta}}_C \mathcal{N}_2''$ with $(\mathcal{N}_1', \mathcal{N}_2'''), (\mathcal{N}_1'', \mathcal{N}_2'') \in \mathcal{R}$. By application of *Par*, $(\nu\ell)\mathcal{N}_2' \Rightarrow (\nu\ell)\mathcal{N}_2'''$ with $((\nu\ell)\mathcal{N}_1', (\nu\ell)\mathcal{N}_2''') \in \mathcal{R}'$. There are two cases to consider:

- $\overline{\eta} = \eta$: Consequently $(\nu\ell)\mathcal{N}_2''' \xrightarrow{\eta'}_{C'} (\nu\ell)\mathcal{N}_2''$.
- $\overline{\eta} \neq \eta$: in this case η is of the form $m(\widehat{u})!\{?\}$, $\eta' = \eta$ and $C' = hide(C, \ell)$. If $\overline{\eta} = \eta[\ell/?]$ then $\overline{\eta}[?/\ell] = \eta$ and $C' = hide(C[\ell/?], \ell)$ hold, otherwise $\overline{\eta}[?/\ell] = \overline{\eta}$ and $C'[\ell'/?] = hide(C[\ell'/?], \ell)$ hold where $\ell' \neq \ell$. Consequently $(\nu\ell)\mathcal{N}_2''' \xrightarrow{\overline{\eta'}}_{C'} (\nu\ell)\mathcal{N}_2''$.

With the above argumentation, there are $\mathcal{N}_2'''$ and $\mathcal{N}_2''$ such that $(\nu\ell)\mathcal{N}_2' \Rightarrow (\nu\ell)\mathcal{N}_2''' \xrightarrow{\overline{\eta'}}_{C'} (\nu\ell)\mathcal{N}_2''$ with $((\nu\ell)\mathcal{N}_1', (\nu\ell)\mathcal{N}_2''') , ((\nu\ell)\mathcal{N}_1'', (\nu\ell)\mathcal{N}_2'') \in \mathcal{R}'$.

Likewise we can prove that $\mathcal{N}_1 \simeq_{rb} \mathcal{N}_2$ implies $(\nu\ell)\mathcal{N}_1 \simeq_{rb} (\nu\ell)\mathcal{N}_2$. To this aim we examine the root condition in Definition 2. Suppose $(\nu\ell)\mathcal{N}_1 \xrightarrow{\eta'}_{C'} (\nu\ell)\mathcal{N}_1'$, with the same argumentation as above, $(\nu\ell)\mathcal{N}_2 \xrightarrow{\overline{\eta'}}_{C'} (\nu\ell)\mathcal{N}_2'$. Since $\mathcal{N}_1' \simeq_b \mathcal{N}_2'$, we proved that $(\nu\ell)\mathcal{N}_1' \simeq_b (\nu\ell)\mathcal{N}_2'$. Concluding $(\nu\ell)\mathcal{N}_1 \simeq_{rb} (\nu\ell)\mathcal{N}_2$.

From the three remaining cases, we focus on the most challenging case, which is the sync operator $|$; the others are proved in a similar fashion. First we prove that if $\mathcal{N}_1 \simeq_b \mathcal{N}_2$, then $\mathcal{N}_1 \parallel \mathcal{N} \simeq_b \mathcal{N}_2 \parallel \mathcal{N}$. Let $\mathcal{N}_1 \simeq_b \mathcal{N}_2$ be witnessed by the branching computed network bisimulation relation $\mathcal{R}$. We define $\mathcal{R}' = \{(\mathcal{N}_1' \parallel \mathcal{N}', \mathcal{N}_2' \parallel \mathcal{N}') | (\mathcal{N}_1', \mathcal{N}_2') \in \mathcal{R}, \mathcal{N}' \text{ any computed network term}\}$. We prove that $\mathcal{R}'$ is a branching computed network bisimulation relation. Suppose $\mathcal{N}_1 \parallel \mathcal{N} \xrightarrow{\eta}_{C^*} \mathcal{N}^*$. There are several cases to consider:

- Suppose η is a send action $m(\widehat{u})!$ performed by an address ℓ. First let it be performed by $\mathcal{N}_1'$, and $\mathcal{N}$ participated in the communication. That is, $\mathcal{N}_1' \xrightarrow{m(\widehat{u})!\{\ell\}}_{C_1} \mathcal{N}_1''$ and $\mathcal{N} \xrightarrow{m(\widehat{u})?}_{C} \mathcal{N}'$ give rise to the transition $\mathcal{N}_1' \parallel \mathcal{N} \xrightarrow{m(\widehat{u})!\{\ell\}}_{C_1 \cup C[\ell/?]} \mathcal{N}_1'' \parallel \mathcal{N}'$. As $(\mathcal{N}_1', \mathcal{N}_2') \in \mathcal{R}$ and $\mathcal{N}_1' \xrightarrow{m(\widehat{u})!\{\ell\}}_{C_1} \mathcal{N}_1''$, there are $\mathcal{N}_2''$ and $\mathcal{N}_2'''$ such that $\mathcal{N}_2' \Rightarrow \mathcal{N}_2''' \xrightarrow{m(\widehat{u})!\{\ell'\}}_{C_1[\ell'/\ell]} \mathcal{N}_2''$, where $(\ell = ? \vee \ell = \ell')$ and $(\mathcal{N}_1', \mathcal{N}_2'''), (\mathcal{N}_1'', \mathcal{N}_2'') \in \mathcal{R}$. Hence $\mathcal{N}_2' \parallel \mathcal{N} \Rightarrow \mathcal{N}_2''' \parallel \mathcal{N} \xrightarrow{m(\widehat{u})!\{\ell'\}}_{C_1 \cup C[\ell'/?]} \mathcal{N}_2'' \parallel \mathcal{N}'$ with $(\mathcal{N}_1' \parallel \mathcal{N}, \mathcal{N}_2''' \parallel \mathcal{N}), (\mathcal{N}_1'' \parallel \mathcal{N}', \mathcal{N}_2'' \parallel \mathcal{N}') \in \mathcal{R}'$.
  Now suppose that the send action was performed by $\mathcal{N}$, and $\mathcal{N}_1'$ participated in the communication. That is, $\mathcal{N}_1' \xrightarrow{m(\widehat{u})?}_{C_1} \mathcal{N}_1''$ and $\mathcal{N} \xrightarrow{m(\widehat{u})!\{\ell\}}_{C} \mathcal{N}'$ give rise to the transition $\mathcal{N}_1' \parallel \mathcal{N} \xrightarrow{m(\widehat{u})!\{\ell\}}_{C_1[\ell/?] \cup C} \mathcal{N}_1'' \parallel \mathcal{N}'$. Since $(\mathcal{N}_1', \mathcal{N}_2') \in \mathcal{R}$ and $\mathcal{N}_1' \xrightarrow{m(\widehat{u})?}_{C_1} \mathcal{N}_1''$, two cases can be considered: either $(\mathcal{N}_1'', \mathcal{N}_2') \in \mathcal{R}$, or there are $\mathcal{N}_2'''$ and $\mathcal{N}_2''$ such that $\mathcal{N}_2' \Rightarrow \mathcal{N}_2''' \xrightarrow{m(\widehat{u})?}_{C_1} \mathcal{N}_2''$ with $(\mathcal{N}_1', \mathcal{N}_2'''), (\mathcal{N}_1'', \mathcal{N}_2'') \in \mathcal{R}$. In the first case, $\mathcal{N}_2' \parallel \mathcal{N} \xrightarrow{m(\widehat{u})!\{\ell\}}_{C_1 \cup C[\ell/?]} \mathcal{N}_2' \parallel \mathcal{N}'$, and $(\mathcal{N}_1'' \parallel \mathcal{N}', \mathcal{N}_2' \parallel \mathcal{N}') \in \mathcal{R}$. In the second case, $\mathcal{N}_2' \parallel \mathcal{N} \Rightarrow \mathcal{N}_2''' \parallel \mathcal{N} \xrightarrow{m(\widehat{u})!\{\ell\}}_{C_1 \cup C[\ell/?]} \mathcal{N}_2'' \parallel \mathcal{N}'$, and $(\mathcal{N}_1' \parallel \mathcal{N}, \mathcal{N}_2''' \parallel \mathcal{N}), (\mathcal{N}_1'' \parallel \mathcal{N}', \mathcal{N}_2'' \parallel \mathcal{N}') \in \mathcal{R}'$.
  The cases where $\mathcal{N}$ or $\mathcal{N}_1$ does not participate in the communication are straightforward.
- The case where η is a receive action $m(\widehat{u})?$ is also straightforward; it originates from $\mathcal{N}_1, \mathcal{N}$, or both.

Likewise we can prove that $\mathcal{N}_1 \simeq_{rb} \mathcal{N}_2$ implies $\mathcal{N} \parallel \mathcal{N}_1 \simeq_{rb} \mathcal{N} \parallel \mathcal{N}_2$.

Now let $\mathcal{N}_1 \simeq_{rb} \mathcal{N}_2$. To prove $\mathcal{N}_1|\mathcal{N} \simeq_{rb} \mathcal{N}_2|\mathcal{N}$, we examine the root condition from Definition 2. Suppose $\mathcal{N}_1|\mathcal{N} \xrightarrow{m(\widehat{u})!\{\ell\}}_{C^*} \mathcal{N}^*$. There are two cases to consider:

- This send action was performed by $\mathcal{N}_1$ at node ℓ, and $\mathcal{N}$ participated in the communication. That is, $\mathcal{N}_1 \xrightarrow{m(\widehat{u})!\{\ell\}}_{C_1} \mathcal{N}_1'$ and $\mathcal{N} \xrightarrow{m(\widehat{u})?}_{C} \mathcal{N}'$, so that $\mathcal{N}_1|\mathcal{N} \xrightarrow{m(\widehat{u})!\{\ell\}}_{C_1 \cup C[\ell/?]} \mathcal{N}_1' \parallel \mathcal{N}'$. Since $\mathcal{N}_1 \simeq_{rb} \mathcal{N}_2$, there is an $\mathcal{N}_2'$ such that $\mathcal{N}_2 \xrightarrow{m(\widehat{u})!\{\ell'\}}_{C_1[\ell'/\ell]} \mathcal{N}_2'$ with $(\ell =? \vee \ell = \ell')$ and $\mathcal{N}_1' \simeq_b \mathcal{N}_2'$. Then $\mathcal{N}_2|\mathcal{N} \xrightarrow{m(\widehat{u})!\{\ell'\}}_{C_1 \cup C[\ell'/?]} \mathcal{N}_2' \parallel \mathcal{N}'$. Since $\mathcal{N}_1' \simeq_b \mathcal{N}_2'$, we proved that $\mathcal{N}_1' \parallel \mathcal{N}' \simeq_b \mathcal{N}_2' \parallel \mathcal{N}'$.
- The send action was performed $\mathcal{N}$ at node ℓ, and $\mathcal{N}_1$ participated in the communication. That is, $\mathcal{N}_1 \xrightarrow{m(\widehat{u})?}_{C_1} \mathcal{N}_1'$ and $\mathcal{N} \xrightarrow{m(\widehat{u})!\{\ell\}}_{C} \mathcal{N}$, so that $\mathcal{N}_1|\mathcal{N} \xrightarrow{m(\widehat{u})!\{\ell\}}_{C_1 \cup C[\ell/?]} \mathcal{N}_1 \parallel \mathcal{N}'$. Since $\mathcal{N}_1 \simeq_{rb} \mathcal{N}_2$, there is an $\mathcal{N}_2'$ such that $\mathcal{N}_2 \xrightarrow{m(\widehat{u})?}_{C_1} \mathcal{N}_2'$ with $\mathcal{N}_1' \simeq_b \mathcal{N}_2'$. Then $\mathcal{N}_2|\mathcal{N} \xrightarrow{m(\widehat{u})!\{\ell\}}_{C_1 \cup C[\ell/?]} \mathcal{N}_2' \parallel \mathcal{N}'$. Since $\mathcal{N}_1' \simeq_b \mathcal{N}_2'$, we have $\mathcal{N}_1' \parallel \mathcal{N}' \simeq_b \mathcal{N}_2' \parallel \mathcal{N}'$.

Finally, the case where $\mathcal{N}_1|\mathcal{N} \xrightarrow{m(\widehat{u})?}_{C^*} \mathcal{N}^*$ can be easily dealt with. This receive action was performed by both $\mathcal{N}_1$ and $\mathcal{N}$.

Concluding, $\mathcal{N}_1|\mathcal{N} \simeq_{rb} \mathcal{N}_2|\mathcal{N}$. Likewise it can be argued that $\mathcal{N}|\mathcal{N}_1 \simeq_{rb} \mathcal{N}|\mathcal{N}_2$. □