Theoretical Computer Science 411 (2010) 1928–1948

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

An operational semantics for a calculus for wireless systems[☆]

Ivan Lanese, Davide Sangiorgi[*]

Focus Research Team, University of Bologna/INRIA, Italy

ARTICLE INFO

Keywords:
Wireless systems
Broadcast
Reduction Semantics
Labelled Transition Semantics
Interference

ABSTRACT

In wireless systems, the communication mechanism combines features of broadcast, synchrony, and asynchrony. We develop an operational semantics for a calculus of wireless systems. We present different Reduction Semantics and a Labelled Transition Semantics and prove correspondence results between them. Finally, we apply CWS to the modelling of the Alternating Bit Protocol, and prove a simple correctness result as an example of the kind of properties that can be formalized in this framework.

A major goal of the semantics is to describe the forms of interference among the activities of processes that are peculiar of wireless systems. Such interference occurs when a location is simultaneously reached by two transmissions. The Reduction Semantics differ on how information about the active transmissions is managed.

We use the calculus to describe and analyse a few properties of a version of the Alternating Bit Protocol.

1. Motivation of the work

Wireless technology is becoming more widespread everyday; its applications span user applications, such as personal area networks, ambient intelligence, and wireless local area networks, and real-time applications, such as cellular networks for mobile telephony and sensor networks.

The goal of this work is to define a basic calculus of wireless systems, and its associated operational semantics, which is the most important contribution of the work. Here we have to deal with the peculiarities of wireless systems, which combine features of broadcast, synchrony, and asynchrony, as we outline below.

Wireless devices communicate by *broadcast* of messages. This is however quite different from the more conventional wired-based broadcast that we find most notably in Ethernet networks and that, from a semantic point of view, is well understood [27,28]. First, in Ethernet-like systems broadcasting is global, i.e., the messages transmitted reach all nodes of the system. By contrast, in wireless systems broadcasting is local, i.e., a transmission spans over a limited area, called a *cell*, and therefore reaches only a – possibly empty – subset of the devices in the system. Second, the communication channel for an Ethernet device is full-duplex; that is, a node can transmit and receive at the same time. As a consequence of full-duplex channels and global broadcasting, interference between two simultaneous transmissions is immediately detected by transmitters. Thus transmitters know that they have to re-transmit their messages, and they do so after a randomly chosen period of time. This means that interference in Ethernet-like systems is easy to repair. In a model of these systems it is therefore reasonable to abstract away from interference, i.e., to assume that it does not exist. By contrast, wireless systems channels are *half-duplex*: on a given channel, a device can either transmit or receive, but cannot do both at the same time. Hence, interference between two transmissions is only possibly detected by receivers located in the intersection of the cells of the two transmitters. Interference is thus a delicate aspect of wireless systems that is handled by means of

[☆] Supported by the EC projects IST-FP6 16004 SENSORIA and IST-FP7-231620 HATS.

[*] Corresponding address: Mura Anteo Zamboni n. 7, I-40127 Bologna, Italy. Tel.: +39 051 2094980; fax: +39 051 2094510.
E-mail addresses: lanese@cs.unibo.it (I. Lanese), sangio@cs.unibo.it (D. Sangiorgi).
URLs: http://www.cs.unibo.it/~lanese (I. Lanese), http://www.cs.unibo.it/~sangio (D. Sangiorgi).

0304-3975/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2010.01.023

ad hoc protocols (e.g., 802.11 [12]). Interference is also one of the key aspects to be described by a model of these systems. A third, but semantically less relevant, difference between Ethernet and wireless broadcast is that only the latter uses explicit communication channels.

Wireless systems also have features of *synchrony* that remind us of synchronous languages (e.g., Esterel [4], Statecharts [11], SCCS [20]). Indeed, in a single time unit multiple events can happen in a wireless system; such a simultaneous execution of the events is different from an interleaving of them: only in the former case, for instance, interference can appear.

Wireless devices locally try to remain synchronized on a common clock so that, for instance, transmissions start at the same time. This is due to the fact that each device incorporates a hardware clock and uses it for physical transmission; however, each hardware clock is generally characterised by a different clock drift with respect to the nominal frequency. Hence, to reduce the effect of the clock drift, each device re-synchronizes its own clock on the transmissions which are performed by the other devices. However, the physical distance between the devices and the partial area covered by each transmission may prevent an exact global agreement among clocks. As a consequence, the clocks of distant devices usually do not coincide. This introduces an additional element of *asynchrony* that affects the way interference appears and is detected, as we discuss below.

Another – though less important – feature of wireless systems (that a semantics should show) is the fact that a transmitter, before initiating a transmission, checks that, locally, the communication channel (i.e. a specific frequency band) is not presently employed for performing other transmissions. This is imposed in order to reduce the possibilities of interference. Notice that there is a short gap between the check for channel availability and the begin of the transmission.

Our calculus is called *Calculus of Wireless Systems* (CWS). It has nodes, which represent the devices of the system, that can be composed in parallel. Inside a node there is a sequential process, which models the behavior of that device. Each node has a location and a radius that define the cell over which that node can transmit. When developing the semantics for CWS we tried hard to adhere to the standard operational semantics of concurrent systems (broadcast systems as CBS [27,28,26], point-to-point systems as CCS [21]) in which each transmission of a message (or communication) is represented precisely by a single transition (or reduction) in the semantics. For instance, in the labelled transition semantics of CCS an event $P \xrightarrow{\bar{a}v} P'$ means that an output process in P transmits value v on channel a and then the process evolves into P'. Only one among the input processes executing in parallel with P receives that value. Similarly, in CBS an event $P \xrightarrow{!v} P'$ means that an output process in P transmits value v and then the process evolves into P'. All the input processes executing in parallel with P catch that transmission.

Our attempt at following this approach was unsuccessful due to the physical aspects of wireless systems outlined above. In particular, we could not maintain transmission atomicity and, at the same time, express all the possible modes in which interference may occur.

A natural alternative would have been to follow the approach of synchronous languages. In these languages an event in the semantics represents a set of transmissions, namely all those that occurred during the same time unit. Unfortunately, we failed also with this approach. The reason is that – as explained earlier – the synchrony among the devices of a wireless system is not perfect: the transmissions of two distant devices may span time units that are only partially overlapping, with strong consequences on the interference that these devices can cause with each other and with other devices.

We thus decided to refine the view of transmissions and observe, for each node, the change of state between transmission and reception (and vice versa), rather than single transmissions. Further, in accordance with physical wireless devices, we assume that when a device is not performing a transmission its antenna is in reception mode. We call *event* the state change that occurs in the network when a device changes the function of its antenna. Specifically, we call *begin transmission* the event which corresponds to a device which initiates a transmission, and we call *end transmission* the event which corresponds to a transmitter which finishes its transmission. This choice has also physical justifications: wireless communication protocols require different signal sequences for indicating begin and end of transmissions.

In concurrency theory, *Labelled Transition Systems* (LTSs) are the most widely adopted means of giving operational semantics. An LTS can be a basis for defining powerful proof techniques: both inductive techniques, because, for instance, the derivation of a transition in the LTS is driven by the structure of the term; and co-inductive techniques, because, for instance, the transitions of an LTS expose the full behavior of the system (its internal activities as well as the interactions with the environment) which is needed for defining behavioral equivalences according to the notion of bisimulation. However, sometimes the rules of an LTS can be hard to understand. This happens, for instance, in calculi for mobility such as the π-calculus [30] or in higher-order calculi such as Ambients [6]. Thus, a different form of operational semantics, called *Reduction Semantics* (RS), has been introduced [22]. An RS only gives meaning to the internal activities of a system. For this reason, the structural operational semantics style is combined with an auxiliary relation of structural congruence that allows the manipulation of the term structure so as to bring potential interacting components into contiguous positions. An RS can make the meaning of a calculus or language easier to grasp; and it can be used to check the correctness of an LTS, by proving consistency with respect to such an LTS. On the other hand, the LTS remains superior for defining reasoning techniques on.

For the reasons above, we define both RS and LTS semantics for CWS. RS rules usually specify how two system components interact, but in our case an unbounded number of components (one sender and many receivers) can be involved in an atomic interaction. Thus we need to use schemas of rules (to be instantiated according to the number of interacting components) instead of just rules. Also, under the RS style we analyse different approaches on how to check for interference. The first one

(Section 3) takes a set of active transmitters (i.e., a set of devices that are currently engaged in a transmission) as a parameter; then, the various possibilities of interference are checked against such a parameter. This is the approach used for the LTS too. In Section 6 instead we propose an approach where the information about the active transmissions is not global, but distributed among the receivers. This allows us to have more local checks. However this semantics appears difficult to extend to deal with operators such as channel switching (Section 8). For this reason a new RS is presented (Section 7), where no information is required in the term and checks for interference are "hardwired" in the rules.

Our main technical result is the equivalence between the LTS and the RS semantics. The proof differs from existing results of this kind in the literature because of the differences between standard LTSs and RSs and ours. We also present correspondence theorems between the different RSs we consider.

Section 8 adds to the basic calculus discussed in the previous sections operators such as handled input, channel switching, conditional and constant definitions.

Section 9 applies CWS to the modelling of a version of the Alternating Bit Protocol, and proves a simple correctness result as an example of the kind of properties that can be formalized in this framework.

Section 10 considers a few other issues. In particular, Section 10.1 updates the semantics by taking into account the gap between the check for availability of a channel and the beginning of the transmission on it, thus allowing to model transmissions that start concurrently. Finally, in Section 10.2 we discuss a few challenging language extensions: local channels, time, and mobility.

Concerning mobility, in particular, we decided to avoid it in the calculus for two reasons. First, mobility is orthogonal to the problem of interference we are mainly interested in, and can be modelled following the approaches in the literature [23,9,17] (see Section 10.2 for the details). Second, movement is not relevant in important classes of wireless systems, most notably sensor networks (not all sensor networks are static, but the static case is predominant). A sensor network [3,2] is a computer network of many, spatially distributed devices using sensors to monitor conditions (e.g., temperature, sound, vibration, pressure, motion) at different locations. Each device is equipped with a radio transceiver, a small micro-controller, and an energy source, usually a battery. Sensors communicate employing a broadcast-based wireless data-link protocol (i.e., 802.15.4 [13]) to collaboratively transport, and possibly aggregate, the collected information to a monitoring computer. Sensor networks are applied in a wide variety of areas: traffic monitoring and video surveillance [16,32], disaster recovery [33,7,24], home monitoring [15], manufacturing and industrial automation [10,31], etc.

This paper is an extended and improved version of [19]. The main novelties concern the RS, which has been simplified and of which three variants are presented, addressing interference from various points of view. However, the RS in [19] uses rules instead of rule schemas. The application to the Alternating Bit Protocol is also an original contribution.

Related work

Probabilistic simulation [5,25,14] is currently the main validation technique used for verification of wireless systems: given a system, pseudo-random number generators are employed to select a finite subset of the possible execution traces and then statistical analysis techniques are applied to obtain probabilistic information on that system.

We discuss below the works on process calculi that are most closely related to ours. Broadcast has been first analysed by Prasad [27,28,26] who models Ethernet-like communications. In particular, in this setting broadcast is global, while this is not true for wireless broadcast. Ene and Muntean [8] add channels in the style of π-calculus [30] to limit the scope of broadcast, but this models logic constraints instead of the physical constraints found in wireless systems. Similarly in MBS [29] processes are allowed to move between different rooms, and broadcast is limited to rooms. Here the emphasis is more on exploring a new programming paradigm based on the interplay between movement and broadcast than on analysing wireless systems.

The only three calculi for wireless broadcast we are aware of are CBS# [23], CMAN [9] and CMN [17]. In all the cases the topology of the system and the behavior of the different components are represented by dedicated constructs, and broadcasted messages are received only by processes near enough to the sender. In CBS# and CMAN the topology of the system is represented by graphs (explicit in CBS#, encoded in the term in CMAN), while in CMN each process has a location and a transmission radius, and a distance function is used. Also, in CBS# mobility is modelled by allowing to change the connecting graph (and the used connecting graph is attached to the transition label) among a set of possible graphs. In CMN nodes may be mobile or stationary, and mobile nodes may change their location at will. Finally, in CMAN connections can be created and destroyed as silent moves.

All the works above treat broadcast as an atomic action, thus in particular abstracting away from interference. This abstraction makes the calculi above interesting for reasoning about high-level properties and applications. Instead we chose to model a wireless system at the lower level, namely that of data-link, which is the lowest level at which wireless systems can be programmed. At this level interference is an essential aspect (indeed, the presence of collisions unresolvable by hardware is one of the most original features of wireless networking).

2. The core language

We present the (core of the) Calculus of Wireless Systems, CWS. The language has two parts: the process part, which describes the possible states of a node, and the network part. For now, we introduce only those primitives that are necessary for communication. The full language is presented in Section 8, and possible extensions are discussed in Section 10.2.

Table 1
Language for the description of wireless networks.

P	$\overset{\text{def}}{=}$	$\mathtt{out}\langle e\rangle.P$	*output*	$\vert$	$\langle v\rangle.P$	*active output*
	$\vert$	$\mathtt{in}(x).P$	*input*	$\vert$	$(x).P$	*active input*
	$\vert$	$\mathbf{0}$	*inactive process*			
N	$\overset{\text{def}}{=}$	$n\,[P]^c_{l,r}$	*node (or device)*	$\vert$	$N\vert N$	*parallel composition*
	$\vert$	$\mathbf{0}$	*empty network*			

Processes are sequential and live within the nodes, which are the basic network elements, and represent single devices. Each node has a location and a transmission radius; they are employed to define the cell over which that node can transmit. Nodes cannot be created or destroyed. We write $n\,[P]^c_{l,r}$ for a node named n, executing process P, synchronized on channel c, and which can transmit over a cell centered in l (i.e., its location) with radius r. The node identifier n represents a logical location — the device network address. By contrast, location l represents a physical location and, together with the radius, is employed for deriving information about the network connectivity.

We do not indicate how locations should be specified; for instance, they could be given by means of a coordinate system. The only assumption we make is the possibility of comparing locations, so to determine whether a node lies or not within the transmission cell of another node. We do so by means of a function d which takes two locations and returns their distance.[1]

Nodes transmit values by broadcasting them via channels. The details about how transmissions work are given in the next section. Two important features of transmissions that are visible in the choice of operators in CWS are the following:

- As motivated in Section 1, in order to identify whether (and where) two transmissions interfere, in CWS a transmission is not atomic, but is modelled by its two boundary events: *begin transmission* and *end transmission*. Physically, these events represent a modification of the antenna communication mode (from reception to transmission mode, and opposite).
- In a wireless system, a transmission is performed in a bit-per-bit fashion. If the transmission aborts, the partial bit-stream is discarded. Abstractly, this corresponds to keeping the channel allocated for the transmission time and revealing the value just before concluding the transmission. We hence assume that a transmitted value becomes visible only once it has been completely transmitted; that is, we make it visible to the recipients with the occurrence of the end transmission event.

Table 1 defines the syntax of CWS. We use $a \ldots d$ for channels; $m \ldots o$ for identifiers; $x \ldots z$ for variables; $u \ldots v$ for the values that can be transmitted over a channel: these include node identifiers, but not channels. Finally, $e \ldots f$ are value expressions, which include values and variables; in the operational semantics, we use also the special symbol $\perp$, which indicates detection of an interference and cannot be transmitted. We do not provide a grammar for values and value expressions. They represent orthogonal choices with respect to the process and network constructs. We assume the existence of an evaluation function $[\![.]\!]$, which returns the value of a closed expression.

In the process constructs, an output $\mathtt{out}\langle e\rangle.P$ is a process willing to initiate the transmission of value $v = [\![e]\!]$; as a result, the process will evolve to $\langle v\rangle.P$. This represents a process that is currently transmitting v; when the transmission is terminated, this process becomes P. An input process $\mathtt{in}(x).P$ represents a process willing to receive; when the begin of a transmission is detected in the clear (i.e., without interference) the process becomes the active input process $(x).P$, representing a process that is currently receiving. This reception succeeds if no collision is detected before the end of the transmission; in this case, x is bound to the received value. Otherwise x is bound to $\perp$, which indicates a failed reception. The inactive process $\mathbf{0}$ is a terminated process. In the syntax for networks, parallel composition represents two networks that execute in parallel. Given a set of indices I, we use $\prod_{i\in I} P_i$ to denote the parallel composition of processes P_i for each $i \in I$. The empty network is a network without any nodes. In $\mathtt{in}(x).P$ and $(x).P$ the displayed occurrence of x is binding with scope P.

We assume that in any network each node identifier is unique, and two different nodes cannot share the same location. Further, no active input or active output can appear underneath a process prefix. We call such networks *well formed*; in the remainder of the paper all networks are well formed. Since new nodes cannot be created, the well-formedness of a network is maintained as the network evolves.

For power saving purposes, when a device is not performing any transmission, it maintains the antenna in reception mode. Hence, in the reminder of this work we call a *transmitter* a node whose process begins with an output or with an active output prefix; in the latter case, we say that the node is an *active transmitter* because it is presently performing a transmission (in practice, this is the only case in which a wireless device employs its antenna in transmit mode).

[1] Actually for our approach a generic function from pairs of locations to real numbers is enough, but we use the term distance since this is the most common case.

3. Reduction Semantics

We present in this section a Reduction Semantics (RS) for CWS. In the literature [22], RS has been mainly employed for describing processes with handshake communications (essentially mobile and higher-order calculi such as the π-calculus). In such systems, a reduction is an interaction between an emitter and a receiver. In our case, an internal activity is a broadcast which, as discussed in Sections 1 and 2, is modelled by the two events begin transmission and end transmission. Thus, in our RS for core CWS, a reduction represents either a begin transmission or an end transmission. We use rule schemas in order to deal with the interaction among an unbounded number of processes. Also, reductions are decorated by (simple) labels. This is necessary since, as opposed to standard handshake communication, a reduction modelling a broadcast cannot be performed inside arbitrary contexts: one should check that the context is not affected by the event. For instance the context must not contain a receiver reached by the transmission. For this, some minimal information about the transmission is attached to the label of the reduction.

To be able to faithfully model interference, we have to check whether or not a node is currently reached by more than one transmission. Different approaches to this problem can be used, differing on how much information is stored in the process representation and how much is computed during each derivation. We present here a centralized approach: all the information concerning the active transmitters (the only ones that may provoke interference) is collected in a set T which appears in any reduction. We have chosen to present this approach first since it has the most intuitive rules and it allows for a direct correspondence with the LTS of Section 4. Other approaches will be considered in the following sections. We write RST (from "RS with parameter T") to identify this form of Reduction Semantics.

RST reductions have the form $T \rhd N \hookrightarrow^{c}_{l,r} N'$ and read *"given the set T of active transmitters, network N can reduce to network N' due to an event from a transmitter located at l, with radius r, and synchronized on channel c"*. The component T is a set of triples (l, r, c), with each triple containing location, radius and channel of an active transmitter. The set T should include the set of active transmitters in N (as a consequence, if a transmitter (l, r, c) performs a begin transmission event then $(l, r, c) \notin T$, while if it performs an end transmission event then $(l, r, c) \in T$). Having T larger than the set of active transmitters in N may be useful, for instance to analyse the evolution of a system in the presence of external transmissions. The two sets should however coincide to know the evolution of a system in isolation.

The semantics does not automatically update the set T after a reduction. Thus, when a reduction is performed, the new T to be used for the next one has to be manually computed. We have preferred to keep in the rules only the information necessary for deriving the reductions; it is easy to modify the rules so that they also produce the new T.

As usual for RS, we exploit a structural congruence relation, denoted $\equiv$, to bring interacting processes into contiguous positions. Here $\equiv$ is the smallest congruence including associativity, commutativity and identity over $\mathbf{0}$ for parallel composition:

$$N|(N'|N'') \equiv (N|N')|N'' \qquad N|N' \equiv N'|N \qquad N|\mathbf{0} \equiv N$$

We introduce some notation useful for networks, which will be used in the rules of RST.

- $T|_{l,c}$ is the subset of the active transmitters T whose transmissions reach a node located in l and synchronized on channel c. Formally,

$$T|_{l,c} = \left\{(l', r', c') \;\middle|\; (l', r', c') \in T \wedge \mathrm{d}(l', l) \leq r' \wedge c' = c\right\}$$

- $(l, r, c) \not\Downarrow_i N$ holds if network N contains no input nodes $n\,[\mathtt{in}(x).P]^{c}_{l',r'}$ or $n\,[(x).P]^{c}_{l',r'}$ for which $\mathrm{d}(l, l') \leq r$ holds (i.e., a transmission on c from l with radius r reaches no input nodes in N).

In the RST, we have two rule schemas, one for begin transmission and one for end transmission, plus two closure rules, one allowing us to make use of structural congruence and one allowing closure under contexts that are not influenced by the reduction.

[RST-BEGIN]

$$\frac{T|_{l,c} = \emptyset \quad \forall h \in I \cup J \cup K . \mathrm{d}(l, l_h) \leq r \quad \forall i \in I . T|_{l_i,c} = \emptyset \quad \forall j \in J . T|_{l_j,c} \neq \emptyset}{\begin{array}{c} T \rhd n\,[\mathtt{out}\langle e\rangle.P]^{c}_{l,r} \mid \prod_{h\in I\cup J} n_h\,[\mathtt{in}(x_h).P_h]^{c}_{l_h,r_h} \mid \prod_{k\in K} n_k\,[(x_k).P_k]^{c}_{l_k,r_k} \hookrightarrow^{c}_{l,r} \\ n\,[\langle [\![e]\!] \rangle.P]^{c}_{l,r} \mid \prod_{i\in I} n_i\,[(x_i).P_i]^{c}_{l_i,r_i} \mid \prod_{j\in J} n_j\,\left[\mathtt{in}(x_j).P_j\right]^{c}_{l_j,r_j} \mid \prod_{k\in K} n_k\,[P_k\{\bot/x_k\}]^{c}_{l_k,r_k} \end{array}}$$

[RST-END]

$$\frac{\forall i \in I . \mathrm{d}(l, l_i) \leq r}{\begin{array}{c} T \rhd n\,[\langle v\rangle.P]^{c}_{l,r} \mid \prod_{i\in I} n_i\,[(x_i).P_i]^{c}_{l_i,r_i} \mid \prod_{j\in J} n_j\,\left[\mathtt{in}(x_j).P_j\right]^{c}_{l_j,r_j} \hookrightarrow^{c}_{l,r} \\ n\,[P]^{c}_{l,r} \mid \prod_{i\in I} n_i\,[P_i\{v/x_i\}]^{c}_{l_i,r_i} \mid \prod_{j\in J} n_j\,\left[\mathtt{in}(x_j).P_j\right]^{c}_{l_j,r_j} \end{array}}$$

$$\begin{aligned}
\emptyset \rhd N &\hookrightarrow^{c}_{l_1,\rho} \; n_1\,[\langle v_1\rangle.\mathbf{0}]^{c}_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^{c}_{l_2,\rho} \,\big|\, m\,[(x).P]^{c}_{l',\rho'} \overset{\text{def}}{=} N_1 \\
\{(l_1,\rho,c)\} \rhd N_1 &\hookrightarrow^{c}_{l_2,\rho} \; n_1\,[\langle v_1\rangle.\mathbf{0}]^{c}_{l_1,\rho} \,\big|\, n_2\,[\langle v_2\rangle.\mathbf{0}]^{c}_{l_2,\rho} \,\big|\, m\,[P\{\perp/x\}]^{c}_{l',\rho'} \overset{\text{def}}{=} N_2 \\
\{(l_1,\rho,c),(l_2,\rho,c)\} \rhd N_2 &\hookrightarrow^{c}_{l_2,\rho} \; n_1\,[\langle v_1\rangle.\mathbf{0}]^{c}_{l_1,\rho} \,\big|\, n_2\,[\mathbf{0}]^{c}_{l_2,\rho} \,\big|\, m\,[P\{\perp/x\}]^{c}_{l',\rho'} \overset{\text{def}}{=} N_3 \\
\{(l_1,\rho,c)\} \rhd N_3 &\hookrightarrow^{c}_{l_1,\rho} \; n_1\,[\mathbf{0}]^{c}_{l_1,\rho} \,\big|\, n_2\,[\mathbf{0}]^{c}_{l_2,\rho} \,\big|\, m\,[P\{\perp/x\}]^{c}_{l',\rho'}
\end{aligned}$$

Fig. 1. Interference.

$$[\text{RST-CONT}]\ \frac{T \rhd N \hookrightarrow^{c}_{l,r} N' \quad (l,r,c) \not\Downarrow_i N''}{T \rhd N|N'' \hookrightarrow^{c}_{l,r} N'|N''}$$

$$[\text{RST-CONGR}]\ \frac{N \equiv N' \quad T \rhd N' \hookrightarrow^{c}_{l,r} N'' \quad N'' \equiv N'''}{T \rhd N \hookrightarrow^{c}_{l,r} N'''}$$

Rule RST-BEGIN is the main rule for deriving begin transmission reductions. It rewrites atomically an output node $n\,[\mathtt{out}\langle e\rangle.P]^{c}_{l,r}$ willing to begin a transmission and all the receiver nodes in its transmission cell (and synchronized on the same channel c). Condition $T|_{l,c} = \emptyset$ checks that the output process is not currently reached by another transmission (note that the triple (l,r,c) is not in T). The output process becomes $\langle [\![e]\!] \rangle.P$ indicating a process that is transmitting the result $[\![e]\!]$ of the evaluation of expression e. The effect of the begin transmission event on each receiver in the transmission cell depends on the structure of each receiver and on the set T of active transmitters. There are three different cases, corresponding to the three sets of indices I, J and K. Processes indexed in I are inputs, and they are not reached by any other transmission since $T|_{l_i,c} = \emptyset$. Thus they start receiving and become active inputs of the form $(x_i).P_i$. The opposite happens for processes indexed in J, which are currently reached by at least one other transmission, since $T|_{l_j,c} \neq \emptyset$. Thus they do not receive the begin transmission event in the clear and they stay idle. Finally processes indexed in K are active inputs. As such, they are currently reached by another transmission, and the new begin transmission event causes interference, represented by the symbol $\perp$ that they receive.

Rule RST-END deals with end transmission events and it has the same structure as RST-BEGIN. There are however a few important differences. First of all, it models an active transmitter $n\,[\langle v\rangle.P]^{c}_{l,r}$ completing the transmission of value v. This case is simpler than that of RST-BEGIN since it cannot originate interference. Simply, all active input processes $(x_i).P_i$ in the transmission range receive the value v becoming $P_i\{v/x_i\}$, while inactive inputs stay idle.

Rules RST-CONGR and RST-CONT give us, respectively, closure under structural congruence, and closure under contexts that do not contain receivers in the transmission cell of the transmitter. The last condition can be checked since the required information, i.e. the location and the radius of the transmitter and the channel used, is recorded in the reduction label.

Example 1 (*Interference*). This example shows the behavior of a receiver $R \overset{\text{def}}{=} m\,[\mathtt{in}(x).P]^{c}_{l',\rho'}$ and two transmitters $S_1 \overset{\text{def}}{=} n_1\,[\mathtt{out}\langle v_1\rangle.\mathbf{0}]^{c}_{l_1,\rho}$ and $S_2 \overset{\text{def}}{=} n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^{c}_{l_2,\rho}$ when transmissions interfere with each other at the receiver site (i.e., the relations $\mathrm{d}(l_1,l_2) > \rho$, $\mathrm{d}(l_1,l') \leq \rho$, and $\mathrm{d}(l_2,l') \leq \rho$ hold). Thus the network is:

$$N \overset{\text{def}}{=} n_1\,[\mathtt{out}\langle v_1\rangle.\mathbf{0}]^{c}_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^{c}_{l_2,\rho} \,\big|\, m\,[\mathtt{in}(x).P]^{c}_{l',\rho'}$$

A possible computation is represented in Fig. 1. At the beginning, both transmitters can begin a transmission since $T = \emptyset$. We assume that S_1 initiates; R detects the transmission since $\mathrm{d}(l_1,l') \leq \rho$ and begins receiving. Then S_2 can initiate its own transmission since $\{(l_1,\rho,c)\}|_{l_2,c} = \emptyset$, provoking interference at R; interference is detected by R, which stores $\perp$ and then continues its execution. The last two reductions model the end transmission events from S_1 and S_2.

Note that for the above interference to occur, condition $\mathrm{d}(l_1,l_2) > \rho$ (ensuring that S_2 is not in the transmission cell of S_1) is necessary. In fact, a process begins transmitting only if it sees the communication channel free.

Example 2 (*Begin Transmission Not in the Clear*). This example shows a receiver $\mathtt{in}(x).P$ which is reached by a begin transmission event while it is already in the transmission cell of another transmission. Thus such a receiver is not able to detect the begin transmission event.

The system is similar to the one above, but now R sends a value v before starting receiving, i.e., $R \overset{\text{def}}{=} m\,[\mathtt{out}\langle v\rangle.\mathtt{in}(x).P]^{c}_{l',\rho'}$. Also, as far as the topology of the system is concerned, $\mathrm{d}(l_1,l_2) > \rho$, $\mathrm{d}(l_1,l') \leq \rho$, $\mathrm{d}(l_2,l') \leq \rho$, $\mathrm{d}(l',l_1) > \rho'$ and $\mathrm{d}(l',l_2) > \rho'$, that is, R is in the transmission cell of both S_1 and S_2 while each of S_1 and S_2 is outside the transmission cells of all the other nodes.

$$N \overset{\text{def}}{=} n_1\,[\mathtt{out}\langle v_1\rangle.\mathbf{0}]^{c}_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^{c}_{l_2,\rho} \,\big|\, m\,[\mathtt{out}\langle v\rangle.\mathtt{in}(x).P]^{c}_{l',\rho'}$$

A possible computation is in Fig. 2. There R starts transmitting first (no node receives its transmission). While R is transmitting, S_1 starts transmitting too, and this is possible since it is outside the transmission cell of R. Then the transmission of R ends. Now R is willing to receive, but it cannot see the begin transmission event of S_2, even if it is in its transmission cell, since it is hidden by the transmission from S_1. The remaining reductions model the completion of the transmissions.

$$\begin{aligned}
\emptyset \rhd N &\hookrightarrow^{c}_{l',\rho'} n_1\,[\mathtt{out}\langle v_1\rangle.\mathbf{0}]^{c}_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^{c}_{l_2,\rho} \,\big|\, m\,[\langle v\rangle.\mathtt{in}(x).P]^{c}_{l',\rho'} \stackrel{\mathrm{def}}{=} N_1\\
\{(l',\rho',c)\} \rhd N_1 &\hookrightarrow^{c}_{l_1,\rho} n_1\,[\langle v_1\rangle.\mathbf{0}]^{c}_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^{c}_{l_2,\rho} \,\big|\, m\,[\langle v\rangle.\mathtt{in}(x).P]^{c}_{l',\rho'} \stackrel{\mathrm{def}}{=} N_2\\
\{(l_1,\rho,c),(l',\rho,c)\} \rhd N_2 &\hookrightarrow^{c}_{l',\rho'} n_1\,[\langle v_1\rangle.\mathbf{0}]^{c}_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^{c}_{l_2,\rho} \,\big|\, m\,[\mathtt{in}(x).P]^{c}_{l',\rho'} \stackrel{\mathrm{def}}{=} N_3\\
\{(l_1,\rho,c)\} \rhd N_3 &\hookrightarrow^{c}_{l_2,\rho} n_1\,[\langle v_1\rangle.\mathbf{0}]^{c}_{l_1,\rho} \,\big|\, n_2\,[\langle v_2\rangle.\mathbf{0}]^{c}_{l_2,\rho} \,\big|\, m\,[\mathtt{in}(x).P]^{c}_{l',\rho'} \stackrel{\mathrm{def}}{=} N_4\\
\{(l_1,\rho,c),(l_2,\rho,c)\} \rhd N_4 &\hookrightarrow^{c}_{l_1,\rho} n_1\,[\mathbf{0}]^{c}_{l_1,\rho} \,\big|\, n_2\,[\langle v_2\rangle.\mathbf{0}]^{c}_{l_2,\rho} \,\big|\, m\,[\mathtt{in}(x).P]^{c}_{l',\rho'} \stackrel{\mathrm{def}}{=} N_5\\
\{(l_2,\rho,c)\} \rhd N_5 &\hookrightarrow^{c}_{l_2,\rho} n_1\,[\mathbf{0}]^{c}_{l_1,\rho} \,\big|\, n_2\,[\mathbf{0}]^{c}_{l_2,\rho} \,\big|\, m\,[\mathtt{in}(x).P]^{c}_{l',\rho'}
\end{aligned}$$

Fig. 2. Begin transmission not in the clear.

Table 2
Labelled transitions for processes.

$\dfrac{[\![e]\!] = v}{\mathtt{out}\langle e\rangle.P \xrightarrow{!} \langle v\rangle.P}$ [PS-OUT$_{begin}$]		$\langle v\rangle.P \xrightarrow{!v} P$ [PS-OUT$_{end}$]
$\mathtt{in}(x).P \xrightarrow{?} (x).P$ [PS-IN$_{begin}$]	$\mathtt{in}(x).P \xrightarrow{?v} \mathtt{in}(x).P$ [PS-WAIT1$_{begin}$]	$\mathtt{in}(x).P \xrightarrow{?\bot} \mathtt{in}(x).P$ [PS-WAIT2$_{begin}$]
$(x).P \xrightarrow{?v} P\{v/x\}$ [PS-IN1$_{end}$]	$(x).P \xrightarrow{?\bot} P\{\bot/x\}$ [PS-IN2$_{end}$]	$\dfrac{\alpha \in \{?, ?v, ?\bot\} \quad P \notin IP}{P \xrightarrow{\alpha} P}$ [PS-NOIN]
		where IP is the set of processes of the form $\mathtt{in}(x).P_1$ or $(x).P_1$

4. Labelled Transition Semantics

Reflecting the structure of the language syntax, the Labelled Transition Semantics (LTS) has two sets of rules: those for the processes and those for the networks. Table 2 presents the LTS for processes. Transitions are of the form $P \xrightarrow{\alpha} P'$, where the grammar for α is

$$\alpha := \; ! \;\;|\;\; ? \;\;|\;\; !v \;\;|\;\; ?v \;\;|\;\; ?\bot$$

Labels ! and $!v$ represent begin transmission and end transmission events emanating from P; and ?, $?v$ represent begin transmission and end transmission events that reach P. Similarly $?\bot$ represents interference that reaches P. We comment on the rules. In PS-OUT$_{begin}$, the process evaluates expression e and initiates the transmission of the resulting value; in PS-OUT$_{end}$, the process terminates the transmission of v; in PS-IN$_{begin}$, the process captures a begin transmission; in PS-WAIT1$_{begin}$, the input process does not react to an end transmission event because it is waiting for a begin transmission event; in PS-WAIT2$_{begin}$, the input process stays idle since it receives a begin transmission which is not in the clear; in PS-IN1$_{end}$, the process receives v, and then continues its execution; in PS-IN2$_{end}$, the process receives noise (i.e., there is interference); PS-NOIN shows that non-input processes never react to the reception of events.

In the network semantics, a transition has the form

$$T \rhd N \xrightarrow{\mu} N'$$

where T is as in Section 3. The transition reads *"given the set T of active transmitters, at the occurrence of event μ, network N becomes network N'"*.

There are four possible kinds of transitions for networks:

(1) $T \rhd N \xrightarrow{c![l,r]} N'$: the node in N at location l becomes active and initiates a transmission over channel c with radius r;
(2) $T \rhd N \xrightarrow{c!v[l,r]} N'$: analogous to the previous one, for an end transmission;
(3) $T \rhd N \xrightarrow{c?[l,r]} N'$: the begin of a transmission over channel c and with radius r, produced by a node that is external to N and located at l, is propagated into N;
(4) $T \rhd N \xrightarrow{c?v[l,r]} N'$: analogous to the previous one, for an end transmission event.

Table 3 presents the LTS for networks. In the rules, the metavariable θ is a notational convention used to have fewer rules. Thus $?\theta$ ranges over ? and $?v$; similarly, $!\theta$ ranges over ! and $!v$.

In the LTS for networks, the key rules concern events reaching a single node (i.e., NS-IN$_1$, NS-IN$_2$, and NS-IN$_3$) and nodes emitting events (i.e., NS-OUT$_{begin}$ and NS-OUT$_{end}$). Rule NS-IN$_1$ shows the behavior of a node that receives an event in the clear. Condition $\mathrm{d}(l, l') \leq r'$ ensures that the node receives the event in the label, whereas $T|_{l,c} = \emptyset$ ensures that the node is not exposed to other transmissions. Rule NS-IN$_2$ shows that a node is not affected by events that do not reach it. Rule NS-IN$_3$ describes a node that detects colliding transmissions; conditions $\mathrm{d}(l, l') \leq r'$ and $T|_{l,c} \neq \emptyset$ ensure that at least two

Table 3
Labelled transitions for networks.

$$\frac{P \xrightarrow{!} P' \quad T|_{l,c} = \emptyset}{T \rhd n\,[P]^c_{l,r} \xrightarrow{c![l,r]} n\left[P'\right]^c_{l,r}}\ [\text{NS-OUT}_{begin}] \qquad \frac{P \xrightarrow{!v} P'}{T \rhd n\,[P]^c_{l,r} \xrightarrow{c!v[l,r]} n\left[P'\right]^c_{l,r}}\ [\text{NS-OUT}_{end}]$$

$$\frac{P \xrightarrow{?\theta} P' \quad \mathrm{d}(l,l') \leq r' \quad T|_{l,c} = \emptyset}{T \rhd n\,[P]^c_{l,r} \xrightarrow{c?\theta[l',r']} n\left[P'\right]^c_{l,r}}\ [\text{NS-IN}_1] \qquad \frac{\mathrm{d}(l,l') > r' \vee c \neq c'}{T \rhd n\,[P]^c_{l,r} \xrightarrow{c'?\theta[l',r']} n\,[P]^c_{l,r}}\ [\text{NS-IN}_2]$$

$$\frac{P \xrightarrow{?\bot} P' \quad \mathrm{d}(l,l') \leq r' \quad T|_{l,c} \neq \emptyset}{T \rhd n\,[P]^c_{l,r} \xrightarrow{c?[l',r']} n\left[P'\right]^c_{l,r}}\ [\text{NS-IN}_3] \qquad \frac{T \rhd N_1 \xrightarrow{c?\theta[l,r]} N_1' \quad T \rhd N_2 \xrightarrow{c!\theta[l,r]} N_2'}{T \rhd N_1|N_2 \xrightarrow{c!\theta[l,r]} N_1'|N_2'}\ [\text{NS-COM}_r]$$

$$\frac{T \rhd N_1 \xrightarrow{c!\theta[l,r]} N_1' \quad T \rhd N_2 \xrightarrow{c?\theta[l,r]} N_2'}{T \rhd N_1|N_2 \xrightarrow{c!\theta[l,r]} N_1'|N_2'}\ [\text{NS-COM}_l] \qquad \frac{T \rhd N_1 \xrightarrow{c?\theta[l,r]} N_1' \quad T \rhd N_2 \xrightarrow{c?\theta[l,r]} N_2'}{T \rhd N_1|N_2 \xrightarrow{c?\theta[l,r]} N_1'|N_2'}\ [\text{NS-COM}_{in}]$$

$$T \rhd \mathbf{0} \xrightarrow{c?\theta[l,r]} \mathbf{0}\ [\text{NS-NULL}]$$

$$\begin{aligned}
\emptyset \rhd N &\xrightarrow{c![l_1,\rho]} n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \big| n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \big| m\,[(x).P]^c_{l',\rho'} \stackrel{\text{def}}{=} N_1 \\
\{(l_1,\rho,c)\} \rhd N_1 &\xrightarrow{c![l_2,\rho]} n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \big| n_2\,[\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \big| m\,[P\{\bot/x\}]^c_{l',\rho'} \stackrel{\text{def}}{=} N_2 \\
\{(l_1,\rho,c),(l_2,\rho,c)\} \rhd N_2 &\xrightarrow{c!v_2[l_2,\rho]} n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \big| n_2\,[\mathbf{0}]^c_{l_2,\rho} \big| m\,[P\{\bot/x\}]^c_{l',\rho'} \stackrel{\text{def}}{=} N_3 \\
\{(l_1,\rho,c)\} \rhd N_3 &\xrightarrow{c!v_1[l_1,\rho]} n_1\,[\mathbf{0}]^c_{l_1,\rho} \big| n_2\,[\mathbf{0}]^c_{l_2,\rho} \big| m\,[P\{\bot/x\}]^c_{l',\rho'}
\end{aligned}$$

Fig. 3. Interference, with LTS.

transmissions reach the node. Rule NS-OUT$_{begin}$ shows that a transmission can be initiated by a node only when that node is not presently reached by another transmission. In rule NS-OUT$_{end}$, the transmission from a node terminates.

The propagation of events through the network is described by rules NS-COM$_r$, NS-COM$_l$, and NS-COM$_{in}$. In NS-COM$_r$, an event generated in a network is propagated to another network that executes in parallel (NS-COM$_l$ is symmetrical to NS-COM$_r$). NS-COM$_{in}$ shows two networks in parallel that detect the same event. Finally, NS-NULL allows the empty network to receive messages.

Now we give some examples. Since the LTS semantics is more complex we start with an example simpler than the ones shown for RST.

Example 3 (*Communication*). We consider a simple network N composed by just a transmitter and a receiver in its transmission cell:

$$N \stackrel{\text{def}}{=} n\,[\mathtt{out}\langle v\rangle.\mathbf{0}]^c_{l,\rho} \big| m\,[\mathtt{in}(x).P]^c_{l',\rho'}$$

At the process level we have transitions $\mathtt{out}\langle v\rangle.\mathbf{0} \xrightarrow{!} \langle v\rangle.\mathbf{0}$ from PS-OUT$_{begin}$ and $\mathtt{in}(x).P \xrightarrow{?} (x).P$ from PS-IN$_{begin}$. These transitions can be lifted to the network level using NS-OUT$_{begin}$ and NS-IN$_1$ respectively, and can then interact via NS-COM$_l$. A similar derivation can be done for the end transmission event. The obtained computation is:

$$\begin{aligned}
\emptyset \rhd N &\xrightarrow{c![l,\rho]} n\,[\langle v\rangle.\mathbf{0}]^c_{l,\rho} \big| m\,[(x).P]^c_{l',\rho'} \stackrel{\text{def}}{=} N_1 \\
\{(l,\rho,c)\} \rhd N_1 &\xrightarrow{c!v[l,\rho]} n\,[\mathbf{0}]^c_{l,\rho} \big| m\,[P\{v/x\}]^c_{l',\rho'}
\end{aligned}$$

The first event is the begin transmission from the transmitter S; this can happen since there are no active transmissions reaching S (i.e., $\emptyset|_{l,c} = \emptyset$). The receiver R, which was not previously reached by any transmission (i.e., $\emptyset|_{l',c} = \emptyset$) receives the transmission event and begins receiving that transmission. In the second event, the triple (l, r, c), representing S, is included in the set of active transmissions. In the final step, the transmitter completes the transmission of value v and R terminates receiving.

Example 4 (*Interference, with LTS*). We show how Example 1 is transposed in the LTS setting. The main difference is the introduction of full labels.

$$N \stackrel{\text{def}}{=} n_1\,[\mathtt{out}\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \big| n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \big| m\,[\mathtt{in}(x).P]^c_{l',\rho'}$$

The computation is shown in Fig. 3. First, S_1 initiates its transmission; R detects the transmission and begins receiving. Then, S_2, which cannot detect the transmission of S_1 (since $\{(l_1,\rho,c)\}|_{l_2,c} = \emptyset$), initiates its own transmission, causing interference at R. As a consequence, R records $\bot$ as result.

5. Harmony theorem

In this section we analyse the relationship between the RST from Section 3 and the LTS from Section 4, and show that they model the same behaviours.

The theorem has three parts. The first shows that the LTS is compatible with structural congruence; that is, application of structural congruence does not affect the possible transitions. The second shows that RST behaves as the LTS; i.e., each reduction in RST has a corresponding transition in the LTS, and the resulting networks are structurally congruent. The third part shows the inverse, which holds only for labels of the form $\xrightarrow{c![l,r]}$ and $\xrightarrow{c!v[l,r]}$, those that correspond to actual activities of the system (other labels are acknowledgments of activities from the environment).

Before proving the correspondence theorem we show some auxiliary lemmas that characterise the shape of processes able to perform a particular labelled transition, and the shape of the derivative process.

Lemma 5. *If $T \rhd N \xrightarrow{c?[l,r]} N'$ then*

$$N \equiv \prod_{i\in I} n_i\,[\mathtt{in}(x_i).P_i]^c_{l_i,r_i} \mid \prod_{j\in J} n_j\left[\mathtt{in}(x_j).P_j\right]^c_{l_j,r_j} \mid \prod_{k\in K} n_k\,[(x_k).P_k]^c_{l_k,r_k} \mid N''$$

where $\forall h \in I\cup J\cup K.\mathrm{d}(l,l_h) \le r$, $\forall i \in I.T|_{l_i,c} = \emptyset$, $\forall j \in J.T|_{l_j,c} \neq \emptyset$ and $(l,r,c) \not\Downarrow_i N''$. Furthermore if $[\![e]\!] = v$ then

$$N' \equiv \prod_{i\in I} n_i\,[(x_i).P_i]^c_{l_i,r_i} \mid \prod_{j\in J} n_j\left[\mathtt{in}(x_j).P_j\right]^c_{l_j,r_j} \mid \prod_{k\in K} n_k\,[P_k\{\perp/x_k\}]^c_{l_k,r_k} \mid N''$$

Proof. The proof is by rule induction on the derivation of $T \rhd N \xrightarrow{c?[l,r]} N'$. We have to consider only the rules able to produce labels of the form $\xrightarrow{c?[l,r]}$.

We consider just a few cases as examples.

Rule NS-IN$_1$ We know from the premise of the rule that $P \xrightarrow{?} P'$, thus by inspection on the LTS for processes we have two cases. Either the transition has been derived using rule PS-IN$_{begin}$ or using rule PS-NOIN. In the first case $P = \mathtt{in}(x_i).P_i$ and $P' = (x_i).P_i$, i.e., the thesis follows with $I = \{i\}$, $J = K = \emptyset$ and $N'' \equiv \mathbf{0}$. In the second case P is not an input, thus the corresponding node can be inserted in N''.

Rule NS-IN$_3$ Here too we have two cases, since $P \xrightarrow{?\perp} P'$ can be derived using either rule PS-WAIT2$_{begin}$ or rule PS-IN2$_{end}$. The two cases correspond to index P with $j \in J$ and with $k \in K$ respectively.

NS-COM$_{in}$ This is the inductive case. It is enough to take the union of the corresponding sets of indices in the two premises and the parallel composition of the non-indexed part of the networks (denoted by N'' in the statement of the lemma) to put the terms in the desired form. □

Lemma 6. *If $T \rhd N \xrightarrow{c![l,r]} N'$ then*

$$N \equiv n\,[\mathtt{out}\langle e\rangle.P]^c_{l,r} \mid \prod_{i\in I} n_i\,[\mathtt{in}(x_i).P_i]^c_{l_i,r_i} \mid \prod_{j\in J} n_j\left[\mathtt{in}(x_j).P_j\right]^c_{l_j,r_j} \mid \prod_{k\in K} n_k\,[(x_k).P_k]^c_{l_k,r_k} \mid N''$$

where $\forall h \in I\cup J\cup K.\mathrm{d}(l,l_h) \le r$, $\forall i \in I.T|_{l_i,c} = \emptyset$, $\forall j \in J.T|_{l_j,c} \neq \emptyset$ and $(l,r,c) \not\Downarrow_i N''$. Furthermore if $[\![e]\!] = v$ then

$$N' \equiv n\,[\langle v\rangle.P]^c_{l,r} \mid \prod_{i\in I} n_i\,[(x_i).P_i]^c_{l_i,r_i} \mid \prod_{j\in J} n_j\left[\mathtt{in}(x_j).P_j\right]^c_{l_j,r_j} \mid \prod_{k\in K} n_k\,[P_k\{\perp/x_k\}]^c_{l_k,r_k} \mid N''$$

Proof. The proof is similar to the one for Lemma 5, and it uses Lemma 5 itself to handle premises which are input transitions. □

Lemma 7. *If $T \rhd N \xrightarrow{c?v[l,r]} N'$ then*

$$N \equiv \prod_{i\in I} n_i\,[(x_i).P_i]^c_{l_i,r_i} \mid \prod_{j\in J} n_j\left[\mathtt{in}(x_j).P_j\right]^c_{l_j,r_j} \mid N''$$

where $\forall h \in I\cup J.\mathrm{d}(l,l_h) \le r$ and $(l,r,c) \not\Downarrow_i N''$. Furthermore

$$N' \equiv \prod_{i\in I} n_i\,[P_i\{v/x_i\}]^c_{l_i,r_i} \mid \prod_{j\in J} n_j\left[\mathtt{in}(x_j).P_j\right]^c_{l_j,r_j} \mid N''$$

Proof. The proof is similar to the one for Lemma 5, using rules for end transmission instead of rules for begin transmission. □

Lemma 8. *If $T \triangleright N \xrightarrow{c!v[l,r]} N'$ then*

$$N \equiv n\,[\langle v\rangle.P]^c_{l,r} \mid \prod_{i\in I} n_i\,[(x_i).P_i]^c_{l_i,r_i} \mid \prod_{j\in J} n_j\left[\mathtt{in}(x_j).P_j\right]^c_{l_j,r_j} |N''$$

where $\forall h \in I \cup J.\mathrm{d}(l, l_h) \le r$ and $(l, r, c) \not\Downarrow_i N''$. Furthermore

$$N' \equiv n\,[P]^c_{l,r} \mid \prod_{i\in I} n_i\,[P_i\{v/x_i\}]^c_{l_i,r_i} \mid \prod_{j\in J} n_j\left[\mathtt{in}(x_j).P_j\right]^c_{l_j,r_j} |N''$$

Proof. The proof is similar to the one for Lemma 6, and it uses Lemma 7 to handle premises which are input transitions. □

Theorem 9 (*Harmony Theorem*). *Let N be a network, and T a set of active transmitters.*

(1) *If $T \triangleright N \xrightarrow{\mu} N'$ and $N \equiv N_1$ then there exists N_1' such that $T \triangleright N_1 \xrightarrow{\mu} N_1' \equiv N'$.*
(2) *If $T \triangleright N \hookrightarrow^c_{l,r} N'$ then*
  (a) *either $T \triangleright N \xrightarrow{c![l,r]} N_1' \equiv N'$;*
  (b) *or there is v such that $T \triangleright N \xrightarrow{c!v[l,r]} N_1' \equiv N'$.*
(3) *The converse also holds, i.e., if $T \triangleright N \xrightarrow{c![l,r]} N'$ or $T \triangleright N \xrightarrow{c!v[l,r]} N'$ then also $T \triangleright N \hookrightarrow^c_{l,r} N'$.*

Proof. We consider the different points in order.

(1) Commutativity is ensured since rules NS-Com$_r$ and NS-Com$_l$ are symmetric while rule NS-Com$_{in}$ is self-symmetric. Identity over the empty network holds since, due to rule NS-Null, the empty network can perform any label of the form $\xrightarrow{c?\theta[l,r]}$, which acts as neutral element of parallel composition. Finally associativity follows since the operations performed on labels by rules for parallel composition are associative and the structure of the network is always preserved.
(2) The proof is by rule induction on the derivation of $T \triangleright N \hookrightarrow^c_{l,r} N'$. Let us consider rule RST-BEGIN. The proof for this case is by induction on the size of $I \cup J \cup K$. The base case is trivial, using rules PS-Out$_{begin}$ and NS-Out$_{begin}$. Let us consider the inductive case. Let us remove one element h from $I \cup J \cup K$ (notice that the three sets are disjoint). By the inductive hypothesis we have a transition with label $\xrightarrow{c![l,r]}$ for the resulting process, which has fewer components (it is easy to check that, since the transition represents a begin transmission event, the other form of label, $\xrightarrow{c!v[l,r]}$, is not possible). We have to consider different cases according to which set h belongs to. Suppose that $h \in I$. Then from rule PS-In$_{begin}$ we have $\mathtt{in}(x_i).P_i \xrightarrow{?} (x_i).P_i$. Then we can apply rule NS-In$_1$ since $\mathrm{d}(l_i, l) \le r$ and $T|_{l_i,c} = \emptyset$, from the precondition of rule RST-BEGIN. The desired transition can thus be proved using rule NS-Com$_l$.

Suppose now that $h \in J$. The above rule can no longer be applied, since the precondition is no longer satisfied. Thus one must use rule PS-Wait2$_{begin}$ to derive $\mathtt{in}(x_j).P_j \xrightarrow{?\perp} \mathtt{in}(x_j).P_j$, and then rule NS-In$_3$ to derive a transition with label $\xrightarrow{c?[l,r]}$. The desired transition can thus be proved using rule NS-Com$_l$.

Finally suppose that $h \in K$. We can use rule PS-In2$_{end}$ to derive $(x_k).P_k \xrightarrow{?\perp} P_k\{\perp/x_k\}$. This transition can be lifted to an input at the network level using rule NS-In$_3$ since $\mathrm{d}(l_k, l) \le r$ from the premise of rule RST-BEGIN and $T|_{l_k,c} \ne \emptyset$ since a process can be an active input only if reached by a transmission.

Rule RST-END can be simulated in a similar way, just note that now the correspondence is the one described in case (b) of the theorem and that rules for end transmission events should be used.

The proof is similar also for rule RST-CONT, since the predicate $(l, r, c) \not\Downarrow_i N''$ guarantees that all the input nodes satisfy the conditions of rule NS-In$_2$. For non-input nodes, either they satisfy the precondition of rule NS-In$_2$, or rule PS-NoIn can be applied, followed by either rule NS-In$_1$ or rule NS-In$_3$.

Finally, rule RST-CONGR can be simulated as by the first part of the theorem.
(3) The proof is based on Lemma 6 for transitions of the form $T \triangleright N \xrightarrow{c![l,r]} N'$ and on Lemma 8 for transitions of the form $T \triangleright N \xrightarrow{c!v[l,r]} N'$. In the first case the desired reduction can be derived using rule RST-BEGIN and RST-CONGR if component N'' in the statement of Lemma 6 is empty. Otherwise N'' can be added using rule RST-CONT. The proof is similar in the second case, using rule RST-END instead of rule RST-BEGIN. □

6. A local Reduction Semantics

Now we give an alternative presentation of the RS in Section 3 which has a more local management of the information concerning the active transmissions. In particular, we put marks $\ominus$ in front of process P, one for each transmission that is currently reaching it, even if P is not an active receiver. This is required to handle situations such as Example 12. For this, we extend the syntax of processes with the additional production

$$P \stackrel{\text{def}}{=} \ominus P$$

We call *marked* these processes and use Q to range over them. To have a correct modelling of process behaviours we have to start with a proper marking of processes: marks are always inserted as prefixes of top level processes, and each process currently reached by n transmissions should be prefixed by n marks. A network N is *coherently marked* if it is structural congruent to a network N' where all the processes are marked as described above. As we show in Lemma 10, this property is preserved by the rules.

We also extend the structural congruence with rule

$$\ominus\langle v\rangle.Q \equiv \langle v\rangle.\ominus Q$$

To avoid ambiguity, we write $\equiv_D$ for this extended structural congruence. The new rule allows active outputs to complete their execution even if reached by other transmissions.

In the rules below we adopt a shortcut similar to $(l, r, c) \not\Downarrow_i N$ used until now: we define $(l, r, c) \not\Downarrow N$ to hold if network N contains no nodes $n\,[P]^c_{l',r'}$ for which $\mathrm{d}(l, l') \leq r$ holds. The need for the different notation occurs because now transmitters in the cell of a transmission are influenced by it, since their marking should be updated.

As in Section 3 we need two rule schemas, one for begin transmission and one for end transmission, and two closure rules. These rules are similar to those in Section 3, the only difference being caused by the different way the checks for interference are performed. The rules exploit the metaoperators $\odot$ and $\otimes$, defined afterward, to compute the effect of a transmission on a process.

[LRS-BEGIN]

$$\frac{\forall i \in I.\mathrm{d}(l, l_i) \leq r \qquad [\![\![e]\!]\!] = v}{n\,[\mathtt{out}\langle e\rangle.Q]^c_{l,r} \mid \prod_{i\in I} n_i\,[Q_i]^c_{l_i,r_i} \longmapsto^c_{l,r} n\,[\langle v\rangle.Q]^c_{l,r} \mid \prod_{i\in I} n_i\,[\otimes Q_i]^c_{l_i,r_i}}$$

[LRS-END]

$$\frac{\forall i \in I.\mathrm{d}(l, l_i) \leq r}{n\,[\langle v\rangle.Q]^c_{l,r} \mid \prod_{i\in I} n_i\,[Q_i]^c_{l_i,r_i} \longmapsto^c_{l,r} n\,[Q]^c_{l,r} \mid \prod_{i\in I} n_i\,[v \odot Q_i]^c_{l_i,r_i}}$$

[LRS-CONT] $\dfrac{N \longmapsto^c_{l,r} N' \quad (l, r, c) \not\Downarrow N''}{N|N'' \longmapsto^c_{l,r} N'|N''}$

[LRS-CONGR] $\dfrac{N \equiv_D N' \longmapsto^c_{l,r} N'' \equiv_D N'''}{N \longmapsto^c_{l,r} N'''}$

The definitions of the metaoperators $\otimes$ and $\odot$ (i.e., their effect on the marking of the processes) is as follows:

$$\begin{aligned}
\otimes\mathtt{in}(x).Q &= \ominus(x).Q \\
\otimes \ominus (x).Q &= \ominus \ominus Q\{\bot/x\} \\
\otimes \ominus Q &= \ominus \ominus Q && \text{if } Q \text{ is not of the form } (x).P \\
\otimes\mathtt{out}\langle e\rangle.Q &= \ominus\mathtt{out}\langle e\rangle.Q \\
\otimes\langle v\rangle.Q &= \ominus\langle v\rangle.Q \\
\\
v \odot \ominus(x).Q &= Q\{v/x\} \\
v \odot \ominus Q &= Q && \text{if } Q \text{ is not of the form } (x).P
\end{aligned}$$

Rule LRS-BEGIN differs with respect to rule RST-BEGIN since the effect of the begin transmission events on the different processes is computed by the operation $\otimes Q_i$, and depends on the marking of Q_i. If Q_i is not marked and is an input then it starts receiving (and a mark is added). If it is an active input then it must already be marked, by the coherence conditions on marking, and thus interference occurs (and again a mark is added). In all the other cases a mark is added. This rule must explicitly consider also other transmitters in the transmission cell, since they should be marked. This is reflected in the context closure rule LRS-CONT, that requires other transmitters to be outside the transmission cell. In contrast, in the rules for RST (Section 3) any transmitter can be added via context closure. Note that we do not need to check that the output process is not reached by a transmission, as this is guaranteed by the absence of marks in the process.

Rule LRS-END is essentially as rule RST-END, the only difference is that a mark is removed from each receiving process.

Rule LRS-CONT allows only processes outside the transmission cell to be added in parallel, since in all the otherprocesses the marking must be updated.

Finally, rule LRS-CONGR also allows to move marks inside active outputs, to make rule LRS-END applicable to active outputs even if they are marked. This is required since end transmission events should be enabled when the transmitter is reached by other transmissions.

Lemma 10. *If network N is coherently marked and $N \longmapsto^c_{l,r} N'$ then N' is coherently marked too.*

Proof. By rule inspection one can check that whenever a begin transmission occurs all the processes in its transmission cell receive an additional mark at the top level, while when an end transmission event occurs a mark is removed from all the processes in its transmission cell. □

$$
\begin{array}{lll}
N & \overset{\text{def}}{=} & n_1\,[\mathtt{out}\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\mathtt{in}(x).P]^c_{l',\rho'} \\
 & \longmapsto^c_{l_1,\rho} & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\otimes\mathtt{in}(x).P]^c_{l',\rho'} \\
 & = & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\ominus(x).P]^c_{l',\rho'} \\
 & \longmapsto^c_{l_2,\rho} & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\otimes\ominus(x).P]^c_{l',\rho'} \\
 & = & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\ominus\ominus P\{\bot/x\}]^c_{l',\rho'} \\
 & \longmapsto^c_{l_2,\rho} & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[v_2 \odot \ominus\ominus P\{\bot/x\}]^c_{l',\rho'} \\
 & = & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\ominus P\{\bot/x\}]^c_{l',\rho'} \\
 & \longmapsto^c_{l_1,\rho} & n_1\,[\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[v_1 \odot \ominus P\{\bot/x\}]^c_{l',\rho'} \\
 & = & n_1\,[\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[P\{\bot/x\}]^c_{l',\rho'}
\end{array}
$$

Fig. 4. Interference, with LRS.

$$
\begin{array}{lll}
N & \overset{\text{def}}{=} & n_1\,[\mathtt{out}\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\mathtt{out}\langle v\rangle.\mathtt{in}(x).P]^c_{l',\rho'} \\
 & \longmapsto^c_{l',\rho'} & n_1\,[\mathtt{out}\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\langle v\rangle.\mathtt{in}(x).P]^c_{l',\rho'} \\
 & \longmapsto^c_{l_1,\rho} & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\otimes\langle v\rangle.\mathtt{in}(x).P]^c_{l',\rho'} \\
 & = & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\ominus\langle v\rangle.(x).P]^c_{l',\rho'} \\
 & \longmapsto^c_{l',\rho'} & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\ominus\mathtt{in}(x).P]^c_{l',\rho'} \\
 & \longmapsto^c_{l_2,\rho} & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\otimes\ominus\,\mathtt{in}(x).P]^c_{l',\rho'} \\
 & = & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\ominus\ominus\,\mathtt{in}(x).P]^c_{l',\rho'} \\
 & \longmapsto^c_{l_1,\rho} & n_1\,[\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[v_1 \odot \ominus\ominus\,\mathtt{in}(x).P]^c_{l',\rho'} \\
 & = & n_1\,[\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\ominus\mathtt{in}(x).P]^c_{l',\rho'} \\
 & \longmapsto^c_{l_2,\rho} & n_1\,[\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[v_2 \odot \ominus\mathtt{in}(x).P]^c_{l',\rho'} \\
 & = & n_1\,[\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\mathtt{in}(x).P]^c_{l',\rho'}
\end{array}
$$

Fig. 5. Begin transmission not in the clear, with LRS.

Example 11 (*Interference, with LRS*). We discuss now how the interference situation in Example 1 is modelled using LRS. The computation is in Fig. 4. When R becomes active, it is marked since it is in the cell of an ongoing transmission. Thus, when S_2 starts its transmission (and this is possible since it is not marked), it provokes interference at R. Now R is reached by two transmissions, so it is marked by $\ominus\ominus$. As transmissions terminate, the marking is removed, thus allowing $P\{\bot/x\}$ to continue its execution.

Example 12 (*Begin Transmission Not in the Clear, with LRS*). We show how LRS can be used in the scenario of Example 2. The corresponding computation is in Fig. 5. Now, when S_2 begins its transmission, R is marked and thus cannot receive the begin transmission event. Notice that R has been marked while it was an output: this scenario explains why a marking is required also for non-input processes. To derive the third reduction the extended structural congruence $\equiv_D$ is required, since R should perform an end transmission event while marked.

We conclude this section with the comparison between the LRS presented here and the RST presented in Section 3. For this, we need to be able to move between the two representations of the information about active transmissions. We introduce two additional notations to this end. Given a set of transmitters T and a network N without marks, we denote as $\mathrm{mark}(N, T)$ the unique network obtained by adding for each transmitter in T, a mark on all processes in its transmission cell. Conversely we write $\mathrm{unmark}(N)$ for the unmarked version of network N, i.e., a network obtained from N by removing all marks. We can now state the correspondence theorem.

Theorem 13. *Let N be a network, and T a set of active transmitters. Then $T \rhd N \hookrightarrow^c_{l,r} N'$ iff* $\mathrm{mark}(N, T) \longmapsto^c_{l,r} N''$ *and* $\mathrm{unmark}(N'') = N'$.

Proof. The proof is obtained by a comparison between corresponding rules in the two RSs, noticing that a process P located at l and synchronized on channel c is marked iff $T|_{l,c} \neq \emptyset$. Also, the check $T|_{l,c} = \emptyset$ for the output process, that appears in

Table 4
Rules for end transmission events.

[GRS-END]
$$\frac{\forall i \in I.\mathrm{d}(l, l_i) \leq r}{n\,[\langle v\rangle.P]^c_{l,r} \mid \prod_{i\in I} n_i\,[(x_i).P_i]^c_{l_i,r_i} \mid \prod_{j\in J} n_j\left[\mathtt{in}(x_j).P_j\right]^c_{l_j,r_j} \rightharpoonup^c_{l,r} n\,[P]^c_{l,r} \mid \prod_{i\in I} n_i\,[P_i\{v/x_i\}]^c_{l_i,r_i} \mid \prod_{j\in J} n_j\left[\mathtt{in}(x_j).P_j\right]^c_{l_j,r_j}}$$

[GRS-END-CONT] $\dfrac{E \rightharpoonup^c_{l,r} E' \quad (l, r, c) \not\Downarrow_i E''}{E|E'' \rightharpoonup^c_{l,r} E'|E''}$

rule RST-BEGIN, is implicit in rule LRS-BEGIN. In fact, if the condition is not satisfied then, in the local setting, the process is marked, and this forbids to apply the rule. Finally the difference between the two context closures is compensated by the fact that rules LRS-BEGIN and LRS-END deal also with transmitters in the transmission cell of the corresponding event. □

7. A more robust Reduction Semantics

The RS presented in Section 6 stores in the process the information about active transmissions. Clearly, this relies on the fact that this information is static. However, if one considers extensions of the language that allow to change the channel used for communication or the location of nodes, then this assumption fails. For instance, if a node changes its communication channel, its entire marking has to be recomputed.

Driven by this observation, we propose a different RS where each reduction directly checks whether or not there are other transmitters interfering with it. This does not rely on additional information such as marks, and can thus be straightforwardly extended to handle more general situations, such as the channel switch primitive in Section 8. To reach this goal we need a few more rules. We call this semantics GRS (generalisable RS).

The begin transmission event is the most complex case since it may originate interference. In GRS it is modelled as a sequence of subreductions of three different kinds: the first one tags the recipients of the communication; while the other two normalize them while checking for interference.

We extend the network terms of Table 1; we call *tagged terms* the extended terms and we use E to range over them:

$$E \stackrel{\text{def}}{=} N \;\mid\; \langle\!| N |\!\rangle_n \;\mid\; E|E$$

where N is as by Table 1 and n is an identifier. The term $\langle\!| N |\!\rangle_n$ represents a network of receivers that has been tagged since it is reached by a begin transmission event that originated at n. Structural congruence and the shortcuts introduced in previous sections are extended to tagged terms in the obvious way.

We need an additional notation. $E \not\Downarrow (l, c)$ holds if network E contains no active transmitter $n\,[\langle v\rangle.P]^c_{l',r}$, for which $d(l, l') \leq r$ holds (i.e., no transmission on c emanating from E presently reaches l).

Formally, a reduction step $N \rightharpoonup^c_{l,r} N'$ occurs if one of the following two cases holds.

(Begin Transmission) there are E and l' such that

$$N \longrightarrow^c_{l,r} E \succ^{c\,*}_{l'} \succ^* N'$$

where $\longrightarrow^c_{l,r}$, $\succ^c_l$ and $\succ$ are defined in Table 5 and $\succ^*$ is the transitive closure of $\succ$ (and similarly for $\succ^c_{l'}$, where l' needs not to be always the same).

(End Transmission) the reduction is derived from the rule schema and the context closure in Table 4.

In each group of rules, for $\longrightarrow^c_{l,r}$, $\succ^c_l$, $\succ$ and $\rightharpoonup^c_{l,r}$, we omit structural closures like RST-CONGR in Section 3

We discuss the rules in detail. Rule GRS-BEGIN is similar to rule LRS-BEGIN used in previous section, but here only receivers are considered. They are just tagged with the name of the node performing the begin transmission event. This is necessary since in this semantics there is no information in the term about active transmissions. Thus we have to introduce some auxiliary relations to perform the required checks for interference. If n is the node that performs the begin transmission event, context closure rule GRS-BEGIN-CONT allows, in the context, only receivers out of the range of n and transmitters that, if active, do not reach n. This last check was not required in LRS: if such a transmitter existed then $\mathtt{out}\langle e\rangle.P$ would be marked and could not reduce.

Rule GRS-LNORM considers the effect of begin transmissions on receivers. The reduction is labelled by the location and the channel of the receiver so that context closure rule (GRS-LNORM-CONT) can check that the receiver is not reached by other transmissions (i.e., the begin transmission event is received in the clear). One such reduction is required for each receiver. We have to include the transmitter m performing the begin transmission in the rule, since its transmission actually reaches the receiver and thus, if m were in the context, then the check in the context closure rule would fail.

The other kind of reduction considers instead receivers that are reached by the begin transmission event but which do not receive it in the clear. In rule GRS-COVER the receiver is already reached by another transmission, thus it will not become active. The condition $m \neq m'$ checks that the two transmissions are distinct. Rule GRS-INTERF concerns an active receiver, which as such is listening to some other transmission, and in this case an interference occurs. The context closure GRS-NORM-CONT has no additional conditions, since other transmissions cannot influence the reduction.

Table 5
Reduction rules for $\longrightarrow^c_{l,r}$, $\succ^c_l$ and $\succ$.

Rules for $\longrightarrow^c_{l,r}$

[GRS-BEGIN]

$$\frac{\forall i \in I.\mathrm{d}(l, l_i) \leq r \qquad \forall j \in J.\mathrm{d}(l, l_j) \leq r \qquad [\![e]\!] = v}{n\,[\mathtt{out}\langle e\rangle.P]^c_{l,r} \mid \prod_{i\in I} n_i\,[(x_i).P_i]^c_{l_i,r_i} \mid \prod_{j\in J} n_j\left[\mathtt{in}(x_j).P_j\right]^c_{l_j,r_j} \longrightarrow^c_{l,r} n\,[\langle v\rangle.P]^c_{l,r} \mid \prod_{i\in I} \llparenthesis n_i\,[(x_i).P_i]^c_{l_i,r_i} \rrparenthesis_n \mid \prod_{j\in J} \llparenthesis n_j\left[\mathtt{in}(x_j).P_j\right]^c_{l_j,r_j} \rrparenthesis_n}$$

[GRS-BEGIN-CONT]

$$\frac{E \longrightarrow^c_{l,r} E' \quad (l,r,c) \not\Downarrow_i E'' \quad E'' \not\downarrow (l,c)}{E|E'' \longrightarrow^c_{l,r} E'|E''}$$

Rules for $\succ^c_l$

[GRS-LNORM]

$$m\,[\langle v_m\rangle.P_m]^c_{l_m,r_m} \mid \llparenthesis n\,[\mathtt{in}(x).P]^c_{l,r} \rrparenthesis_m \succ^c_l m\,[\langle v_m\rangle.P_m]^c_{l_m,r_m} \mid n\,[(x).P]^c_{l,r}$$

[GRS-LNORM-CONT]

$$\frac{E_1 \succ^c_l E'_1 \quad E_2 \not\downarrow (l,c)}{E_1 \mid E_2 \succ^c_l E'_1 \mid E_2}$$

Rules for $\succ$

[GRS-COVER]

$$\frac{\mathrm{d}(l, l_m) \leq r_m \qquad m \neq m'}{m\,[\langle v_m\rangle.P_m]^c_{l_m,r_m} \mid \llparenthesis n\,[\mathtt{in}(x).P]^c_{l,r} \rrparenthesis_{m'} \succ m\,[\langle v_m\rangle.P_m]^c_{l_m,r_m} \mid n\,[\mathtt{in}(x).P]^c_{l,r}}$$

[GRS-INTERF]

$$\llparenthesis n\,[(x).P]^c_{l,r} \rrparenthesis_m \succ n\,[\{\bot/x\}P]^c_{l,r}$$

[GRS-NORM-CONT]

$$\frac{E_1 \succ E'_1}{E_1 \mid E_2 \succ E'_1 \mid E_2}$$

$$\begin{array}{lll}
N & \overset{\text{def}}{=} & n_1\,[\mathtt{out}\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \Big| n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \Big| m\,[\mathtt{in}(x).P]^c_{l',\rho'} \\
& \longrightarrow^c_{l_1,\rho} & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \Big| n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \Big| \llparenthesis m\,[\mathtt{in}(x).P]^c_{l',\rho'} \rrparenthesis_{n_1} \\
& \succ^c_{l'} & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \Big| n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \Big| m\,[(x).P]^c_{l',\rho'} \\
& \longrightarrow^c_{l_2,\rho'} & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \Big| n_2\,[\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \Big| \llparenthesis m\,[(x).P]^c_{l',\rho'} \rrparenthesis_{n_2} \\
& \succ & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \Big| n_2\,[\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \Big| m\,[P\{\bot/x\}]^c_{l',\rho'} \\
& \rightharpoonup^c_{l_1,\rho} & n_1\,[\mathbf{0}]^c_{l_1,\rho} \Big| n_2\,[\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \Big| m\,[P\{\bot/x\}]^c_{l',\rho'} \\
& \rightharpoonup^c_{l_2,\rho'} & n_1\,[\mathbf{0}]^c_{l_1,\rho} \Big| n_2\,[\mathbf{0}]^c_{l_2,\rho} \Big| m\,[P\{\bot/x\}]^c_{l',\rho'}
\end{array}$$

Fig. 6. Interference, with GRS.

Rule GRS-END and the context closure GRS-END-CONT are like the corresponding rules of RST (Section 3), but without T. Indeed, T, which is fundamental to compute the behavior of begin transmission events, is not necessary for end transmissions.

We show now how the examples in the previous sections can be modelled using GRS.

Example 14 (*Interference, with GRS*). In Fig. 6 we revisit the interference scenario from Example 11 using GRS. The main difference is that here, after a whole reduction has been computed, we obtain a network without additional marks. On the other hand, each of the first two reductions requires two (but in general there can be more) steps to be computed. For instance in the first reduction the first step just tags the receiver, which becomes $\llparenthesis m\,[\mathtt{in}(x).P]^c_{l',\rho'} \rrparenthesis_{n_1}$, and another step is required to make it active by checking that no other transmitter is interfering.

In contrast, end of transmission events are modelled essentially as before.

Example 15 (*Begin Transmission Not in the Clear, with GRS*). The scenario considering a begin transmission event not in the clear (Example 12) is modelled in Fig. 7. The most important step is the begin transmission of S_2: using rule GRS-BEGIN the possible receiver R is tagged, but then rule GRS-COVER has to be applied, thus reception will not start. It is not possible to apply rule GRS-LNORM since, when S_1 has to be added to the context, the condition of rule GRS-LNORM-CONT fails.

We conclude this section with the comparison with RST (Section 3).

Theorem 16. *Let N be a network, and T the set of active transmitters in N. Then $T \rhd N \hookrightarrow^c_{l,r} N'$ iff $N \rightharpoonup^c_{l,r} N'$.*

$$\begin{aligned}
N \;&\overset{\text{def}}{=} && n_1\,[\mathtt{out}\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\mathtt{out}\langle v\rangle.\mathtt{in}(x).P]^c_{l',\rho'}\\
&\longrightarrow^c_{l',\rho'} && n_1\,[\mathtt{out}\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\langle v\rangle.\mathtt{in}(x).P]^c_{l',\rho'}\\
&\longrightarrow^c_{l_1,\rho} && n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\langle v\rangle.\mathtt{in}(x).P]^c_{l',\rho'}\\
&\rightharpoonup^c_{l',\rho'} && n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathtt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\mathtt{in}(x).P]^c_{l',\rho'}\\
&\longrightarrow^c_{l_2,\rho} && n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, \langle\!|\, m\,[\mathtt{in}(x).P]^c_{l',\rho'} \,|\!\rangle_{n_2}\\
&\succ && n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\mathtt{in}(x).P]^c_{l',\rho'}\\
&\rightharpoonup^c_{l_1,\rho} && n_1\,[\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\mathtt{in}(x).P]^c_{l',\rho'}\\
&\rightharpoonup^c_{l_2,\rho} && n_1\,[\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\mathtt{in}(x).P]^c_{l',\rho'}
\end{aligned}$$

Fig. 7. Begin transmission not in the clear, with GRS.

Table 6
Extended syntax for processes.

P	$\overset{\text{def}}{=}$	...	*Old processes*	|	$\mathtt{in}(x.)P\lceil Q$	*Handled input*
	|	$\mathtt{if}\ e\ \mathtt{then}\ P\ \mathtt{else}\ Q$	*Conditional*	|	$\mathtt{switch}\ c.P$	*Channel switch*
	|	$C[\vec{v}]$	*Constant invocation*			

Proof. The proof is trivial for end transmission events. For begin transmission events note that all the receivers considered in rule RST-BEGIN are tagged in rule GRS-BEGIN. The check $T|_{l,c} = \emptyset$ for the output process is done by the additional condition in the context closure GRS-BEGIN-CONT. We have to show that all the tagged inputs evolve as in rule RST-BEGIN. For active inputs, rule GRS-INTERF is applied producing interference. For inputs, two cases are possible: either they are reached by no other transmissions, thus rule GRS-LNORM can be applied to make them active since the precondition of context closure GRS-LNORM-CONT is satisfied; or they are reached by at least one other transmission (different from the one beginning in this reduction, since $m \neq m'$) thus one can find a node m to apply rule GRS-COVER and have the input node not evolve. This is exactly what happens using rule RST-BEGIN.

Since all the correspondences analysed are bijective, the reverse implication follows too. □

8. The full language

In the previous sections we have considered the subset of CWS with only the operators necessary for communication. We now discuss the full calculus. A few more challenging extensions are outlined in Section 10.2. The additional productions for processes are in Table 6. The syntax for networks is unchanged.

The first addition is both an extension and a modification of the core calculus: the input construct is replaced by the *handled input* construct $\mathtt{in}(x.)P_1\lceil P_2$. This process can either receive a transmission, or trigger the handler P_2. In the first case, when the reception begins the process becomes an active receiver $(x).P_1$. In the second case, the process evolves to process P_2. The handled input enables the modelling of scenarios in which a device, due to the verification of internal conditions (e.g., timeouts, interrupts, exceptions) stops waiting for incoming messages. This situation is common in modern wireless systems (for instance, in communication protocols that implement techniques for preventing starvation). Note that input can be modelled using handled input.

The other additions include the construct $\mathtt{if}\ e\ \mathtt{then}\ P_1\ \mathtt{else}\ P_2$, representing a process that behaves according to the truth value $[\![e]\!]$ and a construct $\mathtt{switch}\ c.P$ representing a process that decides to change its communication channel to c, and then continues as P.

As further addition we consider constants to model devices with an infinite behavior. The behavior of a constant $C[\vec{v}]$ is defined by a constant definition of the form $C[\vec{x}] \overset{\text{def}}{=} P$. We require constant occurrences in constant definitions to be guarded by prefixes.

We only examine the addition of the new operators to the RST reduction semantics, as this is the simplest one and the one that we will use in the application in Section 9. However the operators can be introduced in a similar way into the other semantics. The only exception is the addition of the channel switch primitive to LRS: this is more difficult since the new primitive changes the whole information about ongoing transmissions, which should thus be recomputed. Instead of the simple rule RST-SWITCH below, one should use a rule schema to be instantiated according to the number i of active transmitters currently transmitting on the chosen channel c' and reaching the node n. This rule should remove from the process at n all the marks (which referred transmissions on the old channel c), and add i marks. Also, the context closure rule for channel switch should forbid contexts containing active transmitters of this kind.

To update the RST semantics one first needs to introduce the rule $C[\vec{v}] \equiv P\{\vec{v}/\vec{x}\}$ if $C[\vec{x}] \stackrel{\text{def}}{=} P$ in the structural congruence for processes, and lift it to the network level.

A main consequence of the additions in the operational semantics is the introduction of transitions representing activities internal to the nodes (the analogous of the τ-steps in languages such as CCS). Examples of internal activities are the switch of a communication channel and the activation of an input handler. We denote these activities with the transition symbol $\hookrightarrow$.

The operational rules below, plus the rules RST-CONT and RST-CONGR from Section 3, define the semantics of the full calculus.

[RST-BEGIN]

$$\frac{T|_{l,c} = \emptyset \quad \forall h \in I \cup J \cup K.\mathrm{d}(l, l_h) \le r \quad \forall i \in I.T|_{l_i,c} = \emptyset \quad \forall j \in J.T|_{l_j,c} \neq \emptyset}{\begin{array}{c} T \rhd n\,[\mathtt{out}\langle e\rangle.P]^c_{l,r} \mid \prod_{h\in I\cup J} n_h\,[\mathtt{in}(x_h.)P_h\lceil Q_h]^c_{l_h,r_h} \mid \prod_{k\in K} n_k\,[(x_k).P_k]^c_{l_k,r_k} \hookrightarrow^c_{l,r} \\ n\,[\langle [\![\![e]\!]\!]\rangle.P]^c_{l,r} \mid \prod_{i\in I} n_i\,[(x_i).P_i]^c_{l_i,r_i} \mid \prod_{j\in J} n_j\left[\mathtt{in}(x_j.)P_j\lceil Q_j\right]^c_{l_j,r_j} \mid \prod_{k\in K} n_k\,[P_k\{\perp/x_k\}]^c_{l_k,r_k}\end{array}}$$

[RST-END]

$$\frac{\forall i \in I.\mathrm{d}(l, l_i) \le r}{\begin{array}{c} T \rhd n\,[\langle v\rangle.P]^c_{l,r} \mid \prod_{i\in I} n_i\,[(x_i).P_i]^c_{l_i,r_i} \mid \prod_{j\in J} n_j\left[\mathtt{in}(x_j.)P_j\lceil Q_j\right]^c_{l_j,r_j} \hookrightarrow^c_{l,r} \\ n\,[P]^c_{l,r} \mid \prod_{i\in I} n_i\,[P_i\{v/x_i\}]^c_{l_i,r_i} \mid \prod_{j\in J} n_j\left[\mathtt{in}(x_j.)P_j\lceil Q_j\right]^c_{l_j,r_j}\end{array}}$$

[RST-HANDLED]

$$T \rhd n\,[\mathtt{in}(x.)P\lceil Q]^c_{l,r} \hookrightarrow n\,[Q]^c_{l,r}$$

[RST-IF-TRUE]

$$\frac{[\![\![e]\!]\!] = true}{T \rhd n\,[\mathtt{if}\ e\ \mathtt{then}\ P\ \mathtt{else}\ Q]^c_{l,r} \hookrightarrow n\,[P]^c_{l,r}}$$

[RST-IF-FALSE]

$$\frac{[\![\![e]\!]\!] = false}{T \rhd n\,[\mathtt{if}\ e\ \mathtt{then}\ P\ \mathtt{else}\ Q]^c_{l,r} \hookrightarrow n\,[Q]^c_{l,r}}$$

[RST-SWITCH]

$$T \rhd n\left[\mathtt{switch}\ c'.P\right]^c_{l,r} \hookrightarrow n\,[P]^{c'}_{l,r}$$

[RST-CONT-INT]

$$\frac{T \rhd N \hookrightarrow N'}{T \rhd N|N'' \hookrightarrow N'|N''}$$

[RST-CONGR-INT]

$$\frac{N \equiv N' \quad T \rhd N' \hookrightarrow N'' \quad N'' \equiv N'''}{T \rhd N \hookrightarrow N'''}$$

Rules RST-BEGIN and RST-END are as before, but with handled input instead of input. Rules RST-HANDLED, RST-IF-TRUE, RST-IF-FALSE and RST-SWITCH account for the different possible internal activities: handler activation, guard evaluation (to true and to false respectively) and channel switch. Finally rules RST-CONT-INT and RST-CONGR-INT are analogous to rules RST-CONT and RST-CONGR, but they consider internal activities.

9. An application: The Alternating Bit Protocol

In this section we show a more substantial example of the use of CWS: the description and the analysis of a few properties of the Alternating Bit Protocol (ABP).

ABP is used when a process P_1 has to transmit a sequence of messages to a second process P_2. Some messages however may not be correctly received by P_2. The objective of the protocol is to impose the re-transmission of these messages, possibly several times, so to guarantee, at the end, a correct delivery. In order to check that each message is received, a one-bit sequence number s is included in the message. Initially s is 0. When P_2 correctly receives a message with sequence number $s = 0$, it sends an acknowledgment containing also s, and starts waiting for a message with the complemented sequence number. When P_1 receives the correct acknowledgment, it sends a new message with the complemented sequence number.

```
$Send[q,s] \overset{\text{def}}{=}$  if empty($q$) then
          out⟨($l_1$, $End$, $s$)⟩.in($x$.)
                    if $x = (l_2, Ack, s)$ then $\mathbf{0}$ else $Send[q,s]$
                    ⌈$Send[q,s]$
      else
          out⟨($l_1$, head($q$), $s$)⟩.in($x$.)
                    if $x = (l_2, Ack, s)$ then $Send$[dequeue($q$), $\neg s$]
                                   else $Send[q,s]$
                    ⌈$Send[q,s]$

$Recv[q,s] \overset{\text{def}}{=}$  in($x$).if fst($x$) $= l_1 \wedge$ trd($x$) $= s$ then
              if snd($x$) $= End$ then
                  $Succ[q]$
              else
                  out⟨$l_2$, $Ack$, $s$⟩.$Recv$[enqueue(fst($s$), $q$), $\neg s$]
            else
              out⟨$l_2$, $Ack$, $\neg s$⟩.$Recv[q,s]$
```

Fig. 8. The Alternating Bit Protocol in CWS.

If an erroneous acknowledgment is received, or a timeout expires, the last message sent is sent again. In contrast with the versions of ABP for point-to-point communications (e.g., [21]), where a node may fail to receive a message because the message has been lost during transmission, here failure is due to interference: the message is correctly transmitted, but may not be read because of other simultaneous transmissions.

P_1 and P_2 can be modelled using the constant definitions in Fig. 8. In our case, the transmitter P_1 is located at l_1 and the receiver P_2 is located at l_2. We assume that the sender has a queue q containing the messages to be sent and the receiver has a queue to store the received messages, with operations head (returning the first element), enqueue (inserting an element), dequeue (extracting the first element) and empty (testing whether the queue is empty). We also use triples as values, with constructor $(\_,\_,\_)$ and add destructors fst, snd and trd, retrieving the first, second and third component respectively, to the expressions. We have no static typing, but we assume that evaluating ill-typed expressions (e.g., evaluating fst(v) where v is not a triple) produces value $\perp$. We indicate with $Succ[q]$ a state in which the receiver has successfully received the message.

We show below that the ABP protocol is robust against interference: messages will be delivered to destination regardless of possible interference. Interference may only cause some messages to be re-transmitted. In contrast, malicious messages such as fake data messages or fake acknowledgments can disrupt the protocol. To distinguish protocol messages from other messages we assume that each message also contains the location of the sender (e.g. a data message is a triple (l, msg, s)), and use locations as unique identifiers for nodes; other means of identifying nodes would be equally acceptable.

The following theorem proves that two nodes n_s and n_r within the transmission range of each other (i.e., with $\mathrm{d}(l_s, l_r) \leq r_s, r_r$ where l_s and l_r are the locations of the two nodes and r_s and r_r are their transmission ranges) are always able to get to a success state where the sequence of messages has been correctly received. In the theorem, we write $\Rightarrow$ for the reflexive and transitive closure of $\hookrightarrow^{\&}$, where $\hookrightarrow^{\&}$ is either $\hookrightarrow$ or $\hookrightarrow^{c}_{l,r}$ for some c, l, and r (i.e, $\hookrightarrow^{\&}$ denotes a generic reduction step).

Theorem 17. *For each network M, let*

$$N = n_s\,[Send[q,s]]^{c}_{l_s,r_s}\ |n_r\,[Recv[\emptyset,s]]^{c}_{l_r,r_r}\ |M$$

and suppose that $\mathrm{d}(l_s,l_r) \leq r_s, r_r$, i.e. the sender and the receiver can communicate with each other. Suppose also that all the transmitters follow the convention of sending their location as first component of each message. Then, whenever $T \rhd N \Rightarrow N'$, there exists N'' such that $T' \rhd N' \Rightarrow N''$ and $N'' = n_r\,[Succ[q]]^{c_r}_{l_r,r_r}\ |M''$ for some M''.

Proof. Let us say that a network N is transmitting q_1 after q_2 iff:

$$N = n_s\,[Send[q_1,s]]^{c_s}_{l_s,r_s}\ |n_r\,[Recv[q_2,s]]^{c_r}_{l_r,r_r}\ |M$$

for some boolean s and network M.

We will show that for each network N transmitting q_1 after q_2 and each $T \rhd N \Rightarrow N'$ there is N'' such that $T' \rhd N' \Rightarrow N''$ and:

(1) either N'' is transmitting q'_1 after q'_2, the concatenation of q_1 and q_2 is equal to the concatenation of q'_1 and q'_2, and q'_2 has less elements than q_2,
(2) or $N'' = n_r\,[Succ[q]]^{c_r}_{l_r,r_r}\ |M''$ for some M'' with q concatenation of q_1 and q_2.

The thesis follows from the facts above, by induction on the length of q_1. The facts above can be proved by case analysis on the possible computations $T \rhd N \Rightarrow N'$. One can easily see that reductions from M that have no effect on nodes n_s and n_r

(e.g., do not send values to them and do not create interference on them) can be disregarded. Also, proving the thesis for a computation proves the thesis also for all its prefixes.

We show a few interesting cases, the others being similar:

empty(q_1), no interference: the sequence of reductions is: empty(q_1) evaluation, sending of *End* message begin and end, evaluation of conditionals in the receiver. After that the system satisfies 2.

¬empty(q_1), no interference: the sequence of reductions is: empty(q_1) evaluation, sending of head(q_1) message begin and end, evaluation of conditionals in the receiver, sending of *Ack* begin and end, testing of *Ack* at sender. After that the system satisfies 1.

¬empty(q_1), interference on data sending: the sequence of reductions is: empty(q_1) evaluation, sending of head(q_1) message begin, interference at receiver, evaluation of conditionals in the receiver, end of transmission from the sender and from the interference generator, sending of previous *Ack* begin and end, testing of *Ack* at sender. After that we are back to the starting system, and case "¬empty(q_1), no interference" provides the desired continuation. The same holds for computations in which interference occurs also on successive sendings.

¬empty(q_1), interference on *Ack*: the sequence of reductions is: empty(q_1) evaluation, sending of head(q_1) message begin and end, evaluation of conditionals in the receiver, sending of *Ack* begin, interference at receiver, testing of *Ack* at sender. After that the system has the form:

$$N = n_s \left[Send[q_1', s]\right]^{c_s}_{l_s,r_s} | n_r \left[Recv[q_2', \neg s]\right]^{c_r}_{l_r,r_r} | M$$

where the concatenation of dequeue(q_1') and q_2' is the concatenation of q_1 and q_2. One can show with another case analysis that from such a state one either goes back to the same state or goes back to a process satisfying the thesis. This last case provides the desired continuation. We are not showing the details. □

The example below shows instead that malicious processes can disrupt the protocol.

Example 18 (*Malicious Process Disrupts ABP*). Consider the process:

$$N = n_s [Send[q, s]]^c_{l_s,r_s} | n_r [Recv[\emptyset, s]]^c_{l_r,r_r} | n_m [\mathtt{out}\langle l_s, msg, s\rangle]^c_{l_m,r_m}$$

and suppose that all processes are within the transmission range of each other.

Suppose that the malicious node n_m performs its communication, which does not follow the convention of having the process location as first element. The message is received by n_r, and interpreted as a message from n_s. Thus the system evolves to:

$$N' = n_s [Send[q, s]]^c_{l_s,r_s} | n_r \left[Recv[q', \neg s]\right]^c_{l_r,r_r} | n_m [\mathbf{0}]^c_{l_m,r_m}$$

where q' contains msg. Not only n_r has received and accepted an erroneous message, but now messages from n_s have s as one-bit sequence number, while n_r is expecting messages with sequence number $\neg s$, thus all messages from n_s will be discarded.

The example above shows that ABP is not robust against attacks from malicious processes. One should use some authentication techniques for messages, relying e.g. on a shared key between n_r and n_s to solve this kind of problems. These are beyond the scope of this paper.

10. Other extensions

10.1. Concurrent begin transmission events

In the models we have considered so far, begin transmission and end transmission events are interleaved, and may never occur at the same time. In comparison with a semantics that allowed simultaneous emission of events, the interleaving ones are much more simpler, and are sufficient to describe the interference phenomenon we were interested in.

The only form of interference that is not captured in the models presented is the one caused by two simultaneous begin transmission events, as in the following scenario. Consider two devices wishing to transmit, and in the transmission range of each other, and a receiver in the transmission range of both. Suppose also no other transmission presently reaches the area. The two transmitters thus can check that the channel is free, and could start transmitting *at the same time*. This results in a signal interference at the receiver node. In the previous models, such an interference is impossible. As the two begin transmission events by the transmitters must be interleaved, only one of them may initially fire; the second cannot occur immediately after the first because the channel then appears as busy; indeed the second begin transmission may only occur when the first transmission has been completed, but then no interference is detected by the receiver.

We have decided to ignore this form of interference in the basic models because it occurs with lower probability than the other forms, and is therefore less relevant. To understand this, consider that: (1) time elapsing between the check of availability of the channel and the begin transmission event is very short (a few microseconds); (2) to further reduce the probability of simultaneous transmissions, the devices use *backoff* techniques, which roughly operate as follows. When

$$\begin{array}{rcl}
\emptyset \rhd N & \hookrightarrow & n_1\,[\texttt{out}^*\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\texttt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\texttt{in}(x).P]^c_{l',\rho'} \overset{\text{def}}{=} N_1 \\
\emptyset \rhd N_1 & \hookrightarrow & n_1\,[\texttt{out}^*\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\texttt{out}^*\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\texttt{in}(x).P]^c_{l',\rho'} \overset{\text{def}}{=} N_2 \\
\emptyset \rhd N_2 & \hookrightarrow^c_{l_1,\rho} & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\texttt{out}^*\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[(x).P]^c_{l',\rho'} \overset{\text{def}}{=} N_3 \\
\{(l_1,\rho,c)\} \rhd N_3 & \hookrightarrow^c_{l_2,\rho} & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[P\{\bot/x\}]^c_{l',\rho'} \overset{\text{def}}{=} N_4 \\
\{(l_1,\rho,c),(l_2,\rho,c)\} \rhd N_4 & \hookrightarrow^c_{l_2,\rho} & n_1\,[\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[P\{\bot/x\}]^c_{l',\rho'} \overset{\text{def}}{=} N_5 \\
\{(l_1,\rho,c)\} \rhd N_5 & \hookrightarrow^c_{l_1,\rho} & n_1\,[\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[P\{\bot/x\}]^c_{l',\rho'}
\end{array}$$

Fig. 9. Concurrent begin of transmission.

a transmitter wishes to transmit and the channel is presently busy, the device first waits for the end of the current transmission. Then, it does not immediately start its own transmission, but waits some random amount of time, at the end of which it checks again the availability of the channel before actually starting the transmission. The probability that this will now conflict with another transmission is extremely low, because different devices will statistically choose different random delays.

In spite of these considerations, one may still be interested in modelling the above interference scenario. We describe below some simple modifications to our semantics that allow us to do so. It is sufficient to use different transitions for the check of availability of the channel and the following begin transmission event. For this, we add the production

$$P \overset{\text{def}}{=} \texttt{out}^*\langle v\rangle$$

to the process syntax, so to represent a process that has checked for the availability of the channel but has not yet started transmitting. We show how to modify the RST semantics of Section 3 to take care of the new phenomenon: we introduce a new rule RST-CHECK, to check that the channel is free, and update rule RST-BEGIN by removing the check for channel availability. Checking channel availability is modelled as an internal action. The two rules are as follows:

[RST-CHECK]

$$\frac{T|_{l,c} = \emptyset}{T \rhd n\,[\texttt{out}\langle e\rangle.P]^c_{l,r} \hookrightarrow n\,[\texttt{out}^*\langle e\rangle.P]^c_{l,r}}$$

[RST-BEGIN]

$$\frac{\forall h \in I \cup J \cup K.\mathrm{d}(l,l_h) \leq r \quad \forall i \in I.T|_{l_i,c} = \emptyset \quad \forall j \in J.T|_{l_j,c} \neq \emptyset}{\begin{array}{c} T \rhd n\left[\texttt{out}^*\langle e\rangle.P\right]^c_{l,r} \mid \prod_{h\in I\cup J} n_h\,[\texttt{in}(x_h).P_h]^c_{l_h,r_h} \mid \prod_{k\in K} n_k\,[(x_k).P_k]^c_{l_k,r_k} \hookrightarrow^c_{l,r} \\ n\,[\langle [\![e]\!] \rangle.P]^c_{l,r} \mid \prod_{i\in I} n_i\,[(x_i).P_i]^c_{l_i,r_i} \mid \prod_{j\in J} n_j\left[\texttt{in}(x_j).P_j\right]^c_{l_j,r_j} \mid \prod_{k\in K} n_k\,[P_k\{\bot/x_k\}]^c_{l_k,r_k} \end{array}}$$

The modifications for the other semantics are similar.

We conclude this section with the example of the interference scenario above, with simultaneous begin of two transmissions.

Example 19 (*Concurrent Begin Transmission Events*). This example shows the behavior of two transmitters $S_1 \overset{\text{def}}{=} n_1\,[\texttt{out}\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho}$ and $S_2 \overset{\text{def}}{=} n_2\,[\texttt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho}$ that start their transmissions concurrently, even if they are in the transmission range of each other. We also consider a receiver $R \overset{\text{def}}{=} m\,[\texttt{in}(x).P]^c_{l',\rho'}$ in the transmission range of both the transmitters.

Thus the network is:

$$N \overset{\text{def}}{=} n_1\,[\texttt{out}\langle v_1\rangle.\mathbf{0}]^c_{l_1,\rho} \,\big|\, n_2\,[\texttt{out}\langle v_2\rangle.\mathbf{0}]^c_{l_2,\rho} \,\big|\, m\,[\texttt{in}(x).P]^c_{l',\rho'}$$

The computation is represented in Fig. 9. At the beginning, both transmitters check whether the channel is free, and the check is satisfied since $T = \emptyset$. Then they both start transmitting. Notice that when S_2 starts the channel is no longer free, but the check has already succeeded. Thus interference occurs at the receiver node. The last two transitions model the end of the two transmissions. This behavior cannot be modelled using the semantics described in the previous sections.

10.2. Language extensions

We discuss here a few further language extensions: mobility, time and local channels.

We regard mobility largely as an orthogonal issue, which does not affect the formulation of our semantics and the treatment of interference (the main topic of this paper). Indeed, with present technologies the communication times are negligible with respect to times for mechanical movements. For instance, consider a transmitter that moves at 5 m/s and that transmits a 2000-byte frame over a channel having a 2 megabit/s bandwidth. The actual transmission would take about 0.008 s; during that time, the transmitter moves only about 4 cm away. In other words, we could assume that the nodes are static when transmitting and receiving, and may change their location only while they are idle. Movement could be determined by a dedicated primitive in the processes in the nodes. We could also follow approaches in the literature such as [23,9,17], where movement of the nodes is completely arbitrary and is externally determined (i.e., the location of a node simply changes, non-deterministically, but the nodes themselves do not contain movement instructions).

If one however wants to model nodes able to move and communicate at the same time, then the following aspects should be taken into account. The rules for movement of processes that are neither active transmitters nor active receivers are straightforward but, if LRS is used, the marking should be recomputed after the transition (as for channel switch transition in Section 8). In contrast, in the case of an active transmitter t moving from l to l', the rule (which is actually a rule schema) should consider three cases: active receivers reachable from l but not from l' should get an error, since they are no more able to receive the transmission; active receivers reachable from l' but not from l (which were receiving a transmission from some other node) should get interference; other nodes are unaffected and can be simply added by a context rule. Similar checks are necessary if an active receiver r moves from l to l': if the transmission is no more receivable, or other transmissions are, then r should get an error or an interference.

Time is another interesting feature: the handled input construct models in fact timeouts, but abstracting away the actual timing information. Placing explicit timing information is straightforward, following one of the approaches in the literature (see [1] for a tutorial, and [28] for an application to wireless systems). Thus the handled input $\mathtt{in}(x).P\lceil Q$ would be replaced by an input with timeout, such as $\mathtt{in}(x).P\lceil^t Q$ where Q triggers after t time units.

Finally, local channels can be used to ensure that only a subset of the receivers is able to interact using a certain channel. This can be modelled by introducing a restriction operator like the one of π-calculus, i.e. by allowing $\nu c\ P$ and $\nu c\ N$ to denote respectively a process P and a network N with a local channel c. Semantically, transmissions on local channels are not visible from the outside, and processes not knowing the channel may not transmit on it (and, in particular, they may not create interference on it). Local transmissions are visible from the outside as internal actions. If channels can also be communicated, then rules for scope extension should be added to the calculus. The details are easy, following the rules of process calculi with extrusion of channels such as the π-calculus.

11. Conclusions and future work

In this paper we have developed an operational semantics for a calculus of wireless systems. We have studied different forms of operational semantics, as Reduction Semantics (RS) and Labelled Transitions Systems (LTS), and proved correspondence results between them. We have first presented a core calculus, and we have then described some extensions. Finally, we have showed our calculus at work on the Alternating Bit Protocol.

In all the semantics, we separate the begin and the end of a transmission; this has been necessary, because of technical difficulties and the peculiarities of wireless systems, in order to capture all possible forms of interference among the processes. We have also analysed different approaches for the check of interference: with centralized information, with local information, and without stored information.

A number of developments are possible. First, further work is needed to evaluate the expressive power of the operators in CWS. We have considered here a few simple examples and a version of the Alternating Bit Protocol; some other examples are presented in [18], but more experiments are necessary. Second, one could try to define appropriate notions of behavioral equivalences and/or logics. A behavioral equivalence is usually defined with respect to an observer that interacts with the system. In wireless systems, each device – and therefore presumably also the observer – has a location and a transmission cell, and it is not clear how to take them into account. By the time the revision of this paper was completed, a few proposals for observational equivalences have appeared [9,17]. All these approaches could be transported to our setting; however, they do not take the above aspects of locality for an observer into account.

We plan to develop an implementation of the calculus which could be used to make simulations of systems or protocols. Finally we would like to study the addition of probabilities to our calculus so as to be able to define the probability with which certain forms of interference may occur.

Acknowledgements

We would like to thank Luciano Bononi for several interesting discussions, Nicola Mezzetti for his participation in the initial phase of this work, and the referees for many useful suggestions.

References

[1] L. Aceto, A. Ingólfsdóttir, K.G. Larsen, J. Srba, Reactive Systems — Modelling, Specification and Verification, Cambridge University Press, 2007.
[2] I.F. Akyildiz, I.H. Kasimoglu, Wireless sensor and actor networks: Research challenges, Ad Hoc Networks 2 (4) (2004) 351–367.
[3] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, A survey on sensor networks, IEEE Communications Magazine 40 (8) (2002) 102–114.
[4] G. Berry, G. Gonthier, The esterel synchronous programming language: Design, semantics, implementation, Science of Computer Programming 19 (2) (1992) 87–152.
[5] A. Boukerche, L. Bononi, Simulation and modeling of wireless, mobile and ad hoc networks, in: Mobile Ad Hoc Networking, IEEE/Wiley, 2004, chapter.
[6] L. Cardelli, A.D. Gordon, Mobile ambients, in: Proceedings of Foundations of Software Science and Computation Structures, FoSSaCS, vol. 1378, Springer, 1998, pp. 140–155.
[7] I. Chatzigiannakis, T. Dimitriou, M. Mavronicolas, S.E. Nikoletseas, P.G. Spirakis, A comparative study of protocols for efficient data propagation in smart dust networks, Parallel Processing Letters 13 (4) (2003) 615–627.
[8] C. Ene, T. Muntean, Expressiveness of point-to-point versus broadcast communications, in: Proceedings of 12th International Symposium on Fundamentals of Computation Theory, FCT'99, in: Lecture Notes in Computer Science, vol. 1684, Springer, 1999, pp. 258–268.
[9] Jens Chr. Godskesen, A calculus for mobile ad hoc networks, in: Proceedings of 9th International Conference on Coordination Models and Languages, COORDINATION'07, in: Lecture Notes in Computer Science, vol. 4467, Springer, 2007, pp. 132–150.
[10] B.G. Goode, Reliable enough for a nuclear plant? Wireless for Industry 3 (2004) 7–9.
[11] D. Harel, Statecharts: A visual formulation for complex systems, Science of Computer Programming 8 (3) (1987) 231–274.
[12] Inst. Elec. Electron. Eng., New York, USA. IEEE Std 802.11 - 1999: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 1999.
[13] Inst. Elec. Electron. Eng., New York, USA. IEEE Std. 802.15.4 - 2003: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPANs), 2003.
[14] Java in simulation time / scalable wireless ad hoc network simulator. http://jist.ece.cornell.edu/.
[15] T. Kevan, Coming soon to your neighborhood, Wireless Sensors 3 (2005) 5–8.
[16] A.M. Mainwaring, D.E. Culler, J. Polastre, R. Szewczyk, J. Anderson, Wireless sensor networks for habitat monitoring, in: Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications, WSNA'02, ACM Press, 2002, pp. 88–97.
[17] M. Merro, An observational theory for mobile ad hoc networks, in: Proceedings of the 23rd Annual Conference on Mathematical Foundations of Programming Semantics, MFPS XXIII, in: Electronic Notes in Theoretical Computer Science, vol. 173, Elsevier, 2007, pp. 275–293.
[18] N. Mezzetti, Trust and wireless communication, Ph.D. Thesis, Computer Science Department, University of Bologna, 2006.
[19] N. Mezzetti, D. Sangiorgi, Towards a calculus for wireless systems, in: Proceedings of the 22nd Annual Conference on Mathematical Foundations of Programming Semantics, MFPS XXII, in: Electronic Notes in Theoretical Computer Science, vol. 158, Elsevier, 2006, pp. 331–353.
[20] R. Milner, Calculi for synchrony and asynchrony, Theoretical Computer Science 25 (1983) 267–310.
[21] R. Milner, Communication and Concurrency, Prentice Hall, 1989.
[22] R. Milner, The polyadic π-calculus: A tutorial, Technical Report ECS–LFCS–91–180, Edinburgh University, 1991. Also in Logic and Algebra of Specification, Springer, 1993.
[23] S. Nanz, C. Hankin, A framework for security analysis of mobile wireless networks, Theoretical Computer Science 367 (1–2) (2006) 203–227.
[24] S.E. Nikoletseas, Models and algorithms for wireless sensor networks (smart dust), in: Proceedings of 32nd Conference on Current Trends in Theory and Practice of Computer Science, SOFSEM'06, in: Lecture Notes in Computer Science, vol. 3831, Springer, 2006, pp. 64–83.
[25] The network simulator. http://nsnam.isi.edu/nsnam/index.php/Main_Page.
[26] K. Ostrovsky, K.V.S. Prasad, W. Taha, Towards a primitive higher order calculus of broadcasting systems, in: Proceedings of the 4th International Conference on Principles and Practice of Declarative Programming, PPDP'02, ACM Press, 2002, pp. 2–13.
[27] K.V.S. Prasad, A calculus of broadcasting systems, Science of Computer Programming 25 (2–3) (1995) 285–327.
[28] K.V.S. Prasad, Broadcasting in time, in: Proceedings of First International Conference on Coordination Languages and Models, COORDINATION'96, in: Lecture Notes in Computer Science, vol. 1061, Springer, 1996, pp. 321–338.
[29] K.V.S. Prasad, A prospectus for mobile broadcasting systems, in: Proceedings of the Workshop on Algebraic Process Calculi: The First Twenty Five Years and Beyond, PA'05, BRICS Press, 2005, pp. 209–212.
[30] D. Sangiorgi, D. Walker, The π-Calculus: A Theory of Mobile Processes, Cambridge University Press, 2001.
[31] D. Senders, S. Neely, J. Lewis, Manufacturing probe needles with vision, Wireless Sensors 3 (2005) 25–28.
[32] R. Szewczyk, J. Polastre, A.M. Mainwaring, D.E. Culler, Lessons from a sensor network expedition, in: Proceedings of First European Workshop on Wireless Sensor Networks, EWSN'04, in: Lecture Notes in Computer Science, vol. 2920, Springer, 2004, pp. 307–322.
[33] B. Warneke, M. Last, B. Liebowitz, K.S.J. Pister, Smart dust: Communicating with a cubic-millimeter computer, IEEE Computer 34 (1) (2001) 44–51.