Università Ca' Foscari di Venezia

Dipartimento di Informatica
Dottorato di Ricerca in Informatica

Ph.D. Thesis

Formal Models for Qualitative and Quantitative Analysis of Mobile Ad Hoc and Sensor Networks

Gallina Lucia

Supervisors
Sabina Rossi and Andrea Marin

PhD Coordinator
Riccardo Focardi

December, 2012

Author's Web Page: http://www.dsi.unive.it/~ gallina/

Author's e-mail: lgallina@dais.unive.it

Author's address:

Dipartimento di Informatica
Università Ca' Foscari di Venezia
Via Torino, 155
30172 Venezia Mestre – Italia
tel. +39 041 2348411
fax. +39 041 2348419
web: http://www.dsi.unive.it

A mio marito Emanuele

Abstract

Mobile ad-hoc networks (MANETs) are systems of mobile devices communicating with each other through wireless links without a pre-established networking infrastructure. The absence of a central infrastructure, together with the heterogeneous nature of the devices, make this kind of networks particularly fit to face critical situations, such as natural disasters or battlefield environments. Connectivity, energy consumption and communication interference are key aspects in mobile ad-hoc networks, due to the dynamic characteristics of the devices. In this thesis we propose a broadcast process algebraic model for the analysis of such aspects of MANETs. In particular we first introduce a non-deterministic calculus, which enables us to define and prove some important connectivity properties of ad hoc networks, and then we introduce two different probabilistic extensions of such calculus, with the aim of providing both a qualitative and quantitative analysis of ad hoc networks. In particular, we concentrate on the energy consumption and the level of interference. Using the model checking technique, we finally provide a framework for automatically evaluate the performances of ad hoc networks, with respects to the metrics previously defined.

Sommario

Le reti ad hoc (MANETs) sono sistemi di dispositivi mobili che comunicano tra loro usando collegamenti wireless, senza alcuna infrastruttura prestabilita. L'assenza di una infrastruttura centrale e la natura eterogenea dei dispositivi rendono questa rete particolarmente adatta a gestire situazioni critiche, come ad esempio le comunicazioni in caso di disastri naturali, o nei campi di battaglia. La connettività, il consumo energetico e l'interferenza nelle comunicazioni sono aspetti chiave nella gestione delle reti ad hoc, date le caratteristiche dinamiche dei dispositivi che le costituiscono. In questa tesi proponiamo un modello formale per l'analisi di queste problematiche. In particolare introduciamo un modello non deterministico che ci permette di definire e provare alcune importanti proprietà legate alla connettività delle reti. Proponiamo poi due differenti estensioni probabilistiche del calcolo introdotto, volte ad un'analisi sia qualitativa che quantitativa delle reti ad hoc. In particolare questa tesi si concentra sullo studio del consumo energetico e del livello di interferenza. Infine, grazie alla tecnica del model checking, viene fornito uno strumento per valutare in modo automatico le prestazioni delle reti ad hoc rispetto alle metriche proposte.

Acknowledgments

First of all, I would like to thank my PhD adivsor, Sabina Rossi, who introduced me to the field of Process Algebra and was a great support for my work throughout my whole thesis. I would like to thank Andrea Marin, assistant professor at Ca' Foscari University of Venice, who sustained my work especially in the last year of my PhD thesis. Marta Kwiatowska, professor at the University of Oxford, and Rocco De Nicola, professor at IMT-Institute for Advanced Studies of Lucca, accepted the role of reviewers of my thesis: I am very proud of that and I thank them for the time spent reading and commenting on my work. I want to thank Hamadou Saradouma, research fellow at INRIA Saclay, LIX Ècole Polytechnique, for his help and his collaboration. I want to thank Tinting Han, research assistant at the University of Oxford, who introduced me to the PRISM Model Checker, and spent a lot of time with me carrying out experiments and studying formulas to find interesting results. I am grateful to my family who have stood by me all through my life. I am particularly grateful to my husband Emanuele, who continuously sustains me and encourages me in my work, and who has made these last three years of my life special.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Key concepts

This thesis deals with the analysis of ad hoc networks [74], which are wireless networks constituted of both mobile and static devices, communicating with each other without a fixed infrastructure.

In Computer Science, the concept of *process algebra* (or process calculus) [60, 41] encloses those mathematical approaches modelling concurrent systems. This thesis presents a process calculus specifically developed to study ad hoc networks.

In particular, we concentrate on the analysis of the main challenging problems affecting the ad hoc networks, such as the scarce availability of energy resources, the frequent link breakages due to the topology changes, and the interference caused by the use of radiofrequency channels for communications.

Model checking [6], is a technique allowing one to automatically verify several kinds of properties of systems that exhibit random or probabilistic behaviour. Usually, the specification of the properties is given by temporal logic formulas. This automatic verification technique has been used to analyse systems of several domains, including communication and security protocols [48], [71], [72].

1.2 Contribution of the thesis

We present a process calculus, named EBUM (Energy aware Broadcast Unicast and Multicast Communications) [26], [25], for formally reasoning about energy-aware broadcast, unicast and multicast communications of mobile ad hoc and sensor networks. Our calculus is built around nodes, representing devices executing a process, and locations, identifying the position cells across which each device may move inside the network. A network is modelled as a parallel composition of nodes. A device can be static or it can arbitrarily change its position inside the network area. Moreover, each node can connect and disconnect with the network, capturing the eventuality of physical damages, or other kinds of problems affecting the devices. Processes describe the input and output actions that a node can perform. In particular, each transmission is associated with a channel and a transmission radius,

which has got the double task of defining the physical area where the transmission will be receivable, and of representing the transmission power used by the sender node to transmit.

We define an observational congruence in the style of [62] to equate networks exhibiting the same connectivity behaviour. In particular, the notion of observability is associated with nodes listening at specific locations in the network, so as to allow a fine grained analysis of connectivity at different areas within a network. We give a coinductive characterization of observational congruence based on a labelled transition semantics. This is a bisimulation-based proof technique in the form of a labelled bisimilarity which is shown to coincide with the observational equivalence.

This model is useful for the definition and the verification of important properties concerning the performances of networks, in terms of energy consumption, connectivity maintenance and interference awareness.

We present also two probabilistic extensions of the EBUM calculus.

The first one [22], [23] extends its predecessor by introducing a probability distribution to model the node mobility. This choice is due to the consideration that usually the node movements are not completely casual, but the devices follow some particular trajectories, due to the presence of physical obstacles in the network area, or to other reasons, depending on the surrounding environment. Working with a calculus where both non-deterministic and probabilistic aspects coexist, allow us to make both qualitative and quantitative analysis of a system performance, in terms of several indexes, such as energy consumption, throughput, interference level, time delay, etc. We also introduce a energy-aware preorder over networks to measure the relative energy costs of different, but behaviourally equivalent, networks. This preorder is defined in two steps: given two networks we first verify their behavioural equivalence, and then we compute the energy consumed at each execution step, calculated looking at the transmission radius used for each communication.

We present another probabilistic extension of the EBUM calculus [24] [11], where again the mobility of the node is modelled with a probability distribution. The novelty with respect to the previous calculus is that the transmissions are modelled with non-atomic actions, following [51]. This new model allows us to capture the collisions occurring when the transmission area of two sender nodes overlap. Again we define a preorder, allowing us to decide, given two networks having the same observational behaviour, whether one of them is able to reduce the number of collisions during the network activity.

Finally, we present a framework [50], based on model checking technique , in order to make automatic verification on ad hoc networks. We exploit the PRISM tool [49] to perform quantitative verification and analysis of wireless networks for a range of performance metrics. Specifically, we develop a parser to translate an EBUM process term, representing a network, into a Marlov Decision Processes expressed in the PRISM language. We used the property specification language of PRISM to formulate the rewards expressing the time and the energy costs of the network behaviours.

1.2.1 Overview

This thesis is organised as follows.

Chapter 2 introduces the main concepts and issues which we deal with.

Chapter 3 introduces the EBUM calculus, with the definition of observational equivalence, bisimulation, and the proof of the complete characterisation of the bisimulation defined. We also define some properties to show the usefulness of the calculus introduced.

Chapter 4 introduces the first probabilistic extension of the calculus, with the definition of probabilistic observational equivalence and of the correspondent probabilsitic bisimulation. We define the energy-aware preorder, and we show its usefulness by introducing two case studies: the first one compares the performances of two ARQ-based communication protocols, with respect to the energy consumption; in the second case study we analyse how the application of a location-based strategy can improve the performances of a routing protocol based on the simple flooding algorithm.

Chapter 5 introduces the second probabilistic extension of the calculus. We show its capability on the analysis of two case studies: the first one compute the level of interference caused by the hidden station problem on a toy network using the alternating bit protocol to communicate, while the second case study analyse the resistance of two different routing strategies to a jamming attack in a in-door environment.

Chapter 6 presents a framework, based on the probabilistic calculus presented in Chapter 4, and on the PRISM Model Checker [49], allowing the automatic verification of several properties of ad hoc networks, such as connectivity maintenance, energy conservation and throughput optimization. We show our framework at work on the analysis of a network communicating using a simple flooding algorithm: in particular we study how the choice of the transmission radius influences the energy consumption of the network.

2
Background

2.1 Introduction

This chapter introduces the key aspects that will be discussed in this thesis, and gives a panoramic of the state of art.

2.2 Wireless Ad Hoc Networks

Wireless ad hoc networks are systems of devices communicating with each other through wireless links without a pre-established network infrastructure.

Hence the connectivity of the network depends on the collaboration among nodes to forward information, since there is no any centralized control.

Wireless Ad-Hoc Networks are built using wireless technology, the devices communicate with each other via radio transceivers, using the protocol IEEE 802.11 (WiFi). This type of communication has a physical scope, because a radio transmission spans over a limited area. Therefore it necessarily must be applied a routing protocol, proper to wireless dynamic systems.

Ad hoc networks can be classified in:

- *Mobile Ad hoc Networks* (Figure 2.1.(a)).

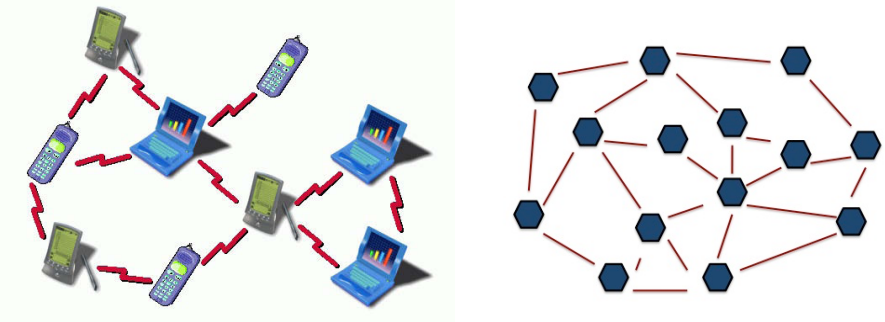
  Mobile ad hoc networks (MANETs) are made of different mobile and static devices, having different battery capacities and a different transmission power.

  Due to their dynamic characteristics, these networks are often used to face critical situations, as the problem of managing the communications during rescue operations for natural emergencies (e.g., the earthquakes), or in military battlefields.

  A particular case of MANETs are the vehicular networks, which exploits the vehicles to create a mobile network and to forward information about traffic.

- *Wireless Sensor Networks* (Figure 2.1.(b)).

  A Wireless Sensor Network (WSN) [3] is a collection of spatially distributed sensors to monitor physical or environmental conditions. Each sensor node



(a) Mobile Ad Hoc Networks (b) Wireless Sensor Networks

Figure 2.1: Structure of Mobile Ad Hoc and Sensor Networks

has a radio transceiver to communicate with the network, a microcontroller (for data survey), an electronic circuit for interfacing with the sensors and an energy source (usually a battery or an embedded form of energy harvesting).

Sensor networks are employed for many different applications as Area Monitoring (over region where some particular phenomenon has to be monitored), Environmental Sensing (e.g., the monitoring of a Volcano's activity, or other natural events), Air Pollution Monitoring or Industrial Monitoring (e.g., the monitoring the machinery's activity).

The main differences between MANETs and WSN are:

- the density of the network nodes: in wireless sensor network, the number of nodes in a given region could be hundreds or thousand higher than the number of devices in a MANET;
- the mobility of the nodes: due to the environmental conditions, the topology changes of a wireless sensor network are more frequent than the topology changes of a mobile ad hoc network (see, e.g., the monitoring of animal behaviours, or the mobility of sensors placed in the sea to monitor the drift);
- the identity of nodes: while in mobile ad hoc networks each device is always univocally identified, e.g., by an IP address, in sensor networks the devices are not necessarily associated with a global identifier, but they may be simply anonymous devices lying in a certain location. This is a consequence of the high number of nodes which can potentially constitute a WSN.

For further details about ad hoc networks see [74].

As ad hoc networks are usually implemented in precarious environments, they are vulnerable to many problems concerning the network connectivity, the energy consumption and the interference, because the nodes only communicate using radio-frequency channels, which can not be private. The dynamic nature of ad hoc net-

works makes the management of the transmissions and of the routing protocols much more complicated. The ad hoc networks are self organized, so the good behaviour of the system depends on the cooperation among the connected nodes: this characteristics can make the network more vulnerable to damages caused, not only by malicious nodes, but also by "lazy" nodes, that are devices which, for power saving, do not cooperate with the other nodes; this bad behavior can originate problems especially in the management of the packets routing. Since ad hoc networks are used to face critical situations, where the transmitted data are often important and confidential, we are now going to enumerate the main properties to be preserved in planning an ad hoc network [95], [9], [78], [3].

Power Conservation : is the property of handling the energy resources of network devices, i.e., the battery waste avoidance. Ad hoc networks are constituted of different kinds of devices (notebook, mobile phones...) which may have scarce energy capacities, and battery with an autonomy of few hours. On the other hand sensor networks are constituted of small nodes with a limited battery. The power limits of a device can be a problem for the whole network lifetime, since nodes communicate thanks to the collaboration of the neighbouts forwarding packets between source and destination. In particular, for sensor networks the power conservation is a challenging issue since, due to the environmental conditions, the battery substitution is not always possible (e.g., the sea monitoring or other critical environment), network lifetime depends on the capacity of conserving as much energy as possible.

Network connectivity maintenance : is the property of guaranteeing at least one path between each pair of nodes in the network. If we consider the necessity of reducing the power consumption of the transmissions, we have always to consider the connectivity of the network nodes: a device cannot use a transmission power that is too small to reach the receivers of its messages. As concerns sensor networks, this property is hard to guarantee, due to the frequent topology changes occurring after sensor disconnections or movements.

Reduction of Interference : is the property of limiting the collisions due to the absence of a fixed infrastructure controlling the network transmissions. Ad hoc networks use radio-frequences to communicate, hence channels are *half-duplex*: a node can not transmit and receive at the same time. Using a too large transmission power will therefore increase the interference level. Interference is usually the effect of the hidden station problem [88], i.e., the presence of multiple transmitters whose transmission area overlap, causing the reception of corrupted messages by the nodes inside the overlapping areas. Interference can also be caused by malicious nodes, intended to disrupt the network traffic with continuous meaningless radio signal.

Fault tolerance : is the property of providing all the network services, regardless of the possible nodes failures and links breakages. Nodes failures can be caused

by physical damages, or by battery depletion, while links breakages are caused by the network topology changes, due to the presence of mobile nodes.

Throughput Optimisation : is the property of providing a good exploitation of the network channel in a given time slot. This property is usually conflicting with power conservation, since, when nodes reduce their power in order to conserve energy, the amount of data sent in a time slot is reduced.

Scalability : is the property of guaranteeing that the functionalities of a given network are preserved even increasing the number of nodes and the size of the network area. This property is particularly relevant for sensor networks, where the number of nodes may reach very high values. Indeed, if in a given network, the density and the physical characteristics of nodes are fixed, the scalability allows one to study some network properties, reducing the analysis to a smaller region.

Time Delay Reduction : is the property of containing the time delays of the transmissions. Unfortunately, as for the throughput optimisation, this property is opposite to power conservation since, choosing low power level for transmissions can critically increase the number of hops to reach a given destination. Indeed, in critical or emergency situations (e.g., military battlefields or natural disasters), minimizing the time spent to send a message become the main goal to pursue. However, due to the scarce power resources of the devices, we have always to take in consideration the problem of reducing the energy consumption during each communication: the best solution is always a trade-off between the time and the energy cost.

Data Integrity : is the property of guaranteeing the correct and complete reception of data by the receivers. Since ad hoc networks communications are realized through multi-hop transmissions, it is important to guarantee that the messages received by the destination are complete. Corrupted messages can arrive due to intermediate forwarding failures.

Network Availability : is the ability of a network to ensure communication services, despite of the problems caused by interference or other environmental obstacles to the traffic flow.

2.2.1 Ad hoc routing protocols

Since ad hoc networks are usually self-organized, one of the most challenging aspects of their implementation is the choice of a routing protocol which preserves the properties described above.

In particular, many research efforts have been devoted to develop energy-aware routing protocols, especially in the case of sensor networks where devices have scarce

Figure 2.2: Equilibrium among different performances indexes

battery resources and where, due to the critical environmental conditions, the substitution of discharged batteries is not always possible. In addition, due to the rapid evolution of modern technologies, which allow the devices to choose among different transmission power levels, several protocols have been implemented, which reduce the total energy consumption by adjusting each node's transmission power just enough to reach up the intended recipients only (see, e.g., [94]).

The concept of *topology control* [81] is the technique used in order to modify the initial network topology to save energy, and to extend the network lifetime. This can be then considered as a trade-off between power saving and network connectivity: when we have to choose the transmission power of each node, we know that, if we assign a low transmission power to a node, we reduce its connectivity within the network, but we also reduce its power consumption. Indeed, reducing the transmission power of the devices will reduce their transmission area, consequently reducing the possibility of collisions occurrences [12]. On the other hand the throughput and the time cost of communications depend on the level of connectivity (see Figure 2.2) : having more links among nodes will reduce the number of hops to reach a destination, reducing the time spent for a transmission, and increasing the throughput. The main goal of topology control is therefore the choice, for each node, of a minimum transmission power reducing the energy consumption and the interference level, but preserving the network connectivity.

Several routing protocols have been specifically developed for ad hoc networks, taking into account the absence of a centralized control and the dynamic nature of the network topology.

The choice of the routing protocol strongly depends on the environmental conditions and on the characteristics of the network topology: this is the reason why it does not exist an algorithm which can be considered the best one, but there will be different *best strategies* for each different network implementation.

Routing protocols can be classified in:

Proactive routing protocols : are protocols that continually exchange routing information about all the nodes, to maintain the routing tables always updated. Among the most used proactive protocols we cite DSDV [75] (Destination-Seqeunced Distance Vector) and WRP (Wireless Routing Protocol) [65].

Reactive routing protocols : are protocols which update the routing table of each node only on-demand. Among the most used reactive protocols we

cite AODV (Ad-Hoc On -Demand Distance Vector) [79], TORA (Temporally-Ordered Routing Algorithm) [73] and DSR (Dynamic Source Routing) [43].

While proactive routing reduces the latency in sending out packets, due to the continuous up-to-date of the routing tables, reactive routing is more efficient in terms of resources usage, since the route tables are updated only on-demand.

Hierarchical routing protocols : are protocols which divide the network in clusters, assigning a manager node for each cluster: in this way the network is no more completely free from a centralized infrastructure, but each network node can refer to its cluster manager node in order to find a path to a given destination. Examples of hierarchical protocols are GLS (Global State Routing) [55], or HSR (Hierarchical State Routing) [28].

Geographical routing protocols : are protocols which exploit the information about the geographical positions of nodes; this strategy can be used in networks where nodes know their locations. Information about geographical positions are usually recovered by help of some common system, as, e.g., the GPS (Global Positioning System) [45]. As an example, LAR (Location Aided Routing) [47] is a protocol which exploits information about the location and the mobility of nodes, to optimize the routes discovery.

Hybrid routing protocols : are protocols similar to the geographic routing because they use information about the geographic positions of nodes and adopts different approaches for the local and for the remote area routing. An example is the ZRP (Zone Routing Protocol) [32], which divides the networks in routing zones, according to the distances between mobile nodes.

2.3 Process Algebras

Process algebras are formal models for concurrent systems. They provide a high level description of interactions, communications and synchronizations among a set of independent agents, and also provide algebraic laws allowing formal reasoning about equivalences of processes.

Among the existing process algebras we cite the Calculus of Communicating Systems (CCS) [60], Communicating Sequential Processes (CSP) [41], the Algebra of Communicating Processes (ACP) [7] and the π-calculus [61]. Comparisons between processes can be made by means of a co-inductive proof methodology called bisimulation [80], associating systems which behave in the same way in the sense that one system simulates the other one and vice-versa.

A distributed version of the Pi-Calculus A particular attention should be paid to the language DPI (Distributed Picalculus) [34] [36], introduced by Hennessy

et al.; DPI is a distributed version of the Pi-calculus where processes may migrate between dynamically created locations. A system is modelled as the parallel composition of processes, denoted with $l[\![P]\!]$, where P is the process/agent, and l a location, meaning that the agents are distributed among different sites where they can use local resources. These resources are modelled using local communication channels. The behaviour of the systems, that is the ability of an agent to interact with other agents, is modelled by means of the definition of *capability types*, which describe the channel access control polices, and the relation among channels and locations. This calculus has been introduced principally to study the behaviour of client-server systems, in the presence of firewalls or other restrictions on the channel access: locations have been introduced just to reason about the accessibility of a resource.

Our work is inspired by the DPI calculus, but locations are used in a different way. In particular, when dealing with wireless channels, locations are used to model the fact that wireless channels are strictly dependent on the physical area of the transmission, and not only on the transmission channel. Moreover, ad hoc networks are characterized by the possibility to perform broadcast transmissions, while DPI only supports the peer-to-peer communication.

2.3.1 Timed Process Algebras

Timed process algebras [69] are extensions of process algebras where actions are associated with time. As an example, SCCS (Synchronous CCS) [59] is a synchronous version of CCS with the addition of clock thicks to control the synchronizations among processes. TCCS (Temporal CCS) [63] is another extension of CCS where the process actions continue to be instantaneous: the peculiarity of this calculus is that there are two kinds of transitions: action transitions and time transitions. Two systems are considered equivalent only if they behave in the same way, with the same time delays. Another timed version of CCS is presented in [35], where time is modelled as clock, i.e., by introducing σ actions: when a process executes a σ action, it means that it is is going to sleep until the next clock time.

2.3.2 Probabilistic and Stochastic Process Algebras

In Process Algebra usually transitions are non-deterministic, meaning that systems at each step can choose among different actions, and the choice is completely casual. Probabilistic process algebras are extensions of process algebras which solve the non-determinism by associating a probability with each possible action a system can perform. Stochastic process algebras associate rates rather then probabilities to each action occurrence.

Probabilistic CCS (PCCS) [44] is an extension of CCS, that is similar to the synchronous CCS but with probabilistic choices replacing non-deterministic ones. As an example of stochastic extensions of CCS we cite [13].

As concerns the π-calculus, there are several probabilistic [37, 31] and stochastic [77] versions that have been proposed in literature.

There are other probabilistic and stochastic process algebras which have gained success: we will discuss them in Section 2.5, while exploring the state of art.

2.4 Model checking and Formal Verification

The concept of *Formal Verification* encloses all the techniques aimed at verifying the correctness of formal methods with respect to certain properties.

One approach is Model Checking [6], which verifies finite models, usually representing systems as finite automata,where states represent the possible system configurations, while transitions show how the system can evolve. Then, by using a temporal logic, we can make a qualitative analysis of several properties, expressed through logic formulas, and we can check if they are are verified for the system under analysis.

Probabilistic Model Checking, in particular, automatically verifies properties of probabilistic systems. Due to the presence of probability, a quantitative analysis can be done, returning the expectation of the formula expressing the probability that the property under analysis is verified for the system.

Usually Probabilistic Model Checking is based on the following probabilistic models:

1. Markov Decision Processes (MDP) : model concurrent probabilistic systems, i.e., systems where both probabilistic and non-deterministic aspect coexists;
2. Discrete Time Markov Chains (DTMC): model fully probabilistic systems;
3. Continuous Time Markov Chains (CTMC): model stochastic systems;
4. Probabilsitic Timed Automata (PTA): model the real time behaviour of systems.

Temporal logic [42] is used to describe the properties to be verified, since it allows to express propositions formulated in terms of time. In particular, Computational Tree Logic (CTL) is a branching-time logic, i.e., its model of time is a tree-like structure describing all possible ways the future would evolve. Linear Temporal Logic (LTL) is a fragment of the Computational Tree Logic, encoding formulas about the future of paths and allowing time-branching and quantifiers.

In the following we cite some of the existing probabilistic model checkers.

MRMC (Markov Reward Model Checker) [46] is a model checker for DTMCs, CTMCs and CTMDPs (Continuous Time Markov Decision Processes) with rewards, supporting probabilistic computation tree logic and continuous stochastic logic. It also supports bisimulation.

The PEPA Plug-in Project [89] is a tool, supporting continuous stochastic logic, specifically developped for model checking PEPA networks (see Section 2.5).

LiQuor [15] is a model checker for markov decision processes expressed in Probmela [5] (a probabilistic extension of the SPIN's Promela language).

PRISM [49] is a probabilistic model checker for discrete time markov chains, markov decision processes and probabilistic automatas. Models are written in the PRISM language, which allows to specifies several properties and rewards, and which supports probabilistic computation tree, continuous stochastic and linear temporal logics.

PRISM model checker is particularly useful for model checking process calculi, since its language supports process algebra operators.

2.5 State of the Art

2.5.1 Process algebras

Probabilistic models are nowadays widely used in the design and verification of complex systems. In the following we give an overview of the systems specifically aimed at modelling mobile ad hoc and sensor networks.

CMN (*Calculus of Mobile Ad Hoc Networks*) Merro introduced CMN [58]: a *value-passing* calculus in CCS style [60], where an ad hoc network is represented as a collection of nodes, running in parallel, and using channels to broadcast messages. The peculiarity of this calculus is that the connectivity is defined by a physical transmission area: each node is associated with a location and a transmission radius which determine the circular area where the message could be received (the author considers omni-directional antennas, abstracting from the fact that the quality of the signal decreases proportionally with the distance from the transmission source). In addition, this calculus models both *mobile* and *stationary* nodes: mobile nodes can move arbitrarily within the networks, while stationary nodes occupy a fixed location.

This calculus presents some limitations: it gives a way of representing mobility of nodes, but the arbitrary connections and disconnections of nodes in the network, due to physical damages, or to intentional switching on/off of the devices, are not allowed. Another limitation is the impossibility of representing multicast and unicast communication: even though mobile ad hoc networks use only radio-frequencies, which do not allow one to make a channel private, in some cases it is necessary to specify a particular set of network locations: one may want to know if a message is receivable at some specific locations, e.g., in case of routing packets exchange.

ω-calculus ω-calculus [84] has been designed by Singh, Ramakrishnan e Smolka and it is a conservative extension of the π-calculus [61]. The key feature of this calculus is the separation of a node's communication and computational behavior from the description of its physical transmission range. The latter is modelled annotating a process with the set of group names which it belongs to. Since the ω-calculus is a conservative extension of the π-calculus, scope extrusion is defined (nodes can create new names and privately share them with other devices).

The peculiarity of this calculus is that not only broadcast transmissions are permitted, but also multicast and unicast communications.

Calculus for Broadcasting Systems (CBS$^{\#}$) CBS$^{\#}$ [66] has been introduced by Nanz and Hankin: it is an extension of CBS (Calculus of Broadcasting Systems) [76]; the peculiarity of this calculus is the choice of representing a node as a pair composed of a process and a store associated to a location. This solution allows the representation in detail of a network, that is usually constituted of devices having an own store.

Nevertheless, the actions executed on the store are internal to the node, so they are not observable actions: their representation make the calculus heavy without a real optimization. The other important peculiarity characterizing the CBS$^{\#}$ is that transmissions are not considered as atomic actions but, when a node executes an output action, the topology of the network may change arbitrarily before the reception of the message by the neighbors of the sender.

A Calculus for Mobile Ad Hoc Networks Jens Chr. Godskesen has proposed CMAN (Calculus for Mobile Ad Hoc Networks) [30], where connections between the nodes of the network is expressed using bidirectional links. A tag is associated to each node, containing the logical locations of the nodes it is connected with. Rules of CMAN allow scope extrusion. Contrarily to the other calculi we found in literature, here the network topology has been represented with bidirectional links between the logical locations of the devices (nodes are not associated to a transmission radius nor a physical location). The author had chosen this solution believing that, dealing with the node's behaviour in its intentions with the network and its neighbours, separately from its physical position, could simplify the model.

AWN (A process algebra for Wireless Mesh Networks) van Glabbeek et al. introduced AWN [19]: a process algebra that has been implemented with the specific aim of modelling wireless mesh routing protocols: it supports the representation of data structures, broadcast and unicast communications, and network topology changes. Since this calculus is specifically aimed at formalising mesh networks, in order to provide a rigorous analysis of routing protocols (see, e.g. , [20]), it is able to provide strongly precise and detailed networks representations.

However, the introduction of data structures and other specific information, make the calculus heavy and not flexible to model other kinds of systems. Moreover, the possibility of topology changes is modelled with connections and disconnections of nodes from each other, meaning that it does not exist any notion of distance or physical location.

CWS (A Calculus for Wireless Systems) CWS [51], introduced by Sangiorgi and Lanese, is a process calculus aimed at studying the problem of interference in mobile ad hoc networks. As in CMN [58], the connectivity of a node is expressed by its location and transmission radius. However, mobility is not modelled. The peculiarity of this calculus is the non-atomic nature of the transmissions, which allows one to capture the collisions which may occur during multiple transmissions by different nodes using the same channel.

The limitation of this calculus is the absence of primitives to deal with mobility.

KLAIM (Kernel Language for Agents Interaction and Mobility) De Nicola et al. presented KLAIM [68]: a process calculus aimed at modelling interactions among mobile agents in any distributed system.

The language consists of a core Linda with multiple tuple spaces and of a set of operators for building processes. It supports the modeling of explicit localities.

The most interesting aspect of this calculus is the concept of *coordination* : the coordination language defines the primitives to model configuration and interaction protocols of sets of software agents. Types are used to express polices of access control [67], and to study the possible access rights violations of mobile agents.

2.5.2 Probabilistic, Timed and Stochastic Process Calculi

Probabilistic extensions of Process Calculi are used to solve the non-determinism, given a quantification to the multiple choices of a systems.

Probabilistic Applied π-calculus Palamidessi et al. introduced the Probabilistic Applied π-calculus [31]: a probabilistic extension of Applied π-calculus [1], where both non-deterministic and probabilistic choices are modelled. The authors define both a static equivalence, and an obervational congruence based on the notion of probabilistic barb, which describes the probability, for a given system, to perform a certain observable action. In order to solve the non-determinism, schedulers (also called polices, or adversaries) have been introduced. They are modelled as functions mapping states into probability distributions.

Moreover, [70] an implementation of both probabilistic and stochastic versions of applied π-calculus has been provided, based on the MMC Model Checker [92] (a logic-programming based model checker for π-calculus), and on the PRISM Model

Checker [49]. The framework has been used to perform analysis of the semantic models derived from both the probabilistic and the stochastic calculus.

aTCWS (Applied Timed Calculus for Wireless Systems) Merro et al. introduced aTCWS [56]: a timed broadcasting process calculus for security analysis of wireless networks, with the assumption that the topology is fixed (it does not model the mobility of nodes), and that all nodes use the same fixed transmission radius (i.e., the same transmission power) and the same channel for their transmissions. The connectivity of the network is expressed by associating with each node a tag containing the list of all its neighbours. The timed model adopted by this calculus is known as the *fictitious clock* approach, and it is based on clock synchronization of nodes.

In [52] the authors propose a probabilistic version of TCWS, aimed at analysing communication protocols in wireless sensor networks. The main peculiarity of this calculus is the definition of a *simulation up to probability* which allows one to compare networks which exhibit the same behaviour up to a certain probability. This is an interesting result with respect, e.g., to the probabilstic applied π-calculus, presented in [31], where two networks can be compared only if they have exactly the same probability of performing observable actions.

The first limitation of the calculus is the absence of mobility, while the dynamic nature of the network topology is one of the most challenging issues of both MANETs and WSNs. The other limitation concerns the absence of channel specifications, while both mobile ad hoc and sensor networks often use radio-frequencies for their communication, which usually allows to choose among different channels.

A Probabilistic Calculus to model Distributed Wireless Networks In [14] Hennessy and Cerone propose a calculus aimed at modelling the high-level behviour of Wireless Systems (i.e., the MAC-layer protocols).

This calculus has been defined with a two-level structure: on one hand, it models both probabilistic and non-deterministic processes behaviour, and communications through a fixed set of channels, on the other hand, the topology is expressed through an undirected graph where each edge represents the direct link between a pair of network nodes. Neither a notion of distance, nor of transmission radius has been introduced. Moreover, modelling the links with an undirected graph presupposes the assumption that all the nodes use the same fixed radius to communicate: this is true for sensor networks, which are constituted of devices with the same characteristics, but it is not true for MANETs, which are constituted of different kinds of devices, having a different physical structure, and different power resources.

Probabilsitic Models for Mobile and Wireless Networks Song and Godskesen [85] propose a probabilistic broadcast calculus for mobile and wireless networks with unreliable connections. The peculiarity of this calculus is the introduction of a

probabilistic mobility function to model the mobility of nodes. Recently, in [86] the same authors propose a new version of their calculus built upon a *stochastic mobility function* to model the stochastic changes of connectivity. As in our works [26, 22, 27] broadcast actions are associated with the locations of the intended recipients of the message. However, differently from our calculus, in [86] any notion of transmission radius is introduced and any performance analysis is considered.

PEPA (Performance Evaluation Process Algebra) Hillston introduced PEPA [40]: the most famous stochastic process calculus: it extends the standard process algebras such as CCS [60] or CSP [41] by associating each action with a rate. It has been developed in order to join the properties of process algebras to model communication and concurrency of networks, with the powerful of the stochastic models such as the continuous-time markov processes, which allows to verify both qualitative and quantitative properties of communication systems.

The main problem of stochastic process algebras is the assignment of a rate to the synchronizing actions. In PEPA a shared activity is enabled only when it is enabled for all its components, and its rate is the minimum among the enabled activities of its type in the syncronizing components.

The authors provided also a tool, the PEPA Workbench [29], which allows a practical use of this process algebra in many applications concerning software architecture and communication protocols.

An interesting modification of PEPA, bio-PEPA [16], has been introduced with the specific aim of studying biological models.

TIPP (Timed Processes and Performability Evaluation) Hermanns et al. introduced TIPP [38]: a process algebra enriched with stochastic timing information. TIPP is similar to PEPA, except for the fact that the rates for the synchronizations are the product of the individual rates. Moreover no assumptions are made about the nature of the distributions.

EMPA$_{gr}$ (Extended Markovian Process Algebra with generative-reactive synchronizations) Bernardo introduced EMPA$_{gr}$[8], inspired by the TIPP calculus of Hermanns et al. and by the Hillston's PEPA calculus. It is aimed at modelling the fact that there are events whose duration influence the system, but also events whose duration is negligible.

The language models both exponentially timed, immediate and passive actions. The choice among immediate or exponentially timed actions is carried out generatively according to their priorities/probabilities or exponentially distributed durations, while the choice among passive actions is carried out reactively, by imposing that an immediate or exponentially timed action can synchronize with passive actions only: actions with the same name can partecipate, but there can be only one

immediate or exponentially timed action, while all the other participating actions must be passive.

Based on EMPA$_{gr}$, an open source software tool has been developped, called TwoTowers [2], modelled in the Æmilia Architecture Description Language [4], which can be used to make functional verification, security analysis and performance evaluation of communication and software systems.

3

A calculus for Energy-aware Multicast Communications

3.1 Introduction

We present a calculus for Energy-aware Broadcast, Unicast and Multicast communications of mobile ad hoc and sensor networks (EBUM) [25, 26, 27]. It allows us to model the ability of a node to broadcast a message to any other node within its physical transmission range, and to move in and out of the transmission range of other nodes in the network. The connectivity of a node is represented by a location and a transmission radius. Broadcast communications are limited to the transmission cell of the sender, while unicast and multicast communications are modelled by specifying, for each output action, the addresses of the intended recipients of the message.

We use the model to define some useful connectivity properties of Wireless Sensor Networks which can be exploited to control the energy consumption of the sensor nodes, increasing the network lifetime.

3.2 The Calculus

We introduce the EBUM calculus, which models mobile ad hoc networks as a collection of nodes, running in parallel, and using channels to broadcast messages. Our calculus supports multicast and unicast communications and it allows one to model the arbitrary and unexpected connections and disconnections of nodes in a network as well as the possibility for a node to administrate the energy consumption by choosing the optimal transmission radius to communicate with the desired recipients.

3.2.1 Syntax

In Table 3.1 we define the syntax of EBUM. This is defined in a two-level structure: the lower one for processes, the upper one for networks. The channel names

Table 3.1: Syntax

Networks	
M,N ::= $\mathbf{0}$	Empty network
$\mid M_1 \mid M_2$	Parallel composition
$\mid (\nu c)M$	Channel restriction
$\mid n[P]_l$	Node (or device)
Processes	
P,Q,R ::= $\mathbf{0}$	Inactive process
$\mid c(\tilde{x}).P$	Input
$\mid \bar{c}_{L,r}\langle\tilde{w}\rangle.P$	Output
$\mid [w_1 = w_2]P, Q$	Matching
$\mid A\langle\tilde{w}\rangle$	Recursion
Channels' description tags	
L ::= $\{l_1, l_2, l_3, ...\}$	Multicast/unicast channel
$\mid$ **Loc**	Broadcast channel
$\mid \epsilon$	Empty channel

set is separated from the names set.

$\mathbf{C}$ = channels, $d, c \in \mathbf{C}$;

$\mathbf{N}$ = names; m, n are used for nodes and r for transmission radii;

$\mathbf{Loc}$ = locations, $l, k, h \in \mathbf{Loc}$;

$\mathbf{X}$ = variables, x, y, z;

$\mathbf{V}$ = values ($\mathbf{X} \cup \mathbf{N} \cup \mathbf{Loc} \in \mathbf{V}$);

Values set includes names, variables and in general, any basic value (integers, booleans, etc.). Letters u, v are used for closed values, and w for open values. A tuple $a_1, ..., a_k$ of names is represented by $\tilde{a}$.

Networks are collections of nodes (which represent devices), running in parallel, using channels to communicate messages. As usual, $\mathbf{0}$ denotes the empty network and $M_1|M_2$ represents the parallel composition of two networks. In $(\nu c)M$ the channel c is private with scope M. We remark that channels are not values, hence they may not be transmitted: furthermore, given the structure of the syntactic productions, channels may not be dynamically created and thus $(\nu c)M$ simply plays the role of a CCS-style hiding operator [1]. We denote by $\mathcal{N}$ the set of all networks.

Processes are sequential and live within the nodes. Process $\mathbf{0}$ denotes the inactive process. Process $c(\tilde{x}).P$ can receive a tuple $\tilde{w}$ of (closed) values via channel c and continue as $P\{\tilde{w}/\tilde{x}\}$, i.e., as P with $\tilde{w}$ substituted for $\tilde{x}$ (where $|\tilde{x}| = |\tilde{w}|$). Process $\bar{c}_{L,r}\langle\tilde{w}\rangle.P$ can send a tuple of (closed) values $\tilde{w}$ via channel c and continue as P.

The tag L is used to list the observation locations of the message transmitted; mobile ad hoc networks use radiofrequences for communications, hence it is not possible to restrict channels. Moreover we would like to observe the behaviour of each transmission with respect to the specific locations it is addressed to. In particular $L = \mathbf{Loc}$ means a broadcast transmission (the message has got no intended recipients), otherwise the message is transmitted with multicast communication (unicast if the set L has only one member). We remark that L is not a set of names, but it is a set of locations. This is due to the fact that one of the main aims of our model is the analysis of protocols for mobile ad hoc networks management (as, e.g. ad hoc routing protocols), where messages are often addressed to a set of physical locations within the network area, rather than to specific devices. The tag r associated to an output action represents the radius (and then the power) used to transmit: we consider that in managing ad hoc networks the choice of the transmission power of each device may depend on precise strategies which are implemented in the communication protocols; then it is reasonable considering transmission range as an

[1] Of course, since channels represent radio frequencies, they may not be hidden in practice. Indeed, the use of the hiding operator is only meant to specialize the verification method to some specific class of contexts as we will see later.

information given by the process running in the sender node. Syntactically, the tags L and r associated to the channel c in an output action may be variables, but they must be instantiated when the output prefix is ready to fire. Note that a node n will never execute any process P requiring transmission radius $r > r_n$.

Process $[w_1 = w_2]P, Q$ behaves as P if $w_1 = w_2$, and as Q otherwise. We write $A\langle\tilde{w}\rangle$ to denote a process defined via a (possibly recursive) definition $A(\tilde{x}) \stackrel{\text{def}}{=} P$, with $|\tilde{x}| = |\tilde{w}|$, where $\tilde{x}$ contains all channels and variables that appear free in P.

Nodes cannot be created or destroyed. We write $n[P]_l$ for a node named n (this is the logical location of the device in the network), located at l (this is the physical location of the node), and executing a process P. We associate to each node identifier n a pair $\langle r_n, \delta_n\rangle$ where r_n represents the maximum transmission radius for n, while δ_n denotes the maximum distance that n can cover in a computational step. We say that $n[P]_l$ is *unpowered* when $r_n = 0$; we say that $n[P]_l$ is *static* when $\delta_n = 0$. The possibility that nodes communicate with each other is verified by looking at the physical locations and the transmission radius of the sender, in other words if a node broadcasts a message, this information will be received only by the nodes that lie in the area delimited by the transmission radius of the sender. In the definition of the operational semantics we then assume 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 $d(\cdot,\cdot)$ which takes two locations and returns their distance.

In the process $c(\tilde{x}).P$ variables $\tilde{x}$ are bound in P, giving rise to the standard notions of α-conversion and free and bound variables, denoted by $fv(\cdot)$ and $fb(\cdot)$ respectively. Similarly, in a network of the form $(\nu c)M$, the channel name c is bound in M and the notions of α-conversion and free and bound channels, $fc(\cdot)$ and $bc(\cdot)$, are defined accordingly.

Processes and networks are then identified up to α-conversion. More formally terms are considered as representatives of their equivalence class with respect to $\equiv_\alpha$, and these representatives will always be chosen so that bound names are distinct from free names. A context $\mathcal{C}[\cdot]$ is defined as a network term with a hole, denoted by $[\cdot]$. Contexts are generated by the following grammar:

$$\mathcal{C}[\cdot] ::= [\cdot] \mid [\cdot]|M \mid M|[\cdot] \mid (\nu c)[\cdot] \tag{3.1}$$

A number of conventions are used to simplify the notation. Parallel composition of networks has lower precedence with respect to restriction. $\prod_{i\in I} M_i$ means the parallel composition of all the networks M_i, $\forall i \in I$. We write $(\nu\tilde{c})M$ as an abbreviation for $(\nu c_1)...(\nu c_k)M$. To denote unicast communication we write c_l for $c_{\{l\}}$. We write $\bar{c}_{L,r}\langle w\rangle$ for $\bar{c}_{L,r}\langle w\rangle.\mathbf{0}$, $\mathbf{0}$ for $n[\mathbf{0}]_l$ and $[w_1 = w_2]P$ as an abbreviation of $[w_1 = w_2]P, \mathbf{0}$. We assume that there are no free variables in a network (while there can be free channels). The absence of free variables is trivially maintained as the network evolves. Moreover, we assume that in any network each node identifier is

$n[\mathbf{0}]_l \equiv \mathbf{0}$	(Struct Zero)
$n[[v = v]P, Q]_l \equiv n[P]_l$	(Struct Then)
$n[[v_1 = v_2]P, Q]_l \equiv n[Q]_l \quad \text{if } v_1 \neq v_2$	(Struct Else)
$n[A\langle\tilde{v}\rangle]_l \equiv n[P\{\tilde{v}/\tilde{x}\}]_l \quad \text{if } A(\tilde{x}) \stackrel{\text{def}}{=} P \wedge \lvert\tilde{x}\rvert = \lvert\tilde{v}\rvert$	(Struct Rec)
$M \mid N \equiv N \mid M$	(Struct Par Comm)
$(M \mid N) \mid M' \equiv M \mid (N \mid M')$	(Struct Par Assoc)
$M \mid \mathbf{0} \equiv M$	(Struct Zero Par)
$(\nu c)\mathbf{0} \equiv \mathbf{0}$	(Struct Zero Res)
$(\nu c)(\nu d)M \equiv (\nu d)(\nu c)M$	(Struct Res Res)
$(\nu c)(M \mid N) \equiv M \mid (\nu c)N \quad \text{If } c \notin fc(M)$	(Struct Res Par)
$M \equiv M$	(Struct Refl)
$N \equiv M \quad \text{if } M \equiv N$	(Struct Symm)
$M \equiv M'' \quad \text{if } M \equiv M' \wedge M' \equiv M''$	(Struct Trans)
$M \mid M' \equiv N \mid M' \quad \forall M' \text{ if } M \equiv N$	(Struct Cxt Par)
$(\nu c)M \equiv (\nu c)N \quad \forall c \quad \text{if } M \equiv N$	(Struct Cxt Res)

Table 3.2: Structural Congruence

unique.

3.2.2 Reduction Semantics

The dynamics of the calculus is specified by the *reduction relation* over networks ($\rightarrow$), described in Table 3.3. As usual, it relies on an auxiliary relation, called structural congruence ($\equiv$), which is the least contextual equivalence relation satisfying the rules defined in Table 3.2.

Rule (R-Bcast) models the transmission of a tuple $\tilde{v}$ through a channel $c_{L,r}$. The set L associated to channel c indicates the set of the receivers lying within the observation locations. Indeed, nodes communicate using radio frequencies that enable only broadcast messages (monopolizing channels is not permitted). However, a node may decide to communicate with a specific node (or group of nodes), this is the reason why we decided to associate to each output action a set of observation recipients. The cardinality of this set indicates the kind of communication that is used: if $L = \mathbf{Loc}$ then the recipient set is the whole network and a broadcast transmission is performed, while if L is a finite set (respectively, a singleton) then a multicast (respectively, a unicast) communication is realized. A radius r is also associated to the channel c, indicating the transmission radius required for that communications which may depend on the power consumption strategy adopted by the surrounding protocol.

Transmissions are defined as *non-blocking actions*: a transmission proceeds even

$$\text{(R-Bcast)}\ \frac{r \leq r_n\ \forall i \in I.d(l,l_i) \leq r,\ r_i \neq 0,\ |\tilde{x_i}| = |\tilde{v}|}{n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid \prod_{i\in I} n_i[c(\tilde{x_i}).P_i]_{l_i} \rightarrow n[P]_l \mid \prod_{i\in I} n_i[P_i\{\tilde{v}/\tilde{x_i}\}]_{l_i}}$$

$$\text{(R-Move)}\ \frac{d(l,k) \leq \delta_n}{n[P]_l \rightarrow n[P]_k} \qquad \text{(R-Par)}\ \frac{M \rightarrow M'}{M|N \rightarrow M'|N}$$

$$\text{(R-Res)}\ \frac{M \rightarrow M'}{(\nu c)M \rightarrow (\nu c)M'} \qquad \text{(R-Struct)}\ \frac{M \equiv N\ N \rightarrow N'\ N' \equiv M'}{M \rightarrow M'}$$

Table 3.3: Reduction Semantics

if there are no nodes listening for messages. The messages transmitted will be received only by those nodes which lie in the transmission area of the sender. It may occur that some recipients within the range of the transmitter do not receive the message. This may be due to several reasons that concern the instability and dynamism of the network. In terms of observation this corresponds to a local activity of the network that is not captured by any external observer.

Rule (R-Move) models arbitrary and unpredictable movements of mobile nodes. As said above, δ_n denotes the maximum distance that node n can cover in a computational step. Movements are atomic actions: while moving, a node cannot do anything else. In our model, due to the interleaving nature of the calculus, only one node can move at each reduction but this does not mean that only one node can move at a time.

Rules (R-Res) and (R-Par) describe the contextuality of the reduction relation.

Finally, (R-Struct) models the reduction closure under the structural congruence (following the rules introduced in Table 3.2).

We denote by $\rightarrow^*$ the reflexive and transitive closure of $\rightarrow$.

3.2.3 Observational Semantics.

In observational semantics two terms are deemed equivalent if they have the same observational behavior in all possible contexts.

The central actions of our calculus are transmission and reception of messages. However, only the transmission of messages can be observed. An observer cannot be sure whether a recipient actually receives a given value. Instead, if a node receives a message, then surely someone must have sent it. Following [62], we use the term *barb* as a synonymous of observable.

In our definition of barb a transmission is observable only if at least one of the observation locations is inside the transmission area.

Definition 3.1 (Barb) *Let* $M \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v} \rangle.P]_l | M')$, *with* $c \notin \tilde{d}$.
If $\exists K \subseteq L \wedge d(l,k) \leqslant r \quad \forall k \in K \wedge K \neq \emptyset$ *then* $M \downarrow_{c@K}$
If $M \longrightarrow^* M' \downarrow_{c@K}$ *then* $M \Downarrow_{c@K}$.

Notice that, if $M \equiv n[\bar{c}_{L,r}\langle \tilde{v} \rangle.P]_l | M'$ and $M \downarrow_{c@K}$ then at least one of the recipients in L is actually able to receive the message.

The concept of observable action is illustrated in Figure 3.1. Consider a node n (the red node in the picture) broadcasting a message which is destined to a specific set L of observation locations. The black circles in the picture represent the network locations not included in L, while the light blue circles represent the locations in L, i.e., the observation locations of the message. Figure 3.1(a) depicts the situation in which there is at least one location in L where a receiver could be reached by the transmission, while Figure 3.1(b) illustrates the case of a non-observable action, where none of the nodes in L is able to receive the message.

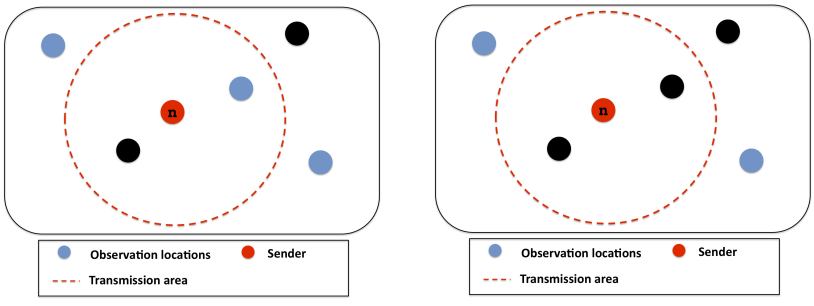
To define our observation equivalence we will ask for the largest relation which satisfies the following properties.

Definition 3.2 *A relation* $\mathcal{R}$ *is* barb preserving *if* $M \mathcal{R} N$ *and* $M \downarrow_{c@K}$ *implies* $N \Downarrow_{c@K}$

Definition 3.3 *A relation* $\mathcal{R}$ *is* reduction closed *if* $M \mathcal{R} N$ *and* $M \longrightarrow M'$ *implies the existence of some* N' *such that* $N \longrightarrow^* N'$ *and* $M' \mathcal{R} N'$

Definition 3.4 *A relation* $\mathcal{R}$ *is* contextual *if* $M \mathcal{R} N$ *implies* $\mathcal{C}[M] \mathcal{R} \mathcal{C}[N]$ *for all contexts* $\mathcal{C}[\cdot]$.

Definition 3.5 (Reduction barbed congruence) *Reduction barbed congruence, written* $\cong$, *is the largest symmetric relation over networks, which is reduction closed, barb preserving, and contextual.*



(a) Observable action (b) Non-observable action

Figure 3.1: Observability

$$\text{(Output)}\ \frac{-}{\bar{c}_{L,r}\langle\tilde{v}\rangle.P \xrightarrow{\bar{c}_{L,r}\tilde{v}} P} \qquad \text{(Input)}\ \frac{-}{c(\tilde{x}).P \xrightarrow{c\tilde{v}} P\{\tilde{v}/\tilde{x}\}}$$

$$\text{(Then)}\ \frac{P \xrightarrow{\eta} P'}{[\tilde{v} = \tilde{v}]P, Q \xrightarrow{\eta} P'} \qquad \text{(Else)}\ \frac{Q \xrightarrow{\eta} Q'}{[\tilde{v}_1 = \tilde{v}_2]P, Q \xrightarrow{\eta} Q'}\ \tilde{v}_1 \neq \tilde{v}_2$$

$$\text{(Rec)}\ \frac{P\{\tilde{v}/\tilde{x}\} \xrightarrow{\eta} P'}{A\langle\tilde{v}\rangle \xrightarrow{\eta} P'}\ A(\tilde{x}) \stackrel{\text{def}}{=} P$$

Table 3.4: LTS rules for Processes

Two networks are related by $\cong$ if they exhibit the same behaviour relative to the corresponding sets of intended recipients. Hereafter we develop a bisimulation-based proof technique for $\cong$. It provides an efficient method to check whether two networks are related by $\cong$.

3.3 A Bisimulation-based Proof Technique

We develop a co-inductive proof technique for the relation $\cong$.

3.3.1 Label Transition Semantics

In this section we describe the LTS (*Label Transition System*) semantics of EBUM. LTS has two sets of rules: one for processes and one for networks.

Table 3.4 presents the LTS for processes. Transitions are of the form $P \xrightarrow{\eta} P'$, where η ranges over input and output actions. More precisely $c\tilde{v}$ and $\bar{c}_{L,r}\tilde{v}$ denote, respectively, input and output of a tuple $\tilde{v}$ of values at channel c. The grammar for η is:

$$\eta ::= c\tilde{v} \mid \bar{c}_{L,r}\tilde{v}. \tag{3.2}$$

Rules for processes are simple and they do not need deeper explanations.

Table 3.5 contains the LTS for networks. Transitions are of the form $M \xrightarrow{\gamma} M'$, where the grammar for γ is:

$$\gamma ::= c?\tilde{v}@l \mid c_L!\tilde{v}[l, r] \mid c!\tilde{v}@K \triangleleft R \mid \tau. \tag{3.3}$$

Rule (Snd) models the sending, with transmission radius r, of the tuple $\tilde{v}$ of values via channel c to the set L of receivers.

Rule (Rcv) models the reception of $\tilde{v}$ at l via channel c.

$$\text{(Snd)}\frac{P \xrightarrow{\bar{c}_{L,r}\tilde{v}} P'}{n[P]_l \xrightarrow{c_L!\tilde{v}[l,r]} n[P']_l} r \leq r_n \qquad \text{(Rcv)}\frac{P \xrightarrow{c\tilde{v}} P'}{n[P]_l \xrightarrow{c?\tilde{v}@l} n[P']_l} r_n \neq 0$$

$$\text{(Bcast)}\ \frac{M \xrightarrow{c_L!\tilde{v}[l,r]} M'\ N \xrightarrow{c?\tilde{v}@l'} N'\ d(l,l') \leq r}{\begin{array}{c} M|N \xrightarrow{c_L!\tilde{v}[l,r]} M'|N' \\ N|M \xrightarrow{c_L!\tilde{v}[l,r]} N'|M' \end{array}}$$

$$\text{(Obs)}\ \frac{M \xrightarrow{c_L!\tilde{v}[l,r]} M'\ R \subseteq \{k : d(l,k) \leq r\}, K = R \cap L, K \neq \emptyset}{M \xrightarrow{c!\tilde{v}@K \triangleleft R} M'}$$

$$\text{(Lose)}\ \frac{M \xrightarrow{c_L!\tilde{v}[l,r]} M'}{M \xrightarrow{\tau} M'} \qquad \text{(Move)}\ \frac{d(l,k) \leq \delta_n}{n[P]_l \xrightarrow{\tau} n[P]_k}$$

$$\text{(Par)}\ \frac{M \xrightarrow{\gamma} M'}{\begin{array}{c} M|N \xrightarrow{\gamma} M'|N \\ N|M \xrightarrow{\gamma} N|M' \end{array}} \qquad \text{(Res)}\ \frac{M \xrightarrow{\gamma} M'\ c \notin fc(\gamma)}{(\nu c)M \xrightarrow{\gamma} (\nu c)M'}$$

Table 3.5: LTS ruels for Networks

Rule (Bcast) models the propagation of broadcast. All the nodes lying within the transmission cell of the transmitter may hear the communication, regardless of the fact that the node hearing the communication is a member of L.

Rule (Obs) models the observability of a transmission: every output action may be detected by any receiver located within the transmission cell of the sender, but it will be observed only by that lying in the observation locations. The action $c!\tilde{v}@K \triangleleft R$ represents the transmission of tuple $\tilde{v}$ of messages via c to the set of recipients of L whose locations lie within the transmission cell of the transmitter ($R \subseteq \{k : d(k,l) \leqslant r\}$ and $K = R \cap L$, where l and r are respectively the location and the transmission radius of the message sender). If $K \neq \emptyset$ this is an observable action corresponding to the barb $\downarrow_{c@K}$.

Rule (Lose) models both message loss and a local activity of the network which an observer is not party to. τ-actions are used, as commonly in process calculi, to denote non-observable actions.

Rule (Move) models migration of a mobile node from a location k to a new location l; again δ_n represents the maximum distance that a node can cover in a single computational step.

In the following we prove that the LTS-based semantics coincides with the re-

duction semantics and the notion of observability given in the previous section.

Lemma 3.1 *Let M be a network.*

1. *If $M \xrightarrow{c?\tilde{v}@l} M'$, then there exist n, $\tilde{x}$, a (possibly empty) sequence $\tilde{d}$ such that $c \notin \tilde{d}$, a process P and a (possibly empty) network M_1 such that*

$$M \equiv (\nu\tilde{d})(n[c(\tilde{x}).P]_l | M_1)$$

and

$$M' \equiv (\nu\tilde{d})(n[P\{\tilde{v}/\tilde{x}\}]_l | M_1).$$

2. *If $M \xrightarrow{c_L!\tilde{v}[l,r]} M'$, then there exist n, a (possibly empty) sequence $\tilde{d}$ such that $c \notin \tilde{d}$, a process P, a (possibly empty) network M_1 and a (possibly empty) set I, such that $\forall i \in I$ with $d(l, l_i) \leq r$, such that:*

$$M \equiv (\nu\tilde{d})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l | \prod_{i\in I} n_i[c(\tilde{x}_i).P_i]_{l_i} \mid M_1)$$

and

$$M' \equiv (\nu\tilde{d})(n[P]_l | \prod_{i\in I} n_i[P_i\{\tilde{v}/\tilde{x}_i\}]_{l_i} \mid M_1).$$

Proof.

By induction on the transition rules of Table 3.5.

Case 1: $M \xrightarrow{c?\tilde{v}@l} M'$.

(Rcv) Let $M \xrightarrow{c?\tilde{v}@l} M'$ inferred by rule (Rcv), then there exist $n, P, \tilde{x}, l, r$ such that
$M \equiv n[c(\tilde{x}).P]_l$ and
$M' \equiv n[P\{\tilde{v}/\tilde{x}\}]_l$.

The Lemma is proved by considering the empty network M_1 and the empty sequence $\tilde{d}$.

(Par) Let $M \xrightarrow{c?\tilde{v}@l} M$ inferred by rule (Par), where $M \equiv M_1 \mid M_2$, $M' \equiv M_1' \mid M_2$ and $M_1 \xrightarrow{c?\tilde{v}@l} M_1'$. By induction hypothesis we have:

$$M_1 \equiv (\nu\tilde{d})(n[c(\tilde{x}).P]_l | N)$$

and

$$M_1' \equiv (\nu\tilde{d})(n[P\{\tilde{v}/\tilde{x}\}]_l | N),$$

for some n, some $\tilde{x}$, some (possibly empty) sequence $\tilde{d}$ with $c \notin \tilde{d}$, some process P and some (possibly empty) network N. By applying rule (Struct Cxt Par), (Struct Par Assoc) and (Struct Res Par) we obtain

$$M_1 \mid M_2 \equiv (\nu\tilde{d})(n[c(\tilde{x}).P]_l | (N \mid M_2))$$

and

$$M_1' \mid M_2 \equiv (\nu\tilde{d})(n[P\{\tilde{v}/\tilde{x}\}]_l | (N \mid M_2)).$$

(Res) Let $M \xrightarrow{c?\tilde{v}@l} M'$ inferred by rule (Res), where $M \equiv (\nu c')M_1$, $M' \equiv (\nu c')M_1'$ and $M_1 \xrightarrow{c?\tilde{v}@l} M_1'$ and $c \neq c'$. By induction hypothesis we have

$$M_1 \equiv (\nu \tilde{d})(n[c(\tilde{x}).P]_l | N)$$

and

$$M_1' \equiv (\nu \tilde{d})(n[P\{\tilde{v}/\tilde{x}\}]_l | N),$$

for some n, some $\tilde{x}$, some (possibly empty) sequence $\tilde{d}$ with $c \notin \tilde{d}$, some process P and some (possibly empty) network N.

If we consider $\tilde{d'} = \tilde{d} \cup c'$, since $c \notin \tilde{d'}$

$$M \equiv (\nu \tilde{d'})(n[c(\tilde{x}).P]_l | N)$$

and

$$M' \equiv (\nu \tilde{d'})(n[P\{\tilde{v}/\tilde{x}\}]_l | N).$$

Case 2: $M \xrightarrow{c_L!\tilde{v}[l,r]} M'$.

(Snd) Let $M \xrightarrow{c_L!\tilde{v}[l,r]} M'$ inferred by rule (Snd), then $M \equiv n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l$ and $M' \equiv n[P]_l$ for some name n, and some process P. Since $\bar{c}_{L,r}\langle\tilde{v}\rangle.P \xrightarrow{\bar{c_{L,r}}} P$, if we suppose I, $\tilde{d}$ and M_1 empty, the lemma is proved because

$$M \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l | \prod_{i \in I} n_i[c(\tilde{x_i})P_i]_{l_i} \mid M_1)$$

and

$$M' \equiv (\nu \tilde{d})(n[P]_l | \prod_{i \in I} n_i[P_i\{\tilde{v}/\tilde{x_i}\}]_{l_i} \mid M_1).$$

(Bcast) Let $M \xrightarrow{c_L!\tilde{v}[l,r]} M'$ because $M \equiv M_1 \mid M_2$, $M' \equiv M_1' \mid M_2'$, $M_1 \xrightarrow{c_L!\tilde{v}[l,r]} M_1'$ and $M_2 \xrightarrow{c?\tilde{v}@l'} M_2'$, with $d(l,l') \leq r$. By induction hypothesis:

$$M_1 \equiv (\nu \tilde{d_1})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l | \prod_{i \in I} n_i[c(\tilde{x_i})P_i]_{l_i} \mid N_1)$$

and

$$M_1' \equiv (\nu \tilde{d_1})(n[P]_l | \prod_{i \in I} n_i[P_i\{\tilde{v}/\tilde{x_i}\}]_{l_i} \mid N_1),$$

for some n, P, $\tilde{v}$, l, some (possibly empty) sequence $\tilde{d_1}$ such that $c \notin \tilde{d_1}$, some (possibly empty) set I such that $d(l,l_i) \leq r \ \forall i \in I$ and some (possibly empty) network N_1, and

$$M_2 \equiv (\nu \tilde{d_2})(m[c(\tilde{x}).Q]_{l'} | N_2)$$

and

$$M_2 \equiv (\nu \tilde{d_2})(m[Q\{\tilde{v}/\tilde{x}\}]_{l'} | N_2),$$

for some m, some process Q, some (possibly empty) sequence $\tilde{d_2}$ such that $c \notin \tilde{d_2}$ and some (possibly empty) network N_2 . By applying rules (Struct Cxt Par), (Struct Par Assoc) and (Struct Res Par), if we consider $\tilde{d} = \tilde{d_1} \cup \tilde{d_2}$, we can assume that $\tilde{d_1} \notin fc(M_2)$ and $\tilde{d_2} \notin fc(M_1)$ and we get:

$$M \mid N \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v} \rangle.P]_l | m[c(\tilde{x}).Q]_{l'} \mid \prod_{i \in I} n_i[c(\tilde{x_i})P_i]_{l_i} \mid (M_1 \mid N_1))$$

and

$$M' \mid N' \equiv (\nu \tilde{d})(n[P]_l | m[Q\{\tilde{v}/\tilde{x}\}]_{l'} \mid \prod_{i \in I} n_i[P_i\{\tilde{v}/\tilde{x}\}]_{l_i} \mid (M_1 \mid N_1)).$$

The proof of the other cases is analogous to the *Case 1.*

We also show that structural congruence respects the transitions of Table 3.5.

Lemma 3.2 *If $M \xrightarrow{\gamma} M'$ and $M \equiv N$, then there exists N' such that $N \xrightarrow{\gamma} N'$ and $M' \equiv N'$.*

Proof.

By induction on the depth of the inference $M \xrightarrow{\gamma} M'$.

There are a lot of cases to consider, following we give only some example.

(Par) Let us consider $M \xrightarrow{\gamma} M'$ where $M \equiv M_1 \mid M_2$, $M' \equiv M_1' \mid M_2$ and $M_1 \xrightarrow{\gamma} M_1'$. There are several rules for structural congruence that can be applied.

Let us consider $M_1 \mid M_2 \equiv M_1 \mid M_3$ because $M_2 \equiv M_3$, by (Struct Cxt Par). By hypothesis $M_1 \xrightarrow{\gamma} M_1'$ and, by an application of rule (Par) we get $M_1 \mid M_3 \xrightarrow{\gamma} M_1' \mid M_3$. But, since $M_2 \equiv M_3$, by applying (Struct Cxt Par) we have that $M_1' \mid M_3 \equiv M_1' \mid M_2$, as required.

Let us consider now $M_1 \mid M_2 \equiv M_2 \mid M_1$ by (Struct Par Com). Again, since $M_1 \xrightarrow{\gamma} M_1'$, by applying rule (Par) we get $M_2 \mid M_1 \xrightarrow{\gamma} M_2 \mid M_1'$, and, by applying again rule (Struct Par Com), $M_2 \mid M_1' \equiv M_1' \mid M_2$, as required.

(Bcast) Let consider $M \xrightarrow{c_L!\tilde{v}[l,r]} M$, where $M \equiv M_1 \mid M_2$, $M' \equiv M_1' \mid M_2'$, $M_1 \xrightarrow{c_L!\tilde{v}[l,r]} M_1'$, $M_2 \xrightarrow{c?\tilde{v}@l'} M_2'$ and $d(l,l') \leq r$.

Again several rules for structural congruence can be applied.

Let us consider, e.g. $M_1 \mid M_2 \equiv M_2 \mid M_1$ by (Struct Par Comm). Since $M_1 \xrightarrow{c_L!\tilde{v}[l,r]} M_1'$, $M_2 \xrightarrow{c?\tilde{v}@l'} M_2'$, and $d(l,l') \leq r$, we can apply rule (Bcast),

obtaining $M_2 \mid M_1 \xrightarrow{c_L!\tilde{v}[l,r]} M_2' \mid M_1'$, and, by applying again (Struct Par Comm) we get $M_2' \mid M_1' \equiv M_1' \mid M_2'$ as required.

Let us consider now $M_3 \mid M_2 \equiv M_1 \mid M_2$ because $M_3 \equiv M_1$, by (Struct Cxt Par). By induction hypothesis $M_3 \xrightarrow{c_L!\tilde{v}[l,r]} M_3'$, and $M_3' \equiv M_1'$. But since $d(l,l') \leq r$ we can apply rule (Bcast), obtaining $M_3 \mid M_2 \xrightarrow{c_L!\tilde{v}[l,r]} M_3' \mid M_2'$. Now, by applying (Struct Cxt Par) we get $M_3' \mid M_2' \equiv M_1' \mid M_2'$, as required.

The proof of the other cases is similar.

The following theorem establishes the relationship between the reduction semantics and the LTS one.

Theorem 3.1 (Harmony) *Let M be a network.*

1. *$M \downarrow_{c@K}$ if and only if there exist $\tilde{v}$, $R \supseteq K$ and $N \equiv M$ such that $N \xrightarrow{c!\tilde{v}@K\triangleleft R}$.*
2. *If $M \xrightarrow{\tau} M'$ then $M \rightarrow M'$.*
3. *If $M \rightarrow M'$ then $\exists N \equiv M$ and $N' \equiv M'$ such that $N \rightarrow N'$.*

Proof.

1. The first part of the theorem follows straightforwardly from Lemma 3.1 and the definition of Barb.

   $\Rightarrow$ If $M \downarrow_{c@K}$, by the definition of Barb:
   $M \equiv (\nu\tilde{d})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid M_1)$, for some n, $\tilde{v}$, L, r, some (possibly empty) sequence $\tilde{d}$ with $c \notin \tilde{d}$, some process P and some (possibly empty) network M_1, with $K \subseteq \{k \in L \text{ s.t. } d(l,k) \leq r\}$ and $K \neq \emptyset$.
   By applying the rules (Snd), (Par) and (Res) (since $c \notin \tilde{d}$) we obtain:

   $$\frac{n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \xrightarrow{c_L!\tilde{v}[l,r]} n[P]_l}{(\nu\tilde{d})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid M_1) \xrightarrow{c_L!\tilde{v}[l,r]} (\nu\tilde{d})(n[P]_l \mid M_1)};$$

   then we can apply rule (Obs):
   $(\nu\tilde{d})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid M_1) \xrightarrow{c!\tilde{v}@K\triangleleft R} (\nu\tilde{d})(n[P]_l \mid M_1)$,
   where $R \subseteq \{l' \in \mathbf{Loc} : d(l,l') \leq r\}$, as required.

   $\Leftarrow$ If $M \xrightarrow{c!\tilde{v}@K\triangleleft R} M'$, we can apply rule (Obs) backwardly and we get $M \xrightarrow{c_L!\tilde{v}![l,r]} M'$, for some set L of locations, some location l, some radius r and, by applying lemma 3.1 then there exists n, some (possibly empty) sequence

$\tilde{d}$ such that $c \notin \tilde{d}$, some process P, some (possibly empty) network M_1 and a (possibly empty) set I, such that $\forall i \in I\ d(l, l_i) \leq r$ and we get:

$$M \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v} \rangle.P]_l | \prod_{i \in I} n_i[c(\tilde{x}_i).P_i]_{l_i} \mid M_1)$$

and

$$M' \equiv (\nu \tilde{d})(n[P]_l | \prod_{i \in I} n_i[P_i\{\tilde{v}/\tilde{x}_i\}]_{l_i} \mid M_1).$$

Since $K \neq \emptyset$, by applying the definition of barb we conclude $M \downarrow_{c@K}$.

2. The second part of the theorem is proved by induction on the derivation $M \xrightarrow{\tau} M'$.
Suppose that $M \xrightarrow{\tau} M'$ is due to an application of the rule (Move), that means $M \equiv n[P]_l$, $M' \equiv n[P]_k$ for some name n, some process P, and some locations l and k such that $d(l, k) \leq \delta_n$, and:

$$\frac{d(l,k) \leq \delta_n}{n[P]_l \xrightarrow{\tau} n[P]_l}.$$

Hence , by applying (R-Move) we get:

$$\frac{d(l,k) \leq \delta_n}{n[P]_l \rightarrow n[P]_l},$$

If $M \xrightarrow{\tau} M'$ is due to an application of (Lose):

$$\frac{M \xrightarrow{c_L ! \tilde{v}[l,r]} M'}{M \xrightarrow{\tau} M'}.$$

By applying Lemma 3.1, there exist n, $\tilde{v}$, a (possibly empty) sequence $\tilde{d}$ such that $c \notin \tilde{d}$, a process P, a (possibly empty) network M_1 and a (possibly empty) set I such that $\forall i \in I\ d(l, l_i) \leq r$, such that:

$$M \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v} \rangle.P]_l | \prod_{i \in I} n_i[c(\tilde{x}_i).P_i]_{l_i} \mid M_1)$$

and

$$M' \equiv (\nu \tilde{d})(n[P]_l | \prod_{i \in i} n_i[P_i\{\tilde{v}/\tilde{x}_i\}]_{l_i} \mid M_1).$$

Finally, by applying rule (R-Bcast), (R-Res) and (R-Struct) we get $M \rightarrow M'$.

Suppose that $M \xrightarrow{\tau} M'$ is due to the application of (Res), that means $M \equiv (\nu c)M_1$, $M' \equiv (\nu c)M_1'$ and:

$$\frac{M_1 \xrightarrow{\tau} M_1'}{(\nu c)M_1 \xrightarrow{\tau} (\nu c)M_1'}.$$

By induction hypothesis $M_1 \rightarrow M_1'$, hence, by applying rule (R-Res) we get $(\nu c)M_1 \rightarrow (\nu c)M_1'$, and by applying rule (R-Struct) we finally get $M \rightarrow M'$.

Finally, suppose that $M \xrightarrow{\tau} M'$ is due to the application of (Par), that means $M \equiv M_1 \mid M_2$, $M' \equiv M_1' \mid M_2$ and:

$$\frac{M_1 \xrightarrow{\tau} M_1'}{M_1|M_2 \xrightarrow{\tau} M_1'|M_2}.$$

By induction hypothesis $M_1 \rightarrow M_1'$, hence, by applying rule (R-Par) we get $M_1|M_2 \rightarrow M_1'|M_2$, and by applying rule (R-Struct) we finally get $M \rightarrow M'$.

3. The third part is proved by induction on the reduction $M \rightarrow M'$.

   Suppose that $M \rightarrow M'$ is due to the application of the rule (R-Move), meaning $M \equiv n[P]_l$, $M' \equiv n[P]_k$ for some name n, some process P and some locations l and k such that $d(l,k) \leq \delta_n$, and:

$$\frac{d(l,k) \leq \delta_n}{n[P]_l \rightarrow n[P]_l}.$$

   We simply apply rule (Move) to obtain:

$$\frac{d(l,k) \leq \delta_n}{n[P]_l \xrightarrow{\tau} n[P]_l},$$

   Suppose that $M \rightarrow M'$ is due to the application of the rule (R-Par), meaning that $M \equiv M_1 \mid M_2$, $M' \equiv M_1' \mid M_2$ and:

$$\frac{M_1 \rightarrow M_1'}{M_1 \mid M_2 \rightarrow M_1' \mid M_2}.$$

   By induction hypothesis $\exists N \equiv M_1$ and $N' \equiv M_1'$ such that $N \xrightarrow{\tau} N'$, then by applying rule (Par) we get:

$$\frac{N \xrightarrow{\tau} N'}{N \mid M_2 \xrightarrow{\tau} N' \mid M_2},$$

and, by applying (Struct Cxt Par) $N \mid M_2 \equiv M_1 \mid M_2$ and $N' \mid M_2 \equiv M_1' \mid M_2$, as required.

Suppose that $M \to M'$ is due to the application of the rule (R-Res), meaning $M \equiv (\nu c)M_1$, $M' \equiv (\nu c)M_1'$ and:

$$\frac{M_1 \to M_1'}{(\nu c)M_1 \to (\nu c)M_1'}.$$

By induction hypothesis $\exists N \equiv M_1$ and $N' \equiv M_1'$ such that $N \xrightarrow{\tau} N'$ with $N' \equiv M_1'$, then by applying rule (Res), since $\texttt{Chan}(\tau) \neq c$ we get:

$$\frac{N \xrightarrow{\tau} N'}{(\nu c)N \xrightarrow{\tau} (\nu c)N'},$$

and, by applying (Struct Cxt Res) and (Struct-Trans) $(\nu c)N \equiv M$ and $(\nu c)N' \equiv M'$ as required.

Suppose that $M \to M'$ is due to the application of the rule (Bcast). It means $M \equiv n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid \prod_{i\in I} n_i[c(\tilde{x_i}).P_i]_{l_i}$, $M' \equiv n[P]_l \mid \prod_{i\in I} n_i[P_i\{\tilde{v}/\tilde{x_i}\}]_{l_i}$, for some name n, some channel c, some set L of locations, some radius r, some tuple $\tilde{v}$ of messages, some process P, some location l and some (possibly empty) set I such that $d(l, l_i) \leq r$, $\forall i \in I$, and:

$$\frac{\forall i \in I.d(l, l_i) \leq r}{n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid \prod_{i\in I} n_i[c(\tilde{x_i}).P_i]_{l_i} \to n[P]_l \mid \prod_{i\in I} n_i[P_i\{\tilde{v}/\tilde{x_i}\}]_{l_i}}.$$

By applying rule (Snd), (Rcv), $\mid I \mid$ times rule (Bcast) and, finally rule (Lose), we get

$$n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid \prod_{i\in I} n_i[c(\tilde{x_i}).P_i]_{l_i} \xrightarrow{\tau} n[P]_l \mid \prod_{i\in I} n_i[P_i\{\tilde{v}/\tilde{x_i}\}]_{l_i},$$

as required.

Finally let suppose that the reduction $M \to M'$ is due to an application of rule (R-Struct):

$$\frac{M \equiv N \quad N \to N' \quad N' \equiv M'}{M \to M'}.$$

By induction hypothesis there exists $N_1 \equiv N$ and $N_2 \equiv N'$ such that $N_1 \xrightarrow{\tau} N_2$. Then, by applying the rule for structural congruence (Struct Trans) we get $M \equiv N \equiv N_1$ and $M' \equiv N' \equiv N_2$, as required.

3.3.2 Simulation and Bisimulation

In this section, using our LTS, we define notions of simulation and bisimulation. Then we prove that bisimulation is a complete characterisation of *reduction barbed congruence*, and hence represents a valid method to prove that two networks are reduction barbed congruent. This property let us deal with all issues that do not permit the correct behaviour of mobile ad hoc networks. We then have to prove both that $\cong \subseteq \approx$ and that $\approx \subseteq \cong$.

For convenience we use metavariable α to range over those actions that will be used in the definition of bisimulation.

$$\alpha ::= c?\tilde{v}@l \mid c!\tilde{v}@K \triangleleft R \mid \tau.$$

Since we are interested in *weak behavioural equivalences*, that abstract over τ-actions, we introduce the notion of *weak action*.

Definition 3.6 (Weak Action) *The definition of* weak action *is the following:*

- $\Longrightarrow$ *is the transitive and reflexive closure of* $\xrightarrow{\tau}$.
- $\overset{c?\tilde{v}@k}{\Longrightarrow}$ *denotes* $\Longrightarrow\xrightarrow{c?\tilde{v}@k}\Longrightarrow$
- $\overset{c!\tilde{v}@K\triangleleft R}{\Longrightarrow}$ *denotes* $\Longrightarrow\xrightarrow{c!\tilde{v}@K\triangleleft R}\Longrightarrow$.

We denote by $\overset{\hat{\alpha}}{\Longrightarrow}$ the weak action $\overset{\alpha}{\Longrightarrow}$ if $\alpha \neq \tau$, and $\Longrightarrow$ otherwise.

In the following we give the definition of labelled bisimilarity. Two conditions are necessary in our definition: the first for output and silent actions and the second for the input ones, since input in our model is not an observable action (if someone receives a message then surely a node must have sent it, while, after a message transmission it is not sure that someone will be able to receive it), hence two systems are considered equivalent even if they do not have the same behavior in terms of transmissions receptions.

Definition 3.7 (Bisimilarity) *A binary relation $\mathcal{R}$ over networks is a* simulation *if $M\mathcal{R}N$ implies:*

- *If $M \xrightarrow{\alpha} M'$, $\alpha \neq c?\tilde{v}@l$, then there exists N' such that $N \overset{\hat{\alpha}}{\Longrightarrow} N'$ and $M'\mathcal{R}N'$*
- *If $M \xrightarrow{c?\tilde{v}@l} M'$ then there exists N' such that:*
  - *either $N \overset{c?\tilde{v}@l}{\Longrightarrow} N'$ and $M'\mathcal{R}N'$*
  - *or $N \Rightarrow N'$ and $M'\mathcal{R}N'$.*

We say that N simulates *M if there is some simulation $\mathcal{R}$ such that $M\mathcal{R}N$. A relation $\mathcal{R}$ is a* bisimulation *if both $\mathcal{R}$ and its converse are simulations. We say that M and N are bisimilar, written $M \approx N$, if there exists some bisimulation $\mathcal{R}$ such that $M\mathcal{R}N$.*

It is easy now to prove that bisimulation is an equivalence relation, because reflexivity and symmetry are trivial, and transitivity follows from definition of $\stackrel{\hat{\alpha}}{\Longrightarrow}$. There are other important properties of bisimulation; here we will prove closure under contexts.

Lemma 3.3 ($\approx$ is contextual) *Let M and N be two networks such that $M \approx N$, then:*

1. *$M|O \approx N|O$, for all networks O;*
2. *$(\nu c)M \approx (\nu c)N$, for all channels c.*

Proof.

As regards the first item we have to prove that the relation

$$\mathcal{R} = \{(M|O, N|O) \mid \forall O, M \approx N\}$$

is a bisimulation. To prove it we do a case analysis on the transition $M|O \stackrel{\alpha}{\rightarrow} \hat{M}$. The interesting cases are when the transition is due to an interaction between M and O, and this happens by an application of rule (Bcast). Let $M|O \xrightarrow{c!\tilde{v}@K\triangleleft R} \hat{M}$ because $M|O \xrightarrow{c_L!\tilde{v}[l,r]} \hat{M}$ for some r, l, with $d(l,k) \leq r$, $\forall k \in R$ and $K = L \cap R$, due to an application of (Bcast). There are then two possibilities:

1. $M|O \xrightarrow{c_L!\tilde{v}[l,r]} \hat{M}$ because $M \xrightarrow{c_L!\tilde{v}[l,r]} M'$ and $O \xrightarrow{c?\tilde{v}@l'} O'$ with $d(l,l') \leq r$ and $\hat{M} = M'|O'$
2. $M|O \xrightarrow{c_L!\tilde{v}[l,r]} \hat{M}$ because $M \xrightarrow{c?\tilde{v}@l'} M'$ and $O \xrightarrow{c_L!\tilde{v}[l,r]} O'$, with $d(l,l') \leq r$ and $\hat{M} = M'|O'$

Case 1.

By applying rule (Obs) we have $M \xrightarrow{c!\tilde{v}@K\triangleleft R} M'$, and since by induction hypothesis $M \approx N$, $N \stackrel{c!\tilde{v}@K\triangleleft R}{\Longrightarrow} N'$ and $N' \approx M'$.

By aplying rule (Bcast) backwardly we have

$N \Longrightarrow N'' \xrightarrow{c_{L''}!\tilde{v}[l'',r'']} N''' \Longrightarrow N'$

with $d(l'',k) \leq r''$ $\forall k \in R$ and $K \subseteq R \cap L''$. Since $l' \in R$, $d(l'',l') \leq r''$ and we can apply rule (Bcast):

$N'' \mid O \xrightarrow{c_{L''}!\tilde{v}[l'',r'']} N''' \mid O'$.

Hence, by application of the rule (Par) we get

$N \mid O \Longrightarrow N'' \mid O \xrightarrow{c_{L''}!\tilde{v}[l'',r'']} N''' \mid O' \Longrightarrow N' \mid O'$.

Finally, by applying rule (Obs) we can turn again the transition $N'' \mid O \xrightarrow{c_{L''}!\tilde{v}[l'',r'']} N''' \mid O$ into $N'' \mid O \xrightarrow{c!\tilde{v}@K\triangleleft R} N''' \mid O'$. This implies $N \mid O \stackrel{c!\tilde{v}@K\triangleleft R}{\Longrightarrow} N' \mid O'$, with $(M' \mid O', N' \mid O') \in \mathcal{R}$ as required.

Case 2.

$M|O \xrightarrow{c_L!\tilde{v}[l,r]} \hat{M}$ because $M \xrightarrow{c?\tilde{v}@l'} M'$ and $O \xrightarrow{c_L!\tilde{v}[l,r]} O'$, with $d(l,l') \leq r$ and $\hat{M} = M'|O'$. As $M \approx N$ then there exists N' such that:

- $N \overset{c?\tilde{v}@l'}{\Longrightarrow} N'$, with $M' \approx N'$; in this case

$$N|O \Rightarrow \xrightarrow{c_L!\tilde{v}[l,r]} \Rightarrow N'|O'$$

and, by an application of rule (Obs), also $N|O \overset{c!\tilde{v}@K\triangleleft R}{\Longrightarrow} N'|O'$, with $(M'|O', N'|O') \in \mathcal{R}$, as required.

- or $N \Longrightarrow N'$, with $M' \approx N'$; in this case by applying rule (Par) and (Obs) to $O \xrightarrow{c_L!\tilde{v}[l,r]} O'$ we obtain

$$N|O \xrightarrow{c!\tilde{v}@K\triangleleft R} N|O'$$

and, by applying rule (Par) to $N \Longrightarrow N'$ we get $N \mid O \xrightarrow{c!\tilde{v}@K\triangleleft R} N \mid O' \Longrightarrow N' \mid O'$, meaning $N|O \overset{c!\tilde{v}@K\triangleleft R}{\Longrightarrow} N'|O'$, with $(M'|O', N'|O') \in \mathcal{R}$, as required.

Let $M \mid O \xrightarrow{\tau} \hat{M}$ because $M \mid O \xrightarrow{c_L!\tilde{v}[l,r]} M' \mid O$. The proof is analogous to the previous case.

The cases where there is no interaction between M and O are easy to deal with. For example if we suppose $M \mid O \xrightarrow{\gamma} \hat{M}$ due to an application of the rule (Par) we have two cases to consider:

1. $M|O \xrightarrow{\gamma} \hat{M}$ because $M \xrightarrow{\gamma} M'$ and $\hat{M} = M' \mid O$. Since $N \approx M$ we know that
   - if $\gamma \neq c?\tilde{v}@l$ for some channel c, some tuple $\tilde{v}$ of messages and some location$l$, $N \overset{\hat{\gamma}}{\Longrightarrow} N'$ and $M' \approx N'$. Hence, by applying rule (Par) we get $N \mid O \overset{\gamma}{\Longrightarrow} N' \mid O$ and $(M' \mid O, N' \mid O) \in \mathcal{R}$ as required.
   - if $\gamma = c?\tilde{v}@l$ for some channel c, some tuple $\tilde{v}$ of messages and some location l, $N \overset{\gamma}{\Longrightarrow} N'$ or $N \Longrightarrow N'$, and $M' \approx N'$. Again, by applying rule (Par) we get $N \mid O \overset{\gamma}{\Longrightarrow} N' \mid O$, or $N \mid O \Longrightarrow N' \mid O$, with $(M' \mid O, N' \mid O) \in \mathcal{R}$ as required.
2. $M|O \xrightarrow{\gamma} \hat{M}$ because $O \xrightarrow{\gamma} O'$ and $\hat{M} = M \mid O'$. Since $M \approx N$, $(M \mid O', N \mid O') \in \mathcal{R}$, as required.

In order to prove the second item of the lemma it suffices to show that the relation

$$\mathcal{S} \overset{\text{def}}{=} \{((\nu c)M, (\nu c)N) : M \approx N, \forall c\}$$

is a bisimulation. The proof follows by a case analysis on the transition $(\nu c)M \xrightarrow{\alpha} O$. The proof is straightforward as channels cannot be transmitted, i.e. there is no *scope extrusion.*

3.3.3 A complete characterisation

We can now demonstrate that our bisimulation is a valid proof method for *reduction barbed congruence*.

Theorem 3.2 (Soundness) *Let M and N be two arbitrary networks, such that $M \approx N$. Then $M \cong N$*

Proof.
We have to prove that the relation $\approx$ is:

1. reduction closed
2. barb preserving
3. contextual

1. *Reduction Closure.* If $M \approx N$ and $M \rightarrow M'$, by the Theorem 3.1, $\exists \hat{M} \equiv M'$ such that $M \xrightarrow{\tau} \hat{M}$, and, by Lemma 3.2, $M' \approx \hat{M}$. Since $M \approx N$, $\exists N'$ such that $N \Longrightarrow N'$ and $\hat{M} \approx N'$. Again, by the Theorem 3.1, $N \rightarrow^* N'$ and, by transitivity of the relation $\approx$, $M' \approx N'$.

2. *Barb preservation.* Suppose $M \downarrow_{c@K}$. By Theorem 3.1, $M \xrightarrow{c!\tilde{v}@K \triangleleft R}$ for some set $R \supseteq K$. Since $M \approx N$, it follows that $N \overset{c!\tilde{v}@K \triangleleft R}{\Longrightarrow}$ too and, by the definition of weak actions, $N \Longrightarrow \hat{N} \xrightarrow{c!\tilde{v}@K \triangleleft R}$. Again, by Theorem 3.1, we get $N \rightarrow^* \hat{N} \downarrow_{c@K}$, that means $N \Downarrow_{c@K}$, as required.

3. *Contextuality.* It follows straightforwardly from Lemma 3.3.

In order to prove the completeness result, we use the following standard property of reduction barbed congruence (see, e.g. , [62]).

Proposition 3.1 *If $M \cong N$ then*

- *$M \Downarrow_{c@K}$ iff $N \Downarrow_{c@K}$*
- *$M \Longrightarrow M'$ implies that there is N' such that $N \Longrightarrow N'$ and $M' \cong N'$*

Theorem 3.3 (Completeness) *Let M and N be two arbitrary networks, such that $M \cong N$. Then $M \approx N$*

Proof.
We prove that the relation $\mathcal{R} = \{(M, N) \mid M \cong N\}$ is a bisimulation. The result will follow by co-induction.

- Suppose that $M \mathcal{R} N$ and $M \xrightarrow{\tau} M'$. By Theorem 3.1, $M \rightarrow M'$. Then there exists N' such that $N \rightarrow^* N'$, hence $N \Longrightarrow N'$ and $M' \cong M'$, as required.

- Suppose $M\mathcal{R}N$ and $M \xrightarrow{c!\tilde{v}@K\triangleleft R} M'$, with $R = \{k_1, ..., k_n\}$ and $K \subseteq R$. As the action $c!\tilde{v}@K \triangleleft R$ can only be generated by an application of rule (Obs), it follows that $M \xrightarrow{c_L!\tilde{v}[l,r]} M'$ for some l, r and L such that $K = R \cap L$. Let us build a context which mimics the effect of the action $c!\tilde{v}@K \triangleleft R$ and also allows us to subsequently compare the residuals of the two systems under consideration. Our context has the form

  $\mathcal{C}[\cdot] \stackrel{\text{def}}{=} [\cdot] \mid \prod_{i=1}^{n}(m_i[c(\tilde{x}_i).[\tilde{x}_i = \tilde{v}]\bar{\mathtt{f}}^{(i)}_{k_i,r_i}\langle\tilde{x}_i\rangle]_{k_i} \mid n_i[\mathtt{f}^{(i)}(\tilde{y}_i).\bar{\mathtt{ok}}^{(i)}_{k_i,r_i}\langle\tilde{y}_i\rangle]_{k_i})$ with $r_{m_i}, r_{n_i} > 0$, $r_i \leq r_{n_i}$ and $r_i \leq r_{m_i}$, names m_i, n_i for $1 \leq i \leq n$ and channels names $\mathtt{f}^{(i)}$, $\mathtt{ok}^{(i)}$ for $1 \leq i \leq n$ fresh. Intuitively, the existence of the barbs on the fresh channels $\mathtt{f}^{(i)}$ indicates that the action has not yet happened, whereas the presence of the barbs on channels $\mathtt{ok}^{(i)}$, together with the absence of the barbs on channels $\mathtt{f}^{(i)}$ ensures that the action has been performed.

  As $\cong$ is preserved by network contexts, $M \cong N$ implies $\mathcal{C}[M] \cong \mathcal{C}[N]$. As $M \xrightarrow{c_L!\tilde{v}[l,r]} M'$ it follows that
  $\mathcal{C}[M] \longrightarrow^* M' \mid \prod_{i=1}^{n}(m_i[\mathbf{0}]_{k_i} \mid n_i[\bar{\mathtt{ok}}^{(i)}_{k_i,r_i}\langle\tilde{v}\rangle]_{k_i}) = \hat{M}$, with $\hat{M} \not\downarrow_{\mathtt{f}^{(i)}@k_i}$ and $\hat{M} \downarrow_{\mathtt{ok}^{(i)}@k_i}$, for $1 \leq i \leq n$.

  The reduction sequence must be matched by a corresponding reduction sequence $\mathcal{C}[N] \longrightarrow^* \hat{N}$ with $\hat{M} \cong \hat{N}$, $\hat{N} \not\Downarrow_{\mathtt{f}^{(i)}@k_i}$ and $\hat{N} \Downarrow_{\mathtt{ok}^{(i)}@k_i}$ for $1 \leq i \leq n$. The constrains on the barbs allow us to deduce the structure of the above reduction sequence

  $\mathcal{C}[N] \longrightarrow^* N' \mid \prod_{i=1}^{n}(m_i[\mathbf{0}]_{k_i} \mid n_i[\bar{\mathtt{ok}}^{(i)}_{k_i,r_i}\langle\tilde{v}\rangle]_{k_i}) = \hat{N}$.

  By barb preservation we know that, since $M \downarrow_{c@K}$, then $N \Downarrow_{c@K}$, while, by the behaviour of the context, we are sure that N can perform an output reachable by all the locations in R. This implies that $N \stackrel{c!\tilde{v}@K\triangleleft R}{\Longrightarrow} N'$. More precisely, the derivative N' might be reached performing several outputs of the message $\tilde{v}$ along the same channel c.

  As $\hat{M} \cong \hat{N}$, and as Reduction Barbed Congruence is preserved by restriction, we have $(\nu\tilde{\mathtt{f}}, \tilde{\mathtt{ok}})\hat{M} \cong (\nu\tilde{\mathtt{f}}, \tilde{\mathtt{ok}})\hat{N}$ (where $\tilde{\mathtt{f}} = \mathtt{f}^{(1)}, \mathtt{f}^{(2)}$ ... and $\tilde{\mathtt{ok}} = \mathtt{ok}^{(1)}, \mathtt{ok}^{(2)}$, ...). As $\mathtt{f}^{(i)}$ and $\mathtt{ok}^{(i)}$ for all $1 \leq i \leq n$ are fresh, by applying structural congruence we have
  $(\nu\tilde{\mathtt{f}}, \tilde{\mathtt{ok}})\hat{M} \equiv M' \mid (\nu\tilde{\mathtt{f}}, \tilde{\mathtt{ok}})(m_i[\mathbf{0}]_{k_i} \mid n_i[\bar{\mathtt{ok}}^{(i)}_{k_i,r_i}\langle\tilde{v}\rangle]_{k_i})$
  $(\nu\tilde{\mathtt{f}}, \tilde{\mathtt{ok}})\hat{N} \equiv N' \mid (\nu\tilde{\mathtt{f}}, \tilde{\mathtt{ok}})(m_i[\mathbf{0}]_{k_i} \mid n_i[\bar{\mathtt{ok}}^{(i)}_{k_i,r_i}\langle\tilde{v}\rangle]_{k_i})$.
  Using bisimilarity definition and soundness theorem we can easily prove that
  $(\nu\tilde{\mathtt{f}}, \tilde{\mathtt{ok}})(m_i[\mathbf{0}]_{k_i} \mid n_i[\bar{\mathtt{ok}}^{(i)}_{k_i,r_i}\langle\tilde{v}\rangle]_{k_i}) \cong \mathbf{0}$.
  As a consequence, it follows that $M' \cong N'$, as required.

- Suppose that $M\mathcal{R}N$ and $M \xrightarrow{c?\tilde{v}@l} M'$.
  The reception of a message cannot be directly observed. So we have to build a context which let the action be observable.

A context associated to the action $M \xrightarrow{c?\tilde{v}@l} M'$ could be:
$\mathcal{C}[\cdot] \stackrel{\text{def}}{=} [\cdot]|n[\bar{c}_{l,r}\langle\tilde{v}\rangle.\bar{\mathtt{f}}_{l,r}\langle\tilde{v}\rangle.\bar{\mathtt{ok}}_{l,r}\langle\tilde{v}\rangle]_k$,
with $\mathtt{f}$ and $\mathtt{ok}$ fresh channels, $r \leq r_n$ and $d(l,k) \leq r$. As $\cong$ is preserved by network contexts, $\mathcal{C}[M] \cong \mathcal{C}[N]$. As $M \xrightarrow{c?\tilde{v}@l} M'$ it follows that
$\mathcal{C}[M] \longrightarrow^* M'|n[\bar{\mathtt{ok}}_{l,r}\langle\tilde{v}\rangle]_k = \hat{M}$
with $\hat{M} \not\downarrow_{\mathtt{f}@l}$ and $\hat{M} \downarrow_{\mathtt{ok}@l}$. The reduction sequence must be matched by a corresponding reduction sequence $\mathcal{C}[N]$, so we have $\mathcal{C}[N] \longrightarrow^* \hat{N}$ and $\hat{M} \cong \hat{N}$, with $\hat{N} \not\Downarrow_{\mathtt{f}@l}$ and $\hat{N} \Downarrow_{\mathtt{ok}@l}$. The constrains on the barb allows us to deduce the structure of the above sequence: $\mathcal{C}[N] \longrightarrow^* N' \mid n[\bar{\mathtt{ok}}_{l,r}\langle\tilde{v}\rangle]_k \equiv \hat{N}$. This reduction does not ensure that N performs $c?\tilde{v}@l$ but there exists N' such that $N \stackrel{c?\tilde{v}@l}{\Longrightarrow} N'$, or $N \Longrightarrow N'$, in case N is not able to perform the input action on the channel c. As $\hat{M} \cong \hat{N}$, and $\cong$ is preserved by restriction, it follows that $(\nu\mathtt{ok})\hat{M} \cong (\nu\mathtt{ok})\hat{N}$, from which we can easily derive (by applying rule (Struct Res Par))
$(\nu\mathtt{ok})\hat{M} \equiv M'|(\nu\mathtt{ok})(n[\bar{\mathtt{ok}}_{l,r}\langle\tilde{v}\rangle]_{k,})$ and
$(\nu\mathtt{ok})\hat{N} \equiv N'|(\nu\mathtt{ok})(n[\bar{\mathtt{ok}}_{l,r}\langle\tilde{v}\rangle]_k)$.
As $(\nu\mathtt{ok})(n[\bar{\mathtt{ok}}_{l,r}\langle\tilde{v}\rangle]_k) \equiv \mathbf{0}$ we obtain $M' \cong N'$, as required.
We have proved that $\cong \subseteq \approx$.

3.4 Connectivity Properties

In this section we show how the EBUM calculus can be used to face the problem of conserving energy while maintaing a good connectivity among nodes.

As already introduced in Chapter 2, the problem of energy conservation is particularly challenging for wireless sensor networks, since sensors have usually limited energy resources, and, due to the critical conditions of the environment where they are installed, the substitution of the devices when battery is spent is not always possible. In the literature there are a lot of communication protocols trying to extend the lifetime of these kinds of networks with energy-aware strategies (see, e.g., [93, 90, 33]).

In the following we define some important properties of sensor networks (which can be verified for mobile ad hoc networks as well) which allow us to analyse the trade-off between connectivity and energy consumption, in order to find the best strategy to manage the power resources of a given network.

Simulation of static nodes

The tag L associated to each output action allows us to express a property of simulation for static devices at different locations. Indeed, two static nodes, placed at different locations (with therefore different neighbours), but communicating with the same set of intended recipients, result to be observational equivalent (see Figure

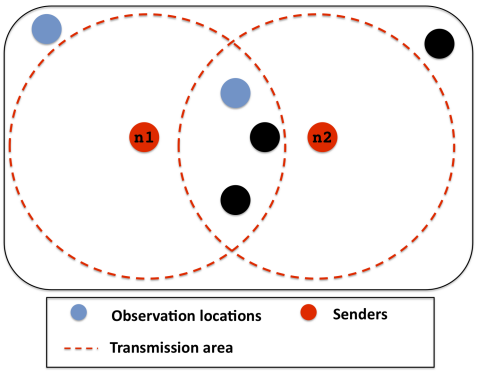


Figure 3.2: Example of simulation of stationary nodes

3.2).

Theorem 3.4 (Simulation of static nodes) *Let $n[P]_{l_n}$ and $m[P]_{l_m}$ be two static nodes with $\delta_n = \delta_m = 0$. Assume $r \leq r_n$ and $r \leq r_m$ for all r associated to the output actions of P, $R = \{k \mid d(l,k) \leq r_n\}$ and $R' = \{k \mid d(l',k) \leq r_m\}$. It holds that:*

1. *If $R' \subseteq R$, then $n[P]_l$ simulates $m[P]_{l'}$;*
2. *if $R = R'$, then $n[P]_l \approx m[P]_{l'}$.*

Proof.

1. We prove that the relation

$$\mathcal{S} = \{(m[P]_{l_m}, n[P]_{l_n}) \mid R' \subseteq R\}$$

is a simulation.

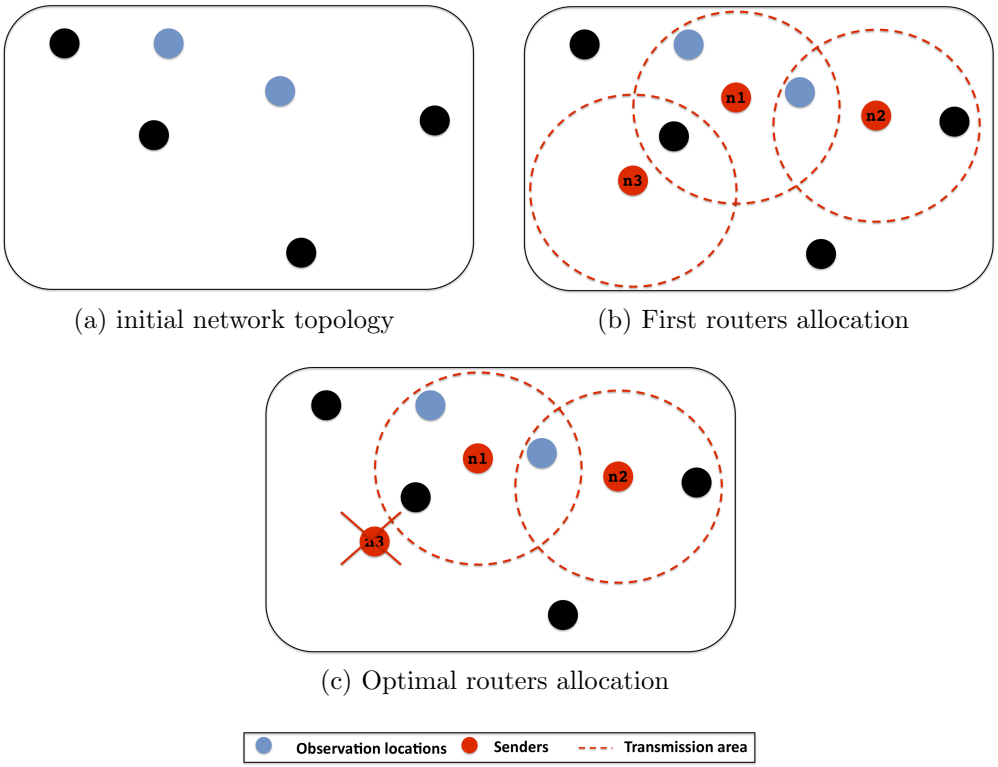
Suppose that $m[P]_{l_m} \xrightarrow{c!\tilde{v}@K \triangleleft \hat{R}} m[P']_{l_m}$ because $m[P]_{l_m} \xrightarrow{c_L!\tilde{v}[l_m,r]} m[P']_{l_m}$ with $\hat{R} \subseteq R'$ and $K = \hat{R} \cap L$. Hence $P \xrightarrow{\bar{c}_{L,r}\tilde{v}} P'$. Since, by hypothesis $r \leq r_n$, by rule (Snd), $n[P]_{l_n} \xrightarrow{c_L!\tilde{v}[l_n,r]} n[P']_{l_n}$. Since $R' \subseteq R$, we have that $\hat{R} \subseteq R$, and hence, by rule (Obs), $n[P]_{l_n} \xrightarrow{c!\tilde{v}@K \triangleleft \hat{R}} n[P']_{l_n}$, and $(m[P']_{l_m}, n[P']_{l_n}) \in \mathcal{S}$ as required.

The other cases are straightforward.

2. If $R = R'$ then $R \subseteq R'$ and $R' \subseteq R$; so, by applying the same reasoning used to prove the first item of this theorem, we can demonstrate that the relation

$$\mathcal{S} = \{(n[P]_{l_n}, m[P]_{l_m}) \mid R = R'\}$$

is a bisimulation.



(a) initial network topology

(b) First routers allocation

(c) Optimal routers allocation

Figure 3.3: Example of optimized routers allocation

This property is useful, e.g., to minimize the number of routers within a network while ensuring the correct communication between a given set of locations. Consider, for instance, the case in which we want to determine the lowest number of routers to be installed in a specific area. If we detect that two different routers result to exhibit the same behaviour then one of them can be turn off, thus allowing us to save both power and physical resources. Figure 3.3 shows an example of optimal routers allocation, by turning off the router `r3`, which is not necessary since it is simulated by `r1`.

This property is particularly useful when dealing with Wireless Sensor Networks, constituted of devices with limited power sources, and it can be used to optimize the scheduling strategies activating and deactivating sensor nodes.

Range repeaters

Range repeaters are static devices which regenerate a network signal in order to extend the range of the existing network infrastructure. Here we generalize the definition of repeater given in [58] and introduce a notion of *complete* range repeater. In the following we consider range repeaters with both one and two channels.

In the following we assume that for each process P executed by a network node,

it is possible to identify the set of all the observation locations that may appear in an output action performed by P. We denote by $\texttt{rcv}(P)$ the minimum set of locations ensuring that for each output action $\bar{c}_{L,r}\langle\tilde{v}\rangle$ performed by P it holds that $L \subseteq \texttt{rcv}(P)$. Indeed, the tag L associated to an output action occurring in P can be either a variable or a set of locations, then we are not able to statically calculate $\texttt{rcv}(P)$. However, since an ad hoc network is usually designed to guarantee the communication within a specific area, we can reasonably assume that the underling protocol will always multicast messages to recipients located within the interested area and we can abstractly represent them by a finite set of locations.

Definition 3.8 (Range repeater with two channels) *Let a and b be two channels, l be a location, r be a transmission radius and L be a set of locations. A repeater with two channels a and b relative to L with transmission radius r is a static device, denoted $rr[a \hookrightarrow_{L,r} b]_l$, where $a \hookrightarrow_{L,r} b$ is a process whose general recursive definition is:*

$$a \hookrightarrow_{L,r} b \stackrel{\text{def}}{=} a(\tilde{x}).\bar{b}_{L,r}\langle\tilde{x}\rangle.a \hookrightarrow_{L,r} b.$$

A range repeater with two channels receives tuples of values through the input channel a and retransmits them through the output channel b to the set of L of observation locations.

A range repeater with one channel operates analogously, but input and output channels coincide.

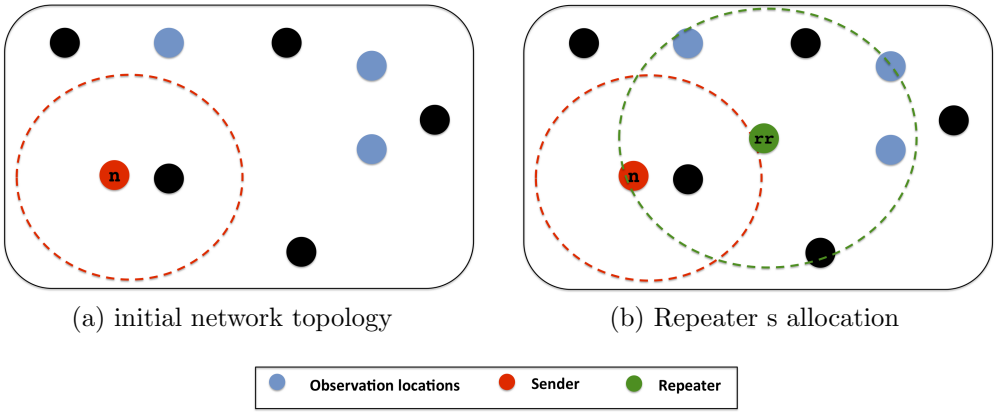
Definition 3.9 (Range repeater with one channel) *Let c be a channel, l be a location, r be a transmission radius and L be a set of locations. A range repeater with one channel c relative to L with transmission radius r is a static device, denoted $rr[c \hookrightarrow_{L,r} c]_{l,r}$ where*

$$c \hookrightarrow_{L,r} c \stackrel{\text{def}}{=} c(\tilde{x}).\bar{c}_{L,r}\langle\tilde{x}\rangle.c \hookrightarrow_{L,r} c.$$

Range repeaters are usually exploited to enlarge the transmission cell of a static node and, if such a node always communicates with the same set of devices, each time through the same channel, by using a range repeater we can simulate the presence of the sender in the location of the repeater.

Theorem 3.5 (Range repeaters with one channel) *Let $n[P]_l$ be a static node such that $\texttt{rcv}(P) = L$. Suppose that P uses exactly one channel c with a fixed transmission radius r (i.e., each output action will be of the form $c_{L',r}$ with $L' \subseteq L$) and $r \leq r_n$. Let $rr[c \hookrightarrow_{L,r} c]_k$ be a range repeater such that $d(l,k) \leq r$ and $r \leq r_{rr}$. Then:*

$$n[P]_l \mid rr[c \hookrightarrow_{L,r} c]_k \text{ simulates } n[P]_k.$$



(a) initial network topology (b) Repeater s allocation

Figure 3.4: Example of repeater allocation

Proof.

It is sufficient to prove that the relation

$\mathcal{S} = \{(n[P]_k, n[P]_l \mid rr[c \hookrightarrow_{L,r} c]_k) : d(l,k) \leq r, \mathtt{rcv}(P) \subseteq L$ and each output action of P is of the form $c_{L',r}\}$

is a simulation.

Suppose that $n[P]_k \xrightarrow{c!\tilde{v}@K \triangleleft R} n[P']_k$ because $n[P]_k \xrightarrow{c_{L'}!\tilde{v}[k,r]} n[P']_k$ with $R \subseteq \{k' : d(k,k') \leq r\}$ and $K = R \cap L'$ and $P \xrightarrow{\bar{c}_{L',r}\tilde{v}} P'$ for some $L' \subseteq L$. Hence, since $n[P]_l \xrightarrow{c_{L'}!\tilde{v}[l,r]} n[P']_l$ and $rr[c \hookrightarrow_{L,r} c]_k \xrightarrow{c?\tilde{v}@k} rr[\bar{c}_{L,r}\langle\tilde{v}\rangle.c \hookrightarrow_{L,r} c]_k$ with $d(l,k) \leq r$, by applying rule (Bcast) we obtain

$$n[P]_l \mid rr[c \hookrightarrow_{L,r} c]_k \xrightarrow{c_{L'}!\tilde{v}[l,r]} n[P']_l \mid rr[\bar{c}_{L,r}\langle\tilde{v}\rangle.c \hookrightarrow_{L,r} c]_k,$$

and, by applying the rule (Lose),

$$n[P]_l \mid rr[c \hookrightarrow_{L,r} c]_k \xrightarrow{\tau} n[P']_l \mid rr[\bar{c}_{L,r}\langle\tilde{v}\rangle.c \hookrightarrow_{L,r} c]_k$$

Since $rr[\bar{c}_{L,r}\langle\tilde{v}\rangle.c \hookrightarrow_{L,r} c]_k \xrightarrow{c_L!\tilde{v}[k,r]} rr[c \hookrightarrow_{L,r} c]_k$ we can deduce that

$rr[\bar{c}_{L,r}\langle\tilde{v}\rangle.c \hookrightarrow_{L,r} c]_k \xrightarrow{c!\tilde{v}@K' \triangleleft R'} rr[c \hookrightarrow_{L,r} c]_k$ for all $R' \subseteq \{k' : d(k,k') \leq r\}$, $K' = R' \cap L'$.

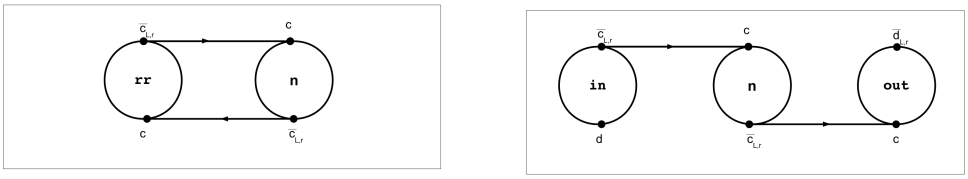
As $L' \subseteq L$ we can infer $rr[\bar{c}_{L,r}\langle\tilde{v}\rangle.c \hookrightarrow_{L,r} c]_k \xrightarrow{c!\tilde{v}@K \triangleleft R} rr[c \hookrightarrow_{L,r} c]_k$ and then

$$n[P]_l \mid rr[\bar{c}_{L,r}\langle\tilde{v}\rangle.c \hookrightarrow_{L,r} c]_k \overset{c!\tilde{v}@K \triangleleft R}{\Longrightarrow} n[P']_l \mid rr[c \hookrightarrow_{L,r} c]_k.$$

By hypothesis $\mathtt{rcv}(P') \subseteq L$ and each output action of P' is of the form $c_{L',r}$ with $L' \subseteq L$, hence $(n[P']_k, n[P']_l \mid rr[c \hookrightarrow_{L,r} c]_k) \in \mathcal{S}$.

Suppose now that $n[P]_k \xrightarrow{c?\tilde{v}@k} n[P']_k$ because $P \xrightarrow{c\tilde{v}} P'$. Hence

$$n[P]_l \mid rr[c \hookrightarrow_{L,r} c]_k \xrightarrow{c?\tilde{v}@k} n[P]_l \mid rr[\bar{c}_{L,r}\langle\tilde{v}\rangle.c \hookrightarrow_{L,r} c]_k.$$



(a) $rr[c \hookrightarrow_{L,r} c]_l$ (b) $\mathtt{in}[d \hookrightarrow_{L,r} c]_l | n[P]_k | \mathtt{out}[c \hookrightarrow_{L,r} d]_l$

Figure 3.5: Range repeaters: interactions between the nodes

Moreover, since $n[P]_l \xrightarrow{c?\tilde{v}@l} n[P']_l$ and $rr[\bar{c}_{L,r}\langle\tilde{v}\rangle.c \hookrightarrow_{L,r} c]_k \xrightarrow{c_L!\tilde{v}[k,r]} rr[c \hookrightarrow_{L,r} c]_k$ with $d(l,k) \leq r$, by applying rule (Bcast) and (Lose) we obtain

$$n[P]_l \mid rr[\bar{c}_{L,r}\langle\tilde{v}\rangle.c \hookrightarrow_{L,r} c]_k \xrightarrow{\tau} n[P']_l \mid rr[c \hookrightarrow_{L,r} c]_k$$

and then

$$n[P]_l \mid rr[c \hookrightarrow_{L,r} c]_k \stackrel{c?\tilde{v}@k}{\Longrightarrow} n[P']_l \mid rr[c \hookrightarrow_{L,r} c]_k,$$

with $(n[P']_k, n[P']_l \mid rr[c \hookrightarrow_{L,r} c]_k) \in \mathcal{S}$, as required.

Finally, the case $n[P]_k \xrightarrow{\tau} n[P']_k$ is trivial.

The simulation just described can be realised also with a range repeater with two channels. Using two channels, however, we need to adopt two range repeaters, respectively one for the input ($\mathtt{in}[d \hookrightarrow_{L,r} c]_l$) and one for the output ($\mathtt{out}[c \hookrightarrow_{L,r} d]_l$) management. The diagrams in Figure 3.5 illustrate the use of the channels and the interaction between the nodes when range repeaters with one or two channels are adopted. This picture is inspired by the diagrams in [60], describing the behaviour of agents and the use of channels for data input and output. We emphasise that this kind of diagrams gives no information about the physical position of the nodes or about the network topology, but they only show the connections through which devices can exchange data.

Theorem 3.6 (Range repeaters with two channels) *Let $n[P]_l$ be a stationary node such that $\mathtt{rcv}(P) = L$. Suppose that P uses exactly one channel c with a fixed transmission radius r (i.e., each output action will be of the form $c_{L',r}$ with $L' \subseteq L$) and $r \leq r_n$. Let $\mathtt{out}[c \hookrightarrow_{L,r} d]_k$ and $\mathtt{in}[d \hookrightarrow_{L,r} c]_k$ be two range repeaters such that $d(l,k) \leq r$ and $r \leq r_{rr}$. Then:*

$$n[P]_l | \mathtt{out}[c \hookrightarrow_{L,r} d]_k | \mathtt{in}[d \hookrightarrow_{L,r} c]_k \text{ simulates } n[P\{d/c\}]_k$$

Proof.

It is sufficient to prove that the following relation

$\mathcal{S} = \{(n[P\{d/c\}]_k, n[P]_l \,|\, \mathtt{out}[c \hookrightarrow_{L,r} d]_k \,|\, \mathtt{in}[d \hookrightarrow_{L,r} c]_k) :$
$d(l,k) \leq r,\ \mathtt{rcv}(P) \subseteq L$ and

each output action of P is of the form $c_{L',r}\}$

is a simulation.

Let $n[P\{d/c\}]_k \xrightarrow{d!\tilde{v}@K\triangleleft R} n[P'\{d/c\}]_k$ because $n[P]_k \xrightarrow{c_{L'}!\tilde{v}[k,r]} n[P']_k$ with $R \subseteq \{k' : d(k',k) \leq r\}$ and $K = R \cap L'$ with $P \xrightarrow{\bar{c}_{L',r}\tilde{v}} P'$ for some $L' \subseteq L$. Hence, from $n[P]_l \xrightarrow{c_{L'}!\tilde{v}[l,r]} n[P']_l$ and $\mathtt{out}[c \hookrightarrow_{L,r} d]_k \xrightarrow{c?\tilde{v}@k} \mathtt{out}[\bar{d}_{L,r}\langle\tilde{v}\rangle.c \hookrightarrow_{L,r} d]_k$ with $d(l,k) \leq r$ by applying rule (Bcast) we obtain

$$n[P]_l \mid \mathtt{out}[c \hookrightarrow_{L,r} d]_k \mid \mathtt{in}[d \hookrightarrow_{L,r} c]_k \xrightarrow{c_{L'}!\tilde{v}[l,r]} n[P']_l \mid \mathtt{out}[\bar{d}_{L,r}\langle\tilde{v}\rangle.c \hookrightarrow_{L,r} d]_k \mid \mathtt{in}[d \hookrightarrow_{L,r} c]_k.$$

By rule (Lose) we have

$$n[P]_l \mid \mathtt{out}[c \hookrightarrow_{L,r} d]_k \mid \mathtt{in}[d \hookrightarrow_{L,r} c]_k \xrightarrow{\tau} n[P']_l \mid \mathtt{out}[\bar{d}_{L,r}\langle\tilde{v}\rangle.c \hookrightarrow_{L,r} d]_k \mid \mathtt{in}[d \hookrightarrow_{L,r} c]_k.$$

By rule (Bcast) and rule (Obs) we get

$$n[P']_l \mid \mathtt{out}[\bar{d}_{L,r}\langle\tilde{v}\rangle.c \hookrightarrow_{L,r} d]_k \mid \mathtt{in}[d \hookrightarrow_{L,r} c]_k \xrightarrow{d!\tilde{v}@K'\triangleleft R'} n[P']_l \mid \mathtt{out}[c \hookrightarrow_{L,r} d]_k \mid \mathtt{in}[d \hookrightarrow_{L,r} c]_k$$

for all $R' \subseteq \{k' : d(k,k') \leq r\}$, $K' = R' \cap L$. Since $L' \subseteq L$ we can infer

$$n[P']_l \mid \mathtt{out}[\bar{d}_{L,r}\langle\tilde{v}\rangle.c \hookrightarrow_{L,r} d]_k \mid \mathtt{in}[d \hookrightarrow_{L,r} c]_k \xrightarrow{d!\tilde{v}@K\triangleleft R} n[P']_l \mid \mathtt{out}[c \hookrightarrow_{L,r} d]_k \mid \mathtt{in}[d \hookrightarrow_{L,r} c]_k$$

and then

$$n[P]_l \mid \mathtt{out}[c \hookrightarrow_{L,r} d]_k \mid \mathtt{in}[d \hookrightarrow_{L,r} c]_k \overset{d!\tilde{v}@K\triangleleft R}{\Longrightarrow} n[P']_l \mid \mathtt{out}[c \hookrightarrow_{L,r} d]_k \mid \mathtt{in}[d \hookrightarrow_{L,r} c]_k.$$

Since $\mathtt{rcv}(P') \subseteq L$ and each output action of P' is of the form $c_{L',r}$ with $L' \subseteq L$ we get $(n[P'\{d/c\}]_k, n[P']_l \mid \mathtt{out}[c \hookrightarrow_{L,r} d]_k \mid \mathtt{in}[d \hookrightarrow_{L,r} c]_k) \in \mathcal{S}$.

The other cases are similar to corresponding cases of Theorem 3.5.

We introduce now the notion of *complete range repeater*, that is a repeater which has a radius large enough to reach all the observation locations.

Definition 3.10 (Complete range repeater) *A range repeater* $rc[c \hookrightarrow_{L,r} c]_l$ *is said* complete *with respect to* L *if* $L \subseteq K$ *where* $K = \{k : d(l,k) \leq r\}$.

Consider the example depicted in Figure 3.4: a sender (the red node in figure) tries to communicate a message to a set of receivers (the light blue nodes), but its radius does not allow it to reach all the observation locations at the same time. The allocation of a repeater (the green node) covering the whole observation area is a good strategy to guarantee the connectivity of the given network.

3.5 Conclusions

In this chapter we introduced the EBUM calculus, which allows us to study the behaviour of mobile ad hoc and sensor networks. In particular we defined an equivalence relation, that is a congruence, allowing us to compare different networks having the same observational behaviour. Finally we addressed the problem of energy conservation in wireless sensor networks and we exploited the EBUM calculus to define some important connectivity properties that can be used to analyse the communication strategies with respect to the energy consumption.

One of the main peculiarities of EBUM calculus is the possibility to represent arbitrary movements of mobile nodes within the network area. Moreover, given a network, the mobility of its nodes is not always completely causal, but it may be influenced by several causes, as the obstacles inside the network area or the speed and the physical characteristics of the devices. In chapter 4 we will extend the EBUM calculus by adding probabilities to obtain a more realistic and precise representation of node mobility.

4

Connectivity and Energy-Aware Preorders for Mobile Ad hoc Networks

4.1 Introduction

In this chapter we introduce a probabilistic extension of the EBUM calculus [22, 21, 23], where probability distributions are used to describe the mobility of nodes. Like its predecessor [27], the calculus is built around nodes, representing the devices of the systems, and locations, identifying the position cells across which each device may move inside the network.

The probabilistic EBUM calculus deals with both nondeterministic and probabilistic choices. The semantics is inspired by Segala's probabilistic automata [83, 82] driven by schedulers to resolve the non-deterministic choice among the probability distributions over target states. We define a probabilistic observational congruence in the style of [62] to equate networks exhibiting the same probabilistic connectivity behaviour.

We also introduce energy-aware preorders over networks to measure the relative energy cost of different, but behaviourally equivalent, networks. As a result, the preorder may be employed to justify the replacement of components to lower the overall energy cost of a network while preserving its connectivity properties.

4.2 The Calculus

In this section we introduce the probabilistic extension of the EBUM calculus.

The syntax of our calculus is the same introduced in Chapter 3 (see Table 3.1). Both an Observational and a Labelled Transition Semantics are defined, based on the introduction of probability distributions associated with nodes movements.

While the syntax for a node n is the same as for the non-deterministic calculus, here we associate n with a pair $< r_n, \mathbf{J}^n >$, where r_n is again the maximum transmission radius for n, while $\mathbf{J}^n$ is the transition matrix of a discrete time Markov

chain: each entry $\mathbf{J}^n_{lk}$ is the probability that the node n located at l moves to location k. Hence, $\sum_{k\in\mathbf{Loc}} \mathbf{J}^n_{lk} = 1$ for all locations $l \in \mathbf{Loc}$. Static nodes inside a network are associated with the identity Markov chain, i.e., the identity matrix $\mathbf{J}^n_{ll} = 1$ for all $l \in \mathbf{Loc}$ and $\mathbf{J}^n_{lk} = 0$ for all $l \neq k$. We denote by μ^n_l the probability distribution associated with node the n located at l, i.e. , the function over **Loc** such that $\mu^n_l(k) = \mathbf{J}^n_{lk}$, for all $k \in \mathbf{Loc}$. We will model the probabilistic evolution of the network according to these distributions.

4.2.1 Probability distributions for networks

Let n be a node of a network M and l its location. We denote by $M\{n : l'/l\}$ the network obtained by replacing l with l' inside the node n and by $[\![M]\!]_{\mu^n_l}$ the probability distribution over the set of networks induced by μ^n_l and defined as follows: $\forall M' \in \mathcal{N}$,

$$[\![M]\!]_{\mu^n_l}(M') \;=\; \begin{cases} \mu^n_l(l') & \text{if } M' = M\{n : l'/l\} \\ 0 & \text{otherwise} \end{cases}$$

Intuitively, $[\![M]\!]_{\mu^n_l}(M')$ is the probability that the network M evolves to M' due to the movement of its node n located at l. We say that M' is in the support of $[\![M]\!]_{\mu^n_l}$ ($M' \in spt([\![M]\!]_{\mu^n_l})$) if $[\![M]\!]_{\mu^n_l}(M') \neq 0$. We write $[\![M]\!]_\Delta$ for the Dirac distribution on network M, namely the probability distribution defined as: $[\![M]\!]_\Delta(M) = 1$ and $[\![M]\!]_\Delta(M') = 0$ for all M' such that $M' \neq M$. Finally, we let θ range over $\{\mu^n_l \mid n \text{ is a node and } l \in \mathbf{Loc}\} \cup \{\Delta\}$.

Example 4.1 *(Probability distributions) Consider a network*

$$M = n_1[\bar{c}_{L,r_1}\langle\tilde{v_1}\rangle.P_1]_{l_1} \mid n_2[\bar{c}_{L,r_2}\langle\tilde{v_2}\rangle.P_2]_{l_2} \mid m[c(\tilde{x}).P_3]_k$$

consisting of two mobile sender nodes, n_1 and n_2, communicating with a static receiver node m. Node n_1 moves back and forth between locations l_1 and l_2 according to the probability distribution defined by the discrete time Markov chain with the following transition matrix

$$\mathbf{J} = \begin{vmatrix} 1-p & p \\ q & 1-q \end{vmatrix},$$

where $0 < p, q < 1$. Similarly, n_2 moves between l_2 and l_1 according to the same transition matrix $\mathbf{J}$. Then the probabilistic mobility of the network induced by the movement of the node n_1 is

$$[\![M]\!]_{\mu^{n_1}_{l_1}}(M') \;=\; \begin{cases} 1-p & \textit{if } M' = M\{n_1 : l_1/l_1\} = M \\ p & \textit{if } M' = M\{n_1 : l_2/l_1\} \\ 0 & \textit{otherwise.} \end{cases}$$

$$\text{(R-Bcast)}\ \frac{}{n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid \prod_{i\in I} n_i[c(\tilde{x}_i).P_i]_{l_i} \rightarrow [\![n[P]_l \mid \prod_{i\in I} n_i[P_i\{\tilde{v}/\tilde{x}_i\}]_{l_i}]\!]_\Delta}$$

where $0 < r \leq r_n$, $\forall i \in I.d(l, l_i) \leq r$, $r_i > 0$ and $|\tilde{x}_i| = |\tilde{v}|$

$$\text{(R-Move)}\ \frac{}{n[P]_l \rightarrow [\![n[P]_l]\!]_{\mu_l^n}} \qquad \text{(R-Par)}\ \frac{M \rightarrow [\![M']\!]_\theta}{M|N \rightarrow [\![M'|N]\!]_\theta}$$

$$\text{(R-Res)}\ \frac{M \rightarrow [\![M']\!]_\theta}{(\nu\tilde{c})M \rightarrow [\![(\nu\tilde{c})M']\!]_\theta} \qquad \text{(R-Struct)}\ \frac{N \equiv M \ \ M \rightarrow [\![M']\!]_\theta \ \ M' \equiv N'}{N \rightarrow [\![N']\!]_\theta}$$

Table 4.1: Reduction Semantics

Similarly for the second node we have

$$[\![M]\!]_{\mu_{l_2}^{n_2}}(M') \ = \ \begin{cases} 1-q & \text{if } M' = M\{n_2 : l_2/l_2\} = M \\ q & \text{if } M' = M\{n_2 : l_1/l_2\} \\ 0 & \text{otherwise.} \end{cases}$$

while for the static receiver we have

$$[\![M]\!]_{\mu_k^m}(M') \ = \ \begin{cases} 1 & \text{if } M' = M\{m : k/k\} = M \\ 0 & \text{otherwise.} \end{cases}$$

Note that for the static node movement, we have $[\![M]\!]_{\mu_k^m} = [\![M]\!]_\Delta$.

4.2.2 Reduction Semantics

The dynamics of the calculus is specified by the *probabilistic reduction relation* over networks ($\rightarrow$), described in Table 4.1. Again, it relies on the structural congruence relation, defined in Table 3.2. The rules are almost the same as in the previous calculus, but reductions lead to probability distributions.

The probabilistic reduction relation takes the form $M \rightarrow [\![M']\!]_\theta$, which describes a transition that leaves from network M and leads to a probability distribution $[\![M']\!]_\theta$.

Rule (R-Move) is the only rule with a different behaviour with respect to the non-deterministic calculus. A node n located at l and executing a move action will reach a location with a probability described by the distribution μ_l^n that depends on the Markov chain $\mathbf{J}^n$ statically associated with n. Again movements are atomic actions.

Since we are dealing with a probabilistic reduction semantics, which reduces networks into probability distributions, we need a way of representing the steps of each probabilistic evolution of a network. Formally, given a network M, we write

$$M \rightarrow_\theta N$$

if $M \rightarrow [\![M']\!]_\theta$, and N is in the support of $[\![M']\!]_\theta$. Following [31], an execution for M is a (possibly infinite) sequence of steps $M \rightarrow_{\theta_1} M_1 \rightarrow_{\theta_2} M_2 ....$

In the rest of the paper, we write $Exec_M$ for the set of all possible executions starting from M, $last(e)$ for the final state of a *finite* execution e, e^j for the prefix $M \rightarrow_{\theta_1} M_1 ... \rightarrow_{\theta_j} M_j$ of length j of the execution e of the form

$$M \rightarrow_{\theta_1} M_1 \cdots \rightarrow_{\theta_j} M_j \rightarrow_{\theta_{j+1}} M_{j+1} \cdots ,$$

and $e\uparrow$ for the set of e' such that $e \leq_{prefix} e'$.

4.2.3 Observational Semantics

As for the EBUM calculus, we formalize the observational semantics in terms of a notion *barb*, that provides the basic unit of observation [62]. As in other calculi for wireless communication, the definition of barb is naturally expressed in terms of message transmission. However, the technical development is more involved, as our calculus presents both non-deterministic and probabilistic aspects, where the non-deterministic choices are among the possible probability distributions that a network may follow and arise from the possibility for the node to perform arbitrary movements. Notice that the fact that a node performs a movement is arbitrary and unpredictable, while the resulting location is surely predictable (it depends on the transition matrix).

We denote by $behave(M) = \{[\![M']\!]_\theta \mid M \rightarrow [\![M']\!]_\theta\}$ the set of the possible behaviours of M. In order to solve the non-determinism in a network execution, we consider each possible probabilistic transition $M \rightarrow [\![M']\!]_\theta$ as arising from a *scheduler* (also called adversary or policy [83]).

Definition 4.1 (Scheduler) *A* scheduler *is a total function F assigning to a finite execution e a distribution $[\![N]\!]_\theta \in behave(last(e))$.*

We denote by *Sched* the set of all schedulers.

Given a network M and a scheduler F, we define the set of executions starting from M and driven by F as:

$$Exec_M^F = \{e = M \rightarrow_{\theta_1} M_1 \rightarrow_{\theta_2} M_2 ... \mid \forall j,\ M_{j-1} \rightarrow [\![M'_j]\!]_{\theta_j},\ [\![M'_j]\!]_{\theta_j} = F(e^{j-1}) \text{ and } M_j \text{ is in the support of } [\![M'_j]\!]_{\theta_j}\}.$$

Formally, given a finite execution $e = M \rightarrow_{\theta_1} M_1 ... \rightarrow_{\theta_k} M_k$ starting from a network M and driven by a scheduler F we define

$$P_M^F(e) = [\![M'_1]\!]_{\theta_1}(M_1) \cdot ... \cdot [\![M'_k]\!]_{\theta_k}(M_k)$$

where $\forall j \leq k$, $[\![M'_j]\!]_{\theta_j} = F(e^{j-1})$. We define the probability space on the executions starting from a given network M as follows. Given a scheduler F, $\sigma Field^F_M$ is the smallest sigma field on $Exec^F_M$ that contains the basic cylinders $e\uparrow$, where $e \in Exec^F_M$. The probability measure $Prob^F_M$ is the unique measure on $\sigma Field^F_M$ such that $Prob^F_M(e\uparrow) = P^F_M(e)$. Given a measurable set of networks H, we denote by $Exec^F_M(H)$ the set of executions starting from M and crossing a state in H. Formally $Exec^F_M(H) = \{e \in Exec^F_M \mid last(e^j) \in H \text{ for some } j\}$. We denote the probability for a network M to evolve into a network in H, according to the policy given by F, as $Prob^F_M(H) = Prob^F_M(Exec^F_M(H))$.

The notion of barb introduced below is the probabilistic extension of Definition 3.1 of Chapter 3 and denotes an observable transmission with a certain probability according to a fixed scheduler.

Definition 4.2 (Probabilistic Barb) *We say that a network M has a* probabilistic barb *with probability p on a channel c to the set K of locations, according to the scheduler F, written $M \Downarrow^F_p c@K$, if $Prob^F_M(\{N | N \downarrow_{c@K}\}) = p$.*

Intuitively, for a given network M and a scheduler F, if $M \Downarrow^F_p c@K$ then p is the positive probability that M, driven by F, performs a transmission on channel c and at least one of the receivers in the observation locations is able to correctly listen to it.

In the following, we introduce a probabilistic observational congruence, in the style of [31].

Schedulers constitute an essential feature for modeling communication protocols as they provide freedom in modelling implementation and incomplete knowledge of a system. However, many schedulers could be in fact unrealistic. Consider for example schedulers giving priority to communication actions over movements of the nodes. Such schedulers cancel the consequence that nodes mobility has on the network behaviour, since no movements can be performed during the process execution.

Therefore our aim is the definition of a relation allowing us to compare networks with respect to a given restricted set of schedulers.

In order to define a congruence relation among networks, we have to select a set of schedulers guaranteeing that, for each behaviour a network can exhibit, the same behaviour can be exhibited by the network in presence of any possible context. The following definition allows us to select the set of schedulers preserving the contextuality, once we have fixed the particular behaviour we want to capture.

Definition 4.3 *Given a scheduler $F \in Sched$, we denote by $F_{\mathcal{C}}$ the set of schedulers F' such that $\forall M_0$, $\forall e \in Exec^F_{M_0}$ of the form*

$e = M_0 \rightarrow_{\theta_1} M_1 \rightarrow_{\theta_2} M_2 ... \rightarrow_{\theta_h} M_h$,

$\forall$ context $C_0[\cdot]$ and $\forall e' \in Exec^{F'}_{C_0[O_0]}$ with $M_0 \equiv O_0$ of the form

$e' = C_0[O_0] \rightarrow_{\theta'_1} C_1[O_1] \rightarrow_{\theta'_2} C_2[O_2] ... \rightarrow_{\theta'_k} C_k[O_k]$,

there exists a monotonic surjective function f from $[0-k]$ to $[0-h]$ such that:

(i) $\forall i \in [0-k]$, $O_i \equiv M_{f(i)}$

(ii) $\forall j \in [1-k]$, $\theta'_j = \theta_{f(j)}$ *when* $M_{f(j-1)} \rightarrow_{\theta_{f(j)}} M_{f(j)}$.

Given a subset $\mathcal{F} \in Sched$ *of schedulers, then we have:*
$\mathcal{F}_{\mathcal{C}} = \bigcup_{F \in \mathcal{F}} F_{\mathcal{C}}$

Example 4.2 *Let* $M_0 \equiv m[\bar{c}_{L,r}\langle v \rangle.P]_l$ *and* $F \in Sched$ *such that*

$$M_0 \rightarrow_\Delta M_1 \in Exec_M^F,$$

with $M_1 \equiv m[P]_l$.

First notice that $F \in F_{\mathcal{C}}$, *since we can take the empty context* $C[\cdot] \equiv \emptyset \mid \cdot$ *and the identity function* f *such that* $f(i) = i$ *for all* $i \in [0-1]$. *In this case* $C[M_i] \equiv M_i$ *for* $i \in \{0,1\}$ *and the property of Definition 4.3 is satisfied.*

Let now consider $N_0 \equiv n[c(x).Q]_k$ *such that* $d(l,k) \leq r$. *All the admissible schedulers allowing* M_0 *and* N_0 *to interact are in* $F_{\mathcal{C}}$. *Indeed, consider* $F_1 \in Sched$ *such that, by applying rules (Struct-Bcast)*

$$M_0 \mid N_0 \rightarrow_\Delta M_1 \mid N_1 \in Exec^{F_1}_{M_0|N_0}$$

with $N_1 \equiv n[Q\{v/x\}]_k$, *and consider also* F_2 *such that, by applying rule (R-Par)*

$$M_0 \mid N_0 \rightarrow_\Delta M_1 \mid N_0 \in Exec^{F_2}_{M_0|N_0}.$$

Both F_1 *and* F_2 *satisfy the properties of Definition 4.3, hence* $F_1, F_2 \in F_{\mathcal{C}}$.

Now consider again the network N_0. *Let* $e' = n[c(x).Q]_k \rightarrow_{\mu_k^n} n[c(x).Q]_{k'} \notin Exec^F_{N_0}$, *then* $\forall \bar{F} \in Sched$ *such that* $e' \in Exec^{\bar{F}}_{N_0}$, $\bar{F} \notin F_{\mathcal{C}}$ *since* $\bar{F}$ *does not satisfy the conditions of Definition 4.3.*

Now we are able to introduce our equivalence relation.

Definition 4.4 *Given a set* $\mathcal{F} \in Sched$ *of scehdulers, and a relation* $\mathcal{R}$ *over networks:*

- Barb preservation. $\mathcal{R}$ *is* barb preserving *w.r.t.* $\mathcal{F}$ *if* $M\mathcal{R}N$ *and* $M \Downarrow_p^F c@K$ *for some* $F \in \mathcal{F}_{\mathcal{C}}$ *implies that there exists* $F' \in \mathcal{F}_{\mathcal{C}}$ *such that* $N \Downarrow_p^{F'} c@K$.
- Reduction closure. $\mathcal{R}$ *is* reduction closed *w.r.t.* $\mathcal{F}$ *if* $M\mathcal{R}N$ *implies that for all* $F \in \mathcal{F}_{\mathcal{C}}$, *there exists* $F' \in \mathcal{F}_{\mathcal{C}}$ *such that for all classes* $\mathcal{C} \in \mathcal{N}/\mathcal{R}$, $Prob_M^F(\mathcal{C}) = Prob_N^{F'}(\mathcal{C})$.
- Contextuality. $\mathcal{R}$ *is* contextual *if* $M\mathcal{R}N$ *implies that for every context* $\mathcal{C}[\cdot]$, *it holds that* $\mathcal{C}[M]\,\mathcal{R}\,\mathcal{C}[N]$.

$$\text{(Snd)}\frac{P \xrightarrow{\bar{c}_{L,r}\tilde{v}} P'}{n[P]_l \xrightarrow{c_L!\tilde{v}[l,r]} [\![n[P']_l]\!]_\Delta} \qquad \text{(Rcv)}\frac{P \xrightarrow{c\tilde{v}} P'}{n[P]_l \xrightarrow{c?\tilde{v}@l} [\![n[P']_l]\!]_\Delta}$$

$$\text{(Bcast)}\ \frac{M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta \quad N \xrightarrow{c?\tilde{v}@l'} [\![N']\!]_\Delta \quad d(l,l') \le r}{\begin{array}{c} M|N \xrightarrow{c_L!\tilde{v}[l,r]} [\![M'|N']\!]_\Delta \\ N|M \xrightarrow{c_L!\tilde{v}[l,r]} [\![N'|M']\!]_\Delta \end{array}}$$

$$\text{(Obs)}\ \frac{M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta \quad R \subseteq \{l' \in \mathbf{Loc} :\ d(l,l') \le r\} \quad K = R \cap L,\ K \neq \emptyset}{M \xrightarrow{c!\tilde{v}@K \lhd R} [\![M']\!]_\Delta}$$

$$\text{(Lose)}\ \frac{M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta}{M \xrightarrow{\tau} [\![M']\!]_\Delta} \qquad \text{(Move)}\ \frac{}{n[P]_l \xrightarrow{\tau} [\![n[P]_l]\!]_{\mu_l^n}}$$

$$\text{(Par)}\ \frac{M \xrightarrow{\gamma} [\![M']\!]_\theta}{\begin{array}{c} M|N \xrightarrow{\gamma} [\![M'|N]\!]_\theta \\ N|M \xrightarrow{\gamma} [\![N|M']\!]_\theta \end{array}} \qquad \text{(Res)}\ \frac{M \xrightarrow{\gamma} [\![M']\!]_\theta \quad \mathtt{Chan}(\gamma) \neq c}{(\nu c)M \xrightarrow{\gamma} [\![(\nu c)M']\!]_\theta}$$

Table 4.2: LTS rules for Networks

Our probabilistic observational congruence with respect to a restricted set $\mathcal{F}$ of schedulers is defined as the largest relation as follows.

Definition 4.5 (Probabilistic Observational Congruence with respect to $\mathcal{F}$) *Given a set $\mathcal{F}$ of schedulers,* Probabilistic observational congruence w.r.t. $\mathcal{F}$, *written $\cong_p^{\mathcal{F}}$, is the largest symmetric relation over networks which is reduction closed, barb preserving and contextual.*

Two networks are related by $\cong_p^{\mathcal{F}}$ if they exhibit the same probabilistic behaviour (communications) relative to the corresponding sets of intended recipients. In the next section we develop a bisimulation-based proof technique for $\cong_p^{\mathcal{F}}$. It provides an efficient method to check whether two networks are related by $\cong_p^{\mathcal{F}}$.

4.2.4 Labelled Transition Semantics

We define a LTS semantics for our calculus, which is built upon two sets of rules: one for processes and one for networks.

Rules for processes are the same defined in Chapter 3, shown in Table 3.4.

Table 4.2 presents the LTS rules for networks. Again Rules are the same defined in Chapter 4 but transitions are of the form $M \xrightarrow{\gamma} [\![M']\!]_\theta$, where M is a network and $[\![M']\!]_\theta$ is a distribution over networks.

Probabilities are used to model the mobility of nodes. Rule (Move) models migration of a mobile node n from a location l to a location k according with the probability distribution μ_l^n, which depends on the Markov chain $\mathbf{J}^n$ statically associated with n.

The other rules have the same behaviour as for Table 3.5.

We prove that the LTS-based semantics coincides with the reduction semantics and the notion of observability (barb) given in the previous section.

We first prove that if $M \xrightarrow{\gamma} [\![M']\!]_\Delta$, then the structure of M and M' can be determined up to structural congruence.

Lemma 4.1 *Let M be a network.*

1. *If $M \xrightarrow{c?\tilde{v}@l} [\![M']\!]_\Delta$, then there exist n, $\tilde{x}$, a (possibly empty) sequence $\tilde{d}$ such that $c \notin \tilde{d}$, a process P and a (possibly empty) network M_1 such that*

$$M \equiv (\nu \tilde{d})(n[c(\tilde{x}).P]_l | M_1)$$

and

$$M' \equiv (\nu \tilde{d})(n[P\{\tilde{v}/\tilde{x}\}]_l | M_1).$$

2. *If $M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta$, then there exist n, a (possibly empty) sequence $\tilde{d}$ such that $c \notin \tilde{d}$, a process P, a (possibly empty) network M_1 and a (possibly empty) set I, with t $d(l, l_i) \leq r$ $\forall i \in I$, such that:*

$$M \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v} \rangle.P]_l | \prod_{i \in I} n_i[c(\tilde{x}_i).P_i]_{l_i} \mid M_1)$$

and

$$M' \equiv (\nu \tilde{d})(n[P]_l | \prod_{i \in I} n_i[P_i\{\tilde{v}/\tilde{x}_i\}]_{l_i} \mid M_1).$$

We also show that structural congruence (Table 3.2) respects the transitions of Table 4.2.

Lemma 4.2 *If $M \xrightarrow{\gamma} [\![M']\!]_\theta$ and $M \equiv N$, then there exists N' such that $N \xrightarrow{\gamma} [\![N']\!]_\theta$ and $M' \equiv N'$.*

Lemmas 4.1 and 4.2 corresponds to Lemmas 3.1 and 3.2 of Chapter 3 and the proofs are similar.

The following theorem establishes the relationship between the reduction semantics and the LTS one.

Theorem 4.1 (Harmony) *Let M be a network.*

1. *If $M \to [\![M']\!]_\theta$ then there exists $N \equiv M$ and $N' \equiv M'$ such that $N \xrightarrow{\tau} [\![N']\!]_\theta$.*
2. *$M \downarrow_{c@K}$ if and only if there exist $\tilde{v}$, $R \supseteq K$ and $N \equiv M$ such that $N \xrightarrow{c!\tilde{v}@K \triangleleft R}$.*
3. *If $M \xrightarrow{\tau} [\![M']\!]_\theta$ then $M \to [\![M']\!]_\theta$.*
4. *If $M \xrightarrow{c!\tilde{v}@K \triangleleft R} [\![M']\!]_\Delta$ then $M \to [\![M']\!]_\Delta$.*

Proof.

1. The first part is proved by induction on the reduction $M \to [\![M']\!]_\theta$

   Suppose that $M \to [\![M']\!]_\theta$ is due to the application of the rule (R-Move). It means that $M \equiv M' \equiv n[P]_l$, for some name n, location l, some (possibly empty) process P, with $\theta = \mu_l^n$. We simply apply (Move) to obtain:

   $$\frac{}{n[P]_l \xrightarrow{\tau} [\![n[P]_l]\!]_{\mu_l^n}}.$$

   Suppose that $M \to [\![M']\!]_\theta$ is due to the application of the rule (R-Par), meaning $M \equiv M_1 \mid M_2$, $M' \equiv M_1' \mid M_2$ and:

   $$\frac{M_1 \to [\![M_1']\!]_\theta}{M_1 \mid M_2 \to [\![M_1' \mid M_2]\!]_\theta}.$$

   By induction hypothesis $\exists N \equiv M_1$ and $N' \equiv M_1'$ such that $N \xrightarrow{\tau} [\![N']\!]_\theta$, then by applying rule (Par) we get:

   $$\frac{N \xrightarrow{\tau} [\![N']\!]_\theta}{N \mid M_2 \xrightarrow{\tau} [\![N' \mid M_2]\!]_\theta},$$

   and, by applying (Struct Cxt Par) and (Struct Trans) $N \mid M_2 \equiv M_1 \mid M_2 \equiv M$ and $N' \mid M_2 \equiv M_1' \mid M_2 \equiv M'$ as required.

   Suppose that $M \to [\![M']\!]_\theta$ is due to the application of the rule (R-Res), meaning $M \equiv (\nu c)M_1$, and $M' \equiv (\nu c)M_1'$, for some channel c and some networks M_1 and M_1'; we get:

   $$\frac{M_1 \to [\![M_1']\!]_\theta}{(\nu c)M_1 \to [\![(\nu c)M_1']\!]_\theta}.$$

By induction hypothesis $\exists N \equiv M_1$ and $N' \equiv M_1'$ such that $N \xrightarrow{\tau} [\![N']\!]_\theta$, then by applying rule (Res), since $\mathtt{Chan}(\tau) \neq c$ we get:

$$\frac{N \xrightarrow{\tau} [\![N']\!]_\theta}{(\nu c)N \xrightarrow{\tau} [\![(\nu c)N']\!]_\theta},$$

and, by applying (Struct Cxt Res) and (Struct Trans) $(\nu c)N \equiv (\nu c)M_1 \equiv M$ and $(\nu c)N' \equiv (\nu c)M_1' \equiv M'$ as required.

Suppose that $M \to [\![M']\!]_\theta$ is due to the application of the rule (R-Bcast). It means:

$M \equiv n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid \prod_{i\in I} n_i[c(\tilde{x}_i).P_i]_{l_i}$,

$M' \equiv n[P]_l \mid \prod_{i\in I} n_i[P_i\{\tilde{v}/\tilde{x}_i\}]_{l_i}$ and

$$\frac{\forall i \in I.d(l,l_i) \leq r}{n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid \prod_{i\in I} n_i[c(\tilde{x}_i).P_i]_{l_i} \to [\![n[P]_l \mid \prod_{i\in I} n_i[P_i\{\tilde{v}/\tilde{x}_i\}]_{l_i}]\!]_\Delta},$$

for some name n, location l, radius r, some set L of locations, some tuple $\tilde{v}$ of messages, some (possibly empty) process P, some (possibly empty) set I of networks. By applying rule (Snd), (Rcv), $\mid I \mid$ times rule (Bcast) and, finally rule (Lose), we get

$$n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid \prod_{i\in I} n_i[c(\tilde{x}_i).P_i]_{l_i} \xrightarrow{\tau} [\![n[P]_l \mid \prod_{i\in I} n_i[P_i\{\tilde{v}/\tilde{x}_i\}]_{l_i}]\!]_\Delta,$$

as required.

Finally let suppose that the reduction $M \to [\![M']\!]_\theta$ is due to an application of rule (R-Struct):

$$\frac{M \equiv N \quad N \to [\![N']\!]_\theta \quad N' \equiv M'}{M \to [\![M']\!]_\theta}.$$

By induction hypothesis there exists $N_1 \equiv N$ and $N_2 \equiv N'$ such that $N_1 \xrightarrow{\tau} [\![N_2]\!]_\theta$. Then, by applying the rule for structural congruence (Struct Trans) we get $M \equiv N \equiv N_1$ and $M' \equiv N' \equiv N_2$, as required.

2. The second part of the theorem follows straightforwardly from Lemma 4.1 and the definition of Barb.

   $\Rightarrow$ If $M \downarrow_{c@K}$, by the definition of Barb:
   $M \equiv (\nu\tilde{d})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid M_1)$, for some n, $\tilde{v}$, L, r, some (possibly empty) sequence $\tilde{d}$ with $c \notin \tilde{d}$, some process P and some (possibly empty) network M_1, with $K \subseteq \{k \in L \text{ s.t. } d(l,k) \leq r\}$ and $K \neq \emptyset$.

By applying the rules (Snd), (Par) and (Res):

$$\frac{n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \xrightarrow{c_L!\tilde{v}[l,r]} [\![n[P]_l]\!]_\Delta}{(\nu\tilde{d})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid M_1) \xrightarrow{c_L!\tilde{v}[l,r]} [\![(\nu\tilde{d})(n[P]_l \mid M_1]\!]_\Delta)};$$

then we can apply rule (Obs):

$n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid M_1 \xrightarrow{c!\tilde{v}@K\triangleleft R} [\![n[P]_l \mid M_1]\!]_\Delta$,
where $R = \{l' \in \mathbf{Loc} : d(l,l') \leq r\}$, and $K \subseteq L \cap R$ as required.

$\Leftarrow$ If $M \xrightarrow{c!\tilde{v}@K\triangleleft R} [\![M']\!]_\Delta$, because $M \xrightarrow{c_L!\tilde{v}![l,r]} [\![M']\!]_\Delta$, by applying lemma 4.1 then there exists n, some (possibly empty) sequence $\tilde{d}$ such that $c \notin \tilde{d}$, some process P, some (possibly empty) network M_1 and a set I, such that $\forall i \in I\ d(l,l_i) \leq r$ such that:

$$M \equiv (\nu\tilde{d})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l | \prod_{i\in I} n_i[c(\tilde{x}_i).P_i]_{l_i} \mid M_1)$$

and

$$M' \equiv (\nu\tilde{d})(n[P]_l | \prod_{i\in I} n_i[P_i\{\tilde{v}/\tilde{x}_i\}]_{l_i} \mid M_1).$$

Since $K \neq \emptyset$, by applying the definition of barb we conclude $M \downarrow_{c@K}$.

3. The third part of the theorem is proved by induction on the derivation $M \xrightarrow{\tau} [\![M']\!]_\theta$.
Suppose that $M \xrightarrow{\tau} [\![M']\!]_\theta$ is due to an application of the rule (Move), that means:

$M \equiv n[P]_l$, $M' \equiv n[P]_l$, for some name n, some (possibly empty) process P. some location l with $\theta = \mu_l^n$ and

$$\frac{}{n[P]_l \xrightarrow{\tau} [\![n[P]_l]\!]_{\mu_l^n}},$$

hence , by applying (R-Move) we get:

$$\frac{}{n[P]_l \rightarrow [\![n[P]_l]\!]_{\mu_l^n}}.$$

If $M \xrightarrow{\tau} [\![M']\!]_\theta$ is due to an application of (Lose):

$$\frac{M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta}{M \xrightarrow{\tau} [\![M']\!]_\Delta},$$

for some channel c, some set L of locations, some tuple $\tilde{v}$ of messages, some location l and radius r. By applying Lemma 4.1, there exist n, $\tilde{v}$, a (possibly

empty) sequence $\tilde{d}$ such that $c \notin \tilde{d}$, a process P, a (possibly empty) network M_1 and a (possibly empty) set I with $d(l, l_i) \leq r \ \forall i \in I$, such that:

$$M \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v} \rangle.P]_l | \prod_{i \in I} n_i[c(\tilde{x}_i).P_i]_{l_i} \mid M_1)$$

and

$$M' \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v} \rangle.P]_l | \prod_{i \in i} n_i[P_i\{\tilde{v}/\tilde{x}_i\}]_{l_i} \mid M_1).$$

Finally, by applying rule (R-Bcast), (R-Res) and (R-Struct) we get $M \rightarrow [\![M']\!]_\theta$.

Suppose that $M \xrightarrow{\tau} [\![M']\!]_\theta$ is due to the application of (Res), that means $M \equiv (\nu c)M_1$ and $M' \equiv (\nu c)[\![M_1']\!]_\theta$, for some channel c and for some networks M_1 and M_1'. Then we have:

$$\frac{M_1 \xrightarrow{\tau} [\![M_1']\!]_\theta}{(\nu c)M_1 \xrightarrow{\tau} [\![(\nu c)M_1']\!]_\theta}.$$

By induction hypothesis $M_1 \rightarrow [\![M_1']\!]_\theta$, hence, by applying rule (R-Res) we get $(\nu c)M_1 \rightarrow [\![(\nu c)M_1']\!]_\theta$.

Finally, suppose that $M \xrightarrow{\tau} [\![M']\!]_\theta$ is due to the application of (Par), meaning $M \equiv M_1 \mid M_2$, $M' \equiv M_1' \mid M_2$ and

$$\frac{M_1 \xrightarrow{\tau} [\![M_1']\!]_\theta}{M_1|M_2 \xrightarrow{\tau} [\![M_1'|M_2]\!]_\theta}.$$

By induction hypothesis $M_1 \rightarrow [\![M_1']\!]_\theta$, hence, by applying rule (R-Par) we get $M_1|M_2 \rightarrow [\![M_1'|M_2]\!]_\theta$.

4. The last part of the theorem follows from the definition of barb and Lemma 4.1. Formally, since $M \xrightarrow{c!\tilde{v}@K \triangleleft R} [\![M']\!]_\Delta$ because $M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta$ for some location l, radius r and set L of intended recipients, by applying Lemma 4.1, exist n, a (possibly empty) sequence $\tilde{d}$ with $c \notin \tilde{d}$, a process P, a (possibly empty) network M_1 and a (possibly empty) set I such that:

$$M \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v} \rangle.P]_l \mid \prod_{i \in I} n_i[c(\tilde{x}_i).P_i]_{l_i} \mid M_1)$$

and

$$M' \equiv (\nu \tilde{d})(n[P]_l \mid \prod_{i \in I} n_i[P_i\{\tilde{v}/\tilde{x}_i\}]_{l_i} \mid M_1).$$

Then, by applying the rule (R-Bcast), (R-Par) and (R-Res) we get:

$$(\nu\tilde{d})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid \prod_{i\in I} n_i[c(\tilde{x}_i).P_i]_{l_i} \mid M_1)$$

$$\rightarrow$$

$$[\![(\nu\tilde{d})(n[P]_l \mid \prod_{i\in I} n_i[P_i\{\tilde{v}/\tilde{x}_i\}]_{l_i} \mid M_1)]\!]_\Delta,$$

and, by applying (R-Struct), we obtain $M \rightarrow [\![M']\!]_\Delta$, as required.

4.2.5 Probabilistic labelled bisimilarity

Based on the LTS semantics, we define a probabilistic labelled bisimilarity that is a complete characterisation of our *probabilistic observational congruence*. It is built upon the following actions:

$$\alpha ::= c?\tilde{v}@l \mid c!\tilde{v}@K \triangleleft R \mid \tau.$$

Again, we write $M \xrightarrow{\alpha}_\theta N$ if $M \xrightarrow{\alpha} [\![M']\!]_\theta$ and N is in the support of $[\![M']\!]_\theta$. A *labelled execution* e of a network M is a finite (or infinite) sequence of steps: $M \xrightarrow{\alpha_1}_{\theta_1} M_1 \xrightarrow{\alpha_2}_{\theta_2} M_2 ... \xrightarrow{\alpha_k}_{\theta_k} M_k$. With abuse of notation, we define $Exec_M$, $last(e)$, e^j and $e\uparrow$ as for unlabeled executions. We denote by $lbehave(M)$ the set of all possible behaviours of M, i.e., $lbehave(M) = \{(\alpha, [\![M']\!]_\theta) \mid M \xrightarrow{\alpha} [\![M']\!]_\theta\}$. Labelled executions arise by resolving the non-determinism of both α and $[\![M]\!]_\theta$. As a consequence, a scheduler[1] for the labelled semantics is a function F assigning to a finite labelled execution e a pair $(\alpha, [\![M]\!]_\theta) \in lbehave(last(e))$. We denote by $LSched$ the set of all schedulers for the LTS semantics. Given a network M and a scheduler F, we define $Exec^F_M$ as the set of all labelled executions starting from M and driven by F.

We denote by $Exec^F_M(\stackrel{\alpha}{\Longrightarrow}, H)$[2] the set of executions that, starting from M, according to the scheduler F, lead to a network in the set H by performing $\stackrel{\alpha}{\Longrightarrow}$. Moreover, we define $Prob^F_M(\stackrel{\alpha}{\Longrightarrow}, H) = Prob^F_M(Exec^F_M(\stackrel{\alpha}{\Longrightarrow}, H))$.

Since we want our bisimulation to be a complete characterisation of our notion of behavioural equivalence, which has been defined with respect to a restricted set of schedulers $\mathcal{F} \subseteq Sched$ on Reduction semantics, we have to define the corresponding set of schedulers for LTS.

Definition 4.6 *Given a scheduler $F \subseteq Sched$, we denote as $\hat{F}_\mathcal{C} \subseteq LSched$ as the set of schedulers $F \in \hat{\mathcal{F}}_\mathcal{C}$ such that:*

$\forall F \in \hat{F}_\mathcal{C}$, $\forall M_0 \in \mathcal{N}$ such that $e \in Exec^F_{M_0}$:

$$e = M_0 \xrightarrow{\alpha_1}_{\theta'_1} M_1 ... \xrightarrow{\alpha_k}_{\theta'_h} M_h$$

[1] With abuse of notation, we still use F to denote a scheduler for the LTS semantics.

[2] For the definition of $\stackrel{\alpha}{\Longrightarrow}$ see Definition 3.6 in Chapter 3

$\exists F' \in F_{\mathcal{C}}$, *a context* C_0 *and* $O_0 \in \mathcal{N}$, *such that* $e' \in Exec^{F'}_{C_0[O_0]}$:

$$e = C_0[O_0] \rightarrow_{\theta_1} C_1[O_1] ... \rightarrow_{\theta_k} C_k[O_k]$$

and there exists a monotone surjective function f *from* $[0-k]$ *to* $[0-h]$ *such that:*

(i) $\forall i \in [1-k]\ O_i \equiv M_{f(i)}$

(ii) $\forall j \in [1-k]$ *if* $M_{f(j-1)} \xrightarrow{\alpha_{f(j)}}_{\theta'_{f(j)}} M_{f(j)}$ *then* $\theta'_{f(j)} = \theta_j$.

Given a set $\mathcal{F} \subseteq Sched$ *of schedulers,* $\hat{\mathcal{F}}_{\mathcal{C}} = \bigcup_{F \in \mathcal{F}} \hat{F}_{\mathcal{C}}$.

Example 4.3 *Consider the networks* M_0 *and* N_0, *and the schedulers* F *and* F_1 *introduced in the Example 4.2. If we take* $\hat{F}_1 \in LSched$ *such that*

$$M_0 \xrightarrow{c_L!v[l,r]}_{\Delta} M_1 \in Exec^{\hat{F}_1}_{M_0},$$

then, since

$$M_0 \rightarrow_{\Delta} M_1 \in Exec^{F}_{M_0}$$

the conditions of Definition 4.6 are satisfied by taking the empty context $C[\cdot] = \mathbf{0} \mid \cdot$ *and the identity function* $f(i) = i$ *for* $i \in \{0,1\}$. *Hence* $\hat{F}_1 \in \hat{F}_{\mathcal{C}}$.

Moreover, if we consider $\hat{F}_2 \in LSched$ *such that*

$$N_0 \xrightarrow{c?v@k}_{\Delta} N_1 \in Exec^{\hat{F}_2}_{N_0},$$

since

$$M_0 \mid N_0 \rightarrow_{\Delta} M_1 \mid N_1 \in Exec^{F_1}_{M_0|N_0}$$

with $F_1 \in F_{\mathcal{C}}$, *by considering the contexts* $C_i[\cdot] \equiv M_i \mid \cdot$ *for* $i \in \{0,1\}$, *and the identity function* $f(i) = i$ *for* $i \in \{0,1\}$ *we get* $\hat{F}_2 \in \hat{F}_{\mathcal{C}}$ *too.*

Proposition 4.1

1. $Sched_{\mathcal{C}} = Sched$
2. $\widehat{Sched_{\mathcal{C}}} = LSched$

Proof.

1. Proof follows straightforwardly from the Definition 4.3.

2. $\forall F \in LSched$, $\forall M_0 \in \mathcal{N}$, where $e \in Exec^F_{M_0}$ is of the form:

   $e = M_0 \xrightarrow{\alpha_1}_{\theta_1} M_1 ... \xrightarrow{\alpha_k}_{\theta_k} M_k$

   we are always able to find a context $C_0[\cdot]$ and a scheduler $F' \in LSched$ such that $e' \in Exec^{F'}_{C_0[M_0]}$:

   $e' = C_0[M_0] \xrightarrow{\tau}_{\theta_1} ...C_1[M_1]... \xrightarrow{\tau}_{\theta_k} C_k[M_k]$

   and by theorem 4.1, there will exists $F'' \in Sched$ such that $e'' \in Exec^{F''}_{C_0[M_0]}$:

   $e'' = C_0[M_0] \rightarrow_{\theta_1} ...C_1[M_1]... \rightarrow C_k[M_k]$,

   meaning $F \in \widehat{Sched}_{\mathcal{C}}$ as required.

Following we will give the definition of probabilistic labelled bisimilarity with respect to a given set of schedulers.

Definition 4.7 (Probabilistic Labelled Bisimulation) *Let M and N be two networks. An equivalence relation $\mathcal{R}$ over networks is a* probabilistic labelled bisimulation *w.r.t. $\mathcal{F}$ if $M\mathcal{R}N$ implies: for all scheduler $F \in \hat{\mathcal{F}}_{\mathcal{C}}$ there exists a scheduler $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that for all α and for all classes $\mathcal{C}$ in $\mathcal{N}/\mathcal{R}$ it holds:*

1. *if $\alpha \neq c?\tilde{v}@l$ then $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = Prob^{F'}_N(\stackrel{\hat{\alpha}}{\Longrightarrow} \mathcal{C})$;*

2. *if $\alpha = c?\tilde{v}@l$ then either $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = Prob^{F'}_N(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C})$ or*
   $$Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = Prob^{F'}_N(\Longrightarrow, \mathcal{C}).$$

Probabilistic labelled bisimilarity*, written $\approx^{\mathcal{F}}_p$, is the largest probabilistic labelled bisimulation w.r.t. $\mathcal{F}$ over networks.*

Following we introduce two important propositions that will help us to prove that our probabilistic bisimulation is a complete characterization of probabilistic barbed congruence.

Proposition 4.2 *Let M and N be two networks. If $M\mathcal{R}N$ for some bisimulation $\mathcal{R}$ w.r.t $\mathcal{F}$, $\forall F \in \hat{\mathcal{F}}_{\mathcal{C}}$ $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that for all α and for all classes $\mathcal{C}$ in $\mathcal{N}/\mathcal{R}$ it holds:*

1. *if $\alpha \neq c?\tilde{v}@l$ then $Prob^F_M(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}) = Prob^{F'}_N(\stackrel{\hat{\alpha}}{\Longrightarrow} \mathcal{C})$;*

2. *if $\alpha = c?\tilde{v}@l$ then either $Prob^F_M(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}) = Prob^{F'}_N(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C})$ or*
   $$Prob^F_M(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}) = Prob^{F'}_N(\Longrightarrow, \mathcal{C}).$$

Proof.

We proceed by induction on the length of the weak transition $\stackrel{\hat{\alpha}}{\Longrightarrow}$.

If M reaches $\mathcal{C}$ in one step then, since $M\mathcal{R}N$, $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that: if $\alpha \neq c?\tilde{v}@l$, $Prob_M^F(\stackrel{\alpha}{\rightarrow}, \mathcal{C}) = Prob_N^{F'}(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C})$,

while, if $\alpha = c?\tilde{v}@l$ $Prob_M^F(\stackrel{\alpha}{\rightarrow}, \mathcal{C}) = Prob_N^{F'}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C})$, or

$Prob_M^F(\stackrel{\alpha}{\rightarrow}, \mathcal{C}) = Prob_N^{F'}(\Longrightarrow, \mathcal{C})$, as required.

If M reaches $\mathcal{C}$ in more steps then we consider two cases:

- The first transition is α, and $M \stackrel{\alpha}{\rightarrow} [\![M']\!]_\theta$.

  $Prob_M^F(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}) = \sum_{\hat{M} \in spt([\![M']\!]_\theta)}(Prob_M^F(\stackrel{\alpha}{\rightarrow}, \hat{M}) \times Prob_{\hat{M}}^F(\Longrightarrow, \mathcal{C}))$.

  Now, if we partition the support of $[\![M']\!]_\theta$ in equivalence classes of $\mathcal{R}$, $\exists I$ such that $\forall i \in I$ $\mathcal{C}_i \in \mathcal{N}/\mathcal{R}$, $spt([\![M']\!]_\theta) \cap \mathcal{C}_i \neq \emptyset$, and $spt([\![M']\!]_\theta) \subseteq \bigcup_{i \in I} \mathcal{C}_i$. We get:

  $Prob_M^F(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}) = \sum_{i \in I}(Prob_M^F(\stackrel{\alpha}{\rightarrow}, \mathcal{C}_i) \times Prob_{Rep_{\mathcal{C}_i}}^F(\Longrightarrow, \mathcal{C}))$,

  where $Rep_{\mathcal{C}_i}$ is a representative element of the equivalence class C_i. Since $M\mathcal{R}N$ $\exists \bar{F} \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that, $\forall i \in I$: if $\alpha \neq c?\tilde{v}@l$ $Prob_M^F(\stackrel{\alpha}{\rightarrow}, \mathcal{C}_i) = Prob_N^{\bar{F}}(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}_i)$, while, if $\alpha = c?\tilde{v}@l$ $Prob_M^F(\stackrel{\alpha}{\rightarrow}, \mathcal{C}_i) = Prob_N^{\bar{F}}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}_i)$, or $Prob_M^F(\stackrel{\alpha}{\rightarrow}, \mathcal{C}_i) = Prob_N^{\bar{F}}(\Longrightarrow, \mathcal{C}_i)$.

  Now, if we take $F' \in LSched$ such that, $\forall e$ such that $e \leq_{prefix} e' \in Exec_N^{\bar{F}}(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}_i)$, $F'(e) = \bar{F}(e)$, and $\forall e$ such that $e \leq_{prefix} e' \in Exec_{Rep_{\mathcal{C}_i}}^F(\Longrightarrow, \mathcal{C})$ $F'(e) = F(e)$, then, since F' is a composition of F and $\bar{F}$, both elements of $\hat{\mathcal{F}}_{\mathcal{C}}$, by Definition 4.6 $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ too and we get, if $\alpha \neq c?\tilde{v}@l$,

  $$Prob_M^F(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}) = \sum_{i \in I}(Prob_M^F(\stackrel{\alpha}{\rightarrow}, \mathcal{C}_i) \times Prob_{Rep_{\mathcal{C}_i}}^F(\Longrightarrow, \mathcal{C}))$$
  $$= \sum_{i \in I}(Prob_N^{F'}(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}_i) \times Prob_{Rep_{\mathcal{C}_i}}^{F'}(\Longrightarrow, \mathcal{C})) = Prob_N^{F'}(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}),$$

  while, if $\alpha = c?\tilde{v}@l$:

  $$Prob_M^F(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}) = \sum_{i \in I}(Prob_M^F(\stackrel{\alpha}{\rightarrow}, \mathcal{C}_i) \times Prob_{Rep_{\mathcal{C}_i}}^F(\Longrightarrow, \mathcal{C}))$$
  $$= \sum_{i \in I}(Prob_N^{F'}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}_i) \times Prob_{Rep_{\mathcal{C}_i}}^{F'}(\Longrightarrow, \mathcal{C})) = Prob_N^{F'}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}),$$

  or

  $$= \sum_{i \in I}(Prob_N^{F'}(\Longrightarrow, \mathcal{C}_i) \times Prob_{Rep_{\mathcal{C}_i}}^{F'}(\Longrightarrow, \mathcal{C})) = Prob_N^{F'}(\Longrightarrow, \mathcal{C}),$$

  as required.

- The first transition is a τ, and $M \stackrel{\tau}{\rightarrow} [\![M']\!]_\theta$.

  The proof is analogous to the first item.

Proposition 4.3 *Let $\mathcal{R} = (\bigcup_{i \in I} \mathcal{R}_i)^*$, where $\mathcal{R}_i$ are Probabilistic Labelled Bisimulations w.r.t. $\mathcal{F}$. Then $\mathcal{R}$ is a Probabilistic Labelled Bisimulation w.r.t. $\mathcal{F}$.*

Proof.

Each relation $\mathcal{R}_i$ partition the set $\mathcal{N}$ in equivalence classes. If $(M,N) \in \mathcal{R}_i$, that means $(M,N) \in \mathcal{R}$, by definition of $\mathcal{R}$. Given then an equivalence class $\mathcal{C}^i \in \mathcal{N}/\mathcal{R}_i$, this is wholly contained in an equivalence class $\mathcal{C} \in \mathcal{N}/\mathcal{R}$. By partitioning the equivalence class $\mathcal{C}$ with a set of equivalence classes for $\mathcal{R}_i$, we can then deduce the existence of a set J such that: $\mathcal{C} = \bigcup_{j\in J} \mathcal{C}^i_j$.

Now, let consider $(M,N) \in \mathcal{R}$. That means $(M,N) \in (\bigcup_{i\in I} \mathcal{R}_i)^*$, and $(M,N) \in (\bigcup_{i\in I} \mathcal{R}_i)^n$ for some $n > 0$.

We will prove by induction over n that $\mathcal{R}$ is a probabilistic labelled bisimulation w.r.t. $\hat{\mathcal{F}}$.

- $n = 1$.

  If $n = 1$, $(M,N) \in (\bigcup_{i\in I} \mathcal{R}_i)^1$ means that for some $i \in I$, $(M,N) \in \mathcal{R}_i$. We have that, $\forall F \in \hat{\mathcal{F}}_{\mathcal{C}}$, $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ s.t. $\forall \alpha$, $\mathcal{C} \in \mathcal{N}/\mathcal{R}$:

  If $\alpha \neq c?\tilde{v}@l$ then $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = \sum_{j\in J} Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}^i_j)$

  $$= \sum_{j\in J} Prob^{F'}_N(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}^i_j) = Prob^{F'}_N(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}),$$

  as required.

  If $\alpha = c?\tilde{v}@l$ then:

  $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = \sum_{j\in J} Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}^i_j)$

  $$= \sum_{j\in J} Prob^{F'}_N(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}^i_j) = Prob^{F'}_N(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}),$$

  or

  $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = \sum_{j\in J} Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}^i_j)$

  $$= \sum_{j\in J} Prob^{F'}_N(\Longrightarrow, \mathcal{C}^i_j) = Prob^{F'}_N(\Longrightarrow, \mathcal{C}),$$

  as required.

- $n > 1$.

  $(M,N) \in (\bigcup_{i\in I} \mathcal{R}_i)^n$ means that $\exists i \in I$ such that $M\mathcal{R}_i(\bigcup_{i\in I} \mathcal{R}_i)^{n-1}N$, and that $\exists O \in \mathcal{N}$ such that, $(M,O) \in \mathcal{R}_i$ and $(O,N) \in (\bigcup_{i\in I} \mathcal{R}_i)^{n-1}$.

  $(M,O) \in \mathcal{R}_i$ implies that, $\forall F \in \hat{\mathcal{F}}_{\mathcal{C}}$ $\exists F_1 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $\forall \mathcal{C} \in \mathcal{N}/\mathcal{R}$ and $\forall \alpha$:

  If $\alpha \neq c?\tilde{v}@l$ then $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = \sum_{j\in J} Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}^i_j)$

  $$= \sum_{j\in J} Prob^{F_1}_O(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}^i_j) = Prob^{F_1}_O(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}),$$

  while, if $\alpha = c?\tilde{v}@l$ then:

  $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = \sum_{j\in J} Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}^i_j)$

  $$= \sum_{j\in J} Prob^{F_1}_O(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}^i_j) = Prob^{F_1}_O(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}),$$

  or

$Prob_M^F(\xrightarrow{\alpha}, \mathcal{C}) = \sum_{j\in J} Prob_M^F(\xrightarrow{\alpha}, \mathcal{C}_j^i)$

$= \sum_{j\in J} Prob_O^{F_1}(\Longrightarrow, \mathcal{C}_j^i) = Prob_O^{F_1}(\Longrightarrow, \mathcal{C})$.

By induction hypothesis, $\forall m < n$, $(\bigcup_{i\in I} \mathcal{R}_i)^m$ is a bisimulation w.r.t. $\mathcal{F}$, hence $(\bigcup_{i\in I} \mathcal{R}_i)^{n-1}$ is a bisimulation w.r.t. $\mathcal{F}$. Again, since for each $(P,Q) \in (\bigcup_{i\in I} \mathcal{R}_i)^{n-1}$, $(P,Q) \in \mathcal{R}$, each equivalence class of $(\bigcup_{i\in I} \mathcal{R}_i)^{n-1}$ is wholly contained in an equivalence class for $\mathcal{R}$ and we can then partition $\mathcal{C}$ with a set of equivalence classes in $(\bigcup_{i\in I} \mathcal{R}_i)^{n-1}$, that means $\exists J'$ such that $\mathcal{C} = \bigcup_{j\in J'} \mathcal{C}_j$ where $\mathcal{C}_j \in \mathcal{N}/(\bigcup_{i\in I} \mathcal{R}_i)^{n-1}$ $\forall j \in J'$.

By Proposition 4.2 and by Definition 4.6 we finally get that $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $\forall \mathcal{C} \in \mathcal{N}/\mathcal{R}$ and $\forall \alpha$, if $\alpha \neq c?\tilde{v}@l$:

$Prob_M^F(\xrightarrow{\alpha}, \mathcal{C}) = Prob_O^{F_1}(\overset{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}) = \sum_{j\in J'} Prob_O^{F_1}(\overset{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}_j)$

$= \sum_{j\in J'} Prob_N^{F'}(\overset{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}_j) = Prob_N^{F'}(\overset{\hat{\alpha}}{\Longrightarrow}, \mathcal{C})$,

while, if $\alpha = c?\tilde{v}@l$ then there are three different possibilities:

$Prob_M^F(\xrightarrow{\alpha}, \mathcal{C}) = Prob_O^{F_1}(\overset{\alpha}{\Longrightarrow}, \mathcal{C}) = \sum_{j\in J'} Prob_O^{F_1}(\overset{\alpha}{\Longrightarrow}, \mathcal{C}_j)$

$= \sum_{j\in J'} Prob_N^{F'}(\overset{\alpha}{\Longrightarrow}, \mathcal{C}_j) = Prob_N^{F'}(\overset{\alpha}{\Longrightarrow}, \mathcal{C})$,

$Prob_M^F(\xrightarrow{\alpha}, \mathcal{C}) = Prob_O^{F_1}(\overset{\alpha}{\Longrightarrow}, \mathcal{C}) = \sum_{j\in J'} Prob_O^{F_1}(\overset{\alpha}{\Longrightarrow}, \mathcal{C}_j)$

$= \sum_{j\in J} Prob_N^{F'}(\Longrightarrow, \mathcal{C}_j) = Prob_O^{F'}(\Longrightarrow, \mathcal{C})$,

or

$Prob_M^F(\xrightarrow{\alpha}, \mathcal{C}) = Prob_O^{F_1}(\Longrightarrow, \mathcal{C}) = \sum_{j\in J'} Prob_O^{F_1}(\Longrightarrow, \mathcal{C}_j)$

$= \sum_{j\in J} Prob_N^{F'}(\Longrightarrow, \mathcal{C}_j) = Prob_O^{F'}(\Longrightarrow, \mathcal{C})$.

4.2.6 A complete characterisation

We show that our probabilistic labelled bisimilarity is a complete characterisation of the probabilistic observational congruence of Definition 4.5.

Theorem 4.2 (Soundness) *Let M and N be two networks. We show that if $M \approx_p^{\mathcal{F}} N$ then $M \cong_p^{\mathcal{F}} N$.*

Proof.

In order to prove that probabilistic labelled bisimilarity $\approx_p^{\mathcal{F}}$ is a sound characterisation of probabilistic observational congruence $\cong_p^{\mathcal{F}}$ we have to prove that $\approx_p^{\mathcal{F}}$ is:

1. probabilistic barb preserving
2. reduction closed

3. contextual.

1. To prove that probabilistic labelled bisimilarity $\approx_p^{\mathcal{F}}$ is *barb preserving* we have to show that if $M \approx_p^{\mathcal{F}} N$ then, for each scheduler $F \in \mathcal{F}_{\mathcal{C}}$, for each channel c and for each set K of locations such that $M \Downarrow_p^F c@K$, there exists $F' \in \mathcal{F}_{\mathcal{C}}$ such that $N \Downarrow_p^{F'} c@K$.

Assume that $M \Downarrow_p^F c@K$ for some $F \in \mathcal{F}_{\mathcal{C}}$. Then, by Definition 4.2 we have $Prob_M^F(H) = p$, where $H = \{M' : M' \downarrow_{c@K}\}$. We can partition H with a set of equivalence classes with respect to $\approx_p^{\mathcal{F}}$. Formally, $\exists J$ such that $H \subseteq \cup_{j \in J} \mathcal{C}_j$, $\forall j \in J$ $\mathcal{C}_j \in \mathcal{N}/\cong_p^{\mathcal{F}}$ and $H \cap \mathcal{C}_j \neq \emptyset$. Hence:

$Prob_M^F(H) = \sum_{e \in Exec_M^F(H)} P_M^F(e) = \sum_{j \in J} Prob_M^F(\mathcal{C}_j) = p$.

By Theorem 4.1 and by Definition 4.6 there exists $\hat{F} \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $\forall j \in J$:

$Prob_M^F(\mathcal{C}_j) = Prob_M^{\hat{F}}(\Longrightarrow, \mathcal{C}'_j)$

where $\mathcal{C}'_j = \mathcal{C}_j \cup \{\hat{M} \mid \exists \hat{M}' \in \mathcal{C}_j \text{ and } \hat{M} \equiv \hat{M}'\}$;

Now, since $\forall \hat{M}$ such that $\hat{M} \equiv \hat{M}' \in \mathcal{C}_j$, by applying rule (R-Struct) and by Definition 4.3 $\hat{M} \cong_p^{\mathcal{F}} \hat{M}'$, we get $\{\hat{M} : \hat{M} \equiv \hat{M}' \in \mathcal{C}_j\} \subseteq \mathcal{C}_j$, that means $\mathcal{C}'_j = \mathcal{C}_j$ $\forall j \in J$. Hence we get:

$\sum_{j \in J} Prob_M^F(\mathcal{C}_j) = \sum_{j \in J} Prob_M^{\hat{F}}(\Longrightarrow, \mathcal{C}_j)$.

Since $M \approx_p^{\mathcal{F}} N$, there exists $\hat{F}' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that, by Proposition 4.2, for all $j \in J$:

$Prob_M^{\hat{F}}(\Longrightarrow, \mathcal{C}_j) = Prob_N^{\hat{F}'}(\Longrightarrow, \mathcal{C}_j)$.

We then have:

$p = \sum_{j \in J} Prob_N^{\hat{F}'}(\Longrightarrow, \mathcal{C}_j)$.

Again, by Theorem 4.1, Proposition 4.2 and Definition 4.3, there exists $F' \in \mathcal{F}_{\mathcal{C}}$ such that for all $j \in J$:

$Prob_N^{\hat{F}'}(\Longrightarrow, \mathcal{C}_j) = Prob_N^{F'}(\mathcal{C}_j)$ and

$p = \sum_{j \in J} Prob_N^{\hat{F}'}(\Longrightarrow, \mathcal{C}_j) = \sum_{i \in J} Prob_N^{F'}(\mathcal{C}_j) = Prob_N^{F'}(H)$, that means $N \Downarrow_p^{F'} c@K$ as required.

2. To prove that probabilistic labelled bisimilarity $\approx_p^{\mathcal{F}}$ is reduction closed, we have to show that if $M \approx_p^{\mathcal{F}} N$, then for all $F \in \mathcal{F}_{\mathcal{C}}$, there exists $F' \in \mathcal{F}_{\mathcal{C}}$ such that for all classes $\mathcal{C} \in \mathcal{N}/\cong_p^{\mathcal{F}}$, $Prob_M^F(\mathcal{C}) = Prob_N^{F'}(\mathcal{C})$.

By Theorem 4.1 and by Definition 4.6 we deduce that $\exists \hat{F} \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $Prob_M^F(\mathcal{C}) = Prob_M^{\hat{F}}(\Longrightarrow, \mathcal{C}')$, where $\mathcal{C}' = \mathcal{C} \cup \{\hat{M} : \hat{M} \equiv \hat{M}' \in \mathcal{C}\}$, but since $\forall \hat{M}$ such that $\hat{M} \equiv \hat{M}' \in \mathcal{C}$, by applying rule (R-Struct) and by Definition 4.3 $\hat{M} \cong_p^{\mathcal{F}} \hat{M}'$ we get $\{\hat{M} : \hat{M} \equiv \hat{M}' \in \mathcal{C}\} \subseteq \mathcal{C}$, that means $\mathcal{C}' = \mathcal{C}$.

By Proposition 4.2 we have that $\exists \hat{F}' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $Prob_M^{\hat{F}}(\Longrightarrow, \mathcal{C}) = Prob_N^{\hat{F}'}(\Longrightarrow, \mathcal{C})$.

Finally, by Theorem 4.1 and by Definitions 4.6 and 4.3, $\exists F' \in \mathcal{F}_{\mathcal{C}}$ such that $Prob_N^{\hat{F}'}(\Longrightarrow, \mathcal{C}) = Prob_N^{F'}(\mathcal{C})$, as required.

3. In order to prove that probabilistic labelled bisimilarity $\approx_p^{\mathcal{F}}$ is contextual we have to prove that, if $M \approx_p^{\mathcal{F}} N$:

1. $M \mid O \approx_p^{\mathcal{F}} N \mid O \ \forall O \in \mathcal{N}$.
2. $(\nu d)M \approx_p^{\mathcal{F}} (\nu d)N \ \forall d \in \mathbf{C}$.

Case 1.

Let consider the relation $\mathcal{R} = \{(M \mid O, N \mid O) : M \approx_p^{\mathcal{F}} N\}$.

We prove that for all scheduler $F \in \hat{\mathcal{F}}_{\mathcal{C}}$ there exists a scheduler $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that for all α and for all classes $\mathcal{C}$ in $\mathcal{N}/\approx_p^{\mathcal{F}}$:

1. if $\alpha = \tau$ then

   $Prob^{F}_{M|O}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F'}_{N|O}(\Longrightarrow, \mathcal{C})$.

   If $P, Q \in \mathcal{C}$, then, by definition of $\mathcal{R}$, $P \equiv \bar{P} \mid \bar{O}$, $Q \equiv \bar{Q} \mid \bar{O}$ and $\bar{P} \approx_p^{\mathcal{F}} \bar{Q}$. But then there exists $\mathcal{D} \in \mathcal{N}/\approx_p^{\mathcal{F}}$ such that $\mathcal{D} = \{\bar{P} : \bar{P} \mid \bar{O} \in \mathcal{C}\}$. Now we have three cases to consider:

   (i) if $M \mid O \xrightarrow{\tau} [\![M \mid O']\!]_\theta$ because $O \xrightarrow{\tau} [\![O']\!]_\theta$ the proof is simple, because for all $\bar{M}$ in the support of $[\![M \mid O']\!]_\theta$ such that $\bar{M} \in \mathcal{C}$, it holds $\bar{M} \equiv M \mid O''$ and, since $M \approx_p^{\mathcal{F}} N$, $N \mid O'' \in \mathcal{C}$ too, by definition of $\mathcal{R}$. By Definition 4.3 there exists $\bar{F} \in \mathcal{F}_{\mathcal{C}}$ such that, by applying rule (R-Par) to the reduction $O \to [\![O']\!]_\theta$, $N \mid O \to [\![O' \mid N]\!]_\theta \in Exec^{\bar{F}}_{N|O}$. By Theorem 4.1 and by Definition 4.6 $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $Prob^{\bar{F}}_{N|O}(\mathcal{C}) = Prob^{F'}_{N|O}(\Longrightarrow, \mathcal{C})$, hence:

   $$Prob^{F}_{M|O}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F'}_{N|O}(\Longrightarrow, \mathcal{C}).$$

   as required.

   (ii) If $M \mid O \xrightarrow{\tau} [\![M' \mid O]\!]_\theta$ because $M \xrightarrow{\tau} [\![M']\!]_\theta$, by Definition 4.6 there exists a scheduler $F_1 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $Prob^{F}_{M|O}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F_1}_{M}(\xrightarrow{\tau}, \mathcal{D})$. But since $M \approx_p^{\mathcal{F}} N$, there exists $F_2 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $Prob^{F_1}_{M}(\xrightarrow{\tau}, \mathcal{D}) = Prob^{F_2}_{N}(\Longrightarrow, \mathcal{D})$. For each execution:

   $$N \xrightarrow{\tau}_{\theta_1} N_1 ... \xrightarrow{\tau}_{\theta_k} N_k \in Exec^{F_1}_{N}(\Longrightarrow, \mathcal{D}),$$

   there exists a scheduler $\bar{F} \in \mathcal{F}_{\mathcal{C}}$ such that

   $$N \to_{\theta_1} N_1 ... \to_{\theta_k} N_k \in Exec^{\bar{F}}_{N}.$$

   By Definition 4.3, since $\mathcal{F}_{\mathcal{C}}$ captures the interactions of N with any context, $\exists \bar{F}' \in \mathcal{F}_{\mathcal{C}}$ such that, by applying rule (R-Par) to each step in e:

   $$N \mid O \to_{\theta_1} N_1 \mid O ... \to_{\theta_k} N_k \mid O \in Exec^{\bar{F}'}_{N|O}.$$

   By Definition 4.6 we finally get $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that:

   $$Prob^{F_2}_{N}(\Longrightarrow, \mathcal{D}) = Prob^{\bar{F}}_{N}(\mathcal{D}) = Prob^{\bar{F}'}_{N|O}(\mathcal{C}) = Prob^{F'}_{N|O}(\Longrightarrow, \mathcal{C}).$$

(iii) If $M \mid O \xrightarrow{\tau} [\![M' \mid O']\!]_\Delta$ due to a synchronization between M and O, then there are two cases to consider.
If $M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta$ and $O \xrightarrow{c?\tilde{v}@k} [\![O']\!]_\Delta$, for some tuple $\tilde{v}$ of messages, channel c, locations l, k and radius r, such that $d(l,k) \leq r$, we can apply rule (Obs) obtaining $M \xrightarrow{c!\tilde{v}@K\triangleleft R} [\![M']\!]_\Delta$ for some $R = \{l' \mid d(l,l') \leq r\}$ with $k \in R$ and $K = L \cap R$. Therefore, by Definition 4.6 there exists $F_1 \in \hat{\mathcal{F}}_\mathcal{C}$ such that:

$$Prob^F_{M|O}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F_1}_M(\xrightarrow{c!\tilde{v}@K\triangleleft R}, \mathcal{D}).$$

Since $N \approx^\mathcal{F}_p M$, there exists $F_2 \in \hat{\mathcal{F}}_\mathcal{C}$ such that

$$Prob^{F_1}_M(\xrightarrow{c!\tilde{v}@K\triangleleft R}, \mathcal{D}) = Prob^{F_2}_N(\overset{c!\tilde{v}@K\triangleleft R}{\Longrightarrow}, \mathcal{D}),$$

where each execution e in $Exec^{F_2}_N(\overset{c!\tilde{v}@K\triangleleft R}{\Longrightarrow}, \mathcal{D})$ is of the form

$$e = N \xrightarrow{\tau}_{\theta_1} N_1 \xrightarrow{\tau}_{\theta_2} ...N_{i-1} \xrightarrow{c!\tilde{v}@K\triangleleft R}_\Delta N_i \xrightarrow{\tau}_{\theta_{i+1}} ... N',$$

with $k \in R$, and, by applying rule (Obs) backwardly, $N_{i-1} \xrightarrow{c!\tilde{v}[l',r']}_\Delta N_i$ for some l' and r' such that $d(l',k) \leq r'$. We can apply rule (Bcast) obtaining $N_{i-1} \mid O \xrightarrow{c!\tilde{v}[l',r']}_\Delta N_i \mid O'$ without changing probability. Finally if we take $F' \in LSched$ which applies rule (Lose) to the output action, we obtain the required result:

$$Prob^{F_2}_N(\overset{c!\tilde{v}@K\triangleleft R}{\Longrightarrow}, \mathcal{D}) = Prob^{F'}_{N|O}(\Longrightarrow, \mathcal{C}).$$

We have finally to prove that $F' \in \hat{\mathcal{F}}_\mathcal{C}$. We start by the consideration that, by Definition 4.1, for any execution of the form $\overset{\alpha}{\Longrightarrow}$ in $\hat{\mathcal{F}}_\mathcal{C}$, where α is a silent or an output action there exists a correspondent reduction in $\mathcal{F}_\mathcal{C}$. Since by Definition 4.3, for any context, there exists a scheduler in $\mathcal{F}_\mathcal{C}$ mimicking the behaviour exhibited by N when interacting with the given context, we can affirm that $\exists \bar{F} \in \mathcal{F}_\mathcal{C}$ such that $Exec^{\bar{F}}_{N|O}$ contains all the reductions corresponding to the executions of $Exec^{F'}_{N|O}$. Hence, by Definition 4.6, $F' \in \hat{\mathcal{F}}_\mathcal{C}$, as required.
If $M \xrightarrow{c?\tilde{v}@k} [\![M']\!]_\Delta$ and $O \xrightarrow{c_L!\tilde{v}[l,r]} [\![O']\!]_\Delta$, for some message $\tilde{v}$, channel c, locations l, k and radius r, such that $d(l,k) \leq r$, then by Definition 4.6 $\exists F_1 \in \hat{\mathcal{F}}_\mathcal{C}$ such that:

$$Prob^F_{M|O}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F_1}_M(\xrightarrow{c?\tilde{v}@k}, \mathcal{D}),$$

and, since $M \approx^\mathcal{F}_p N$, there exists $F_2 \in \hat{\mathcal{F}}_\mathcal{C}$ such that

$$Prob^{F_1}_M(\xrightarrow{c?\tilde{v}@k}, \mathcal{D}) = Prob^{F_2}_N(\overset{c?\tilde{v}@k}{\Longrightarrow}, \mathcal{D})$$

or

$$Prob_M^{F_1}(\xrightarrow{c?\tilde{v}@k}, \mathcal{D}) = Prob_N^{F_2}(\Longrightarrow, \mathcal{D}).$$

In the first case, since by hypothesis $k \in R$, also N is able to synchronize with O, for all executions

$$e = N \xrightarrow{\tau}_{\theta_1} N_1 \xrightarrow{\tau}_{\theta_2} ...N_{i-1} \xrightarrow{c?\tilde{v}@k}_{\Delta} N_i \xrightarrow{\tau}_{\theta_{i+1}} ...N' \in Exec_N^{F_2}(\overset{c?\tilde{v}@k}{\Longrightarrow}, \mathcal{D}),$$

since by hypothesis $d(l,k) \leq r$, by applying rule (Bcast) we get $N_{i-1} \mid O \xrightarrow{c_L!\tilde{v}[l.r]} N_i \mid O'$, and there exists a matching execution:

$$N \mid O \xrightarrow{\tau}_{\theta_1} N_1 \mid O \xrightarrow{\tau}_{\theta_2} ...N_{i-1} \mid O \xrightarrow{c_L!\tilde{v}[l,r]}_{\Delta} N_i \mid O' \xrightarrow{\tau}_{\theta_{i+1}} ...N' \mid O'.$$

By applying the rule (Lose) to the action $N_{i-1} \mid O \xrightarrow{c_L!\tilde{v}[l,r]}_{\Delta} N_i \mid O'$ and by Definition 4.3 $\exists \bar{F}' \in \mathcal{F}_{\mathcal{C}}$ such that,

$$Prob_{N|O}^{\bar{F}'}(\mathcal{C}) = Prob_N^{F_2}(\mathcal{D}).$$

By Definition 4.6 there exists $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that,

$$Prob_{N|O}^{F'}(\Longrightarrow, \mathcal{C}) = Prob_{N|O}^{\bar{F}'}(\mathcal{C}).$$

If N is not able to receive the message the proof is analogous, because $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that, for each execution of $Exec_N^{F_1}(\Longrightarrow, \mathcal{D})$:
$N \xrightarrow{\tau}_{\theta_1} N_1 ... \xrightarrow{\tau}_{\theta_k} N_k$,
by applying rule (Par) to each step:
$N \mid O \xrightarrow{\tau}_{\theta_1} N_1 \mid O ... \xrightarrow{\tau}_{\theta_k} N_k \mid O$,
and by applying rule (Bcast) and (Lose) to O, and then (Par) to $N_k \mid O$, we get:
$N \mid O \xrightarrow{\tau}_{\theta_1} N_1 \mid O ... \xrightarrow{\tau}_{\theta_k} N_k \mid O \xrightarrow{\tau}_{\Delta} N_k \mid O' \in Exec_{N|O}^{F'}(\Longrightarrow, \mathcal{C})$,
hence, since the output of O does not change the probabilities of the executions, we get:

$$Prob_{M|O}^{F}(\Longrightarrow, \mathcal{C}) = Prob_M^{F_1}(\Longrightarrow, \mathcal{D}) = Prob_N^{F_2}(\Longrightarrow, \mathcal{D}) = Prob_{N|O}^{F'}(\Longrightarrow, \mathcal{C}).$$

2. if $\alpha = c!\tilde{v}@K \triangleleft R$ then $Prob_{M|O}^{F}(\xrightarrow{c!\tilde{v}@K \triangleleft R}, \mathcal{C}) = Prob_{N|O}^{F'}(\overset{c!\tilde{v}@K \triangleleft R}{\Longrightarrow}, \mathcal{C})$.

   The proof is analogous to point *(iii)* of the previous item.

3. if $\alpha = c?\tilde{v}@k$ then $Prob_{M|O}^{F}(\xrightarrow{\alpha}, \mathcal{C}) = Prob_{N|O}^{F'}(\overset{\alpha}{\Longrightarrow}, \mathcal{C})$

   or

   $Prob_{M|O}^{F}(\xrightarrow{\alpha}, \mathcal{C}) = Prob_{N|O}^{F'}(\Longrightarrow, \mathcal{C})$.

   If $P, Q \in \mathcal{C}$, then by definition of $\mathcal{R}$, $P \equiv \bar{P} \mid \bar{O}$, $Q \equiv \bar{Q} \mid \bar{O}$ and $\bar{P} \approx_p^{\mathcal{F}} \bar{Q}$. But then there exists $\mathcal{D} \in \mathcal{N}/\approx_p^{\mathcal{F}}$ such that $\mathcal{D} = \{\bar{P} : \bar{P} \mid \bar{O} \in \mathcal{C}\}$. Now we have two cases to consider:

(i) The transition is due to an action performed by O, hence $O \xrightarrow{\alpha}_{\Delta} O'$ and $M \mid O' \in \mathcal{C}$. But since $M \approx_p^{\mathcal{F}} N$, then also $N \mid O' \in \mathcal{C}$, and, by Definition 4.6 there exists $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that by applying rule (Par) to $O \xrightarrow{\alpha} O'$, we get $N \mid O \xrightarrow{\alpha} N \mid O'$ obtaining:

$$Prob^{F}_{M|O}(\xrightarrow{\alpha}, \mathcal{C}) = Prob^{F'}_{N|O}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}).$$

(ii) The transition is due to an action performed by M, in this case, by Definition 4.6 $\exists F_1 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that:
$Prob^{F}_{M|O}(\xrightarrow{\alpha}, \mathcal{C}) = Prob^{F_1}_{M}(\xrightarrow{\alpha}, \mathcal{D})$.
Since $M \approx_p^{\mathcal{F}} N$, there exists $F_2 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that
$Prob^{F_1}_{M}(\xrightarrow{\alpha}, \mathcal{D}) = Prob^{F_2}_{N}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{D})$,
or
$Prob^{F_1}_{M}(\xrightarrow{\alpha}, \mathcal{D}) = Prob^{F_2}_{N}(\Longrightarrow, \mathcal{D})$.
In both cases, for each $e \in Exec^{F_1}_{N}(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{D})$:
$e = N \xrightarrow{\alpha_1}_{\theta_1} N_1 ... \xrightarrow{\alpha_k}_{\theta_k} N_k$
by applying rule (Par) to each step we get:
$N \mid O \xrightarrow{\alpha_1}_{\theta_1} N_1 \mid O ... \xrightarrow{\alpha_k}_{\theta_k} N_k \mid O$.
Hence, $\exists F' \in LSched$ such that:
$Prob^{F_2}_{N}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{D}) = Prob^{F'}_{N|O}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C})$,
or
$Prob^{F_2}_{N}(\Longrightarrow, \mathcal{D}) = Prob^{F'}_{N|O}(\Longrightarrow, \mathcal{C})$.

In order to prove that $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$, we start by the consideration that, by Definition 4.6 there exists at least a context $C[\cdot]$ and $\exists \bar{F} \in \mathcal{F}_{\mathcal{C}}$ such that $C[N] \to C'[N']$, and, by the reduction rules we get:

$$C[\cdot] \equiv (\nu \tilde{d}) m[\bar{c}_{L,r}\langle \tilde{v} \rangle.P]_l \mid M_1$$

for some $\tilde{d}$ such that $c \notin \tilde{d}$, some m, some set L of locations, some process P, some (possibly empty) network M_1, some location l and some radius r such that $d(l,k) \leq r$. Then, by Definition 4.3 we have that there exists a scheduler allowing $m[\bar{c}_{L,r}\langle \tilde{v} \rangle.P]_l \to [\![m[P]_l]\!]_{\Delta}$, and again by Definition 4.3 there exists a scheduler allowing the reduction $m[\bar{c}_{L,r}\langle \tilde{v} \rangle.P]_l \mid N \mid O \to^* [\![m[P]_l \mid N' \mid O']\!]_{\Delta}$, meaning, by Definition 4.6, $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ as required.

Case 2.

Let consider now the relation $\mathcal{S} = \{((\nu d)M, (\nu d)N) : M \approx_p^{\mathcal{F}} N\}$.

Let consider $\mathcal{C}$: if $P, Q \in \mathcal{C}$, then by definition of $\mathcal{S}$ $P \equiv (\nu \bar{d})\bar{P}$, $Q \equiv (\nu \bar{d})\bar{Q}$ and $\bar{P} \approx_p^{\mathcal{F}} \bar{Q}$. But then $\exists \mathcal{D} \in \mathcal{N}/\approx_p^{\mathcal{F}}$ such that $\mathcal{D} = \{\bar{P} : (\nu \bar{d})\bar{P} \in \mathcal{C}\}$.

We have to prove that, $\forall F \in \hat{\mathcal{F}}_{\mathcal{C}}$, $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ sucht that, $\forall \mathcal{C} \in \mathcal{N}/\mathcal{S}$, $\forall \alpha$:

1. $\alpha = \tau$ implies that $Prob^{F}_{(\nu d)M}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F'}_{(\nu d)N}(\Longrightarrow, \mathcal{C})$.

   Since $\mathtt{Chan}(\tau) = \perp$, by Definition 4.6 $\exists F_1 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $Prob^{F}_{(\nu d)M}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F_1}_{M}(\xrightarrow{\tau}, \mathcal{D})$ and, since $M \approx^{\mathcal{F}}_{p} N$ $\exists F_2 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that: $Prob^{F_1}_{M}(\xrightarrow{\tau}, \mathcal{D}) = Prob^{F_2}_{N}(\Longrightarrow, \mathcal{D})$.

   Finally we can take $F' \in LSched$ mimicking the executions in the set $Exec^{F_2}_{N}(\Longrightarrow, \mathcal{D})$, when applying the restriction on N. Hence:

   $$Prob^{F_2}_{N}(\Longrightarrow, \mathcal{D}) = Prob^{F'}_{(\nu d)N}(\Longrightarrow, \mathcal{C}).$$

   In order to prove that $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$, we start by the consideration that, by Definition 4.3, for any context there exists a scheduler in $\mathcal{F}_{\mathcal{C}}$ mimicking the behaviour of N when interacting with the given context. Hence $\exists \bar{F} \in \mathcal{F}_{\mathcal{C}}$ such that $Exec^{\bar{F}}_{(\nu d)N}$ contains all the reductions corresponding to the executions in $Exec^{F'}_{(\nu d)N}$, meaning, by Definition 4.6, $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ as required.

2. $\alpha = c!\tilde{v}@K \triangleleft R$

   Since $\mathtt{Chan}(c!\tilde{v}@K \triangleleft R) \neq d$, by Definition 4.6 $\exists F_1 \in \hat{\mathcal{F}}_{\mathcal{C}}$

   such that $Prob^{F}_{(\nu d)M}(\xrightarrow{\alpha}, \mathcal{C}) = Prob^{F_1}_{M}(\xrightarrow{\alpha}, \mathcal{D})$, then since $M \approx^{\mathcal{F}}_{p} N$, $\exists F_2 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that

   $Prob^{F_1}_{M}(\xrightarrow{\alpha}, \mathcal{D}) = Prob^{F'}_{N}(\overset{\alpha}{\Longrightarrow}, \mathcal{D})$.

   Therefore, since $\mathtt{Chan}(\alpha) \neq d$, $\exists F' \in LSched$ such that:

   $$Prob^{F_2}_{N}(\overset{\alpha}{\Longrightarrow}, \mathcal{D}) = Prob^{F_2}_{(\nu d)N}(\overset{\alpha}{\Longrightarrow}, \mathcal{C}).$$

   We prove that $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ as for the previous cases.

3. $\alpha = c?\tilde{v}@k$

   Again, since $\mathtt{Chan}(c?\tilde{v}@k) \neq d$, by Definition 4.6 $\exists F_1 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that

   $Prob^{F}_{(\nu d)M}(\xrightarrow{\alpha}, \mathcal{C}) = Prob^{F_1}_{M}(\xrightarrow{\alpha}, \mathcal{D})$. Since $M \approx^{\mathcal{F}}_{p} N$, there exists $F_2 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that

   $Prob^{F_1}_{M}(\xrightarrow{\alpha}, \mathcal{D}) = Prob^{F_2}_{N}(\overset{\alpha}{\Longrightarrow}, \mathcal{D})$ or $Prob^{F_1}_{M}(\xrightarrow{\alpha}, \mathcal{D}) = Prob^{F_2}_{N}(\Longrightarrow, \mathcal{D})$, if N is not able to receive $\tilde{v}$. In both cases we can apply rule (Res) to N, since $\mathtt{Chan}(\tau) = \perp$ and $\mathtt{Chan}(c?\tilde{v}@k) \neq d$. Therefore, there exists $F' \in LSched$ such that the required result holds, that is $Prob^{F_2}_{N}(\overset{\alpha}{\Longrightarrow}, \mathcal{D}) = Prob^{F'}_{(\nu d)N}(\overset{\alpha}{\Longrightarrow}, \mathcal{C})$ or $Prob^{F_2}_{N}(\Longrightarrow, \mathcal{D}) = Prob^{F'}_{(\nu d)N}(\Longrightarrow, \mathcal{C})$.

   Again, we prove that $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ as for the previous cases.

We finally prove that probabilistic labelled bisimilarity w.r.t. $\mathcal{F}$ is a complete characterisation of the probabilistic observational congruence w.r.t. $\mathcal{F}$ of Definition 4.5.

Theorem 4.3 (Completeness) *If $M \cong_p^{\mathcal{F}} N$ then $M \approx_p^{\mathcal{F}} N$.*

Proof.

In order to prove the completeness of probabilistic labelled bisimilarity we show that the relation $\mathcal{R} = \{(M,N) : M \cong_p^{\mathcal{F}} N\}$ is a Probabilistic Labelled Bisimulation. This follows from Theorem 4.1 and Proposition 4.2.

We have to prove that, $\forall F \in \hat{\mathcal{F}}_{\mathcal{C}}\ \exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that, $\forall \mathcal{C} \in \mathcal{N}/\mathcal{R}$, $\forall \alpha$:

if $\alpha = \tau$ then $Prob_M^F(\xrightarrow{\tau}, \mathcal{C}) = Prob_N^{F'}(\Longrightarrow, \mathcal{C})$.

By Theorem 4.1 and by Definition 4.6 we know that $\exists \bar{F} \in \mathcal{F}_{\mathcal{C}}$ such that $Prob_M^F(\xrightarrow{\tau}, \mathcal{C}) = Prob_M^{\bar{F}}(\mathcal{C})$, and, since $M \cong_p^{\mathcal{F}} N$, $\exists \bar{F}' \in \mathcal{F}_{\mathcal{C}}$ such that $Prob_M^{\hat{F}}(\mathcal{C}) = Prob_N^{\bar{F}'}(\mathcal{C})$. Again by Theorem 4.1 and by Definition 4.6 $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $Prob_N^{\hat{F}'}(\mathcal{C}) = Prob_N^{F'}(\Longrightarrow, \mathcal{C} \cup \{\bar{N} \equiv N' \in \mathcal{C}\})$, but since $\cong_p^{\mathcal{F}}$ is closed under structural equivalence, $\forall \bar{N} \equiv N' \in \mathcal{C}$, $\bar{N} \in \mathcal{C}$, hence: $Prob_M^F(\xrightarrow{\tau}, \mathcal{C}) = Prob_N^{F'}(\Longrightarrow, \mathcal{C})$.

if $\alpha = c!\tilde{v}@K \triangleleft R$ then $Prob_M^F(\xrightarrow{\alpha}, \mathcal{C}) = Prob_N^{F'}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C})$.

First we notice that $Prob_M^F(\xrightarrow{c!\tilde{v}@K\triangleleft R}, \mathcal{C})$ is either 0 or 1.

If $Prob_M^F(\xrightarrow{c!\tilde{v}@K\triangleleft R}, \mathcal{C}) = 0$ we are done, because it will be enough to take any scheduler $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ not allowing observable output actions on the channel c, and we get $Prob_M^F(\xrightarrow{c!\tilde{v}@K\triangleleft R}, \mathcal{C}) = Prob_N^{F'}(\stackrel{c!\tilde{v}@K\triangleleft R}{\Longrightarrow}, \mathcal{C})$.

If $Prob_M^F(\xrightarrow{c!\tilde{v}@K\triangleleft R}, \mathcal{C}) = 1$, by Theorem 4.1 and by Definition 4.6 $\exists \bar{F} \in \mathcal{F}_{\mathcal{C}}$ such that $M \Downarrow_1^{\bar{F}} c@K$, and it means that $\exists \bar{F}' \in \mathcal{F}_{\mathcal{C}}$ such that $N \Downarrow_1^{\bar{F}'} c@K$, hence, again by Theorem 4.1 and by Definition 4.6 there exist $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ and R' such that $K \subseteq R'$ and $Prob_N^{\bar{F}'}(\mathcal{C}) = Prob_N^{F'}(\stackrel{c!\tilde{v}@K\triangleleft R'}{\Longrightarrow}, \mathcal{C})$.

We proved that $\exists R'$ such that $Prob_M^F(\xrightarrow{c!\tilde{v}@K\triangleleft R}, \mathcal{C}) = Prob_N^{F'}(\stackrel{c!\tilde{v}@K\triangleleft R'}{\Longrightarrow}, \mathcal{C})$, now we want to show that $R' = R$. In order to mimic the effect of the action $c!\tilde{v}@K \triangleleft R$, we build the following context

$$C[\cdot] = \prod_{i=1}^{n}(n_i[c(\tilde{x_i}).[\tilde{x_i} = \tilde{v}]\bar{\mathtt{f}}_{k_i,r}^{(i)}\langle\tilde{x_i}\rangle]_{k_i} \mid m_i[\mathtt{f}^{(i)}(\tilde{y_i}).\bar{\mathtt{ok}}_{k_i,r}^{(i)}\langle\tilde{y_i}\rangle]_{k_i}),$$

where $R = \{k_1, ..., k_n\}$, n_i, m_i, $\mathtt{ok}^{(i)}$ and $\mathtt{f}^{(i)}$ fresh $\forall i \in [1-n]$. Since $M \xrightarrow{c!\tilde{v}@K\triangleleft R}$, then the message is reachable by all nodes n_i, hence, by Definition 4.3 $\exists \bar{F}_1 \in \mathcal{F}_{\mathcal{C}}$ such that $C[M] \rightarrow^* \hat{M}$, where

$$\hat{M} \equiv M' \mid \prod_{i=1}^{n}(n_i[\mathbf{0}]_{k_i} \mid m_i[\bar{\mathtt{ok}}_{k_i,r}^{(i)}\langle\tilde{v_i}\rangle]_{k_i}$$

$$\equiv M' \mid \prod_{i=1}^{n}(m_i[\bar{\mathtt{ok}}_{k_i,r}^{(i)}\langle\tilde{v_i}\rangle]_{k_i},$$

with $\hat{M} \not\downarrow_{\mathtt{f}^{(i)}@R}$ and $\hat{M} \Downarrow_1^{\bar{F}_1} \mathtt{ok}^{(i)}@R$, $\forall i \in [1-n]$.

The absence of the barb on the channels $\mathtt{f}^{(i)}$ together with the presence of the barb on the channels $\mathtt{ok}^{(i)}$ ensures that all the locations in R have been able to receive the message. Since $C[M] \cong_p^{\mathcal{F}} C[N]$, $\exists \bar{F}_2 \in \mathcal{F}_{\mathcal{C}}$ such that $Prob^{\bar{F}_1}_{C[M]}(\mathcal{C}') = Prob^{\bar{F}_2}_{C[N]}(\mathcal{C}')$ where $\hat{M} \in \mathcal{C}'$.

Therefore, $C[N] \rightarrow^* \hat{N}$ with $\hat{N} \not\downarrow_{\mathtt{f}^{(i)}@R}$ and $\hat{N} \Downarrow_1^{\bar{F}_2} \mathtt{ok}^{(i)}@R$. The constrains on the barbs allow us to deduce that

$$\hat{N} \equiv N' \mid \prod\nolimits_{i=1}^{n} (n_i[\mathbf{0}]_{k_i} \mid m_i[\bar{\mathtt{ok}}^{(i)}_{k_i,r} \tilde{v}_i]_{k_i})$$

$$\equiv N' \mid \prod\nolimits_{i=1}^{n} (m_i[\bar{\mathtt{ok}}^{(i)}_{k_i,r} \tilde{v}_i]_{k_i}),$$

which implies $N \stackrel{c!\tilde{v}@K \triangleleft R}{\Longrightarrow} N'$, or $N \Longrightarrow N'$ in case (Lose) has been applied to the output action on the channel c. Since $\hat{M}, \hat{N} \in \mathcal{C}$, then $\hat{M} \cong_p^{\mathcal{F}} \hat{N}$, and since $\cong_p^{\mathcal{F}}$ is contextual, it results $(\nu \mathtt{ok}^{(1)} ... \mathtt{ok}^{(n)})\hat{M} \cong_p^{\mathcal{F}_{\mathcal{M}}} (\nu \mathtt{ok}^{(1)} ... \mathtt{ok}^{(n)})\hat{N}$. By applying (Struct Res Par):

$$(\nu \mathtt{ok}^{(1)} ... \mathtt{ok}^{(n)})\hat{M} \equiv M' \mid (\nu \mathtt{ok}^{(1)} ... \mathtt{ok}^{(n)}) \prod\nolimits_{i=1}^{n} (m_i[\bar{\mathtt{ok}}^{(i)}_{k_i,r} \langle \tilde{v}_i \rangle]_{k_i}) \equiv M'$$

and

$$(\nu \mathtt{ok}^{(1)} ... \mathtt{ok}^{(n)})\hat{N} \equiv N' \mid (\nu \mathtt{ok}^{(1)} ... \mathtt{ok}^{(n)}) \prod\nolimits_{i=1}^{n} (m_i[\bar{\mathtt{ok}}^{(i)}_{k_i,r} \langle \tilde{v}_i \rangle]_{k_i}) \equiv N',$$

and, since the network $(\nu \mathtt{ok}^{(1)} ... \mathtt{ok}^{(n)}) \prod_{i=1}^{n} (m_i[\bar{\mathtt{ok}}^{(i)}_{k_i,r} \langle \tilde{v}_i \rangle]_{k_i})$ is silent, we can derive that $M' \cong_p^{\mathcal{F}} N'$. But since $N' \in \mathcal{C}$ and $N \stackrel{c!\tilde{v}@K \triangleleft R}{\Longrightarrow} N'$, by Definition 4.6 $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that:

$$Prob^{F'}_{N}(\stackrel{c!\tilde{v}@K \triangleleft R}{\Longrightarrow}, \mathcal{C}) = 1 = Prob^{F}_{M}(\stackrel{c!\tilde{v}@K \triangleleft R}{\Longrightarrow}, \mathcal{C}),$$

as required.

if $\alpha = c?\tilde{v}@k$ then $Prob^{F}_{M}(\stackrel{\alpha}{\rightarrow}, \mathcal{C}) = Prob^{F'}_{N}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C})$ or $Prob^{F'}_{N}(\Longrightarrow, \mathcal{C})$.

We notice that $Prob^{F}_{M}(\stackrel{c?\tilde{v}@k}{\longrightarrow}, \mathcal{C})$ is either 0 or 1.

If $Prob^{F}_{M}(\stackrel{c?\tilde{v}@k}{\longrightarrow}, \mathcal{C}) = 0$ we are done, because it will be enough to take any scheduler $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ not allowing input actions on the channel c, and we get $Prob^{F}_{M}(\stackrel{c?\tilde{v}@k}{\longrightarrow}, \mathcal{C}) = Prob^{F'}_{N}(\stackrel{c?\tilde{v}@k}{\Longrightarrow}, \mathcal{C})$.

If $Prob^{F}_{M}(\stackrel{c?\tilde{v}@k}{\longrightarrow}, \mathcal{C}) = 1$, because $M \stackrel{c?\tilde{v}@k}{\longrightarrow} [\![M']\!]_{\Delta}$, by Definition 4.3 there exists at least a context $C[\cdot]$ and $\exists \bar{F} \in \mathcal{F}_{\mathcal{C}}$ such that $C[M] \rightarrow C'[M']$, and by Theorem 4.1 we deduce that:

$$C[\cdot] \equiv (\nu \tilde{d}) m[\bar{c}_{L,r} \langle \tilde{v} \rangle . P]_l \mid M_1,$$
$$C'[\cdot] \equiv (\nu \tilde{d}) m[P]_l \mid M_1',$$

for some m, some tuple $\tilde{d}$ of channel such that $c \notin \tilde{d}$, dome set L of messages, some radius r, some process P, some location l such that $d(l,k) \leq r$ and some (possibly empty) network M_1 and M_1'.

By Definition 4.3, for any context there exists a scheduler in $\mathcal{F}_\mathcal{C}$ allowing m to perform the output when interacting with any context. Hence we can build the following context:

$$C_1[\cdot] = \cdot \mid m[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid m_1[c(\tilde{x}).\bar{\mathtt{f}}_{k,r'}\langle\tilde{x}\rangle.\bar{\mathtt{ok}}_{k,r'}\langle\tilde{x}\rangle]_k,$$

in order to mimic the behaviour of the networks, with m static, $\mathtt{f}$ and $\mathtt{ok}$ fresh, $r' > 0$ and $d(l,k) > r'$ $\forall l \in \mathbf{Loc}$ s.t. $l \neq k$. There exists a scheduler $\bar{F}_1 \in \mathcal{F}_\mathcal{C}$ such that:

$$C_1[M] \rightarrow^* M' \mid m[P]_l \mid m_1[\bar{\mathtt{ok}}_{k,r'}\langle\tilde{v}\rangle]_k \in Exec^{\bar{F}_1}_{C[M]},$$

with $M' \mid m[P]_l \mid m[\bar{\mathtt{ok}}_{k,r'}\langle\tilde{v}\rangle]_k \not\downarrow_{\mathtt{f}@k}$ and
$M' \mid m[P]_l \mid m[\bar{\mathtt{ok}}_{k,r'}\langle\tilde{v}\rangle]_k \Downarrow_1^{\bar{F}_1} \mathtt{ok}@k$.

The reduction sequence above must be matched by a corresponding reduction sequence $C_1[N] \rightarrow^* N' \mid m[P]_l \mid m[\bar{\mathtt{ok}}_{k,r'}\langle\tilde{v}\rangle]_k$, with

$M' \mid m[P]_l \mid m[\bar{\mathtt{ok}}_{k,r'}\langle\tilde{v}\rangle]_k \cong_p N' \mid m[P]_l \mid m[\bar{\mathtt{ok}}_{k,r'}\langle\tilde{v}\rangle]_k$,

$N' \mid m[P]_l \mid m[\bar{\mathtt{ok}}_{k,r'}\langle\tilde{v}\rangle]_k \not\downarrow_{\mathtt{f}@k}$ and
$N' \mid m[P]_l \mid m[\bar{\mathtt{ok}}_{k,r'}\langle\tilde{v}\rangle]_k \Downarrow_1^{\hat{F}_2} \mathtt{ok}@k$ for some $\bar{F}_2 \in \mathcal{F}_\mathcal{C}$.

This does not ensure that N actually performed the input action, but we can conclude that there exists $F' \in LSched$ and N' such that either $N \stackrel{c?\tilde{v}@k}{\Longrightarrow} N'$ or $N \Longrightarrow N'$. Since $M' \mid m[P]_l \mid m[\bar{\mathtt{ok}}_{k,r'}\langle\tilde{v}\rangle]_k \cong_p N' \mid m[P]_l \mid m[\bar{\mathtt{ok}}_{k,r'}\langle\tilde{v}\rangle]_k$ and $\cong_p^\mathcal{F}$ is is a contextual relation, we can easily derive $M' \cong_p^\mathcal{F} N'$ (applying rules for structural equivalence), that means $M', N' \in \mathcal{C}$ and $\exists F' \in LSched$ such that:

$$Prob^F_M(\xrightarrow{c?\tilde{v}@k}, \mathcal{C}) = 1 = Prob^{F'}_N(\stackrel{c?\tilde{v}@k}{\Longrightarrow}, \mathcal{C})$$
$$\text{or}$$
$$Prob^F_M(\xrightarrow{c?\tilde{v}@k}, \mathcal{C}) = 1 = Prob^{F'}_N(\Longrightarrow, \mathcal{C}).$$

Now we have only to prove that $F' \in \hat{\mathcal{F}}_\mathcal{C}$, but this follows straightforwardly by Definition 4.6, since $\bar{F}_2 \in \mathcal{F}_\mathcal{C}$.

Proposition 4.4 $\cong_p^{Sched} = \approx_p^{Sched}$.

Proof.
It follows straightforwardly from Proposition 4.1 and from Theorems 4.2 and 4.3.

4.3 Introduction of a Cost Preorder

In this section, based on the reduction semantics, we define a preorder over networks which allows us to study the performances of different networks, in terms of several kinds of metrics, but exhibiting the same connectivity behaviour.

We first associate a cost function with each reduction as follows:

$\mathbf{Cost}^f(M,N) = f(M,N)$, where $M \to [\![N']\!]_\delta$, with N in the support of $[\![N']\!]_\delta$.

If $e = M_0 \to_{\theta_1} M_1 \to_{\theta_2} M_2 ... \to_{\theta_k} M_k$

is an execution then $\mathbf{Cost}^f(e) = \sum_{i=1}^{k} \mathbf{Cost}(M_{i-1}, M_i)$.

Let H be a set of networks, we denote by $Paths_M^F(H)$ the set of all executions from M ending in H and driven by F which are not prefix of any other execution ending in H. More formally, $Paths_M^F(H) = \{e \in Exec_M^F(H) \mid last(e) \in H$ and $\forall e'$ such that $e <_{prefix} e'$, $e' \notin Paths_M^F(H)\}$.

Now, we are ready to define the average cost of reaching a set of networks H from the initial network M according to the scheduler F.

Definition 4.8 *Let H be a set of networks. The average cost of reaching H from M according to the scheduler F is*

$$\mathbf{Cost}^f{}_M^F(H) = \frac{\sum_{e \in Paths_M^F(H)} \mathbf{Cost}^f(e) \times P_M^F(e)}{\sum_{e \in Paths_M^F(H)} P_M^F(e)}.$$

The average cost is computed by weighting the cost of each execution by its probability according to F and normalized by the overall probability of reaching H. The following definition provides an efficient method to perform both qualitative and quantitative analyses of mobile networks.

Definition 4.9 *Let $\mathcal{H}$ be a countable set of sets of networks and let $\mathcal{F} \subseteq Sched$ a set of schedulers. We say that N is* more efficient than *M with respect to the cost function f, in the context of $\mathcal{H}$ and $\mathcal{F}$ denoted*

$$N \sqsubseteq^f_{\langle \mathcal{H}, \mathcal{F} \rangle} M,$$

if $N \cong_p^{\mathcal{F}} M$ and, for all schedulers $F \in \mathcal{F}_\mathcal{C}$ and for all $H \in \mathcal{H}$, there exists a scheduler $F' \in \mathcal{F}_\mathcal{C}$ such that $\mathbf{Cost}^f{}_N^{F'}(H) \leq \mathbf{Cost}^f{}_M^F(H)$.

4.3.1 Energy Cost Preorder

As already seen in Chapter 2, the performances improving of ad hoc networks can be summarized in finding a good trade-off between energy conservation and connectivity. Since Probabilistic EBUM allows us to compare different networks with the same behaviours, with respect to the transmissions, i.e. to the connectivity, we can define a cost function in terms of energy consumptions.

We define our energy cost function based on the common assumption that the transmission power can be abstracted by taking the transmission radius [91, 12] (a larger transmission power will enlarge the transmission area, while a smaller transmission power will reduce it).

$$\mathsf{e}(M,N) = \begin{cases} r & \text{if } M \to [\![N]\!]_\Delta \text{ with } M \equiv \bar{c}_{L,r}\langle\tilde{v}\rangle.M' \mid M_1 \\ & \text{and } N \equiv M' \mid M_1 \text{for some } c,\ L\,\tilde{v} \text{ and } M_1 \\ 0 & \text{otherwise.} \end{cases}$$

We can use our function of energy cost in many practical cases, since it gives a way to select among several protocols having always the same connectivity behaviours, the most energy efficient one: due to the compositionality of our equivalence relation, this property can be used to replace a network component with a less consuming one, in terms of the given metrics, while maintaining connectivity.

4.4 Analysing the SW-ARQ and GBN-ARQ Protocols

High speed data transmission is rapidly becoming an essential requirement of today's wireless networks. Consequently, adaptive modulation and coding (AMC) techniques are increasingly being used in most of 2.5/3g wireless networks in order to increase the transmission rate. At the same time, a wireless channel is error prone due to fading and other propagation impairments. To address this issue, many control schemes have been proposed. In particular, the automatic repeat request (ARQ)-based error control is considered as very attractive to counteract the residual errors without using costly error correction codes at the physical layer (see, e.g., [96, 53]). However, portable communication devices must rely on batteries with limited energy to conduct communication.

There are three main ARQ protocols: *stop-and-wait (SW)*, *go-back-N (GBN)* and *selective repeat (SR)*. In this section, we use our framework to analyse both SW-ARQ and GBN-ARQ protocols. First, we show that the protocols exhibit the same probabilistic observational behaviour, i.e., they are bisimilar. Then, we compute and compare their energy consumption under various scenarios depending on the stability of the wireless channel.

4.4.1 Protocol description

In the following we briefly recall the salient features of SW-ARQ and GBN-ARQ protocols. In SW-ARQ protocol, the sender pushes a packet into the channel with a delay that is given by ratio between the packet size and the channel bandwidth (pushing time). Once the packet is in the channel we observe two delays: one is that required to reach the destination and the other one is that required for the

acknowledge packet (ACK) to go back to the transmitter. The sum of the two is known as the round trip time. In SW-ARQ protocol the sender sends a packet only once the acknowledge of the previous one has been received. If the round trip time (or an upper bound) is known by the protocol designer, a possible error in the transmission is detected by a timeout mechanism, i.e., if the sender does not receive an ACK from the receiver before a deadline, then it assumes that an error occurred and sends again the same packet. If the round trip time is much higher than the pushing time, then SW-ARQ protocols are very inefficient and exploit only a minimal part of the channel capacity. With respect to SW protocols, GBN takes advantage of the pipelining of the packets, i.e., a sequence of n packets can be sent without receiving any confirmation. This widely used technique is known to highly improve the throughput of the sender, but it is expensive from the energy consumption point of view (see, e.g., [53]) since correctly received packets may be required to be resent. Indeed, once the sender realizes that a packet p has not been received (using a timeout), it has to resend all the packets already sent starting from p. In this way, it can be shown that throughput is really improved and the protocol can use the full channel capacity.

4.4.2 Assumptions on the models

In this case study, we consider a single transmitter node using ARQ-based error recovery protocol to communicate with a receiver node over a wireless channel. Transmissions occur in fixed-size time slots whose size is the time required by the sender to push a packet into the channel. We assume the round trip time to be a multiple of the time slot. For both SW and GBN protocols, the transmitter continuously sends packets until it detects a transmission error. Notice that although in actual implementations of the ARQ protocols errors are usually detected by means of a timeout mechanism, in this context we use negative-acknowledge (NACK) feedbacks which simplify the protocol encodings and are equivalent for the analysis purposes if we assume to know the number of slots that the round trip time consists of. Here, we consider an error-free feedback channel [3] and assume that the ACK or NACK of each transmitted packet arrives at the sender node one slot after the beginning of its transmission slot. Therefore, the feedback of a packet is received exactly after its transmission for the SW-protocol and in case of a failure (NACK), the packet is automatically resent. Instead for the GBN protocol, a feedback for the ith packet arrives exactly after the transmission of the $(i+n-1)$th packet and in case of a failure the transmission restarts from the ith packet. We model both SW-ARQ and GBN-ARQ-based protocols for a communication channel of capacity $n = 3$ in our framework. Observe that in this way we do not take into account the round trip time for SW-ARQ protocols, however this does not affect the analysis that we will carry out later, i.e., the expected energy cost for each packed correctly received.

[3] A very standard assumption [53].

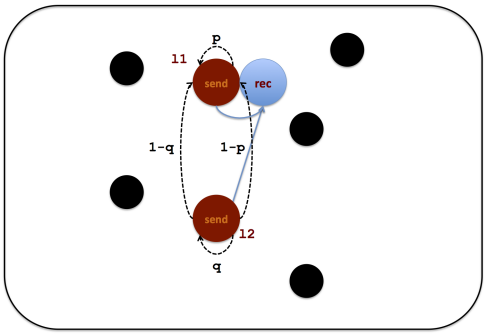


Figure 4.1: Topology of the network and mobility of the sender

We consider a unique static receiver $rec < 0, I^4 >$. We model the transmitter as a mobile node $send\,(< r, J^s >)$ whose reachable locations are l_1, which represents the "good state" of the channel, where the receiver lies within the transmission radius of the channel and l_2 the "bad state", where the destination is no longer reachable (see Figure 4.1). The mobility of the sender is modelled by the two state Markov chain with the following transition probability matrix

$$J^s = \begin{vmatrix} p & 1-p \\ 1-q & q \end{vmatrix},$$

where p and q are the probabilities of the stability of the node in two successive time slots in its good and bad states, respectively.

4.4.3 Modelling the Protocols

In our analysis, we assume that the energy consumption of the feedback messages is negligible. Therefore, they are sent over channels with zero radius. For this reason the static receiver rec is located at l_1, i.e., at the same location of the sender in its good state, so that the feedback will be received with no cost. Note that the sender still transmits over channels with radius r and thus it consumes an amount of energy equal to r for each fired packet.

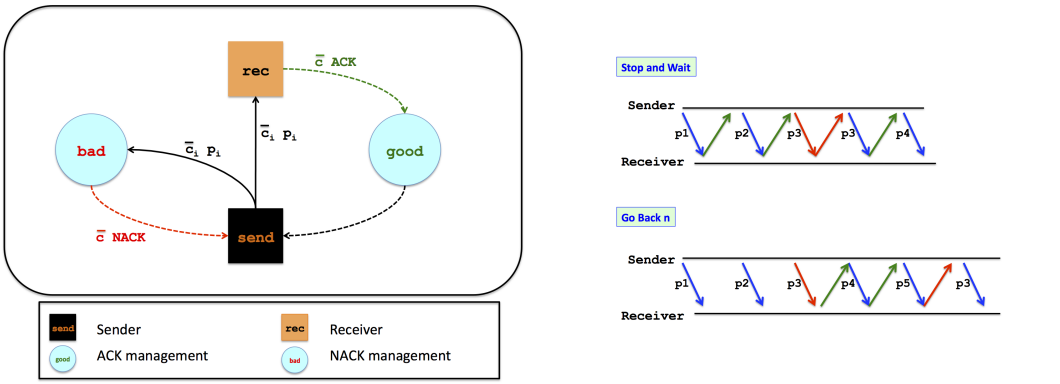
The process executed by rec, the receiver node, is the same for both protocols and modelled as the process

$$REC\langle i \rangle = c^{(i)}(x).\bar{c}_{l_1,0}\langle ACK(i) \rangle.REC\langle i+1 \rangle$$

which, upon receiving packet p_i over the channel $c^{(i)}$, sends $ACK(i)$ over the channel c and waits for the next packet on $c^{(i+1)}$.

For each channel $c^{(i)}$, we use a static auxiliary node $b_i(\langle 0, I \rangle)$ located at l_2, the bad state of the sender, capturing bad transmissions over $c^{(i)}$. It executes the following

[4] i.e. the Identity Matrix.



(a) Structure of the communications (b) Example of GBN and SW behaviour

Figure 4.2: Description and example of the network communications

process which upon receiving packet p_i over the channel $c^{(i)}$, sends $NACK(i)$ over the channel c:

$$BAD\langle i\rangle = c^{(i)}(x).\bar{c}_{l_2,0}\langle NACK(i)\rangle.BAD\langle i\rangle.$$

GBN-ARQ.

Now we introduce the full model of the protocol GBN-ARQ.

We start by modelling its sender node. Recall that, as a simplifying assumption, the channel capacity is 3. It executes the following process:

$$\begin{aligned} GB\langle i\rangle \quad = \quad & \bar{c}^{(i)}_{l_1,r}\langle p_i\rangle.c(x_1)\bar{c}^{(i+1)}_{l_1,r}\langle p_{i+1}\rangle.c(x_2)\bar{c}^{(i+2)}_{l_1,r}\langle p_{i+2}\rangle.c(x_3) \\ & [x_1 = NACK(i)]GB\langle i\rangle, SEND\langle i+3, x_2, x_3\rangle \end{aligned}$$

where the process $SEND$ is defined as follows.

$$SEND\langle i, x, y\rangle = \bar{c}^{(i)}_{l_1,r}\langle p_i\rangle.c(z)[x = NACK(i-2)]GB\langle i-2\rangle, SEND\langle i+1, y, z\rangle.$$

Though that the feedback of a packet is received after the transmission of its two successors, for practical reason, we read a feedback of a packet right after sending it. Indeed, since we do not want feedback to be costly, both sender and receiver must be located at the same place when the feedback is sent. However, the sender node will verify it only after having sent the following two packets.

Recall that the receiver node in our modelling above, reads each packet p_i on its specific channel $c^{(i)}$. Thus, in the GBN, if the transmitter sends p_1 while being in its good state, then moves to bad and sends p_2 and finally moves back to the good state and sends p_3, then the later packet will not be read by the receiver as it is blocked on $c^{(2)}$. Then, the firing on $c^{(3)}$ is lost and this models the fact that packets sent after a bad packet is just a wasting of energy. But since the sender process $GB\langle i\rangle$ is

blocked on the feedback channel c, we introduce a static auxiliary node $lose(\langle 0, I\rangle)$ located at l_1 and executing the process:

$$WAST = \bar{c}_{\emptyset,0}\langle LOST\rangle.WAST$$

SW-ARQ.

Now on to the SW-ARQ-based protocol. This is very simple since it always sends one packet and waits for its feedback. The sender process is defined as follows.

$$SW\langle i\rangle = \bar{c}^{(i)}_{l_1,r}\langle p_i\rangle.c(x)[x = NACK(i)]SW\langle i\rangle, SW\langle i+1\rangle.$$

The full protocols are then modelled as the network

$$GBN = (\nu c^{(1)}, c^{(2)}...)(send[GB\langle 1\rangle]_{l_1} \mid rec[REC\langle 1\rangle]_{l_1} \mid lose[WAST]_{l_1} \mid \prod_{i\geq 1} b_i[BAD\langle i\rangle]_{l_2})$$

and

$$SW = (\nu c^{(1)}, c^{(2)}...)(send[SW\langle 1\rangle]_{l_1} \mid rec[REC\langle 1\rangle]_{l_1} \mid \prod_{i\in I} b_i[BAD\langle i\rangle]_{l_2}).$$

4.4.4 Measuring the Energy Cost of the Protocols.

This section analyzes the energy consumption of the above ARQ-based protocols. In order to compare the observational behaviours of the protocols, we assume that the communications over the feedback channel are observable for any observer node located at l_1. Thus the protocols are equivalent with respect to a set of schedulers $\mathcal{F}$ if for all schedulers F in $\mathcal{F}$ driving one of the protocols, there exists a scheduler F' in $\mathcal{F}$ driving the other one such that both protocols correctly transmit the same packets with the same probabilities. Therefore, we consider the following set of schedulers denoted $\mathcal{F}_{alt}$ which:

1. always alternates between sending packets and node's movement so that at each interaction of the transmitter with the channel, the later can be either good or bad;
2. gives priority to acknowledgment actions (ACK and NACK) to model the standard assumption of an error-free feedback channel;
3. allows interaction with the outside environment only through its observable actions so that we capture exactly the observable behaviour of the protocol.

Notice that the assumptions on the schedulers would be stricter if one desires to carry out an analysis of the throughput. Under these assumptions, we can prove the following results which shows that, the SW-ARQ protocol is more energy efficient of the GBN-ARQ one.

Proposition 4.5 *$GBN \cong_p^{\mathcal{F}_{alt}} SW$.*

Proof.

In order to prove the observational congruence it is enough to prove $GBN \approx_p^{\mathcal{F}_{alt}} SW$. We give here a sketch of the proof. For each sender's window size we will choose, the only observable actions are the acknowledgments sent by the stationary node *rec*. All other actions are silent, since we apply the restriction on each $c^{(i)}$. For all $i \geq 1$ $rec[REC\langle i\rangle]_{l_1}$ sends the acknowledgment $ACK(i)$ if and only if the relative packet p_i has been correctly received, hence, all the executions performed by GBN and SW are of the form:

$$\Longrightarrow \xrightarrow{c!ACK(1)@\{l1\}\triangleleft\{l1\}} \Longrightarrow \xrightarrow{c!ACK(2)@\{l1\}\triangleleft\{l1\}} \Longrightarrow \ldots$$

Since the number of transmissions performed by the sender do not affect the probabilities, the bisimulation between the two different protocols can be proved.

We compare their energy efficiency in the context of the set $\mathcal{H} = \{H_k \,|\, k \geq 1\}$ where H_k means that all the packets up to k have been correctly transmitted and is defined as $H_k = H_k^1 \cup H_K^2$ where

$$H_k^1 = \{M \,|\, M \equiv send[c^{(\bar{k+1})}_{\emptyset,r}\langle p_{k+1}\rangle.P]_{l_1} \mid rec[REC\langle k+1\rangle]_{l_1} \\ \mid loose[WAST]_{l_1} \mid \textstyle\prod_{i\geq 1} b_i[BAD\langle i\rangle]_{l_2}\}$$

for some process P and

$$H_k^2 = \{N \,|\, N \equiv send[SW\langle i+1\rangle]_{l_1} \mid rec[REC\langle k+1\rangle]_{l_1} \mid \prod_{i\in I} b_i[BAD\langle i\rangle]_{l_2}\}.$$

Then, we compute the energy consumption of the protocols assuming that we start by a move action at the good state so that the first message could be lost if it moves to the bad state[5]. The results are summarized in the following propositions and illustrated in Figure 4.3.

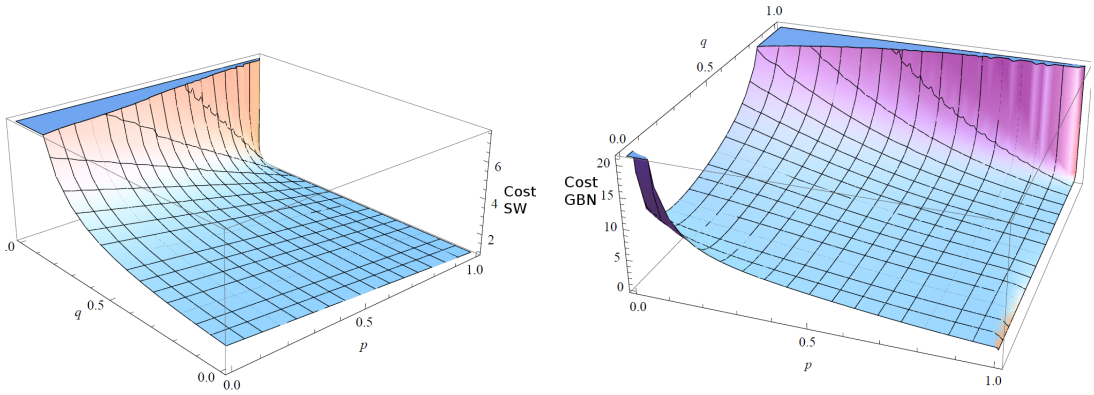
Proposition 4.6 *If $q \neq 1$ then for all $F \in \mathcal{F}_{alt}$*

$$\mathbf{Cost}^{\mathbf{e}F}_{SW}(H_k) = \left(1 + \frac{1-p}{1-q}\right) kr$$

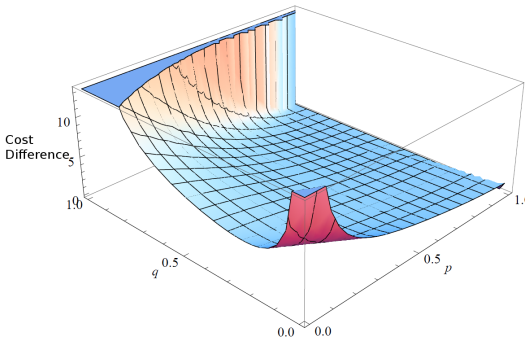
Proposition 4.7 *If $q \neq 1$ then for all $F \in \mathcal{F}_{alt}$*

$$\mathbf{Cost}^{\mathbf{e}F}_{GBN}(H_k) = kr\Big(p + \frac{(p-1)}{(-1+q)(1+p^2-q+q^2-p+2pq)} \\ \cdot \frac{1-2p^2+2p^2q+4q-4q^2+2q^3+2p-6pq+4pq^2}{-p^2+p^2+(-p+pq)(-1+2q)+q(2+-2q+q^2)}\Big)$$

[5]The analysis for the other case is similar.



(a) SW protocol (b) GBN protocol



(c) $\text{cost}_{GBN}(p,q) - \text{cost}_{SW}(p,q)$

Figure 4.3: Energy cost functions for SW and GBN and their comparison.

These results can be derived by applying the Chapman-Kolmogorv's forward equations to compute the probability of consecutive failures in the sending of the same packet. Each of these failures (except the first) causes the waste of a number of sent packets equals to the window size. It can be observed that the number of wasted windows has a geometric distribution. Then, the mean of total packets sent to obtain a success, can be straightforwardly derived.

To conclude this section, we note that while both protocols increasingly enjoy bad performance in term of energy consumption when the channel deteriorates, i.e., when q is increasing (see Figures 4.3-(a) and 4.3-(b)), the GBN protocol deteriorates faster. Indeed, as illustrated by Figure 4.3-(c) as the channel deteriorates the additional energy required by GBN protocol to correctly transmit the same number of packets increases to infinite. Thus, the gain of having a high throughput results in a very high energy consumption.

Finally we can conclude that the GBN protocol is much more energy consuming than SW.

Theorem 4.4 *It holds that* $SW \sqsubseteq^{\mathsf{e}}_{\langle \mathcal{H}, \mathcal{F}_{alt} \rangle} GBN$.

Proof.

The proof follows straightforwardly from Propositions 4.5, 4.6 and 4.7.

4.5 Analysis of a location based routing protocol

In Chapter 2 we discussed how important is the choice of a good routing protocol in the design of an ad hoc network, due to the dynamic nature of the network, and to the limited power sources of its nodes.

As a case study, in order to show how to use the probabilistic EBUM calculus to face the problem of choosing the best routing protocol for a given network, we analyze the performance of LAR (Location Aided Routing)[47] : a location based strategy which improves the traditional flooding algorithm exploiting the possibility for the network nodes, to know their geographical locations, due to common existing technologies such as GPS (Global Position System)[45].

4.5.1 Protocol Description

Informally, location based routing algorithms assume that each node of the wireless network is aware of its own location thanks to a Global Positioning System (GPS) device or thanks to other mechanisms such as the knowledge of the distances between its location at a given epoch and some other static stations. The main idea behind the development of these algorithms is that in very large mobile networks using a flooding policy in an AODV style [79] may turn out to be very expensive in terms of number of sent packets and hence of energy consumption. Location based routing algorithms aim at controlling the flooding by guessing the possible location of the destination node. The guess can be driven by several factors, such as the knowledge of the destination node's location in the latest communication joint with some assumptions on the node's maximum movement speed. In this section, we show our framework at work on a simplified version of the LAR protocol, and prove that, under mild assumptions on the node mobility, it is equivalent to the flooding algorithm in terms of the probability of discovering a path. Obviously, it is not possible to establish a general energy-aware preorder between the two protocols, but this can be done (algorithmically) for specific instances of wireless networks.

4.5.2 Simple flooding: description

Protocol LAR extends the route discovery based on flooding by exploiting information about locations within the network. The simplest route discovery algorithm based on flooding consists of three simple packets: request, reply and error [87], which are forwarded within the network. They are structured as follows:

- *Route Request* packet (RREQ) has the form:

$$(S, Bid, D, \texttt{seq\#}_S, \texttt{hop\_counter}),$$

where S is the permanent source address, Bid is the Request Id (unique identifier), D is the permanent address of the destination, $\texttt{seq\#}_S$ denotes the sequence number of the source, and `hop_counter` is the number of hops to reach the destination (which is initially set to 0 and then incremented at each request forwarding).

- *Route Reply* packet (RREP) has the form:

$$(S, Bid, D, \texttt{seq\#}_D, \texttt{hop\_counter}, \texttt{Lifetime}),$$

where S, Bid and D are as above, $\texttt{seq\#}_D$ is the sequence number of the destination, `hop_counter` is the number of hops to reach the destination and `Lifetime` is the duration of the route validity.

- *Route Error* packet (RERR) has the form:

$$(S, D, \texttt{seq\#}_D),$$

where S D and $\texttt{seq\#}_D$ are as in the previous case.

Normally, a node looking for a path to a given destination, simply broadcasts a RREQ within the network. Having sent the packet, the node sets a timeout to manage the cases when the destination does not receive the request, or the reply packet is lost. If the timeout expires, the node broadcasts a new request, using a different sequence number to avoid loops. When the destination finally receives the RREQ, it immediately sends back the corresponding RREP, using unicast communication, i.e., each intermediate node forwards the RREP using the information in its routing table. When, during a communication, a node realizes that a link failed, it broadcasts a RERR and each node will update its routing table.

4.5.3 Exploiting location data: the LAR policy

LAR extends the simple flooding algorithm described above by directing the propagation of the discovery packets to a particular network area based on the expected locations of the destination node. In the LAR specification, the *Expected Zone* is the network area where the source expects to find the destination node. This is determined by means of the information that the source has previously retrieved about the destination location. In practice, if node S knows that destination node D was located at location l_1 at epoch t, and it moves with a speed v, then it can calculate the circle area centered at l_1, with radius $v(t' - t)$, where t' is the current epoch. If S does not know anything about D, then the *Expected Zone* coincides with the entire network.

The *Request Zone* is the network area that the source defines to specify a candidate route to the destination. An intermediate node forwards a route request only if it is within the *Request Zone*. There are different ways to define a *Request Zone*:

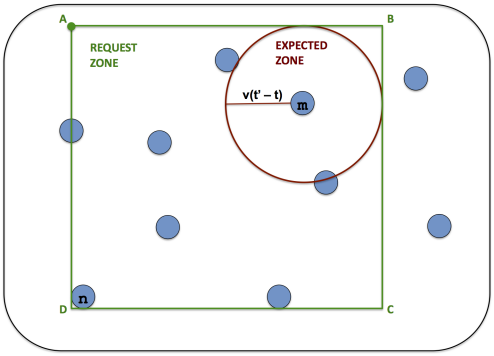


Figure 4.4: *Expected* and *Request Zones* in the LAR protocol

(a) A simple *route request* packet (b) A*route request* packet with location information

Figure 4.5: Different *route request* packets of LAR - *Scheme 1*

usually choosing a smaller area reduces the message overhead (because it reduces the number of forwarded packets), while a larger area reduces the latency of the route discovery because the network finds a path with higher probability.

LAR behaves similarly to the simple flooding, with the difference that a node that is not inside the *Request Zone* does not forward the request. LAR can use two different policies for determining the *Request Zone*: we focus on the first such policy, known as *LAR Scheme 1*.

LAR Scheme 1 uses a rectangular *Request Zone*, depending on the position of the source with respect to the *Expected Zone*. In particular, the *Request Zone* will be the smallest rectangle containing both the *Expected Zone* and the position of the source node, as shown in Figure 4.4.

Let (X_S, Y_S) and (X_D, Y_D) the Cartesian coordinates of S and D, and R the radius of the *Expected Zone*. If S is outside the *Expected Zone*, the coordinates of the rectangle area are:

A: $\rightarrow (X_S, Y_D + R)$ B: $\rightarrow (X_D + R, Y_D + R)$

C: $\rightarrow (X_D + R, Y_S)$ D: $\rightarrow (X_S, Y_S)$

If S falls inside the *Expected Zone*, the coordinates of the rectangle area are:

A: $\rightarrow (X_D - R, Y_D + R)$ B:$\rightarrow (X_D + R, Y_D + R)$

C: $\rightarrow (X_D - R, Y_D - R)$ D: $\rightarrow (X_D + R, Y_D - R)$

When S broadcasts its request, it includes the coordinates of the *Request Zone* rectangle (see Figure 4.5). Once an intermediate node receives a RREQ, this is discarded if its location does not fall within the rectangle specified in the packet.

To take into account the location measuring error, a positive value e is added to the radius of the *Expected Zone*, consequently enlarging also the *Request Zone*.

4.5.4 Modelling the network

We encode the simple flooding and the LAR protocols using PEBUM. We abstract out all details about how the *Expected Zone* and *Request Zone* are determined, by using pre-defined functions that are implemented according to the specifications of LAR Scheme 1.

We first introduce some auxiliary functions to simplify the protocol specification:

- `gps` : returns the actual geographical position of the node executing the process (by means, e.g., of GPS technology);
- `dist`(l) : returns the distance from location l and the location of the node executing the process;
- `self` : returns the name (permanent address) of the node executing the process;
- `geq`(k, l) = `true` if $k \geq l$, `false` otherwise;
- `inside`(s, A) = `true` if $s \in A$, `false` otherwise;
- `unable`(n) = refreshes the route table, removing the existing path to n;
- `find_path`(n) = `true` if there exists a valid path for n in the route table of the node executing the process;
- `newBid`: generates a new unique Bid identifier for a packet;
- `lastBid`: returns the latest generated Bid identifier;
- `control`(Bid) = `true` if the request associated with Bid has been already received by the node executing the process.

Each node maintains a *routing table* containing information about the paths to the other nodes in the network. Each entry has the following form:

$$(d, \texttt{seq\#}_d, \texttt{next\_hop}_d, \texttt{hopcount}_d, \texttt{loc}_d, \texttt{v}_d, \texttt{timeout}) ,$$

where d is the destination name, $\texttt{seq\#}_d$ is the sequence number of the route to d, $\texttt{next\_hop}_d$ is the name of the next node to reach d, $\texttt{hopcount}_d$ is the number of hops to reach d, $\texttt{loc}_d$ is the last location known for d, $\texttt{v}_d$ is the average speed of d and `timeout` is the timeout associated with the entry.

Each node is also associated with a *request table* containing the list of all the requests already processed by the node; this is needed to prevent loops during the route request forwarding. For brevity, we model a network in which all the nodes use a common transmission radius r.

Let's now consider $N = (\nu c)(n[P]_l \mid \prod_{i \in I} n_i[Q\_SIMPLE]_{l_i})$ where a node n broadcasts a route request using the simple flooding algorithm to find a path to

m in the network $\prod_{i\in I} n_i$, and $M = (\nu c)(n[P]_l \mid \prod_{i\in I} n_i[Q\_LAR1]_{l_i})$ which is the same network but with nodes in I using the LAR protocol (Scheme 1) instead of the simple flooding algorithm.

The process executed by node n simply broadcasts a RREQ packet for node m and waits for a RREP packet until a timeout expires. The timeout is modelled using the operator $\oplus$ that behaves as the non-deterministic choice and can be implemented in our calculus by means of the parallel composition is the standard way. In case of timeout, a new RREQ is sent.

$$\begin{aligned} P &= \bar{c}_{\emptyset,r}\langle(\texttt{rreq}, n, \texttt{newBid}, m, \texttt{Request\_Zone}, \texttt{seq\#}_n, 0)\rangle.P' \\ P' &= P \oplus c(x_1,x_2,x_3,x_4,x_5,x_6,x_7).[x_1 = \texttt{rrep}][x_2 = n][x_3 = \texttt{lastBid}] \\ &\quad [x_4 = m][\texttt{geq}(\texttt{hop\_count}_m, x_7)]\bar{\texttt{ok}}_{\texttt{gps},r}\langle\texttt{route\_found}\rangle\,, P' \end{aligned}$$

where $m = n_i$ for some $i \in I$, and $x_7 = \texttt{hop\_count}$ in the RREP packet received. Basically, once a route is found, n broadcasts on channel $\texttt{ok}$ a packet that signals this event. Therefore, we consider that the two networks are probabilistic equivalent with respect to their ability to find a route to m if we observe this transmission with the same probability. Notice that, the output on channel c will not be observed by any location because we want to allow the route discovery packets used in the two networks to be arbitrary different.

Hereafter, we use $X \in \{SIMPLE, LAR1\}$ to denote the simple flooding or LAR Scheme 1. The $RREQ\_SIMPLE$ and the $RREQ\_LAR1$ subprocess are defined as shown by Table 4.3.

In order to compare the behaviour of the protocols, we focus our attention on the following restricted set $\mathcal{F} \subseteq Sched$ of admissible schedulers:

1. the timeout for a RREQ identified by Bid occurs when in the networks there are no packets related to Bid;
2. nodes' movements are allowed at least every time a timeout occurs;

Condition 1 on $\mathcal{F}$ is a requirement inherited by the protocol design; the timeout is usually set by knowing the physical dimension of the network. Roughly speaking, we aim at preventing that in the analysis we consider unrealistic schedulers that always choose the timeout option too quickly and hence a route to the destination is never found and those schedulers that wait for an answer indefinitely long. Condition 2 is needed because we do not want to consider those schedulers that never allow for node movements.

Proposition 4.8 (Energy Efficiency of LAR) *Let M, N, $\mathcal{M} = \{\bar{M} : M \rightarrow^* \bar{M}\} \cup \{\bar{N} : N \rightarrow^* \bar{N}\}$ and $\mathcal{F}$ as above. A sufficient condition for*

$$M \sqsubseteq^{\mathsf{e}}_{\langle\mathcal{M},\mathcal{F}\rangle} N.$$

is that the Markov chains $\mathbf{J}^{n_i}$ associated with the mobile nodes n_i ($i \in I$) are ergodic.

$Q\_X = c(x_1, x_2, x_3, x_4, x_5, x_6, x_7).$
$[x_1 = \texttt{rreq}]([\texttt{control}(x_3) = \texttt{false}]([x_4 = \texttt{self}]$
$\bar{c}_{\texttt{next\_hop}_{x_2},r}\langle(\texttt{rrep}, s, Bid, d, \texttt{seq\#}_s, \texttt{hop\_counter})\rangle.Q\_X, RREQ\_X\langle\tilde{x}\rangle), Q\_X),$
$[x_1 = \texttt{rrep}]([\texttt{x}_2 = \texttt{self}]\texttt{out}\langle ud_{\texttt{gps},r}, x_2, x_3, x_4, x_5, x_6, x_7\rangle,$
$\bar{c}_{\texttt{next\_hop}_{x_2},r}\langle(\texttt{rrep}, s, Bid, d, \texttt{seq\#}_s, \texttt{hop\_counter})\rangle.Q\_X),$
$[x_1 = \texttt{rerr}]\texttt{unable}(x_4).Q\_X, Q\_X$

$RREQ\_SIMPLE\langle(\texttt{rreq}, s, Bid, d, \texttt{seq\#}_s, \texttt{hop\_counter})\rangle =$
$[\texttt{find\_path}(d) = \texttt{true}].$
$\bar{c}_{\texttt{next\_hop}_d,r}\langle(\texttt{rrep}, s, Bid, d, \texttt{seq\#}_d, \texttt{hop\_counter} + 1 + \texttt{hopcount}_d, \texttt{timeout})\rangle,$
$\bar{c}_{\texttt{Request\_Zone},r}\langle(\texttt{rreq}, s, Bid, d, \texttt{seq\#}_s, (\texttt{hop\_counter}) + 1)\rangle.Q\_SIMPLE$

$RREQ\_LAR1\langle(\texttt{rreq}, s, Bid, d, \texttt{Request\_Zone}, \texttt{seq\#}_s, \texttt{hop\_counter})\rangle =$
$([\texttt{inside}(\texttt{gps}, \texttt{Request\_Zone}) = true]($
$[\texttt{find\_path}(d) = \texttt{true}]$
$\bar{c}_{\texttt{next\_hop}_d,r}\langle(\texttt{rrep}, s, Bid, d, \texttt{seq\#}_d, \texttt{hop\_counter} + 1 + \texttt{hopcount}_d, \texttt{timeout})\rangle,$
$\bar{c}_{\texttt{Request\_Zone},r}\langle(\texttt{rreq}, s, Bid, d, \texttt{Request\_Zone}, \texttt{seq\#}_s, (\texttt{hop\_counter}) + 1)\rangle))).Q\_LAR1$

Table 4.3: Process specifications used in the case study of Section 4.5

Proof.
We have to prove that:

1. $N \cong_p^{\mathcal{F}} M$
2. for all schedulers $F \in \mathcal{F}_{\mathcal{C}}$ and for all $H \in \mathcal{M}$, there exists a scheduler $F' \in \mathcal{F}_{\mathcal{C}}$ such that $\mathbf{Cost}^{\mathbf{e}}{}_M^{F'}(H) \leq \mathbf{Cost}^{\mathbf{e}}{}_N^{F}(H)$.

1. It is sufficient to prove $M \approx_p^{\mathcal{F}} N$. We have to find a relation containing the pair (M, N) that is a probabilistic bisimulation relative to $\mathcal{F}$. Let consider $Z_i \in \{RREQ, Q\}$, $\bar{P} \in \{P' : P \rightarrow^* P'\}$ and
$\mathcal{R} = \{(n[\bar{P}]_l \mid \prod_{i\in I} n_i[Z_i\_SIMPLE]_{l_i}, n[\bar{P}]_l \mid \prod_{i\in I} n_i[Z_i\_LAR1]_{l_i}) :$
$N \rightarrow^* n[\bar{P}]_l \mid \prod_{i\in I} n_i[Z_i\_SIMPLE]_{l_i}\}.$
In order to prove that $\mathcal{R} \subseteq \approx_p^{\mathcal{F}}$ we have to show that, for all pairs $(\bar{N}, \bar{M}) \in \mathcal{R}$ and for all schedulers $F \in \hat{\mathcal{F}}_{\mathcal{C}}$ there exists a scheduler $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that for all α and for all classes $\mathcal{C}$ in $\mathcal{N}/\mathcal{R}$ it holds:

1. if $\alpha \neq c?\tilde{v}@l$ then $Prob_{\bar{N}}^{F}(\xrightarrow{\alpha}, \mathcal{C}) = Prob_{\bar{M}}^{F'}(\stackrel{\hat{\alpha}}{\Longrightarrow} \mathcal{C})$;
2. if $\alpha = c?\tilde{v}@l$ then either $Prob_{\bar{N}}^{F}(\xrightarrow{\alpha}, \mathcal{C}) = Prob_{\bar{M}}^{F'}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C})$ or
$$Prob_{\bar{N}}^{F}(\xrightarrow{\alpha}, \mathcal{C}) = Prob_{\bar{M}}^{F'}(\Longrightarrow, \mathcal{C}).$$

We start from τ actions and consider $\bar{N} \xrightarrow{\tau} [\![\bar{N}']\!]_\theta$. Then, $\forall \mathcal{C} \in \mathcal{N}/\mathcal{R}$, we have: $Prob_{\bar{N}}(\xrightarrow{\tau}, \mathcal{C}) = \sum_{\hat{N}\in spt([\![\bar{N}']\!]_\theta)\cap\mathcal{C}} [\![\bar{N}']\!]_\theta(\hat{N})$.

If the action has been determined by the application of rule (Move) we are done, because, for each pair $(\bar{N}, \bar{M}) \in \mathcal{R}$, $\bar{M}$ can perform exactly the same movements as $\bar{N}$, hence there will exists $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that: $Prob^{F}_{\bar{N}}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F'}_{\bar{M}}(\xrightarrow{\tau} \mathcal{C})$, and we are done.

If the action is the result of the application of rule (Lose), by applying rule (Bcast) backwardly we get $\bar{N} \xrightarrow{c_K!\tilde{v}[l,r]} [\![\bar{N}']\!]_{\Delta}$.

If $l \in$ `Request_Zone` then we are done, because, by the analysis of the process P_LAR1 with respect to P_SIMPLE we realize that the protocol packets are forwarded exactly in the same way inside the `RequestZone`.

If $l \notin$ `Request_Zone`, then we are sure that $\bar{M} \not\xrightarrow{c_K!\tilde{v}[l,r]}$ because the routing protocol packets are forwarded only inside the `Request_Zone`. However, this does not means that $\bar{M}$ will not reach an equivalent state with the same probability. By the initial hypothesis that all the Markov matrices are ergodic, $\bar{M}$ can enter the `Request_Zone` with probability 1, send the message, and come back to the previous location again with probability 1, and we get

$$Prob^{F}_{\bar{N}}(\xrightarrow{\tau}, \mathcal{C}) = 1 = Prob^{F}_{\bar{M}}(\Longrightarrow, \mathcal{C})$$

as required.

As concerns the input and the observable actions the proof is trivial, since the input actions are the same for both protocols, and we applied the restriction to channel c, hence the only observable output is the transmission of `route_found` through the channel `ok` by the node n, which behaves in the same way for both protocols.

2. The proof of this second point is easy since we are sure that M performs at least exactly the same output actions as N (the borderline case is when all nodes are inside the `Expected_Zone`, i.e., the request forwarding are the same for both networks). Otherwise, we are sure that M performs less output actions then N, and, since all the nodes use exactly the same transmission radius for their communications, the Energy Cost will be proportional to the number of the outputs.

4.6 Conclusions

In this chapter we introduced a new version of the EBUM calculus, where probabilistic and non-determinstic aspects co-exist. This probabilistic calculus allows us to make both qualitative and quantitative analysis, and to compare ad hoc connectivity protocols having the same observational behaviours but different performances in terms of several metrics, as e.g., energy consumption and throughput.

However, this calculus is not able to cover the collisions occurring during the networks transmissions, which can be due to both traffic congestion or attacks of malicious nodes intended to disturb the network communications. Mobile ad hoc networks are particularly vulnerable to these kinds of interference, due to the nature

of the channels used (radio-frequencies), and to the absence of a fixed infrastructure to control the flow of communication inside the network.

In Chapter 5 we introduce a new version of the EBUM calculus where, in addition to the introduction of probability distributions to model nodes mobility, we define a new semantic, where input and output are modelled as non-atomic actions. The new version of the calculus allows us to capture the collisions which may occur when the transmission areas of different senders overlap.

5

Interference-sensitive Preorders for Mobile Ad hoc Networks

5.1 Introduction

In this chapter we introduce a new extension of the EBUM calculus [11, 24, 10] aimed at modelling the collisions which may occur during the network communications. Like its predecessor [22, 21], it deals with both nondeterministic and probabilistic choices. Its semantics is inspired by Segala's probabilistic automata [83] driven by schedulers to resolve the nondeterministic choice among the probability distributions over target states.

The peculiarity of this new version of the EBUM calculus is that, due to the non-atomic nature of input and output actions, it can capture the case where multiple nodes may simultaneously transmit along the same channel, over overlapping areas: as in [51], the calculus provides an explicit representation of the collisions that may occur at the receivers which lie within the transmission range of different senders.

5.2 The Calculus

5.2.1 Syntax

The syntax of our calculus is the same as the one introduced in chapter 3, (see Table 3.1) except for the input and output actions of the inactive processes, reported in Table 5.1: $\mathtt{in}(c, \tilde{x}).P$ is ready to listen to a transmission, while $\mathtt{out}\langle c_{L,r}, \tilde{w}\rangle.P$ is ready to transmit.

Two further process forms arise as a result of reduction, due to our characterization of communication. In particular, processes that are ready to send or receive evolve into active senders and receivers:

P, Q	::=	$\ldots$	As in Table 5.1
	$\mid$	$c(\tilde{x}).P$	Active input
	$\mid$	$\bar{c}_{L,r}\langle\tilde{w}\rangle.P$	Active output

Table 5.1: Syntax

Processes		
P,Q,R	$::= \mathbf{0}$	Inactive process
	$\mid \mathtt{in}(c, \tilde{x}).P$	Input
	$\mid \mathtt{out}\langle c_{L,r}, \tilde{w}\rangle.P$	Output
	$\mid [w_1 = w_2]P, Q$	Matching
	$\mid A\langle \tilde{w}\rangle$	Recursion

Here, $c(\tilde{x}).P$ is actively receiving a tuple $\tilde{w}$ of (closed) values via channel c and continues as $P\{\tilde{w}/\tilde{x}\}$, i.e., as P with $\tilde{w}$ substituted for $\tilde{x}$ (where $|\tilde{x}| = |\tilde{w}|$). Dually, $\bar{c}_{L,r}\langle\tilde{w}\rangle.P$ is transmitting a tuple of values $\tilde{w}$ via channel c and then continues as P. We say that a process P is *active* if it is in prefix form, with the prefix denoting an active input or output action. Predicate $\mathtt{Active}(P)$ is $\mathtt{true}$ when P is active, and $\mathtt{A}(M)$ denotes the network composed of all the active nodes in M, i.e., all nodes $n[P]_l$ in M with P active.

5.2.2 Reduction Semantics

The dynamics of the calculus is specified by the *probabilistic reduction relation* ($\rightarrow$), described in Table 5.2: it takes the form $M \rightarrow [\![M']\!]_\theta$, denoting a transition that leaves from M and leads to a probability distribution $[\![M']\!]_\theta$. As usually it relies on the structural congruence relation, defined in Table 3.2. .

The synchronization over a wireless channel is described by the two rules (R-Bgn-Bcast) and (R-End-Bcast). (R-Bgn-Bcast) models the beginning of a transmission, with node n transiting from ready to active state to transmit a tuple $\tilde{v}$ of messages on the channel c with radius r. The state change in n may cause a collision, which the rule captures as follows. The premise of the rule describes a situation in which, for $i \in I \cup K$ nodes n_i are actively involved in a synchronization, while node n and the nodes in the set J are in (output and input, respectively) ready state. Given that all the active transmitters are out of n's range (because $d(l, l_i) > r_i \forall i \in I$), n transits into active state: this awakes the n_j with $j \in J$, as they are now in range of an active transmitter, and at the same time causes a collision at the nodes in the set K, which also are in range and were already active on input: as a result, $\forall i \in K$, n_i exits its active state, receiving the error signal $\perp$. All the remaining active receivers that do not sense a collision, and are in the range of an active sender may conclude the synchronisation, as described by the (R-End-Bcast) rule.

(R-Bgn-Bcast)
$$\frac{\forall i \in I.d(l,l_i) > r_i \,\forall i \in I\, \forall j \in J.d(l_i,l_j) > r_i \,\forall h \in (J \cup K).d(l,l_h) \leq r}{n[\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_l \mid M \rightarrow [\![n[\bar{c}_{L,r}\langle \tilde{v}\rangle.P]_l \mid M']\!]_\Delta}$$
where $M \equiv \prod_{i\in I} n_i[\bar{c}_{L_i,r_i}\langle \tilde{v_i}\rangle.P_i]_{l_i} \mid \prod_{j\in J} n_j[\mathtt{in}(c,\tilde{x_j}).P_j]_{l_j} \mid \prod_{k\in K} n_k[c(\tilde{x_k}).P_k]_{l_k}$,
$M' \equiv \prod_{i\in I} n_i[\bar{c}_{L_i,r_i}\langle \tilde{v_i}\rangle.P_i]_{l_i} \mid \prod_{j\in J} n_j[c(\tilde{x_j}).P_j]_{l_j} \mid \prod_{k\in K} n_k[P_k\{\bot/\tilde{x_i}\}]_{l_k}$

(R-End-Bcast) $$\frac{\forall i \in I.d(l,l_i) \leq r}{n[\bar{c}_{L,r}\langle \tilde{v}\rangle.P]_l \mid \prod_{i\in I} n_i[c(\tilde{x_i}).P_i]_{l_i} \rightarrow [\![n[P]_l \mid \prod_{i\in I} n_i[P_i\{\tilde{v}/\tilde{x_i}\}]_{l_i}]\!]_\Delta}$$

(R-Res) $$\frac{M \rightarrow [\![M']\!]_\theta}{(\nu c)M \rightarrow [\![(\nu c)M']\!]_\theta}$$ (R-Move) $$\frac{\mathtt{Active}(P) = \mathtt{false}}{n[P]_l \rightarrow [\![n[P]_l]\!]_{\mu_l^n}}$$

(R-Par) $$\frac{M \rightarrow [\![M']\!]_\theta}{M|N \rightarrow [\![M'|N]\!]_\theta}$$ (R-Struct) $$\frac{N \equiv M \quad M \rightarrow [\![M']\!]_\theta \quad M' \equiv N'}{N \rightarrow [\![N']\!]_\theta}$$

Table 5.2: Reduction Semantics

Example 5.1 *(Interference) Consider a network*

$$M = n_1[\mathtt{out}\langle c_{L,r_1}, \tilde{v_1}\rangle.P_1]_{l_1} \mid n_2[\mathtt{out}\langle c_{L,r_2}, \tilde{v_2}\rangle.P_2]_{l_2} \mid m[\mathtt{in}(c,\tilde{x}).P_3]_k$$

consisting of two mobile sender nodes, n_1 and n_2, communicating with a static receiver node m, where the two sender nodes are not within the radius of each other, i.e., $d(l_1,l_2) > \max(r_1,r_2)$, and they are both able to reach the receiver, i.e., $d(l_1,k) \leq r_1$ and $d(l_2,k) \leq r_2$. Then the following reductions, obtained by applying rule (R-Bgn-Bcast), lead to a state where an interference is caused at the receiver node:

$$M \;\rightarrow\; [\![n_1[\bar{c}_{L,r_1}\langle \tilde{v_1}\rangle.P_1]_{l_1} \mid n_2[\mathtt{out}\langle c_{L,r_2}, \tilde{v_2}\rangle.P_2]_{l_2} \mid m[c(\tilde{x}).P_3]_k]\!]_\Delta$$

and if $M' = n_1[\bar{c}_{L,r_1}\langle \tilde{v_1}\rangle.P_1]_{l_1} \mid n_2[\mathtt{out}\langle c_{L,r_2}, \tilde{v_2}\rangle.P_2]_{l_2} \mid m[c(\tilde{x}).P_3]_k$ then

$$M' \;\rightarrow\; [\![n_1[\bar{c}_{L,r_1}\langle \tilde{v_1}\rangle.P_1]_{l_1} \mid n_2[\bar{c}_{L,r_2}\langle \tilde{v_2}\rangle.P_2]_{l_2} \mid m[P_3\{\bot/\tilde{x}\}]_k]\!]_\Delta.$$

The first sender node starts broadcasting on the channel c causing the receiver to become active. Then the second sender being too far away from n_1 to notice that the channel is occupied starts broadcasting on the same channel and hence causes an interference at the receiver side.

Rule (R-Move) describes node mobility, and it is the same as the one described in the previous chapter.

All the remaining rules are standard, but a further remark is in order about the (R-Par) rule and its interaction with the rules that govern synchronization. In fact, such interactions may give rise to inconsistent network configurations. To see that, observe that an application of the (R-Par) rule may cause messages to be lost by active receivers located within the range of an active sender, even when there is no interference. Similarly, an application of (R-Par) may exclude any set of active sender and/or receiver from a synchronization: in both cases, the network is left in an inconsistent state, with active senders (dually receivers) and no receiver (sender) in range.

Example 5.2 *(Inconsitent networks) Consider again the network of the previous example where now the two sender nodes are within the radius of each other, that is* $d(l_1, l_2) \leq \min(r_1, r_2)$*. By applying rule (R-Bgn-Bcast) we obtain*

$$M \;\rightarrow\; [\![n_1[\mathtt{out}\langle c_{L,r_1}, \tilde{v_1}\rangle.P_1]_{l_1} \mid n_2[\bar{c}_{L,r_2}\langle \tilde{v_2}\rangle.P_2]_{l_2} \mid m[c(\tilde{x}).P_3]_k]\!]_\Delta.$$

Now let $M' = n_1[\mathtt{out}\langle c_{L,r_1}, \tilde{v_1}\rangle.P_1]_{l_1} \mid n_2[\bar{c}_{L,r_2}\langle \tilde{v_2}\rangle.P_2]_{l_2} \mid m[c(\tilde{x}).P_3]_k$*. The following reduction obtained by applying rule (R-Par)*

$$M' \;\rightarrow\; [\![n_1[\bar{c}_{L,r_1}\langle \tilde{v_1}\rangle.P_1]_{l_1} \mid n_2[\bar{c}_{L,r_2}\langle \tilde{v_2}\rangle.P_2]_{l_2} \mid m[c(\tilde{x}).P_3]_k]\!]_\Delta$$

leads to an inconsitent state where both sender nodes are broadcasting on the same channel while being within a reachable distance of each other. Similarly, consider the following application of rule (R-Bgn-Bcast):

$$M \;\rightarrow\; [\![n_1[\bar{c}_{L,r_1}\langle \tilde{v_1}\rangle.P_1]_{l_1} \mid n_2[\mathtt{out}\langle c_{L,r_2}, \tilde{v_2}\rangle.P_2]_{l_2} \mid m[c(\tilde{x}).P_3]_k]\!]_\Delta$$

If $M'' = n_1[\bar{c}_{L,r_1}\langle \tilde{v_1}\rangle.P_1]_{l_1} \mid n_2[\mathtt{out}\langle c_{L,r_2}, \tilde{v_2}\rangle.P_2]_{l_2} \mid m[c(\tilde{x}).P_3]_k$ *then by an application of rule (R-Par) we obtain*

$$M'' \;\rightarrow\; [\![n_1[P_1]_{l_1} \mid n_2[\mathtt{out}\langle c_{L,r_2}, \tilde{v_2}\rangle.P_2]_{l_2} \mid m[c(\tilde{x}).P_3]_k]\!]_\Delta$$

leading to an inconsitent state where m *is activelly receiving a message while there is no active sender.*

While it would be possible to rectify the problem by including conditions to exclude critical pairs for the (R-Par) and synchronization rules, it is technically more convenient to simply disregard any undesired reduction. This is achieved in our definition of observational semantics (to be discussed shortly) by resorting to the notion of "admissible scheduler" to guide the dynamics of networks through "well-formed" executions.

5.2.3 Observational Semantics

As for its predecessor, we introduce the notion of barb denoting an observable transmission with a certain probability according to a fixed scheduler. Schedulers are total functions discribed in chapter 4 (see the Definition 4.1).

As we anticipated, we restrict the class of all networks (resp. executions) to the class of well-formed networks (resp. executions) where, (1) a transmitter, before transiting in active state checks that, locally, the communication channel is not presently busy with other transmissions, and (2) each active receiver in the network is in the transmission cell of exactly one transmitter. In order to restrict the set of all executions to the set of well-formed executions, we restrict the set of all schedulers to the following set of *admissible* schedulers. For this purpose, we remind the reader that $\mathtt{A}(M)$ is network composed only by its currently active nodes in M and introduce the auxiliary operator $\mathtt{Top}(\cdot)$ over networks. A channel c is at the top level of a network M, denoted $c \in \mathtt{Top}(M)$, if $M \equiv (\nu \tilde{d})(n[P]_l \mid N)$ and P is of the form $\mathtt{in}(c, \tilde{x}).Q$; $c(\tilde{x}).Q$; $\mathtt{out}\langle c_{L,r}, \tilde{w}\rangle.Q$; or $\bar{c}_{L,r}\langle \tilde{w}\rangle.Q$.

Definition 5.1 (Well-formed network) *A network M is* well-formed *if either $\mathtt{A}(M) \equiv \emptyset$ or $M \equiv M_1 \mid N$ such that:*

1. $\mathtt{A}(M1) \equiv (\nu \tilde{d})\big(\prod_{i\in I} n_i[\bar{c}_{L_i,r_i}\langle \tilde{v}_i\rangle.P_i]_{l_i} \mid \prod_{j\in J} n_j[c(\tilde{x}_j).P_j]_{l_j}\big)$ *and the following conditions hold:*
   - $\forall i, i' \in I . d(l_i, l_{i'}) > max(r_i, r_{i'})$,
   - $\forall j \in J . \exists! i \in I$ *such that* $d(l_j, l_i) \leq r_i$,
2. $c \notin \mathtt{Top}(N)$, *and N is well-formed.*

Definition 5.2 (Admissible scheduler) *A scheduler F is* admissible *if for all executions e and for all networks M in the support of $F(e)$, M is well-formed.*

We shall denote the set of admissible schedulers by *Sched* and, unless otherwise stated, assume that all schedulers are admissible.

As we have done in the Chapter 4, we want to define a relation among networks, related to a specific set of schedulers [1].

Definition 5.3 *Given an admissible scheduler $F \in Sched$, we denote by $F_{\mathcal{C}}$ the set of admissible schedulers F' such that $\forall M_0$, $\forall e \in Exec^F_{M_0}$ of the form*

$e = M_0 \rightarrow_{\theta_1} M_1 \rightarrow_{\theta_2} M_2 ... \rightarrow_{\theta_h} M_h$,

$\forall$ context $C_0[\cdot]$ and $\forall e' \in Exec^{F'}_{C_0[O_0]}$ with $M_0 \equiv O_0$ of the form

$e' = C_0[O_0] \rightarrow_{\theta'_1} C_1[O_1] \rightarrow_{\theta'_2} C_2[O_2] ... \rightarrow_{\theta'_k} C_k[O_k]$,

there exists a monotonic surjective function f from $[0-k]$ to $[0-h]$ such that:

[1] The following definitions are redundant with respect to Chapter 4 but we report them integrally in order to make the chapter more readable and understandable.

(i) $\forall i \in [0-k]$, $O_i \equiv M_{f(i)}$

(ii) $\forall j \in [1-k]$, $\theta'_j = \theta_{f(j)}$ *when* $M_{f(j-1)} \rightarrow_{\theta_{f(j)}} M_{f(j)}$.

Given a subset $\mathcal{F} \in Sched$ *of schedulers, we define* $\mathcal{F}_{\mathcal{C}} = \bigcup_{F \in \mathcal{F}} F_{\mathcal{C}}$.

Example 5.3 *Let* $M_0 \equiv m[\mathtt{out}\langle c_{L,r}, v\rangle.P]_l$ *and* $F \in Sched$ *such that*

$$M_0 \rightarrow_\Delta M_1 \rightarrow_\Delta M_2 \in Exec_M^F,$$

with $M_1 \equiv m[\bar{c}_{L,r}\langle v\rangle.P]_l$ *and* $M_2 \equiv m[P]_l$.

First notice that $F \in F_{\mathcal{C}}$, *since we can take the empty context* $C[\cdot] \equiv \emptyset \mid \cdot$ *and the identity function* f *such that* $f(i) = i$ *for all* $i \in [0-2]$. *In this case* $C[M_i] \equiv M_i$ *for all* $i \in [0-2]$ *and the property of Definition 5.3 is satisfied.*

Let now consider $N_0 \equiv n[\mathtt{in}(c, x).Q]_k$ *such that* $d(l, k) \leq r$. *All the admissible schedulers allowing* M_0 *and* N_0 *to interact are in* $F_{\mathcal{C}}$. *Indeed, consider* $F_1 \in Sched$ *such that, by applying rules (Struct-Bgn-Bcast) and (Struct-End-Bcast)*

$$M_0 \mid N_0 \rightarrow_\Delta M_1 \mid N_1 \rightarrow_\Delta M_2 \mid N_2 \in Exec^{F_1}_{M_0|N_0}$$

with $N_1 \equiv n[c(x).Q]_k$ *and* $N_2 \equiv n[Q\{v/x\}]_k$, *and consider also* F_2 *such that, by applying rule (R-Par)*

$$M_0 \mid N_0 \rightarrow_\Delta M_1 \mid N_0 \rightarrow_\Delta M_2 \mid N_0 \in Exec^{F_2}_{M_0|N_0}.$$

Both F_1 *and* F_2 *satisfy the properties of Definition 5.3, hence* $F_1, F_2 \in F_{\mathcal{C}}$.

Now consider again the network N_0. *Let* $e' = n[Q]_k \rightarrow_{\mu_k^n} n[Q]_{k'} \notin Exec^F_{N_0}$ *then* $\forall \bar{F} \in Sched$ *such that* $e' \in Exec^{\bar{F}}_{N_0}$, $\bar{F} \notin F_{\mathcal{C}}$ *since* $\bar{F}$ *does not satisfy the conditions of Definition 5.3.*

Now we are able to introduce our equivalence relation.

Again, the notion of barb introduced below is the probabilistic extension of Definition 3.1 of Chapter 3 and denotes an observable transmission with a certain probability according to a fixed admissible scheduler.

Definition 5.4 (Probabilistic Barb) *We say that a network* M *has a* probabilistic barb *with probability* p *on a channel* c *to the set* K *of locations, according to the admissible scheduler* F, *written* $M \Downarrow_p^F c@K$, *if* $Prob_M^F(\{N | N \downarrow_{c@K}\}) = p$.

Intuitively, for a given network M and scheduler F, if $M \Downarrow_p^F c@K$ then p is the positive probability that M, driven by F, performs a transmission on channel c and at least one of the intended recipients is able to correctly listen to it.

In the following, we introduce a probabilistic observational congruence, in the style of [31], which, as for the previous version of the calculus, is parametric to a restricted set of schedulers.

Definition 5.5 *Given a set of networks, and a set $\mathcal{F} \in Sched$ of scehdulers, and a relation $\mathcal{R}$ over networks:*

- Barb preservation. $\mathcal{R}$ *is* barb preserving *w.r.t.* $\mathcal{F}$ *if* $M\mathcal{R}N$ *and* $M\Downarrow_p^F c@K$ *for some* $F \in \mathcal{F}_\mathcal{C}$ *implies that there exists* $F' \in \mathcal{F}_\mathcal{C}$ *such that* $N\Downarrow_p^{F'} c@K$.
- Reduction closure. $\mathcal{R}$ *is* reduction closed *w.r.t.* $\mathcal{F}$ *if* $M\mathcal{R}N$ *implies that for all* $F \in \mathcal{F}_\mathcal{C}$, *there exists* $F' \in \mathcal{F}_\mathcal{C}$ *such that for all classes* $\mathcal{C} \in \mathcal{N}/\mathcal{R}$, $Prob_M^F(\mathcal{C}) = Prob_N^{F'}(\mathcal{C})$.
- Contextuality. $\mathcal{R}$ *is* contextual *if* $M\mathcal{R}N$ *implies that for every context* $\mathcal{C}[\cdot]$, *it holds that* $\mathcal{C}[M]\,\mathcal{R}\,\mathcal{C}[N]$.

Our probabilistic observational congruence with respect to a restricted set $\mathcal{F}$ of schedulers is defined as the largest relation as follows.

Definition 5.6 (Probabilistic Observational Congruence with respect to $\mathcal{F}$) *Given a set $\mathcal{F}$ of schedulers,* Probabilistic observational congruence w.r.t. $\mathcal{F}$, *written $\cong_p^\mathcal{F}$, is the largest symmetric relation over networks which is reduction closed, barb preserving and contextual.*

Two networks are related by $\cong_p^\mathcal{F}$ if they exhibit the same probabilistic behaviour (communications) relative to the corresponding sets of intended recipients. In the next section we develop a bisimulation-based proof technique for $\cong_p^\mathcal{F}$. It provides an efficient method to check whether two networks are related by $\cong_p^\mathcal{F}$.

5.3 A Bisimulation-based Proof Technique

In this section we develop a co-inductive proof technique for the relation $\cong_p^\mathcal{F}$.

5.3.1 Labelled Transition Semantics

As for its predecessor, we define a LTS semantics for our calculus, which is built upon two sets of rules: one for processes and one for networks. Table 5.3 presents the LTS rules for processes. Transitions are of the form $P \xrightarrow{\eta} P'$, where η ranges over input and output actions of the form:

$$\eta ::= c \,|c\vartheta\, |\bar{c}_{L,r} \mid \bar{c}_{L,r}\tilde{v} \quad \text{with} \quad \vartheta ::= \tilde{v} \mid \bot.$$

Rules (Beg-Out) and (End-Out) model the beginning and the end of an output action. Rule (Beg-In) models a process beginning listening to a channel in order to receive a value. Rule (End-In) models either the correct reception of a message or the reception of a $\bot$ due to a collision. All the remaining rules are standard.

$$\text{(Beg-Out)}\ \frac{-}{\texttt{out}\langle c_{L,r}, \tilde{v}\rangle.P \xrightarrow{\bar{c}_{L,r}} \bar{c}_{L,r}\langle\tilde{v}\rangle.P} \qquad \text{(End-Out)}\ \frac{-}{\bar{c}_{L,r}\langle\tilde{v}\rangle.P \xrightarrow{\bar{c}_{L,r}\tilde{v}} P}$$

$$\text{(Beg-In)}\ \frac{-}{\texttt{in}(c,\tilde{x}).P \xrightarrow{c} c(\tilde{x}).P} \qquad \text{(End-In)}\ \frac{-}{c(\tilde{x}).P \xrightarrow{c\vartheta} P\{\vartheta/\tilde{x}\}}$$

$$\text{(Then)}\ \frac{P \xrightarrow{\eta} P'}{[\tilde{v}\ =\ \tilde{v}]P, Q \xrightarrow{\eta} P'} \qquad \text{(Else)}\ \frac{Q \xrightarrow{\eta} Q' \quad \tilde{v_1} \neq \tilde{v_2}}{[\tilde{v_1}\ =\ \tilde{v_2}]P, Q \xrightarrow{\eta} Q'}$$

$$\text{(Rec)}\ \frac{P\{\tilde{v}/\tilde{x}\} \xrightarrow{\eta} P' \quad A(\tilde{x}) \stackrel{\text{def}}{=} P}{A\langle\tilde{v}\rangle \xrightarrow{\eta} P'}$$

Table 5.3: LTS rules for Processes

Table 5.4 presents the LTS rules for networks. The transitions are of the form $M \xrightarrow{\gamma} [\![M']\!]_\theta$, where M is a network, $[\![M']\!]_\theta$ is a distribution over networks, and γ ranges over the following labels:

$$\gamma\ ::=\ c?@l\ |c?\vartheta@l\ |\ c_L![l,r]\ |c_L!\tilde{v}[l,r]\ |\ c!\tilde{v}@K \triangleleft R\ |\ \tau.$$

We denote by $\mathtt{A}^s_M(c,l)$ the set of active senders of M on channel c reaching l, i.e., if $\mathtt{A}(M) \equiv (\nu\tilde{d})\big(\prod_{i\in I} n_i[\bar{c}_{L_i,r_i}\langle\tilde{v_i}\rangle.P_i]_{l_i} \mid \prod_{j\in J} n_j[c(\tilde{x}_j).P_j]_{l_j} \mid N\big)$ and $c \notin \mathtt{Top}(N)$ then $\mathtt{A}^s_M(c,l) = \{n_i\ : i \in I,\ d(l,l_i) \leq r_i\}$.

Rules (Beg-Snd) and (End-Snd) model the transmission of a message $\tilde{v}$ through channel c with radius r to the set L of observers. Transmissions are non-atomic actions: indeed, since mobile ad-hoc networks are not controlled by any fixed infrastructure, we have to take into account the possibility for nodes to be not perfectly synchronized with each other. (Beg-Rcv) models the beginning of a message reception, while (End-Rcv) models both the successful reception of a message or the reception of a failure message (denoted by $\perp$) due to an interference.

Rule (Beg-Bcast) models the beginning of a broadcast message propagation: all the nodes lying within the transmission cell of the sender may begin to receive a message (regardless of the fact that they are in L).

Rule (Coll-Bcast) models the collision occurred at the location of a receiver lying within the intersection of the transmission area of different nodes transmitting simultaneously through the same channel.

Rule (End-Bcast) models the conclusion of a broadcast message propagation: all the nodes lying within the transmission cell of the sender will successfully receive a message.

Rule (Obs) models the observability of a transmission: every transmission may be detected (and hence *observed*) by any recipient located within the transmission cell of one sender and outside the "interference area", that is the intersection of the transmission areas of the active senders of the network. The label $c!\tilde{v}@K \triangleleft R$ represents the transmission of the tuple $\tilde{v}$ of messages via c to the subset K of observers inside the reachable locations R within the transmission cell of the sender. Notice that collisions are not observable and only a correctly ended transmission may be observed.

Rule (Par) is standard.

Rule (Move) models migration of a mobile node n from a location l to a location k according to the probability distribution μ_l^n, which depends on the Markov chain $\mathbf{J}^n$ statically associated with n. Nodes can move only if they are not executing any active action (i.e., nodes cannot move while transmitting or receiving).

Rules (Lose1) and (Lose2) model both message loss and a local activity of the network which an observer is not party to. As usual, τ-transitions are used to denote non-observable actions. Finally, rule (Res) models the standard channel restriction, where $\mathtt{Chan}(\gamma) = c$ if γ is of the form $c?@l$; $c?\vartheta@l$; $c_L![l,r]$; $c_L!\tilde{v}[l,r]$; or $c!\tilde{v}@K \triangleleft R$, and $\mathtt{Chan}(\tau) = \bot$.

We prove that the LTS-based semantics coincides with the reduction semantics and the notion of observability (barb) given in the previous section.

We first prove that if $M \xrightarrow{\gamma} [\![M']\!]_\Delta$, then the structure of M and M' can be determined up to structural congruence.

We introduce the following auxiliary lemma in order to prove our harmony theorem.

Lemma 5.1 *Let M be a network.*

1. *If $M \xrightarrow{c?@l} [\![M']\!]_\Delta$, then there exist n, $\tilde{x}$, a (possibly empty) sequence $\tilde{d}$ such that $c \notin \tilde{d}$, a process P and a (possibly empty) network M_1 such that:*

$$M \equiv (\nu\tilde{d})(n[\mathtt{in}(c,\tilde{x})P]_l \mid M_1)$$

and

$$M' \equiv (\nu\tilde{d})(n[c(x).P]_l | M_1).$$

2. *If $M \xrightarrow{c?\vartheta@l} [\![M']\!]_\Delta$, then there exist n, $\tilde{x}$, $\tilde{d}$ such that $c \notin \tilde{d}$ and P such that:*

$$M \equiv (\nu\tilde{d})(n[c(x).P]_l | M_1)$$

and

$$M' \equiv (\nu\tilde{d})(n[P\{\vartheta/\tilde{x}\}]_l | M_1).$$

3. *If* $M \xrightarrow{c_L![l,r]} [\![M']\!]_\Delta$, *then there exist* n, $\tilde{v}$, *a (possibly empty) sequence* $\tilde{d}$ *such that* $c \notin \tilde{d}$, *a process* P, *two (possibly empty) sets* J *and* K *such that* $\forall h \in J \cup K$ $d(l, l_h) \leq r$ *and a (possibly empty) network* M_1 *such that:*

$$M \equiv (\nu \tilde{d})(n[\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_l \mid \prod\nolimits_{j\in J} n_j[\mathtt{in}(c, \tilde{x}_j).P_j]_{l_j} \mid \prod\nolimits_{k\in K} n_k[c(\tilde{x}_k).P_k]_{l_k} \mid M_1)$$

and

$$M' \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v}\rangle.P]_l \mid \prod\nolimits_{j\in J} n_j[c(\tilde{x}_j).P_j]_{l_j} \mid \prod\nolimits_{k\in K} n_K[P_k\{\perp/\tilde{x}_k\}]_{l_k} \mid M_1).$$

4. *If* $M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta$, *then there exist* n, *a (possibly empty) sequence* $\tilde{d}$ *such that* $c \notin \tilde{d}$, *a process* P, *a (possibly empty) set* J, *such that* $\forall j \in J$ $d(l, l_j) \leq r$ *and a (possibly empty) network* M_1 *such that:*

$$M \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v}\rangle.P]_l | \prod\nolimits_{j\in J} n_j[c(\tilde{x}_j).P_j]_{l_j} \mid M_1)$$

and

$$M \equiv (\nu \tilde{d})(n[P]_l | \prod\nolimits_{j\in J} n_j[P_j\{\tilde{v}/\tilde{x}_j\}]_{l_j} \mid M_1).$$

Proof.

The proof obtained by induction on the transition rules of Table 5.

Case 1: $M \xrightarrow{c?@l} [\![M']\!]_\Delta$

(Beg-Rcv) Let $M \xrightarrow{c?@l} [\![M']\!]_\Delta$ inferred by rule (Beg-Rcv), then there exist n, P, l, r such that $M \equiv n[P]_l$, $M' \equiv n[P']_l$ and $P = \mathtt{in}(c, \tilde{x}).Q$ and $P' = c(\tilde{x}).Q$. If we consider the empty network M_1 and the empty sequence $\tilde{d}$ lemma is proved.

(Par) Let $M \xrightarrow{c?@l} [\![M']\!]_\Delta$ inferred by rule (Par), where $M \equiv M_1 \mid N$, $M' \equiv M_1' \mid N$ and $M_1 \xrightarrow{c?@l} [\![M_1']\!]_\Delta$. By induction hypothesis we have

$$M_1 \equiv (\nu \tilde{d})(n[\mathtt{in}(c, \tilde{x}).P]_l | M_2)$$

and

$$M_1' \equiv (\nu \tilde{d})(n[c(\tilde{x}).P]_l | M_2),$$

for some n, P, $\tilde{x}$, (possibly empty) $\tilde{d}$ such that $c \notin \tilde{d}$, and (possibly empty) M_2. By applying rule (Struct Cxt Par), (Struct Par Assoc), (Struct Res Par) and (Struct Trans) we obtain

$$M \equiv M_1 \mid N \equiv (\nu \tilde{d})(n[\mathtt{in}(c, \tilde{x}).P]_l | (M_2 \mid N))$$

and

$$M' \equiv M_1' \mid N \equiv (\nu \tilde{d})(n[\mathtt{in}(c, \tilde{x}).P]_l | (M_2 \mid N)),$$

as required.

(Res) Let $M \xrightarrow{c?@l} [\![M']\!]_\Delta$ inferred by rule (Res), where $M \equiv (\nu d)M_1$, $M' \equiv (\nu d)M_1'$ and $M_1 \xrightarrow{c?@l} [\![M_1']\!]_\Delta$ and $c \neq d$. By induction hypothesis we have

$$M_1 \equiv (\nu \tilde{d'})(n[\mathtt{in}(c, \tilde{x}).P]_l | M_2)$$

and

$$M_1' \equiv (\nu \tilde{d'})(n[c(\tilde{x}).P]_l | M_2)$$

for some n, P, $\tilde{x}$, (possibly empty) $\tilde{d'}$ such that $c \notin \tilde{d'}$, and (possibly empty) M_2.

If we consider $\tilde{d''} = \tilde{d'} \cup d$, since $c \notin \tilde{d''}$, we get:

$$M \equiv (\nu \tilde{d''})(n[\mathtt{in}(c, \tilde{x}).P]_l | M_2)$$

and

$$M' \equiv (\nu \tilde{d''})(n[\mathtt{in}(c, \tilde{x}).P]_l | M_2).$$

Case 2: $M \xrightarrow{c?\tilde{x}@l} [\![M']\!]_\Delta$

The proof of this case is analogous to the previous one.

Case 3: $M \xrightarrow{c_L![l,r]} [\![M']\!]_\Delta$

(Beg-Snd) Let $M \xrightarrow{c_L![l,r]} [\![M']\!]_\Delta$ inferred by rule (Beg-Snd), then exist $\tilde{v}$ and P such that: $M \equiv n[\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_l$. Since $\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P \xrightarrow{c_{\bar{L},r}} \bar{c}_{L,r}\langle\tilde{v}\rangle.P$, if we suppose $\tilde{d}$, J, K and M_1 empty, lemma is proved because

$$M \equiv (\nu \tilde{d})(n[\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_l | \prod_{j \in J} n_j[\mathtt{in}(c, \tilde{x_j})P_j]_{l_j} | \prod_{k \in K} n_k[c(\tilde{x_k})P_k]_{l_k} \mid M_1)$$

and

$$M' \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l | \prod_{j \in J} n_j[c(\tilde{x_j})P_j]_{l_j} | \prod_{k \in K} n_k[P_k\{\bot/\tilde{x_k}\}]_{l_k} \mid M_1).$$

(Beg-Bcast) Let $M \xrightarrow{c_L![l,r]} [\![M']\!]_\Delta$ because $M \equiv M_1 \mid N$, $M' \equiv M_1' \mid N'$, $M_1 \xrightarrow{c_L![l,r]} [\![M_1']\!]_\Delta$ and $N \xrightarrow{c?@l'} [\![N']\!]_\Delta$, with $d(l, l') \leq r$. By induction hypothesis:

$$M_1 \equiv (\nu \tilde{d_1})(n[\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_l | \prod_{j \in J} n_j[\mathtt{in}(c, \tilde{x_j})P_j]_{l_j} | \prod_{k \in K} n_k[c(\tilde{x_k})P_k]_{l_k} \mid M_2)$$

and

$$M_1' \equiv (\nu \tilde{d_1})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l | \prod_{j \in J} n_j[c(\tilde{x_j})P_j]_{l_j} | \prod_{k \in K} n_k[P_k\{\bot/\tilde{x_k}\}]_{l_k} \mid M_2),$$

for some n, P, $\tilde{v}$, l, some (possible empty) sequence $\tilde{d_1}$ such that $c \notin \tilde{d_1}$, some (possibly empty) sets J and K and some (possibly empty) network M_2, and by indcution hypothesis we get:

$$N \equiv (\nu \tilde{d_2})(m[\mathtt{in}(c, \tilde{x}).Q]_{l'} | N_1)$$

and

$$N' \equiv (\nu \tilde{d_2})(m[c(\tilde{x}).Q]_{l'} | N_1),$$

for some m, Q, $\tilde{x}$, some (possible empty) sequence $\tilde{d_2}$ such that $c \notin \tilde{d_2}$ and (possibly empty) network N_1 . By applying rules (Struct Cxt Par), (Struct Par Assoc), (Struct Res Par) and (Struct Trans), if we consider $\tilde{d} = \tilde{d_1} \cup \tilde{d_2}$, we can assume $\tilde{d_1} \notin fc(N)$ and $\tilde{d_2} \notin fc(M)$ and we get:

$$M \equiv (\nu \tilde{d})(n[\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_l | m[\mathtt{in}(c, \tilde{x}).Q]_{l'} | \prod_{j \in J} n_j[\mathtt{in}(c, \tilde{x_j})P_j]_{l_j}$$

$$| \prod_{k \in K} n_k[c(\tilde{x_k})P_k]_{l_k} \mid (M_2 \mid N_1))$$

and

$$M' \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v}\rangle.P]_l | m[c(\tilde{x}).Q]_{l'} \mid \prod_{j \in J} n_j[c(\tilde{x_j})P_j]_{l_j}$$

$$| \prod_{k \in K} n_k[P_k\{\perp/\tilde{x_k}\}]_{l_k} \mid (M_2 \mid N_1)).$$

(Coll) Let $M \xrightarrow{c_L![l,r]} [\![M']\!]_\Delta$ because $M \equiv M_1 \mid N$, $M' \equiv M_1' \mid N'$, $M_1 \xrightarrow{c_L![l,r]} [\![M_1']\!]_\Delta$ and $N \xrightarrow{c?\perp@l'} [\![N']\!]_\Delta$, with $d(l, l') \leq r$. By induction hypothesis:

$$M_1 \equiv (\nu \tilde{d_1})(n[\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_l | \prod_{j \in J} n_j[\mathtt{in}(c, \tilde{x_j})P_j]_{l_j} | \prod_{k \in K} n_k[c(\tilde{x_k})P_k]_{l_k} \mid M_2)$$

and

$$M_1' \equiv (\nu \tilde{d_1})(n[\bar{c}_{L,r}\langle \tilde{v}\rangle.P]_l | \prod_{j \in J} n_j[c(\tilde{x_j})P_j]_{l_j} | \prod_{k \in K} n_k[P_k\{\perp/\tilde{x_k}\}]_{l_k} \mid M_2),$$

for some n, P, some (possibly empty) sequence $\tilde{d_1}$ such that $c \notin \tilde{d_1}$, some (possibly empty) sets J and K and some (possibly empty) network M_2, and by induction hypothesis we get:

$$N \equiv (\nu \tilde{d_2})(m[c(\tilde{x}).Q]_{l'} | N_1)$$

and

$$N' \equiv (\nu \tilde{d_2})(m[Q\{\perp/\tilde{x}\}]_{l'} | N_1),$$

for some m, Q, $\tilde{x}$, some (possibly empty) sequence $\tilde{d_2}$ such that $c \notin \tilde{d_2}$ and (possibly empty) network N_1 . By applying rules (Struct Cxt Par), (Struct Par Assoc), (Struct Res Par) and (Struct Trans), if we consider $\tilde{d} = \tilde{d_1} \cup \tilde{d_2}$, we can assume $\tilde{d_1} \notin fc(N)$, $\tilde{d_2} \notin fc(M)$ and we get:

$$M \equiv (\nu \tilde{d})(n[\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_l | \prod_{j\in J} n_j[\mathtt{in}(c, \tilde{x_j})P_j]_{l_j}$$

$$| \prod_{k\in K} n_k[c(\tilde{x_k})P_k]_{l_k} \mid m[c(\tilde{x}).Q]_{l'} | (M_2 \mid N_1))$$

and

$$M' \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v}\rangle.P]_l | \prod_{j\in J} n_j[c(\tilde{x_j})P_j]_{l_j}$$

$$| \prod_{k\in K} n_k[P_k\{\perp/\tilde{x_k}\}]_{l_k} m[Q\{\perp/\tilde{x}\}]_{l'} \mid (M_2 \mid N_1)).$$

The proof of the other cases is analogous to the first part of the lemma.

Case 4: $M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta$

(End-Snd) Let $M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta$ inferred by rule (End-Snd), then exists P such that $M \equiv n[\bar{c}_{L,r}\langle \tilde{v}\rangle.P]_l$. Since $\bar{c}_{L,r}\langle \tilde{v}\rangle.P \xrightarrow{c_{\bar{L},r}\tilde{v}} P$, if we suppose J, $\tilde{d}$ and M_1 empty, lemma is proved because

$$M \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v}\rangle.P]_l | \prod_{j\in J} n_j[c(\tilde{x_j})P_j]_{l_j} \mid M_1)$$

and

$$M' \equiv (\nu \tilde{d})(n[P]_l | \prod_{j\in J} n_j[P_j\{\tilde{v}/\tilde{x_j}\}]_{l_j} \mid M_1).$$

(End-Bcast) Let $M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta$ because $M \equiv M_1 \mid N$, $M' \equiv M_1' \mid N'$, $M_1 \xrightarrow{c_L!\tilde{v}[l,r]} [\![M_1']\!]_\Delta$ and $N \xrightarrow{c?\tilde{v}@l'} [\![N']\!]_\Delta$, with $d(l, l') \leq r$. By induction hypothesis:

$$M_1 \equiv (\nu \tilde{d_1})(n[\bar{c}_{L,r}\langle \tilde{v}\rangle.P]_l | \prod_{j\in J} n_j[c(\tilde{x_j})P_j]_{l_j} \mid M_2)$$

and

$$M_1' \equiv (\nu \tilde{d_1})(n[P]_l | \prod_{j\in J} n_j[P_j\{\tilde{v}/\tilde{x_j}\}]_{l_j} \mid M_2),$$

for some n, P, some (possibly empty) sequence $\tilde{d_1}$ such that $c \notin \tilde{d_1}$, some (possibly empty) set J and some (possibly empty) network M_2, and by induction hypothesis we get:

$$N \equiv (\nu \tilde{d_2})(m[c(\tilde{x}).Q]_{l'} | N_1)$$

and

$$N' \equiv (\nu \tilde{d_2})(m[Q\{\tilde{v}/\tilde{x}\}]_{l'}|N_1),$$

for some m, Q, $\tilde{x}$, some (possibly empty) sequence $\tilde{d_2}$ such that $c \notin \tilde{d_2}$ and (possibly empty) network N_1 . By applying rules (Struct Cxt Par), (Struct Par Assoc), (Struct Res Par) and (Struct Trans), if we consider $\tilde{d} = \tilde{d_1} \cup \tilde{d_2}$, we can assume $\tilde{d_1} \notin fc(N)$, $\tilde{d_2} \notin fc(M)$ and we get:

$$M \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l|m[c(\tilde{x}).Q]_{l'} \mid \prod_{j\in J} n_j[c(\tilde{x_j})P_j]_{l_j} \mid (M_1 \mid N_1))$$

and

$$M' \equiv (\nu \tilde{d})(n[P]_l|m[Q\{\tilde{v}/\tilde{x}\}]_{l'} \mid \prod_{j\in J} n_j[P_j\{\tilde{v}/\tilde{x}\}]_{l_j} \mid (M_1 \mid N_1)).$$

The proof of the other cases is analogous to the first part of the lemma.

We also show that structural congruence respects the transitions of Table 5.4.

Lemma 5.2 *If $M \xrightarrow{\gamma} [\![M']\!]_\theta$ and $M \equiv N$, then there exists N' such that $N \xrightarrow{\gamma} [\![N']\!]_\theta$ and $M' \equiv N'$.*

Proof.

By induction on the depth of the inference $M \xrightarrow{\gamma} [\![M']\!]_\theta$.

There are a lot of cases to consider, following we give only some example.

(Par) Let consider $M \xrightarrow{\gamma} [\![M]\!]_\theta$ because $M \equiv M_1 \mid N$, $M' \equiv M_1' \mid N$ and $M \xrightarrow{\gamma} [\![M']\!]_\theta$. There are several rules for structural congruence that can be applied.

Let consider $M_1 \mid N \equiv M_1 \mid N'$ because $N' \equiv N$, by (Struct Cxt Par). By hypothesis $M_1 \xrightarrow{\gamma} [\![M_1']\!]_\theta$ and, by an application of rule (Par) we get $M_1 \mid N' \xrightarrow{\gamma} [\![M_1' \mid N']\!]_\theta$. But, since $N' \equiv N$, by applying (Struct Cxt Par) we have that $M_1' \mid N' \equiv M_1' \mid N$.

Let consider now $M_1 \mid N \equiv N \mid M_1$ by (Struct Par Com). Again, since $M_1 \xrightarrow{\gamma} [\![M_1']\!]_\theta$, by applying rule (Par) we get $N \mid M_1 \xrightarrow{\gamma} [\![N \mid M_1']\!]_\theta$, and, by applying again rule (Struct Par Com), $N \mid M_1' \equiv M_1' \mid N$, as required.

(End-Bcast) Let consider $M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta$, because $M \equiv M_1 \mid N$, $M' \equiv M_1' \mid N'$ and $M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta$ and $N \xrightarrow{c?\tilde{v}@l'} [\![N']\!]_\Delta$, with $d(l,l') \leq r$.

Again several rules for structural congruence can be applied.

Let consider, e.g. $M_1 \mid N \equiv N \mid M_1$ by (Struct Par Comm). Again, since $M_1 \xrightarrow{c_L!\tilde{v}[l,r]} [\![M_1']\!]_\Delta$, $N \xrightarrow{c?\tilde{v}@l'} [\![N']\!]_\Delta$, and $d(l,l') \leq r$, we can apply rule (Bcast), obtaining $N \mid M_1 \xrightarrow{c_L!\tilde{v}[l,r]} [\![N' \mid M_1']\!]_\Delta$, and, by applying again (Struct Par Comm) we get $N' \mid M_1' \equiv M_1' \mid N'$ as required. Let consider now $M_2 \mid N \equiv$

$M_1 \mid N$ because $M_2 \equiv M_1$, by (Struct Cxt Par). By induction hypothesis $M_2 \xrightarrow{c_L!\tilde{v}[l,r]} [\![M_2']\!]_\Delta$, and $M_2' \equiv M_1'$. But since $d(l,l') \leq r$ we can apply rule (Bcast), obtaining $M_2 \mid N \xrightarrow{c_L!\tilde{v}[l,r]} [\![M_2' \mid N']\!]_\Delta$. Now, by applying (Struct Cxt Par) we get $M_2' \mid N' \equiv M_1' \mid N'$, as required.

The proof of the other cases is similar.

The following theorem establishes the relationship between the reduction semantics and the LTS one.

Theorem 5.1 (Harmony) *Let M be a network.*

1. *If $M \to [\![M']\!]_\theta$ then there exist N and N' such that $N \xrightarrow{\tau} [\![N']\!]_\theta$, $M \equiv N$ and $M' \equiv N'$.*
2. *$M \downarrow_{c@K}$ iff M is well-formed and $N \xrightarrow{c!\tilde{v}@K \triangleleft R} [\![N']\!]_\Delta$ for some R, $\tilde{v}$, $N \equiv M$ and M'.*
3. *If $M \xrightarrow{\tau} [\![M']\!]_\theta$ then $M \to [\![M']\!]_\theta$.*
4. *If $M \xrightarrow{c!\tilde{v}@K \triangleleft R} [\![M']\!]_\Delta$ then $M \to [\![M']\!]_\Delta$.*

Proof.

1. The first part is proved by induction on the reduction $M \to [\![M']\!]_\theta$

   Suppose that $M \to [\![M']\!]_\theta$ is due to the application of the rule (R-Move). We deduce $M \equiv M' \equiv n[P]_l$ and $\theta = \mu_l^n$, for some name n, some location l and some process P. We simply apply (Move) to obtain:

   $$\frac{\texttt{Active}(P) = \texttt{false}}{n[P]_l \xrightarrow{\tau} [\![n[P]_l]\!]_{\mu_l^n}}.$$

   Suppose that $M \to [\![M']\!]_\theta$ is due to the application of the rule (R-Par). If we consider $M \equiv M_1 \mid M_2$ and $M' \equiv M_1' \mid M_2$

   $$\frac{M_1 \to [\![M_1']\!]_\theta}{M_1 \mid M_2 \to [\![M_1' \mid M_2]\!]_\theta},$$

   By induction hypothesis $M_1 \xrightarrow{\tau} [\![M_1']\!]_\theta$, then by applying rule (Par) we get:

   $$\frac{M_1 \xrightarrow{\tau} [\![M_1']\!]_\theta}{M_1 \mid M_2 \xrightarrow{\tau} [\![M_1' \mid M_2]\!]_\theta}.$$

Suppose that $M \rightarrow [\![M']\!]_\theta$ is due to the application of the rule (R-Res). If we consider $M \equiv (\nu c)M_1$ and $M' \equiv (\nu c)M_1'$:

$$\frac{M_1 \rightarrow [\![M_1']\!]_\theta}{(\nu c)M_1 \rightarrow [\![(\nu c)M_1']\!]_\theta},$$

by induction hypothesis $M_1 \xrightarrow{\tau} [\![M_1']\!]_\theta$, then by applying rule (Res), since $\mathtt{Chan}(\tau) \neq c$ we get:

$$\frac{M_1 \xrightarrow{\tau} [\![M_1']\!]_\theta}{(\nu c)M_1 \xrightarrow{\tau} [\![(\nu c)M_1']\!]_\theta}.$$

Suppose that $M \rightarrow [\![M']\!]_\theta$ is due to the application of the rule (R-Bgn-Bcast). It means:

$$M \equiv (\nu \tilde{d})(n[\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_l \mid \textstyle\prod_{i\in I} n_i[\bar{c}_{L_i,r_i}\langle \tilde{v}_i\rangle.P_i]_{l_i} \\ \mid \textstyle\prod_{j\in J} n_j[c(x_j).P_j]_{l_j} \mid \textstyle\prod_{k\in K} n_k[\mathtt{in}(c, x_k).P_k]_{l_k})$$

and

$$M' \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v}\rangle.P]_l \mid \textstyle\prod_{i\in I} n_i[\bar{c}_{L_i,r_i}\langle \tilde{v}_i\rangle.P_i]_{l_i} \\ \mid \textstyle\prod_{j\in J} n_j[P_j\{\perp/\tilde{x}_j\}]_{l_j} \mid \textstyle\prod_{k\in K} n_k[c(x_k).P_k]_{l_k}),$$

for some n, some process P, some channel c, some set L of locations, some radius r, some tuple $\tilde{v}$ of messages, some tuple $\tilde{d}$ of channels, and some (possibly empty) sets I, J and K of networks. Then, by applying rule (Beg-Snd), (Beg-Rcv), and then $\mid K \mid$ times rule (Beg-Bcast), $\mid J \mid$ times rule (Coll-Bcast), and, finally rules (Res), (Lose1) and (Par), we obtain:

$$M \xrightarrow{\tau} \equiv$$

$$[\![(\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v}\rangle.P]_l \mid \textstyle\prod_{i\in I} n_i[\bar{c}_{L_i,r_i}\langle \tilde{v}_i\rangle.P_i]_{l_i} \mid \textstyle\prod_{j\in J} n_j[P_j\{\perp/\tilde{x}_j\}]_{l_j} \mid \textstyle\prod_{k\in K} n_k[c(x_k).P_k]_{l_k})]\!]_\theta$$

as required.

Suppose that $M \rightarrow [\![M']\!]_\theta$ is due to the application of the rule (R-End-Bcast). It means:

$$M \equiv (\nu \tilde{d})n[\bar{c}_{L,r}\langle \tilde{v}\rangle.P]_l \mid \textstyle\prod_{j\in J} n_j[c(\tilde{x_j}).P_j]_{l_j}$$

$$M' \equiv (\nu \tilde{d})n[P]_l \mid \textstyle\prod_{j\in J} n_j[P_j\{\tilde{v}/\tilde{x_j}\}]_{l_j}$$

for some channel c, some tuple $\tilde{d}$ of channels such that $c \notin \tilde{d}$, some node n, some process P, some tuple $\tilde{v}$ of messages, some location l, some set L of locations, some radius r, some process P, and some (possibly empty set) J such that $d(l, l_j) \leq r \ \forall j \in J$.

Then, by applying rule (End-Snd), (End-Rcv), $\mid I \mid$ times rule (End-Bcast), $\mid \tilde{d} \mid$ times (Res) and finally rule (Lose2), we get:

$$(\nu\tilde{d})n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid \prod_{j\in J} n_j[c(\tilde{x_j}).P_j]_{l_j} \xrightarrow{\tau} [\![(\nu\tilde{d})n[P]_l \mid \prod_{j\in J} n_j[P_j\{\tilde{v}/\tilde{x_j}\}]_{l_j}]\!]_\Delta$$

as required.

Finally let suppose that the reduction $M \to [\![M']\!]_\theta$ is due to an application of rule (R-Struct):

$$\frac{M \equiv N \quad N \to [\![N']\!]_\theta \quad N' \equiv M'}{M \to [\![M']\!]_\theta}.$$

By induction hypothesis there exists $N_1 \equiv N$ and $N_2 \equiv N'$ such that $N_1 \xrightarrow{\tau} [\![N_2]\!]_\theta$. Then, by applying the rule for structural congruence (Struct Trans) we get $M \equiv N \equiv N_1$ and $M' \equiv N' \equiv N_2$, as required.

2. The second part of the theorem follows from Lemma 5.1 and the definition of Barb. If $M \downarrow_{c@K}$, by the definition of Barb there exists $\tilde{v}$, L, r, a (possibly empty) sequence $\tilde{d}$ such that $c \notin \tilde{d}$, a process P, a (possibly empty) network M_1 such that:

   $M \equiv (\nu\tilde{d})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid M_1)$, with $K \subseteq \{k \in L \text{ s.t. } d(l,k) \leq r\}$ and $K \neq \emptyset$.

   By applying the rules (End-Snd) and (Par) we obtain:

$$\frac{n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \xrightarrow{c_L!\tilde{v}[l,r]} [\![n[P]_l]\!]_\Delta}{n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid M_1 \xrightarrow{c_L!\tilde{v}[l,r]} [\![n[P]_l \mid M_1]\!]_\Delta};$$

   then, since $K \subseteq \{l' \mid d(l,l') \leq r\} \cap L$ and $K \neq \emptyset$, we can apply rule (Obs):

   $n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid M_1 \xrightarrow{c!\tilde{v}@K\triangleleft R} [\![n[P]_l \mid M_1]\!]_\Delta$,

   where $R \subseteq \{l' \in \mathbf{Loc} : d(l,l') \leq r\}$, as required.

   If $M \xrightarrow{c!\tilde{v}@K\triangleleft R} [\![M']\!]_\Delta$, because $M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta$, by applying Lemma 5.1 then there exist n, a (possibly empty) sequence $\tilde{d}$ such that $c \notin \tilde{d}$, P, a (possibly empty) network M_1 and a (possibly empty) set J, such that $\forall j \in J\ d(l,l_j) \leq r$ and:

$$M \equiv (\nu\tilde{d})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l | \prod_{j\in J} n_j[c(\tilde{x}_j).P_j]_{l_j} \mid M_1)$$

   and

$$M \equiv (\nu\tilde{d})(n[P]_l | \prod_{j\in J} n_j[P_j\{\tilde{v}/\tilde{x_j}\}]_{l_j} \mid M_1).$$

   By applying the definition of barb we conclude $M \downarrow_{c@K}$.

3. The third part of the theorem is proved by induction on the derivation $M \xrightarrow{\tau} [\![M']\!]_\theta$.
Suppose that $M \xrightarrow{\tau} [\![M']\!]_\theta$ is due to an application of the rule (Move), that means $M \equiv M' \equiv n[P]_l$, $\theta = \mu_l^n$ for some node n, some process P and some location l, and :

$$\frac{\texttt{Active}(P) = \texttt{false}}{n[P]_l \xrightarrow{\tau} [\![n[P]_l]\!]_{\mu_l^n}}.$$

Hence , by applying (R-Move) we get:

$$\frac{\texttt{Active}(P) = \texttt{false}}{n[P]_l \rightarrow [\![n[P]_l]\!]_{\mu_l^n}}.$$

If $M \xrightarrow{\tau} [\![M']\!]_\theta$ is due to an application of (Lose1):

$$\frac{M \xrightarrow{c_L![l,r]} [\![M']\!]_\Delta}{M \xrightarrow{\tau} [\![M']\!]_\Delta},$$

then, by applying Lemma 5.1, there exists n, $\tilde{v}$, P $\tilde{d}$ such that $c \notin \tilde{d}$ and P, a (possibly empty) network M_1 and two (possibly empty) sets J and K such that $\forall i \in J \cup K$ $d(l, l_i) \leq r$, such that:

$$M \equiv (\nu \tilde{d})(n[\texttt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_l | \prod\nolimits_{j\in J} n_j[\texttt{in}(c, \tilde{x}_j).P_j]_{l_j} \mid \prod\nolimits_{k\in K} n_k[c(\tilde{x}_k).P_j]_{l_j} \mid M_1)$$

and

$$M' \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle \tilde{v}\rangle.P]_l | \prod\nolimits_{j\in J} n_j[c(\tilde{x}_j).P_j]_{l_j} \mid \prod\nolimits_{k\in K} n_k[P_k\{\perp/\tilde{x}_k\}]_{l_k} \mid M_1).$$

Finally, by applying rule (R-Bgn-Bcast), (R-Res) and (R-Struct) we get $M \rightarrow [\![M']\!]_\theta$.

For the application of the rule (Lose2) the proof is analogous to the previous one.

Suppose that $M \xrightarrow{\tau} [\![M']\!]_\theta$ is due to the application of (Res), we have $M \equiv (\nu c)M_1$, $M' \equiv (\nu c)M_1'$ and

$$\frac{M_1 \xrightarrow{\tau} [\![M_1']\!]_\theta}{(\nu c)M_1 \xrightarrow{\tau} [\![(\nu c)M_1']\!]_\theta}.$$

By induction hypothesis $M_1 \rightarrow [\![M_1']\!]_\theta$, hence, by applying rule (R-Res) we get $(\nu c)M_1 \rightarrow [\![(\nu c)M_1']\!]_\theta$.

Finally, suppose that $M \xrightarrow{\tau} [\![M']\!]_\theta$ is due to the application of (Par), we have $M \equiv M_1 \mid N$, $M' \equiv M_1' \mid N$ and:

$$\frac{M_1 \xrightarrow{\tau} [\![M_1']\!]_\theta}{M_1|N \xrightarrow{\tau} [\![M_1'|N]\!]_\theta},$$

by induction hypothesis $M_1 \to [\![M_1']\!]_\theta$, hence, by applying rule (R-Par) we get $M_1|N \to [\![M_1'|N]\!]_\theta$.

4. The last part of the theorem follows from definition of barb and Lemma 5.1. Formally, since $M \xrightarrow{c!\tilde{v}@K \triangleleft R} [\![M']\!]_\Delta$ because $M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta$ for some location l, radius r and set L of intended recipients, by applying Lemma 5.1:

$$M \equiv (\nu \tilde{d})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid \prod\nolimits_{j\in J} n_j[c(\tilde{x}_j).P_j]_{l_j} \mid M_1)$$

and

$$M' \equiv (\nu \tilde{d})(n[P]_l \mid \prod\nolimits_{j\in J} n_j[P_j\{\tilde{v}/\tilde{x}_j\}]_{l_j} \mid M_1)$$

for some n, P, for some (possibly empty) sequence $\tilde{d}$ such that $c \notin \tilde{d}$, some (possibly empty) set J, and some (possibly empty) network M_1. Then, by applying the rule (R-End-Bcast), (R-Par) and (R-Res) we get

$$(\nu \tilde{d})(n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid \textstyle\prod_{j\in J} n_j[c(\tilde{x}_j).P_j]_{l_j} \mid M_1) \to [\![(\nu \tilde{d})(n[P]_l \mid \prod_{j\in J} n_j[P_j\{\tilde{v}/\tilde{x}_j\}]_{l_j} \mid M_1)]\!]_\Delta,$$

and, by applying (R-Struct), we obtain $M \to [\![M']\!]_\Delta$, as required.

5.3.2 Probabilistic Labelled Bisimilarity

As for the previous versions of the calculus, we define a probabilistic labelled bisimilarity that is a complete characterisation of our *probabilistic observational congruence*. It is built upon the following actions:

$$\alpha \ ::= c?@l \mid c?\vartheta@l \mid c!\tilde{v}@K \triangleleft R \mid \tau.$$

Again, we write $M \xrightarrow{\alpha}_\theta N$ if $M \xrightarrow{\alpha} [\![M']\!]_\theta$ and N is in the support of $[\![M']\!]_\theta$. Moreover we write $M \xrightarrow{\alpha} N$ if $M \xrightarrow{\alpha}_\theta N$ for some θ. A labelled *execution* e of a network M is a finite (or infinite) sequence of steps: $M \xrightarrow{\alpha_1}_{\theta_1} M_1 \xrightarrow{\alpha_2}_{\theta_2} M_2 ... \xrightarrow{\alpha_k}_{\theta_k} M_k$. With abuse of notation, we define $Exec_M$, $last(e)$, e^j and $e \uparrow$ as for unlabeled executions. We denote by $lbehave(M)$ the set of all possible behaviors of M, i.e., $lbehave(M) = \{(\alpha, [\![M']\!]_\theta) \mid M \xrightarrow{\alpha} [\![M']\!]_\theta\}$. Labelled executions arise by resolving the non-determinism of both α and $[\![M]\!]_\theta$. As a consequence, a scheduler[2] for the

[2]We abuse notation and still use F to denote a scheduler for the LTS semantics.

labelled semantics is a function F assigning a pair $(\alpha, [\![M]\!]_\theta) \in lbehave(last(e))$ with a finite labelled execution e. We denote by $LSched$ the set of (admissible) schedulers for the LTS semantics, i.e., the set of all the schedulers F such that, for each network M in the support of F, M is well formed. Given a network M and a scheduler F, we define $Exec_M^F$ as the set of all labelled executions starting from M and driven by F.

Since we are interested in weak observational equivalences, that abstract over τ-actions, we introduce the notion of *weak action*.

Definition 5.7 *(Weak Action) We denote by $\Longrightarrow$ the transitive and reflexive closure of $\xrightarrow{\tau}$ and by $\stackrel{\alpha}{\Longrightarrow}$ the weak action $\Longrightarrow\xrightarrow{\alpha}\Longrightarrow$. We denote by $\stackrel{\hat{\alpha}}{\Longrightarrow}$ the weak action $\stackrel{\alpha}{\Longrightarrow}$ if $\alpha \neq \tau$, and $\Longrightarrow$ otherwise.*

In the following we will give the definition of probabilistic labelled bisimilarity with respect to a given set of schedulers

Definition 5.8 *Given an admissible scheduler $F \in Sched$, we denote by $\hat{F}_{\mathcal{C}} \subseteq LSched$ the set of admissible schedulers $\hat{F} \in LSched$ such that $\forall M_0$, $\forall e \in Exec_{M_0}^{\hat{F}}$ of the form*

$e = M_0 \xrightarrow{\alpha_1}_{\theta_1} M_1 ... \xrightarrow{\alpha_h}_{\theta_h} M_h$

$\exists F' \in F_{\mathcal{C}}$, a context C_0 and $e' \in Exec_{C_0[O_0]}^{F'}$ with $O_0 \equiv M_0$ such that

$e' = C_0[O_0] \rightarrow_{\theta'_1} C_1[O_1] ... \rightarrow_{\theta'_k} C_k[O_k]$

and there exists a monotone surjective function f from $[0-k]$ to $[0-h]$ such that:

(i) $\forall i \in [1-k]$, $O_i \equiv M_{f(i)}$

(ii) $\forall j \in [1-k]$, $\theta'_j = \theta_{f(j)}$ *when* $M_{f(j-1)} \xrightarrow{\alpha_{f(j)}}_{\theta_{f(j)}} M_{f(j)}$.

Given a set $\mathcal{F} \subseteq Sched$ of schedulers, we define $\hat{\mathcal{F}}_{\mathcal{C}} = \bigcup_{F \in \mathcal{F}} \hat{F}_{\mathcal{C}}$.

Example 5.4 *Consider the networks M_0 and N_0, and the schedulers F and F_1 introduced in the Example 5.3. If we take $\hat{F}_1 \in LSched$ such that*

$$M_0 \xrightarrow{c_L![l,r]}_\Delta M_1 \xrightarrow{c_L!v[l,r]}_\Delta M_2 \in Exec_{M_0}^{\hat{F}_1},$$

then, since

$$M_0 \rightarrow_\Delta M_1 \rightarrow_\Delta M_2 \in Exec_{M_0}^{F}$$

the conditions of Definition 5.8 are satisfied by taking the empty context $C[\cdot] = \mathbf{0} \mid \cdot$ and the identity function $f(i) = i$ for $i \in [0-2]$. Hence $\hat{F}_1 \in \hat{F}_{\mathcal{C}}$.

Moreover, if we consider $\hat{F}_2 \in LSched$ such that

$$N_0 \xrightarrow{c?@k}_\Delta N_1 \xrightarrow{c?v@k}_\Delta N_2 \in Exec_{N_0}^{\hat{F}_2},$$

since

$$M_0 \mid N_0 \rightarrow_\Delta M_1 \mid N_1 \rightarrow_\Delta M_2 \mid N_2 \in Exec_{M_0|N_0}^{F_1}$$

with $F_1 \in F_{\mathcal{C}}$, by considering the contexts $C_i[\cdot] \equiv M_i \mid \cdot$ for $i \in [0-2]$, and the identity function $f(i) = i$ for $i \in [0-2]$ we get $\hat{F}_2 \in \hat{F}_{\mathcal{C}}$ too.

Proposition 5.1

1. $Sched_{\mathcal{C}} = Sched$
2. $\widehat{Sched}_{\mathcal{C}} = LSched$

Proof.

1. The Proof follows straightforwardly from Definition 5.3.
2. $\forall F \in LSched$, $\forall M_0 \in \mathcal{N}$ and $\forall e \in Exec^F_{M_0}$ of the form:

   $e = M_0 \xrightarrow{\alpha_1}_{\theta_1} M_1 ... \xrightarrow{\alpha_k}_{\theta_k} M_k$
   it is always possible to find a context $C_0[\cdot]$ and a scheduler $F' \in LSched$ such that $e' \in Exec^{F'}_{C_0[M_0]}$ with

   $e' = C_0[M_0] \xrightarrow{\tau}_{\theta_1} ...C_1[M_1] ... \xrightarrow{\tau}_{\theta_k} C_k[M_k]$.

   By Theorem 5.1, $\exists F'' \in Sched$ such that $e'' \in Exec^{F''}_{C_0[M_0]}$ with

   $e'' = C_0[M_0] \rightarrow_{\theta_1} ...C_1[M_1] ... \rightarrow C_k[M_k]$,

   meaning that $F \in \widehat{Sched}_{\mathcal{C}}$ as required.

Definition 5.9 (Probabilistic Labelled Bisimulation) *Let M and N be two networks. An equivalence relation $\mathcal{R}$ over networks is a* probabilistic labelled bisimulation *w.r.t. $\mathcal{F}$ if $M\mathcal{R}N$ implies: for all scheduler $F \in \hat{\mathcal{F}}_{\mathcal{C}}$ there exists a scheduler $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that for all α and for all classes $\mathcal{C}$ in $\mathcal{N}/\mathcal{R}$ it holds:*

1. *if $\alpha = \tau$ or $\alpha = c!\tilde{v}@K \triangleleft R$ then $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = Prob^{F'}_N(\stackrel{\hat{\alpha}}{\Longrightarrow} \mathcal{C})$;*
2. *if $\alpha = c?@l$ or $\alpha = c?\vartheta@l$ then either $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = Prob^{F'}_N(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C})$ or $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = Prob^{F'}_N(\Longrightarrow, \mathcal{C})$.*

Probabilistic labelled bisimilarity*, written $\approx^{\mathcal{F}}_p$, is the largest probabilistic labelled bisimulation w.r.t. $\mathcal{F}$ over networks.*

In the following we prove that our probabilistic labelled bisimulation is a complete characterisation of our notion of probabilistic barbed congruence.

The following propositions will be useful.

Proposition 5.2 *Let M and N be two networks. If $M\mathcal{R}N$ for some bisimulation $\mathcal{R}$ w.r.t. $\mathcal{F}$, then for all scheduler $F \in \hat{\mathcal{F}}_{\mathcal{C}}$ there exists a scheduler $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that for all α and for all classes $\mathcal{C}$ in $\mathcal{N}/\mathcal{R}$ it holds:*

1. *if $\alpha = \tau$ or $\alpha = c!\tilde{v}@K \triangleleft R$ then $Prob^F_M(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}) = Prob^{F'}_N(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C})$;*

2. *if* $\alpha = c?@l$ *or* $\alpha = c?\vartheta@l$ *then either* $Prob^F_M(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}) = Prob^{F'}_N(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C})$ *or* $Prob^F_M(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}) = Prob^{F'}_N(\Longrightarrow, \mathcal{C})$.

Proof.

We proceed by induction on the length of the weak transition $\stackrel{\hat{\alpha}}{\Longrightarrow}$.

If M reaches $\mathcal{C}$ in one step then, since $M\mathcal{R}N$, $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that:

if $\alpha = \tau$ or $c!\tilde{v}@K \triangleleft R$, $Prob^F_M(\stackrel{\alpha}{\rightarrow}, \mathcal{C}) = Prob^{F'}_N(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C})$, while, if $\alpha = c?\vartheta@l$ or $c?@l$ $Prob^F_M(\stackrel{\alpha}{\rightarrow}, \mathcal{C}) = Prob^{F'}_N(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C})$, or $Prob^F_M(\stackrel{\alpha}{\rightarrow}, \mathcal{C}) = Prob^{F'}_N(\Longrightarrow, \mathcal{C})$, as required.

If M reaches $\mathcal{C}$ in more steps then we consider two cases:

- The first transition is α, and $M \stackrel{\alpha}{\rightarrow} [\![M']\!]_\theta$.

  $Prob^F_M(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}) = \sum_{\hat{M} \in spt([\![M']\!]_\theta)} (Prob^F_M(\stackrel{\alpha}{\rightarrow}, \hat{M}) \times Prob^F_{\hat{M}}(\Longrightarrow, \mathcal{C}))$.

  Now, if we partition the support of $[\![M']\!]_\theta$ in equivalence classes of $\mathcal{R}$, $\exists I$ such that $\forall i \in I$ $\mathcal{C}_i \in \mathcal{N}/\mathcal{R}$, $spt([\![M']\!]_\theta) \cap \mathcal{C}_i \neq \emptyset$, and $spt([\![M']\!]_\theta) \subseteq \bigcup_{i \in I} \mathcal{C}_i$. We get:

  $Prob^F_M(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}) = \sum_{i \in I} (Prob^F_M(\stackrel{\alpha}{\rightarrow}, \mathcal{C}_i) \times Prob^F_{Rep_{\mathcal{C}_i}}(\Longrightarrow, \mathcal{C}))$,

  where $Rep_{\mathcal{C}_i}$ is a representative element of the equivalence class C_i. Since $M\mathcal{R}N$, $\exists \hat{F} \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that, $\forall i \in I$: if $\alpha = \tau$ or $c!\tilde{v}@K \triangleleft R$, $Prob^F_M(\stackrel{\alpha}{\rightarrow}, \mathcal{C}_i) = Prob^{\hat{F}}_N(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}_i)$, while, if $\alpha = c?\vartheta@l$ or $c?@l$ $Prob^F_M(\stackrel{\alpha}{\rightarrow}, \mathcal{C}_i) = Prob^{\hat{F}}_N(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}_i)$, or $Prob^F_M(\stackrel{\alpha}{\rightarrow}, \mathcal{C}_i) = Prob^{\hat{F}}_N(\Longrightarrow, \mathcal{C}_i)$.

  If we take $F' \in LSched$ such that, $\forall e$ such that $e \leq_{prefix} e' \in Exec^{\hat{F}}_N(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}_i)$, $F'(e) = \hat{F}(e)$, and $\forall e$ such that $e \leq_{prefix} e' \in Exec^F_{Rep_{\mathcal{C}_i}}(\Longrightarrow, \mathcal{C})$ $F'(e) = F(e)$ we get, if $\alpha = \tau$ or $c!\tilde{v}@K \triangleleft R$,

  $$\begin{aligned} Prob^F_M(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}) &= \sum_{i \in I} (Prob^F_M(\stackrel{\alpha}{\rightarrow}, \mathcal{C}_i) \times Prob^F_{Rep_{\mathcal{C}_i}}(\Longrightarrow, \mathcal{C})) \\ &= \sum_{i \in I} (Prob^{F'}_N(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}_i) \times Prob^{F'}_{Rep_{\mathcal{C}_i}}(\Longrightarrow, \mathcal{C})) = Prob^{F'}_N(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}), \end{aligned}$$

  while, if $\alpha = c?\vartheta@l$ or $c?@l$:

  $$\begin{aligned} Prob^F_M(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}) &= \sum_{i \in I} (Prob^F_M(\stackrel{\alpha}{\rightarrow}, \mathcal{C}_i) \times Prob^F_{Rep_{\mathcal{C}_i}}(\Longrightarrow, \mathcal{C})) \\ &= \sum_{i \in I} (Prob^{F'}_N(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}_i) \times Prob^{F'}_{Rep_{\mathcal{C}_i}}(\Longrightarrow, \mathcal{C})) = Prob^{F'}_N(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}), \end{aligned}$$

  or

  $$= \sum_{i \in I} (Prob^{F'}_N(\Longrightarrow, \mathcal{C}_i) \times Prob^{F'}_{Rep_{\mathcal{C}_i}}(\Longrightarrow, \mathcal{C})) = Prob^{F'}_N(\Longrightarrow, \mathcal{C}),$$

  and, by Definition 5.8, since both $\hat{F}$, $F \in \hat{\mathcal{F}}_{\mathcal{C}}$, $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$, as required.

- The first transition is a τ, and $M \stackrel{\tau}{\rightarrow} [\![M']\!]_\theta$.

  The proof is analogous to the first item.

Proposition 5.3 *Let* $\mathcal{R} = (\bigcup_{i \in I} \mathcal{R}_i)^*$, *where* $\mathcal{R}_i$ *are Probabilistic Labelled Bisimulations w.r.t.* $\mathcal{F}$. *Then* $\mathcal{R}$ *is a Probabilistic Labelled Bisimulation w.r.t.* $\mathcal{F}$.

Proof.

Each relation $\mathcal{R}_i$ partition the set $\mathcal{N}$ in equivalence classes. If $(M,N) \in \mathcal{R}_i$, that means $(M,N) \in \mathcal{R}$, by definition of $\mathcal{R}$. Given then an equivalence class $\mathcal{C}^i \in \mathcal{N}/\mathcal{R}_i$, this is wholly contained in an equivalence class $\mathcal{C} \in \mathcal{N}/\mathcal{R}$. By partitioning the equivalence class $\mathcal{C}$ with a set of equivalence classes for $\mathcal{R}_i$, we can then deduce the existence of a set J such that: $\mathcal{C} = \bigcup_{j\in J} \mathcal{C}^i_j$.

Now, let consider $(M,N) \in \mathcal{R}$. That means $(M,N) \in (\bigcup_{i\in I} \mathcal{R}_i)^*$, and $(M,N) \in (\bigcup_{i\in I} \mathcal{R}_i)^n$ for some $n > 0$.

We will prove by induction over n that $\mathcal{R}$ is a probabilistic labelled bisimulation.

- $n = 1$.

  If $n = 1$, $(M,N) \in (\bigcup_{i\in I} \mathcal{R}_i)^1$ means that for some $i \in I$, $(M,N) \in \mathcal{R}_i$. We have that $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ s.t. $\forall \alpha, \mathcal{C} \in \mathcal{N}/\mathcal{R}$:

  If $\alpha = \tau$ or $c!\tilde{v}@K \triangleleft R$ then $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = \sum_{j\in J} Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}^i_j)$

  $$= \sum_{j\in J} Prob^{F'}_N(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}^i_j) = Prob^{F'}_N(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}),$$

  as required.

  If $\alpha = c?\vartheta$ or $c?@l$ then:

  $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = \sum_{j\in J} Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}^i_j)$

  $$= \sum_{j\in J} Prob^{F'}_N(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}^i_j) = Prob^{F'}_N(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}),$$

  or

  $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = \sum_{j\in J} Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}^i_j)$

  $$= \sum_{j\in J} Prob^{F'}_N(\Longrightarrow, \mathcal{C}^i_j) = Prob^{F'}_N(\Longrightarrow, \mathcal{C}),$$

  as required.

- $n > 1$.

  $(M,N) \in (\bigcup_{i\in I} \mathcal{R}_i)^n$ means that $\exists i \in I$ such that $M\mathcal{R}_i(\bigcup_{i\in I} \mathcal{R}_i)^{n-1}N$, and that $\exists O \in \mathcal{N}$ such that, $(M,O) \in \mathcal{R}_i$ and $(O,N) \in (\bigcup_{i\in I} \mathcal{R}_i)^{n-1}$.

  $(M,O) \in \mathcal{R}_i$ implies that, $\forall F \in \hat{\mathcal{F}}_{\mathcal{C}}$ $\exists F_1 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $\forall \mathcal{C} \in \mathcal{N}/\mathcal{R}$ and $\forall \alpha$:

  If $\alpha = \tau$ or $c!\tilde{v}@K \triangleleft R$ then $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = \sum_{j\in J} Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}^i_j)$

  $$= \sum_{j\in J} Prob^{F_1}_O(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}^i_j) = Prob^{F_1}_O(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}),$$

  while, if $\alpha = c?\vartheta@l$ or $c?@l$ then:

  $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = \sum_{j\in J} Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}^i_j)$

  $$= \sum_{j\in J} Prob^{F_1}_O(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}^i_j) = Prob^{F_1}_O(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}),$$

  or

  $Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}) = \sum_{j\in J} Prob^F_M(\xrightarrow{\alpha}, \mathcal{C}^i_j)$

$$= \sum_{j\in J} Prob_O^{F_1}(\Longrightarrow, \mathcal{C}_j^i) = Prob_O^{F_1}(\Longrightarrow, \mathcal{C}).$$

By induction hypothesis, $\forall m < n$, $(\bigcup_{i\in I} \mathcal{R}_i)^m$ is a bisimulation, hence $(\bigcup_{i\in I} \mathcal{R}_i)^{n-1}$ is a bisimulation. Again, since for each $(P,Q) \in (\bigcup_{i\in I} \mathcal{R}_i)^{n-1}$, $(P,Q) \in \mathcal{R}$, and each equivalence class of $(\bigcup_{i\in I} \mathcal{R}_i)^{n-1}$ is wholly contained in an equivalence class for $\mathcal{R}$, we can then partition $\mathcal{C}$ with a set of equivalence classes in $(\bigcup_{i\in I} \mathcal{R}_i)^{n-1}$ that means $\exists J'$ such that $\mathcal{C} = \bigcup_{j\in J'} \mathcal{C}_j$ where $\mathcal{C}_j \in \mathcal{N}/(\bigcup_{i\in I} \mathcal{R}_i)^{n-1}$ $\forall j \in J'$.

By proposition 5.2 we finally get that $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $\forall \mathcal{C} \in \mathcal{N}/\mathcal{R}$ and $\forall \alpha$:

If $\alpha = \tau$ or $c!\tilde{v}@K \triangleleft R$ then

$$Prob_M^F(\xrightarrow{\alpha}, \mathcal{C}) = Prob_O^{F_1}(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}) = \sum_{j\in J'} Prob_O^{F_1}(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}_j)$$
$$= \sum_{j\in J'} Prob_N^{F'}(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}_j) = Prob_N^{F'}(\stackrel{\hat{\alpha}}{\Longrightarrow}, \mathcal{C}),$$

while, if $\alpha = c?\vartheta@l$ or $c?@l$ then there are three different possibilities:

$$Prob_M^F(\xrightarrow{\alpha}, \mathcal{C}) = Prob_O^{F_1}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}) = \sum_{j\in J'} Prob_O^{F_1}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}_j)$$
$$= \sum_{j\in J'} Prob_N^{F'}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}_j) = Prob_N^{F'}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}),$$
$$Prob_M^F(\xrightarrow{\alpha}, \mathcal{C}) = Prob_O^{F_1}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}) = \sum_{j\in J'} Prob_O^{F_1}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}_j)$$
$$= \sum_{j\in J} Prob_N^{F'}(\Longrightarrow, \mathcal{C}_j) = Prob_O^{F'}(\Longrightarrow, \mathcal{C}),$$

or

$$Prob_M^F(\xrightarrow{\alpha}, \mathcal{C}) = Prob_O^{F_1}(\Longrightarrow, \mathcal{C}) = \sum_{j\in J'} Prob_O^{F_1}(\Longrightarrow, \mathcal{C}_j)$$
$$= \sum_{j\in J} Prob_N^{F'}(\Longrightarrow, \mathcal{C}_j) = Prob_O^{F'}(\Longrightarrow, \mathcal{C}).$$

Theorem 5.2 (Soundness) *Let M and N be two networks. We show that if $M \approx_p^{\mathcal{F}} N$ then $M \cong_p^{\mathcal{F}} N$.*

Proof.

In order to prove that bisimulation is a sound characterisation of Probabilistic Barbed Congruence we have to prove that $\approx_p^{\hat{\mathcal{F}}}$ is:

1. reduction closed w.r.t. $\mathcal{F}$
2. probabilistic barb preserving w.r.t. $\mathcal{F}$
3. contextual

Probabilistic Labelled Bisimulation is reduction closed.

We have to prove that if $M \approx_p^{\mathcal{F}} N$, then for all $F \in \mathcal{F}_{\mathcal{C}}$, there exists $F' \in \mathcal{F}_{\mathcal{C}}$ such that for all classes $\mathcal{C} \in \mathcal{N}/\mathcal{R}$, $Prob_M^F(\mathcal{C}) = Prob_N^{F'}(\mathcal{C})$.

By Theorem 5.1 there exists an admissible scheduler $\hat{F} \in LSched$ such that $Prob_M^F(\mathcal{C}) = Prob_M^{\hat{F}}(\Longrightarrow, \mathcal{C}')$, where $\mathcal{C}' = \mathcal{C} \cup \{M' : M' \equiv M'' \in \mathcal{C}\}$, but since

$\forall M'$ such that $M' \equiv M'' \in \mathcal{C}$, by applying rule (R-Struct) $M' \cong_p^{\mathcal{F}} M''$, we get $\{M' : M' \equiv M'' \in \mathcal{C}\} \subseteq \mathcal{C}$, that means $\mathcal{C}' = \mathcal{C}$.

By Definition 5.8 we deduce $\hat{F} \in \hat{\mathcal{F}}_{\mathcal{C}}$, since for all the executions in $Exec_M^{\hat{F}}(\Longrightarrow, \mathcal{C})$, the correspondent reduction executions are allowed by F, which is an element of $\mathcal{F}_{\mathcal{C}}$.

By Proposition 5.2 we have that $\exists \hat{F}' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $Prob_M^{\hat{F}}(\Longrightarrow, \mathcal{C}) = Prob_N^{\hat{F}'}(\Longrightarrow, \mathcal{C})$.

Finally, by Theorem 5.1, $\exists F' \in Sched$ such that $Prob_N^{\hat{F}'}(\Longrightarrow, \mathcal{C}) = Prob_N^{F'}(\mathcal{C})$. Finally, we can deduce $F' \in \mathcal{F}_{\mathcal{C}}$ by applying Definitions 5.8 and 5.3.

Probabilsitc Lablelled Bisimulation is Probabilistic barb preserving.

To prove that bisimulation is Probabilistic barb preserving we have to show that, if $M \approx_p^{\mathcal{F}} N$, then, for each scheduler $F \in \mathcal{F}_{\mathcal{C}}$, for each channel c, and for each set K of locations such that $M \Downarrow_p^F c@K$, then $\exists F' \in \mathcal{F}_{\mathcal{C}}$ such that $N \Downarrow_p^{F'} c@K$.

Let $M \Downarrow_p^F c@K$ for some channel c, some set K of locations, and scheduler $F \in \mathcal{F}_{\mathcal{C}}$. It means that $Prob_M^F(H) = p$, where $M' \in H$ iff $M' \downarrow_{c@K}$. We can partition H in a set of equivalence classes for $\approx_p^{\mathcal{F}}$. Hence $\exists I$ such that $\forall i \in I\ \mathcal{C}_i \in \mathcal{N}/\approx_p^{\mathcal{F}}, \mathcal{C}_i \cap H \neq \emptyset$, and $H \subseteq \bigcup_{i \in I} \mathcal{C}_i$. We get:

$Prob_M^F(H) = \sum_{e \in Exec_M^F(H)} P_M^F(e) = \sum_{i \in I} Prob_M^F(\mathcal{C}_i) = p$.

by Theorem 5.1, $\exists \hat{F} \in LSched$ such that $\forall i \in I$:

$Prob_M^F(\mathcal{C}_i) = Prob_M^{\hat{F}}(\Longrightarrow, \mathcal{C}_i')$,

where $\mathcal{C}_i' = \mathcal{C}_i \cup \{M' \mid \exists M'' \in \mathcal{C}_i \text{ and } M' \equiv M''\}$, but, since $\cong_p^{\mathcal{F}}$ is closed under structural congruence, $\forall M' \equiv M'' \in \mathcal{C}_i$, $M' \cong_p^{\mathcal{F}} M''$, hence $\{M' : M' \equiv M'' \in \mathcal{C}_i\} \subseteq \mathcal{C}_i$, that means $\mathcal{C}_i' = \mathcal{C}_i$. Now we have to prove that $\hat{F} \in \hat{\mathcal{F}}_{\mathcal{C}}$, but this follows straightforwardly by Definition 5.8. Hence:

$Prob_M^F(\mathcal{C}_i) = Prob_M^{\hat{F}}(\Longrightarrow, \mathcal{C}_i)\ \forall i \in I$ and

$\sum_{i \in I} Prob_M^F(\mathcal{C}_i) = \sum_{i \in I} Prob_M^{\hat{F}}(\Longrightarrow, \mathcal{C}_i)$.

Since $M \approx_p^{\mathcal{F}} N$, $\exists \hat{F}' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that, $\forall i \in I$:

$Prob_M^{\hat{F}}(\Longrightarrow, \mathcal{C}_i) = Prob_N^{\hat{F}'}(\Longrightarrow, \mathcal{C}_i)$.

Again by Lemma 5.1 $\exists F' \in LSched$ such that:

$p = \sum_{i \in I} Prob_N^{\hat{F}'}(\Longrightarrow, \mathcal{C}_i) = \sum_{i \in I} Prob_N^{F'}(\mathcal{C}_i) = Prob_N^{F'}(H)$,

that means $N \Downarrow_p^{F'} c@K$.

By Definition 5.8 we finally deduce $F' \in \mathcal{F}_{\mathcal{C}}$, as required. *Probabilistic Labelled Bisimulation is contextual*

We start with the *Parallel Composition.* Let $\mathcal{R}$ be the following relation:

$$\mathcal{R} = \{(M \mid O, N \mid O) : M, N, M \mid O, N \mid O \text{ are well-formed, and }, M \approx_p^{\mathcal{F}} N\}.$$

We will prove that it is a Probabilistic Labelled bisimulation w.r.t. $\mathcal{F}$. For this purpose, we need to prove that, $\forall F \in \hat{\mathcal{F}}_{\mathcal{C}}\ \exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that, $\forall \mathcal{C} \in \mathcal{N}/\mathcal{R}$, $\forall \alpha$:

1. $\alpha = \tau$ then $Prob_{M|O}^F(\xrightarrow{\tau}, \mathcal{C}) = Prob_{N|O}^{F'}(\Longrightarrow, \mathcal{C})$.

If $P, Q \in \mathcal{C}$, then, by definition of $\mathcal{R}$, $P \equiv \bar{P} \mid \bar{O}$, $Q \equiv \bar{Q} \mid \bar{O}$ and $\bar{P} \approx_p^{\mathcal{F}} \bar{Q}$. But then there exists $\mathcal{D} \in \mathcal{N}/\approx_p^{\mathcal{F}}$ such that $\mathcal{D} = \{\bar{P} : \bar{P} \mid \bar{O} \in \mathcal{C}\}$. Now we have three cases to consider:

(i) if $M \mid O \xrightarrow{\tau} [\![M \mid O']\!]_\theta$ the proof is simple, because we have, $\forall \bar{M}$ in the support of $[\![M \mid O']\!]_\theta$, such that $\bar{M} \in \mathcal{C}$, $\bar{M} \equiv M \mid O''$ and, since $M \approx_p^{\mathcal{F}} N$, $N \mid O'' \in \mathcal{C}$ too, by definition of $\mathcal{R}$. Hence (by applying rule (Par) to the action $O \xrightarrow{\tau} [\![O']\!]_\theta$), since $N \mid O$ is well-formed, $\exists F' \in LSched$ such that

$$Prob^{F}_{M|O}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F'}_{N|O}(\Longrightarrow, \mathcal{C}).$$

We have only to prove that $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$, but the proof follows straightforwardly by the Definitions 5.3 and 5.8.

(ii) If $M \mid O \xrightarrow{\tau} [\![M' \mid O]\!]_\theta$, since M is well-formed, by Definition 5.8 $\exists F_1 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $Prob^{F}_{M|O}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F_1}_{M}(\xrightarrow{\tau}, \mathcal{D})$. But since $M \approx_p^{\mathcal{F}} N$, and N is well-formed, $\exists F_2 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $Prob^{F_1}_{M}(\xrightarrow{\tau}, \mathcal{D}) = Prob^{F_2}_{N}(\Longrightarrow, \mathcal{D})$. Again, since the network $N \mid O$ is well-formed, $\exists F' \in LSched$ such that, by applying rule (Par) to the executions in $Exec^{F_2}_{N}(\Longrightarrow, \mathcal{D})$, we get

$$Prob^{F_2}_{N}(\Longrightarrow, \mathcal{D}) = Prob^{F'}_{N|O}(\Longrightarrow, \mathcal{C}).$$

Since by Definitions 5.8 each execution in the set $Exec^{F_2}_{N}(\Longrightarrow, \mathcal{D})$ has a correspondent reduction execution allowed by $\mathcal{F}_{\mathcal{C}}$, and by Definition 5.3 we know that the same executions can be performed by N when interacting with any context, we can finally deduce, by applying again Definition 5.8, that $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$, as required.

(iii) If $M \mid O \xrightarrow{\tau} M' \mid O'$ due to a synchronization between M and O, then there are two cases to consider.
If $M \xrightarrow{c!\tilde{v}[l,r]} [\![M']\!]_\Delta$ and $O \xrightarrow{c?\tilde{v}@k} [\![O']\!]_\Delta$, for some message $\tilde{v}$, channel c, locations l, k and radius r, such that $d(l,k) \leq r$, we can apply rule (Obs) obtaining $M \xrightarrow{c!\tilde{v}@K \triangleleft R} M'$ for some $K \subseteq L$ and for some R, with $k \in R$. Therefore, $\exists F_1 \in LSched$ such that:

$$Prob^{F}_{M|O}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F_1}_{M}(\xrightarrow{c!\tilde{v}@K \triangleleft R}, \mathcal{D}).$$

By Definition 5.8 we deduce $F_1 \in \hat{\mathcal{F}}_{\mathcal{C}}$ and, since $N \approx_p^{\mathcal{F}} M$, $\exists F_2 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that

$$Prob^{F_1}_{M}(\xrightarrow{c!\tilde{v}@K \triangleleft R}, \mathcal{D}) = Prob^{F_2}_{N}(\overset{c!\tilde{v}@K \triangleleft R}{\Longrightarrow}, \mathcal{D}),$$

where each execution e in $Exec^{F_2}_{N}(\overset{c!\tilde{v}@K \triangleleft R}{\Longrightarrow}, \mathcal{D})$ is of the form

$$e = N \xrightarrow{\tau}_{\theta_1} N_1 \rightarrow ... N_{i-1} \xrightarrow{c!\tilde{v}@K \triangleleft R}_{\Delta} N_i \rightarrow ... N',$$

and, by applying rule (Obs) backwardly, $N_{i-1} \xrightarrow{c!\tilde{v}[l',r']}_{\Delta} N_i$ for some l' and r' such that $d(l',k) \leq r'$. We can apply rule (Bcast) obtaining $N_{i-1} \mid O \xrightarrow{c!\tilde{v}[l',r']}_{\Delta} N_i \mid O'$ without changing probability. Finally if we take $F' \in LSched$ which applies rule (Lose2) to the output action, we obtain the required result:

$$Prob_N^{F_2}(\stackrel{c!\tilde{v}@K\triangleleft R}{\Longrightarrow}, \mathcal{D}) = Prob_{N|O}^{F'}(\Longrightarrow, \mathcal{C}).$$

We have finally to prove that $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$. We start by the consideration that, by Definition 5.1, for any execution of the form $\stackrel{\alpha}{\Longrightarrow}$ in $\hat{\mathcal{F}}_{\mathcal{C}}$, where α is a silent or an output action there exists a correspondent reduction in $\mathcal{F}_{\mathcal{C}}$. Since by Definition 5.3, for any context, there exists a scheduler in $\mathcal{F}_{\mathcal{C}}$ mimicking the behaviour exhibited by N when interacting with the given context, we can affirm that $\exists \bar{F} \in \mathcal{F}_{\mathcal{C}}$ such that $Exec_{N|O}^{\bar{F}}$ contains all the reductions corresponding to the executions of $Exec_{N|O}^{F'}$. Hence, by Definition 5.8, $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$, as required.

If $M \xrightarrow{c?\tilde{v}@k} [\![M']\!]_{\Delta}$ and $O \xrightarrow{c_L!\tilde{v}[l,r]} [\![O']\!]_{\Delta}$, for some message $\tilde{v}$, some set L of locations, some channel c, some locations l, k and radius r, such that $d(l,k) \leq r$, then $\exists F_1 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that:

$$Prob_{M|O}^{F}(\xrightarrow{\tau}, \mathcal{C}) = Prob_M^{F_1}(\xrightarrow{c?\tilde{v}@k}, \mathcal{D}).$$

Since $N \approx_p^{\mathcal{F}} M$, $\exists F_2 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that:

$$Prob_M^{F_1}(\xrightarrow{c?\tilde{v}@k}, \mathcal{D}) = Prob_N^{F_2}(\stackrel{c?\tilde{v}@k}{\Longrightarrow}, \mathcal{D})$$

or

$$Prob_M^{F_1}(\xrightarrow{c?\tilde{v}@k}, \mathcal{D}) = Prob_N^{F_2}(\Longrightarrow, \mathcal{D}).$$

In the first case, since by hypothesis $k \in R$ and $N \mid O$ is well-formed, also N is able to synchronize with O. Hence $\exists F' \in LSched$ such that for all

$$e = N \xrightarrow{\tau}_{\theta_1} N_1 \rightarrow ...N_{i-1} \xrightarrow{c?\tilde{v}@k} N_i \rightarrow ...N' \in Exec_N^{F_2}(\stackrel{c?\tilde{v}@k}{\Longrightarrow}, \mathcal{D})$$

there exists a matching execution such that, by applying rule (Bcast) $N_{i-1} \mid O \xrightarrow{c!\tilde{v}[l,r]} N_i \mid O$, and by applying rule (Lose2), we get:

$$e' = N \mid O \xrightarrow{\tau}_{\theta_1} N_1 \mid O \rightarrow ...N_{i-1} \mid O \xrightarrow{\tau} N_i \mid O' \rightarrow ...N' \mid O'$$

in $Exec_{N|O}^{F'}(\Longrightarrow, \mathcal{C})$. Hence,

$$Prob_N^{F_2}(\stackrel{c?\tilde{v}@k}{\Longrightarrow}, \mathcal{D}) = Prob_{N|O}^{F'}(\Longrightarrow, \mathcal{C}).$$

In order to prove $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$, we start by the consideration that, since $O \xrightarrow{c_L!\tilde{v}[l,r]} [\![O']\!]_\Delta$, by Definition 5.3, for any context, there exists a scheduler in $\mathcal{F}_{\mathcal{C}}$ mimicking the behaviour of O in its interaction with the given context. Then we can affirm that $\exists \bar{F} \in \mathcal{F}_{\mathcal{C}}$ such that $Exec^{\bar{F}}_{N|O}$ contains all the reductions corresponding to the executions of $Exec^{F'}_{N|O}$. Hence, by Definition 5.8, $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$, as required.
If N is not able to receive the message the proof is analogous: it is sufficient to apply the rule (Par) to $O \xrightarrow{c!\tilde{v}@K \triangleleft R} [\![O']\!]_\Delta$, obtaining:

$$Prob^{F_2}_{N}(\Longrightarrow, \mathcal{D}) = Prob_{N|O}(\Longrightarrow, \mathcal{C}).$$

2. $\alpha = c!\tilde{v}@K \triangleleft R$

   The proof is analogous to the point **(iii)** of the previous item.

3. $\alpha = c?@k$ then
   $Prob^{F}_{M|O}(\xrightarrow{\alpha}, \mathcal{C}) = Prob^{F'}_{N|O}(\overset{\alpha}{\Longrightarrow}, \mathcal{C})$ or $Prob^{F}_{M|O}(\xrightarrow{\alpha}, \mathcal{C}) = Prob^{F'}_{N|O}(\Longrightarrow, \mathcal{C})$.

   If $P, Q \in \mathcal{C}$, then $P \equiv \bar{M} \mid \bar{O}$, $Q \equiv \bar{N} \mid \bar{O}$ and $\bar{M} \approx^{\mathcal{F}}_{p} \bar{N}$. But then $\exists \mathcal{D} \in \mathcal{N}/\approx^{\mathcal{F}}_{p}$ such that $\mathcal{D} = \{\bar{M} : \bar{M} \mid \bar{O} \in \mathcal{C}\}$. Now we have two cases to consider:

   (i) The transition is due to an action performed by O, hence $O \xrightarrow{\alpha}_\Delta O'$ and $M \mid O' \in \mathcal{C}$. But since $M \approx^{\mathcal{F}}_{p} N$, $N \mid O' \in \mathcal{C}$ too, $\exists F' \in LSched$ such that by applying parallel composition to the input of O, we obtain the desired result:

   $$Prob^{F}_{M|O}(\xrightarrow{\alpha}, \mathcal{C}) = Prob^{F'}_{N|O}(\overset{\alpha}{\Longrightarrow}, \mathcal{C}).$$

   Finally, by Definition 5.8 we deduce $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$, as required.

   (ii) The transition is due to an action performed by M, in this case, by Definition 5.8 $\exists F_1 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that:
   $Prob^{F}_{M|O}(\xrightarrow{\alpha}, \mathcal{C}) = Prob^{F_1}_{M}(\xrightarrow{\alpha}, \mathcal{D})$. Since $M \approx^{\mathcal{F}}_{p} N$ $\exists F_2 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that:
   $Prob^{F_1}_{M}(\xrightarrow{\alpha}, \mathcal{D}) = Prob^{F_2}_{N}(\overset{\alpha}{\Longrightarrow}, \mathcal{D})$,
   or
   $Prob^{F_1}_{M}(\xrightarrow{\alpha}, \mathcal{D}) = Prob^{F_2}_{N}(\Longrightarrow, \mathcal{D})$.
   In both cases, since $N \mid O$ is well-formed, $\exists F' \in LSched$ such that by applying parallel composition, we have:
   $Prob^{F_2}_{N}(\overset{\alpha}{\Longrightarrow}, \mathcal{D}) = Prob^{F'}_{N|O}(\overset{\alpha}{\Longrightarrow}, \mathcal{C})$,
   or
   $Prob^{F_2}_{N}(\Longrightarrow, \mathcal{D}) = Prob^{F'}_{N|O}(\Longrightarrow, \mathcal{C})$.

In order to prove that $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$, we start by the consideration that, by Definition 5.8 there exists at least a context $C[\cdot]$ and $\exists \bar{F} \in \mathcal{F}_{\mathcal{C}}$ such that $C[N] \rightarrow C'[N']$, and, by the reduction rules we get:

$$C[\cdot] \equiv (\nu \tilde{d}) m[\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_l \mid M_1$$

for some $\tilde{d}$ such that $c \notin \tilde{d}$, some m, some set L of locations, some process P, some (possibly empty) network M_1, some location l and some radius r such that $d(l,k) \leq r$. Then, by Definition 5.3 we have that there exists a scheduler allowing $m[\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_l \rightarrow [\![m[P]_l]\!]_\Delta$, and again by Definition 5.3 there exists a scheduler allowing the reduction $m[\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_l \mid N \mid O \rightarrow^* [\![m[P]_l \mid N' \mid O']\!]_\Delta$, meaning, by Definition 5.8, $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ as required.

4. $\alpha = c?\vartheta@k$ the proof is analogous as for $\alpha = c?@k$.

Now we proceed with the *Restriction.* Let $\mathcal{R} = \{((\nu d)M, (\nu d)N) : M \approx_p^{\mathcal{F}} N\}$ be a relation. We need to prove that it is a Probabilistic Labelled bisimulation w.r.t. $\mathcal{F}$.

Let consider $\mathcal{C}$: if $P, Q \in \mathcal{C}$, by definition of $\mathcal{R}$, $P \equiv (\nu \bar{d})\bar{P}$, $Q \equiv (\nu \bar{d})\bar{Q}$ and $\bar{P} \approx_p^{\mathcal{F}} \bar{Q}$. But then $\exists \mathcal{D} \in \mathcal{N}/\approx_p^{\mathcal{F}}$ such that $\mathcal{D} = \{\tilde{P} : (\nu \bar{d})\bar{P} \in \mathcal{C}\}$.

We have to prove that, $\forall F \in \hat{\mathcal{F}}_{\mathcal{C}}\ \exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that, $\forall \mathcal{C} \in \mathcal{N}/\mathcal{R}$, $\forall \alpha$:

1. $\alpha = \tau$ implies that $Prob^{F}_{(\nu d)M}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F'}_{(\nu d)N}(\Longrightarrow, \mathcal{C})$.

   Since $\mathtt{Chan}(\tau) = \perp$, by Definition 5.8 $\exists F_1 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $Prob^{F}_{(\nu d)M}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F_1}_{M}(\xrightarrow{\tau}, \mathcal{D})$ and, since $M \approx_p^{\mathcal{F}} N$ $\exists F_2 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that: $Prob^{F_1}_{M}(\xrightarrow{\tau}, \mathcal{D}) = Prob^{F_2}_{N}(\Longrightarrow, \mathcal{D})$.

   Finally we can take $F' \in LSched$ mimicking the executions in $Exec^{F_2}_{N}(\Longrightarrow, \mathcal{D})$, when applying the restriction on N. Hence:

   $$Prob^{F_2}_{N}(\Longrightarrow, \mathcal{D}) = Prob^{F'}_{(\nu d)N}(\Longrightarrow, \mathcal{C}).$$

   In order to prove that $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$, we start by the consideration that, by Definition 5.3, for any context there exists a scheduler in $\mathcal{F}_{\mathcal{C}}$ mimicking the behaviour of N when interacting with the given context. Hence $\exists \bar{F} \in \mathcal{F}_{\mathcal{C}}$ such that $Exec^{\bar{F}}_{(\nu d)N}$ contains all the reductions corresponding to the executions in $Exec^{F'}_{(\nu d)N}$, meaning, by Definition 5.8, $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ as required.

2. $\alpha = c!\tilde{v}@K \triangleleft R$

   Since $d \neq c$, by Definition 5.8 $\exists F_1 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $Prob^{F}_{(\nu d)M}(\xrightarrow{\alpha}, \mathcal{C}) = Prob^{F_1}_{M}(\xrightarrow{\alpha}, \mathcal{D})$, then since $M \approx_p^{\mathcal{F}} N$, $\exists F_2 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that

   $Prob^{F_1}_{M}(\xrightarrow{\alpha}, \mathcal{D}) = Prob^{F_2}_{N}(\xRightarrow{\alpha}, \mathcal{D})$.

Therefore, since $\mathtt{Chan}(\alpha) \neq d$, $\exists F' \in LSched$ such that:

$$Prob_N^{F_2}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{D}) = Prob_{(\nu d)N}^{F'}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C}).$$

Again, we prove that $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ as for the previous case.

3. $\alpha = c?@k$

   Again, since $d \neq c$, by Definition 5.8 $\exists F_1 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $Prob_{(\nu d)M}^{F}(\stackrel{\alpha}{\rightarrow}, \mathcal{C}) = Prob_M^{F_1}(\stackrel{\alpha}{\rightarrow}, \mathcal{D})$. Since $M \approx_p^{\mathcal{F}} N$, there exists $F_2 \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that

   $Prob_M^{F_1}(\stackrel{\alpha}{\rightarrow}, \mathcal{D}) = Prob_N^{F_2}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{D})$ or

   $Prob_M^{F_1}(\stackrel{\alpha}{\rightarrow}, \mathcal{D}) = Prob_N^{F_2}(\Longrightarrow, \mathcal{D})$.

   In both cases we can apply rule (Res) to N, since $\mathtt{Chan}(\tau) \neq \mathtt{Chan}(\alpha) \neq d$. Therefore, there exists $F' \in LSched$ such that the required result holds, that is

   $Prob_N^{F_2}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{D}) = Prob_{(\nu d)N}^{F'}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C})$ or

   $Prob_N^{F_2}(\Longrightarrow, \mathcal{D}) = Prob_{(\nu d)N}^{F'}(\Longrightarrow, \mathcal{C})$.

   In order to prove that $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ we proceed as for the previous cases.

4. $\alpha = c?\vartheta@k$

   The proof is analogous as for $\alpha = c?@k$.

Theorem 5.3 (Completeness) *If $M \cong_p^{\mathcal{F}} N$ then $M \approx_p^{\mathcal{F}} N$.*

Proof.

In order to prove the completeness of bismulation we show that the relation $\mathcal{R} = \{(M, N) : M \cong_p^{\mathcal{F}} N\}$ is a Probabilistic Labelled Bisimulation. We have to prove that, $\forall F \in \hat{\mathcal{F}}_{\mathcal{C}}$ $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that, $\forall \mathcal{C} \in \mathcal{N}/\mathcal{R}$, $\forall \alpha$:

if $\alpha = \tau$ then $Prob_M^{F}(\stackrel{\tau}{\rightarrow}, \mathcal{C}) = Prob_N^{F'}(\Longrightarrow, \mathcal{C})$.

By Theorem 5.1 we know that there exists a scheduler $\bar{F} \in Sched$ such that $Prob_M^{F}(\stackrel{\tau}{\rightarrow}, \mathcal{C}) = Prob_M^{\bar{F}}(\mathcal{C})$, and, by Definition 5.8 we deduce $\bar{F} \in \mathcal{F}_{\mathcal{C}}$. Since $M \cong_p^{\mathcal{F}} N$, $\exists \bar{F}' \in \mathcal{F}_{\mathcal{C}}$ such that $Prob_M^{\bar{F}}(\mathcal{C}) = Prob_N^{\bar{F}'}(\mathcal{C})$. Again by Theorem 5.1 and Definition 5.8, there exists $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $Prob_N^{\bar{F}'}(\mathcal{C}) = Prob_N^{F'}(\Longrightarrow, \mathcal{C} \cup \{\bar{N} \equiv N' \in \mathcal{C}\})$, but since $\cong_p^{\mathcal{F}}$ is closed under structural equivalence, $\forall \bar{N} \equiv N' \in \mathcal{C}$, $\bar{N} \in \mathcal{C}$, hence: $Prob_M^{F}(\stackrel{\tau}{\rightarrow}, \mathcal{C}) = Prob_N^{F'}(\Longrightarrow, \mathcal{C})$.

if $\alpha = c!\tilde{v}@K \triangleleft R$ then $Prob_M^{F}(\stackrel{\alpha}{\rightarrow}, \mathcal{C}) = Prob_N^{F'}(\stackrel{\alpha}{\Longrightarrow}, \mathcal{C})$.

First we notice that $Prob_M^{F}(\xrightarrow{c!\tilde{v}@K \triangleleft R}, \mathcal{C})$ is either 0 or 1.

If $Prob^F_M(\xrightarrow{c!\tilde{v}@K\triangleleft R}, \mathcal{C}) = 0$ we are done, because it will be enough to take any scheduler $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ not allowing observable output actions on the channel c, and we get $Prob^F_M(\xrightarrow{c!\tilde{v}@K\triangleleft R}, \mathcal{C}) = Prob^{F'}_N(\stackrel{c!\tilde{v}@K\triangleleft R}{\Longrightarrow}, \mathcal{C})$.

If $Prob^F_M(\xrightarrow{c!\tilde{v}@K\triangleleft R}, \mathcal{C}) = 1$, then, by Definition 5.8 there exists a scheduler $\bar{F} \in \mathcal{F}_{\mathcal{C}}$ such that $M \Downarrow^{\bar{F}}_1 c@K$, and it means that $\exists \bar{F}' \in \mathcal{F}_{\mathcal{C}}$ such that $N \Downarrow^{\bar{F}'}_1 c@K$, hence $\exists R'$ such that $K \subseteq R'$ and $N \stackrel{c!\tilde{v}@K\triangleleft R'}{\Longrightarrow}$. Now in order to mimic the effect of the action $c!\tilde{v}@K \triangleleft R$, we build the following context

$$C[\cdot] = \prod\nolimits_{i=1}^{n} (n_i[\mathtt{in}(c, \tilde{x_i}).[\tilde{x_i} = \tilde{v}]\mathtt{out}\langle \mathtt{f}_{ik_i,r}, \tilde{x_i}\rangle]_{k_i} \mid m_i[\mathtt{in}(\mathtt{f}_i, \tilde{y_i}).\mathtt{out}\langle \mathtt{ok}_{ik_i,r}, \tilde{y_i}\rangle]_{k_i}),$$

where $R = \{k_1, ..., k_n\}$ and $\mathtt{f}_i$ and $\mathtt{ok}_i$ fresh $\forall i \in [1 - n]$.

Since $M \xrightarrow{c!\tilde{v}@K\triangleleft R}$, then the message is reachable by all nodes n_i, hence, by Definition 5.3, which captures the behaviour of a network when interacting in any context, since $C[M]$ is well-formed, $\exists \bar{F}_1 \in \mathcal{F}_{\mathcal{C}}$ such that $C[M] \rightarrow^* \bar{M}$, where

$$\bar{M} \equiv M' \mid \prod\nolimits_{i=1}^{n} (n_i[\mathbf{0}]_{k_i} \mid m_i[\mathtt{out}\langle \mathtt{ok}_{ik_i,r}, \tilde{v_i}\rangle]_{k_i}),$$

with $\bar{M} \not\downarrow_{\mathtt{f}_i@R}$ and $\bar{M} \Downarrow^{\bar{F}_1}_1 \mathtt{ok}_i@R$, $\forall i \in [1 - n]$.

The absence of the barb on the channels $\mathtt{f}_i$ together with the presence of the barb on the channels $\mathtt{ok}_i$ ensures that all the locations in R have been able to receive the message. Since $C[M] \cong^{\mathcal{F}}_p C[N]$, $\exists \bar{F}_2 \in \mathcal{F}_{\mathcal{C}}$ such that $Prob^{\bar{F}_1}_{C[M]}(\mathcal{C}') = Prob^{\bar{F}_2}_{C[N]}(\mathcal{C}')$ where $\bar{M} \in \mathcal{C}'$.

Therefore, $C[N] \rightarrow^* \bar{N}$ with $\bar{N} \not\downarrow_{\mathtt{f}@R}$ and $\bar{N} \Downarrow^{\bar{F}_2}_1 \mathtt{ok}@R$. The constrains on the barbs allow us to deduce that

$$\bar{N} \equiv N' \mid \prod\nolimits_{i=1}^{n} (n_i[\mathbf{0}]_{k_i} \mid m_i[\mathtt{out}\langle \mathtt{ok}_{ik_i,r}, \tilde{v_i}\rangle]_{k_i})$$

which implies $N \stackrel{c!\tilde{v}@K\triangleleft R}{\Longrightarrow} N'$, or $N \Longrightarrow N'$ in case (Lose2) has been applied to the output action on the channel c. Since $\bar{M}, \bar{N} \in \mathcal{C}$, then $\bar{M} \cong^{\mathcal{F}}_p \bar{N}$. Since $\cong^{\mathcal{F}}_p$ is contextual, it results $(\nu \mathtt{ok})\bar{M} \cong^{\mathcal{F}}_p (\nu \mathtt{ok})\bar{N}$, from which we can derive that $M' \cong^{\mathcal{F}}_p N'$. But since $N' \in \mathcal{C}$ and $N \stackrel{c!\tilde{v}@K\triangleleft R}{\Longrightarrow} N'$, then, by Definition 5.8 $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that:

$$Prob^{F'}_N(\stackrel{c!\tilde{v}@K\triangleleft R}{\Longrightarrow}, \mathcal{C}) = 1 = Prob^F_M(\stackrel{c!\tilde{v}@K\triangleleft R}{\Longrightarrow}, \mathcal{C}).$$

if $\alpha = c?@k$ then we notice that $Prob^F_M(\xrightarrow{c?\tilde{v}@k}, \mathcal{C})$ is either 0 or 1.

If $Prob^F_M(\xrightarrow{c?@k}, \mathcal{C}) = 0$ we are done, because it will be enough to take any scheduler $F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ not allowing input actions on the channel c, and we get $Prob^F_M(\xrightarrow{c?@k}, \mathcal{C}) = Prob^{F'}_N(\stackrel{c?@k}{\Longrightarrow}, \mathcal{C})$.

If $Prob_M^F(\xrightarrow{c?@k}, \mathcal{C}) = 1$, because $M \xrightarrow{c?@k} [\![M']\!]_\Delta$, by Definition 5.3 there exists at least a context $C[\cdot]$ and $\exists \bar{F} \in \mathcal{F}_\mathcal{C}$ such that $C[M] \to C'[M']$, and by Theorem 5.1 we deduce that:

$$C[\cdot] \equiv (\nu \tilde{d}) m[\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_l \mid M_1,$$

$$C'[\cdot] \equiv (\nu \tilde{d}) m[\bar{c}_{L,r}\langle \tilde{v}\rangle.P]_l \mid M_1',$$

for some m, some tuple $\tilde{d}$ of channel such that $c \notin \tilde{d}$, dome set L of messages, some radius r, some process P, some location l such that $d(l,k) \leq r$ and some (possibly empty) network M_1 and M_1'.

By Definition 5.3, for any context there exists a scheduler in $\mathcal{F}_\mathcal{C}$ allowing m to perform the output when interacting with any context. Hence we can build the following context:

$$C_1[\cdot] = \cdot \mid m[\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_l \mid m_1[\mathtt{in}(c, \tilde{x}).\mathtt{out}\langle \mathtt{f}_{k,r'}, \tilde{x}\rangle.\mathtt{out}\langle \mathtt{ok}_{k,r'}, \tilde{x}\rangle]_k,$$

in order to mimic the behaviour of the networks, with m static, $\mathtt{f}$ and $\mathtt{ok}$ fresh, $r' > 0$ and $d(l,k) > r'$ $\forall l \in \mathbf{Loc}$ s.t. $l \neq k$. There exists a scheduler $\bar{F}_1 \in \mathcal{F}_\mathcal{C}$ such that:

$$C_1[M] \to^* M' \mid m[P]_l \mid m_1[\mathtt{out}\langle \mathtt{ok}_{k,r'}, \tilde{v}\rangle]_k \in Exec^{\bar{F}_1}_{C[M]},$$

with $M' \mid m[P]_l \mid m[\mathtt{out}\langle \mathtt{ok}_{k,r'}, \tilde{v}\rangle]_k \not\downarrow_{\mathtt{f}@k}$ and $M' \mid m[P]_l \mid m[\mathtt{out}\langle \mathtt{ok}_{k,r'}, \tilde{v}\rangle]_k \Downarrow_1^{\bar{F}_1} \mathtt{ok}@k$.

The reduction sequence above must be matched by a corresponding reduction sequence $C_1[N] \to^* N' \mid m[P]_l \mid m[\mathtt{out}\langle \mathtt{ok}_{k,r'}, \tilde{v}\rangle]_k$, with

$M' \mid m[P]_l \mid m[\mathtt{out}\langle \mathtt{ok}_{k,r'}, \tilde{v}\rangle]_k \cong_p N' \mid m[P]_l \mid m[\mathtt{out}\langle \mathtt{ok}_{k,r'}, \tilde{v}\rangle]_k$,

$N' \mid m[P]_l \mid m[\mathtt{out}\langle \mathtt{ok}_{k,r'}, \tilde{v}\rangle]_k \not\downarrow_{\mathtt{f}@k}$ and $N' \mid m[P]_l \mid m[\mathtt{out}\langle \mathtt{ok}_{k,r'}, \tilde{v}\rangle]_k \Downarrow_1^{\hat{F}_2} \mathtt{ok}@k$ for some $\bar{F}_2 \in \mathcal{F}_\mathcal{C}$.

This does not ensure that N actually performed the input action, but we can conclude that there exists $F' \in LSched$ and N' such that either $N \overset{c?@k}{\Longrightarrow} N'$ or $N \Longrightarrow N'$. Since $M' \mid m[P]_l \mid m[\mathtt{out}\langle \mathtt{ok}_{k,r'}, \tilde{v}\rangle]_k \cong_p N' \mid m[P]_l \mid m[\mathtt{out}\langle \mathtt{ok}_{k,r'}, \tilde{v}\rangle]_k$ and $\cong_p^\mathcal{F}$ is preserved by the parallel composition, we can easily derive $M' \cong_p^\mathcal{F} N'$ (applying rules for structural equivalence), that means $M', N' \in \mathcal{C}$ and $\exists F' \in LSched$ such that:

$$Prob_M^F(\xrightarrow{c?@k}, \mathcal{C}) = 1 = Prob_N^{F'}(\overset{c?@k}{\Longrightarrow}, \mathcal{C})$$

or

$$Prob_M^F(\xrightarrow{c?@k}, \mathcal{C}) = 1 = Prob_N^{F'}(\Longrightarrow, \mathcal{C}).$$

Now we have only to prove that $F' \in \hat{\mathcal{F}}_\mathcal{C}$, but this follows straightforwardly by Definition 5.8, since $\bar{F}_2 \in \mathcal{F}_\mathcal{C}$.

if $\alpha = c?\vartheta@k$ the proof is analogous as for $\alpha = \alpha = c?@k$.

Proposition 5.4 $\cong_p^{Sched} = \approx_p^{LSched}$.

Proof.
It follows straightforwardly from Proposition 5.1 and from Theorems 5.2 and 5.3.

5.4 Introduction of a Cost Preorder

As for the other version of the calculus we define a preorder over networks which allows us to analyse ad hoc networks in terms of several kinds of metrics. This property can be used to replace a network component with a more efficient one, while maintaining the connectivity.

We first associate a cost function with each reduction as follows:
$\mathbf{Cost}^f(M, N) = f(M, N)$, where $M \to [\![N']\!]_\theta$, with N in the support of $[\![N']\!]_\theta$.
If $e = M_0 \to_{\theta_1} M_1 \to_{\theta_2} M_2 ... \to_{\theta_k} M_k$
is an execution then $\mathbf{Cost}^f(e) = \sum_{i=1}^{k} \mathbf{Cost}(M_{i-1}, M_i)$.

Let H be a set of networks, we denote by $Paths_M^F(H)$ the set of all executions from M ending in H and driven by F which are not prefix of any other execution ending in H. More formally, $Paths_M^F(H) = \{e \in Exec_M^F(H) \mid last(e) \in H$ and $\forall e'$ such that $e <_{prefix} e'$, $e' \notin Paths_M^F(H)\}$.

Now, we are ready to define the average cost of reaching a set of networks H from the initial network M according to the scheduler F.

Definition 5.10 *Let H be a set of networks. The average cost of reaching H from M according to the scheduler F is*

$$\mathbf{Cost}^f{}_M^F(H) = \frac{\sum_{e \in Paths_M^F(H)} \mathbf{Cost}^f(e) \times P_M^F(e)}{\sum_{e \in Paths_M^F(H)} P_M^F(e)}.$$

The average cost is computed by weighting the cost of each execution by its probability according to F and normalized by the overall probability of reaching H. The following definition provides an efficient method to perform both qualitative and quantitative analyses of mobile networks.

Definition 5.11 *Let $\mathcal{H}$ be a countable set of sets of networks and let $\mathcal{F} \subseteq Sched$ a set of schedulers. We say that N is* more efficient than *M with respect to the cost function f, in the context of $\mathcal{H}$ and $\mathcal{F}$ denoted*

$$N \sqsubseteq^f_{\langle \mathcal{H}, \mathcal{F} \rangle} M,$$

if $N \cong_p^{\mathcal{F}} M$ and, for all schedulers $F \in \mathcal{F}_\mathcal{C}$ and for all $H \in \mathcal{H}$, there exists a scheduler $F' \in \mathcal{F}_\mathcal{C}$ such that $\mathbf{Cost}^f{}_N^{F'}(H) \leq \mathbf{Cost}^f{}_M^F(H)$.

5.4.1 Measuring the interference level of the protocols.

In the following we introduce a cost function in order to measure the interference level of a network.

First we give the definition of cost. We define two interference metrics. The first one focuses on the senders and counts how many currently broadcasting nodes might interfere with each other due to the overlapping communication ranges. The second metric puts the emphasis on the receiver nodes and counts the number of active receivers which are simultaneously reached by two (or more) transmissions.

Sender-based interference Let M be a network. Given a channel c, we denote by $\mathbf{Overlap^s}(M,c)$ the set of nodes currently broadcasting over c and whose transmission areas are overlapping at some locations. Formally, let

$\mathtt{A}(M) \equiv (\nu\tilde{d})\big(\prod_{i\in I} n_i[\bar{c}_{L_i,r_i}\langle\tilde{v}_i\rangle.P_i]_{l_i} \mid \prod_{j\in J} n_j[c(\tilde{x}_j).P_j]_{l_j} \mid M'\big)$ be the active nodes of M, where $c \notin \mathtt{Top}(M')$, then

$$\mathbf{Overlap^s}(M,c) \;=\; \{n_i \,|\, i\in I,\ \exists i'(\neq i)\in I. d(l_i,l_{i'}) \le r_i + r_{i'}\}.$$

For example, consider the following network

$$\begin{aligned}\hat{M} \;=\;& n_1[\mathtt{out}\langle c_{L_1,r_1}, \tilde{v_1}\rangle.P_1]_{l_1} \mid n_2[\bar{c}_{L_2,r_2}\langle\tilde{v_2}\rangle.P_2]_{l_2} \mid n_3[\bar{c}_{L_3,r_3}\langle\tilde{v_3}\rangle.P_3]_{l_3}\\ & \mid n_4[\bar{a}_{L,r}\langle\tilde{v}\rangle.P_4]_{l_4} \mid n_5[c(\tilde{x}).P_5]_{l_5} \mid n_6[\mathtt{in}(c,\tilde{y}).P_6]_{l_6}\end{aligned}$$

where $d(l_i,l_j) > r_i$ for all $i,j \in \{1,2,3\}$, i.e., the nodes n_1, n_2, and n_3 are all far enough away from each other and can broadcast at the same time over the channel c. In this case, function $\mathbf{Overlap^s}(\hat{M},c)$ is defined as follows: for all $c' \neq c$ (e.g., $c' = a$) $\mathbf{Overlap^s}(\hat{M},c') = \emptyset$, while

$$\mathbf{Overlap^s}(\hat{M},c) = \begin{cases} \{n_2,n_3\} & \text{if } d(l_2,l_3) \le r_2 + r_3\\ \emptyset & \text{otherwise.}\end{cases}$$

We define the sender-based level of interference induced by a probabilistic action as follows:

$$\mathbf{s}(M,N) = \begin{cases} |\mathbf{Overlap^s}(N,c)| - |\mathbf{Overlap^s}(M,c)| & \\ \qquad \text{if } M \equiv (\nu\tilde{d})n[\mathtt{out}\langle c_{L,r},\tilde{v}\rangle.P]_l \mid M_1, & \\ \qquad \text{for some } \tilde{d},n,L,l,r,P,M_1, & \\ \qquad N \equiv (\nu\tilde{d})n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid M_1', \text{ and} & \\ \qquad M \to [\![N]\!]_\Delta; & \\ 0 \qquad \text{otherwise.} & \end{cases}$$

Consider again the above network $\hat{M}$. Since $d(l_1, l_j) > r_1$ for j in $\{2, 3\}$, we have $\hat{M} \to [\![\hat{N}]\!]_\Delta$, where

$$\begin{array}{rcl} \hat{N} & = & n_1[\bar{c}_{L_1,r_1}\langle\tilde{v_1}\rangle.P_1]_{l_1} \mid n_2[\bar{c}_{L_2,r_2}\langle\tilde{v_2}\rangle.P_2]_{l_2} \mid n_3[\bar{c}_{L_3,r_3}\langle\tilde{v_3}\rangle.P_3]_{l_3} \\ & & \mid n_4[\bar{a}_{L,r}\langle\tilde{v}\rangle.P_4]_{l_4} \mid n_5[P'_5]_{l_5} \mid n_6[P'_6]_{l_6} \end{array}$$

The sender-based level of interference induced by $\hat{M} \to [\![\hat{N}]\!]_\Delta$ is, e.g.:

- If n_1 is too far away from both n_2 and n_3, i.e., $d(l_1, l_j) > r_1 + r_j$ for j in $\{2, 3\}$, then $\mathbf{Overlap^s}(\hat{N}, c) = \mathbf{Overlap^s}(\hat{M}, c)$. Hence, $\mathbf{Cost^s}(\hat{M}, \hat{N}) = 0$.
- If n_2 and n_3 were already overlapping, i.e., $d(l_2, l_3) \leq r_2 + r_3$ and n_1 is not too far away of at least one of them, i.e., $d(l_1, l_2) \leq r_1 + r_2$ or $d(l_1, l_3) \leq r_1 + r_3$ then $\mathbf{Overlap^s}(\hat{N}, c) = \{n_1, n_2, n_3\}$. Therefore, $\mathbf{Cost^s}(\hat{M}, \hat{N}) = 1$.

Receiver-based interference. We denote by $\mathbf{Coll^r}(M, c, l, r)$ the set of nodes in M which are currently listening over channel c and lie in the transmission range of a sender located at l with radius r. Formally, let $\mathtt{A}(M) \equiv (\nu\tilde{d})\big(\prod_{i\in I} n_i[\bar{c}_{L_i,r_i}\langle\tilde{v}_i\rangle.P_i]_{l_i} \mid \prod_{j\in J} n_j[c(\tilde{x}_j).P_j]_{l_j} \mid M'\big)$ be the active nodes of M, where $c \notin \mathtt{Top}(M')$, then

$$\mathbf{Coll^r}(M, c, l, r) \quad = \quad \{n_j, j \in J \mid d(l, l_j) \leq r\}.$$

The number of receiver-based interferences induced by a probabilistic step is:

$$\mathbf{r}(M, N) = \begin{cases} |\mathbf{Coll^r}(M, c, l, r)| & \text{if} \\ & M \equiv (\nu\tilde{d})n[\mathtt{out}\langle c_{L,r}, \tilde{v}\rangle.P]_k \mid M_1, \\ & \text{for some } \tilde{d}, n, L, P, M_1, \\ & N \equiv (\nu\tilde{d})n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid M'_1, \text{ and} \\ & M \to [\![N]\!]_\Delta; \\ 0 & \text{otherwise.} \end{cases}$$

For instance, if we consider again our previous networks $\hat{M}$ and $\hat{N}$, assuming that n_1 can reach both l_5 and l_6 then $P'_5 = P_5\{\perp/\tilde{x}\}$ and $P'_6 = c(\tilde{y}).P_6$. Then, $\mathbf{Coll^r}(\hat{M}, c, l_1, r_1) = \{n_5\}$. Hence $\mathbf{Cost^r}(\hat{M}, \hat{N}) = 1$.

5.5 The Alternating Bit Protocol

In the following we analyse the performances of the Alternating Bit Protocol (ABP), in terms of the interference caused by the networks transmissions.

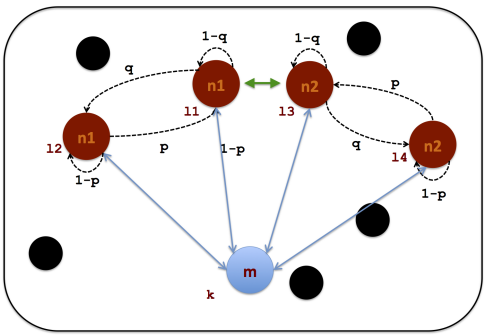


Figure 5.1: Topology of the network and mobility of devices

5.5.1 Introduction to the protocol

The alternating bit protocol (ABP) is a simple network protocol designed to achieve a point to point reliable transmission on unreliable channels. Messages are sent from a transmitter to a receiver and include the payload (i.e., the meaningful data) and some control information (e.g., the address identifying the destination, a checksum for the integrity checks, etc.). Among the control information, there is packet sequence number of 1 bit. When the sender sends a message with sequence number b, it waits for an acknowledge (`ack`) identified with the same sequence number from the receiver. If the `ack` does not arrive before a given deadline then the sender assumes that the packet has been lost and tries to resent it. The deadline is chosen according to the channel characteristics and must be greater than its round trip time. When the `ack` is correctly received, the sender flips the sequence number and starts a new transmission.

We consider a network consisting of two mobile sender nodes, n_1 and n_2, communicating with a static receiver node m. Node n_1 moves back and forth between locations l_1 and l_2 according to the probability distribution defined by the discrete time homogeneous Markov chain with the following transition matrix:

$$\mathbf{J} = \begin{vmatrix} 1-p & p \\ q & 1-q \end{vmatrix},$$

where $0 < p, q < 1$. Node n_2 moves similarly between l_3 and l_4 according to a discrete time Markov chain with the same transition matrix $\mathbf{J}$. We also assume that the receiver node is always in the transmission range of both senders (and that the senders are always in the range of the receiver) regardless of where the senders are located. This guarantees that m receives any packet from the senders (unless a collision occurs), and that both senders receive any `ack` sent by m.

Furthermore, we assume that the transmission ranges of the senders overlap only when n_1 is at l_1 and n_2 is at l_3. As a result, unless n_1 is at l_1 and n_2 is at l_3, the senders are in the condition to attempt a simultaneous transmission (as they don't sense each other) leading to an interference (see Figure 5.1): in literature,

this is known as the *hidden station problem*. Notice that while communications can be damaged by many factors, we shall consider only the interference factor in this analysis.

Table 5.5 shows an encoding of the sender and receiver processes. SND_j runs inside node n_j, sending a queue of messages T_j with sequence bit b_j; RCV, in turn, runs inside the receiver node m, expecting messages with sequence bits b_1 and b_2 from n_1 and n_2, respectively. We presuppose few auxiliary functions: `empty`(), `dequeue`() and `head`() implement the standard queue operations, while $\neg b$ flips the value of the bit b. Finally, `ok` is a channel name and a location introduced for the purposes of our analysis.

In order to compare the observational behaviours of the protocols, we assume that a successful end of transmission of the packets by a sender, indicated by broadcasting the message "END" over the channel `ok`, is observable for any observer node located at k. In this analysis, we are only interested in the levels of interference due to the internal nodes of the protocols. Therefore, we restrict communications over the channel c to the internal nodes of the protocols.

5.5.2 Interference cancellation scheme for CDMA

The CDMA/CA [64] transmission scheme protocol we consider in this section does not avoid collisions, but it allows a receiver to solve collisions without forcing the retransmissions of all the packets involved in the interference. The successive interference cancellation (SIC) method used by CDMA exploits the mathematical properties between vectors representing the data strings. Each sender uses a code orthogonal to the others' codes to modulate their signal. In order to obtain a set of codes preserving orthogonality, CDMA represents each bit as a vector of 8 chips, and assigns a different chip sequence to each device, according with the *Walsh matrix*, that means that for each couple of vectors the number of identical chips are equal to the number of complementary chips. When two devices transmit at the same time, each receiver is able to obtain the desired message by calculating the *dot product* between the received signal and the expected signal. This is due to the properties of orthogonal vectors. A receiver is able to decode a message only if it knows the sender's code.

In this example, we sketch a simplified successive interference cancellation (SIC) method. Assume that nodes n_1 and n_2 cause an interference at m by sending packets encoded by signals x_A and x_B. m receives the signal $y_1 = x_A + x_B$, detects the interference and stores y_1 in memory. In the successive slot, n_1 successfully resends x_A, i.e., m receives $y_2 = x_A$ and sends an ack to n_1. Now, x_B may be extracted from y_1 by m without further retransmissions as the result of $y_1 - x_A$. Although in practice this procedure is not always successful, we assume, for the sake of clarity, that messages can always be recovered correctly.

In modelling this protocol, the sender processes remain the same as in the simple ABP protocol defined in Table 5.5, while the receiver process is defined as shown

in Table 5.6.

In order to compare the observational behaviours of the protocols, we consider the following set $\mathcal{F}_{\mathbf{fas}}$ of *fair alternating schedulers* which:

1. always alternate between sending packets and node movements so that at each interaction of the transmitters with the receiver, the formers could be far enough of each other to cause interference or not;
2. give priority to acknowledgment actions (ACK and NACK) to model our assumption of an error-free feedback channel;
3. give priority to begin broadcasting actions (Beg-Bcast) over end broadcasting actions (End-Bcast) so that whenever an interference is possible, it is automatically created.

We know introduce some propositions in order to prove that applying the SIC method to the alternating bit protocol reduces the interference. We first prove that the two networks exhibit the same observable behaviour relative to $\mathcal{F}_{\mathbf{fas}}$.

Proposition 5.5 $ABP \approx_p^{\mathcal{F}_{\mathbf{fas}}} SIC\_ABP$.

Proof.

For the sake of brevity, we give just a sketch of the proof. In both protocols, the only observable actions, are the final messages sent by n_1 and n_2 through the channel `ok`, that occur when all the messages of their respective queues are completely and correctly received by m, since the other actions are either silent, or hidden by the restriction operator applied to the channel c. Hence, in both protocols the only observable actions are of the form:

$$\Longrightarrow \xrightarrow{\mathtt{ok}!(n_1,END)@k \lhd k} \Longrightarrow,$$

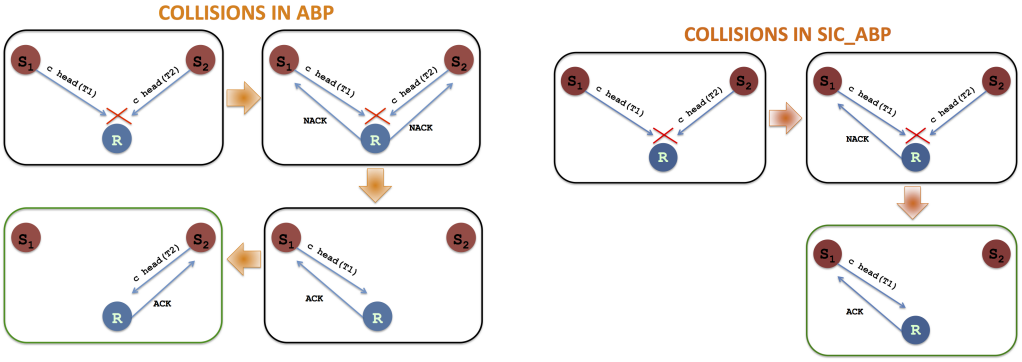
or

$$\Longrightarrow \xrightarrow{\mathtt{ok}!(n_2,END)@k \lhd k} \Longrightarrow .$$

We can conclude that ABP and SIC_ABP are probabilistic bisimilar, because they exhibit the same behavior, with the same probability. Indeed, the characteristics of matrix $\mathbf{J}$ ensures that for both the protocols the probability of eventually transmitting the whole queue of messages in 1.

Now let T_1 and T_2 be the queues of messages to be transmitted by the senders. We compare the interference efficiency of the protocols in the context of the set $\mathcal{H}(T_1, T_2) = \{H_\rho(T_1, T_2) \mid \rho \leq \max(|T_1|, |T_2|)\}$ where $H_\rho(T_1, T_2)$ means that all the packets up to ρ have been correctly transmitted by both senders and is defined as $H_\rho(T_1, T_2) = H_\rho^1(T_1, T_2) \cup H_\rho^2(T_1, T_2)$ where

$$\begin{aligned} H_\rho^1(T_1, T_2) = \{M \mid M \equiv (\nu c)\big(& n_1[SND_1\langle b_1, \mathtt{dequeue}^\rho(T_1)\rangle]_{l'} \\ & \mid n_2[SND_2\langle b_2, \mathtt{dequeue}^\rho(T_2)\rangle]_{k'} \mid m[RCV\langle b_1, b_2\rangle]_k\big)\} \end{aligned}$$



(a) Alternating Bit Protocol (b) CDMA/CA SIC Method

Figure 5.2: Description of the protocols

with the assumption that $\mathtt{dequeue}(\emptyset) = \emptyset$, and b_1, $b_2 \in \{0, 1\}$. Similarly

$$H^2_\rho(T_1, T_2) = \{N \mid N \equiv (\nu c)\big(n_1[SND_1\langle b_1, \mathtt{dequeue}^\rho(T_2)\rangle]_{l''} \\ \mid n_2[SND_2\langle b_2, \mathtt{dequeue}^\rho(T_2)\rangle]_{k''} \mid m[RCV_{SIC}\langle b_1, b_2\rangle]_k\big)\}$$

with b_1 and b_2 in $\{0, 1\}$, l', l'' in $\{l_1, l_2\}$, and k', k'' in $\{l_3, l_4\}$. Then, we compute the interference level of the protocols assuming that we start by a move action for each sender node so that their first transmissions could create an interference if they move too far away from each other[3]. The results are summarized in the following propositions.

Proposition 5.6 *For all F in $\mathcal{F}_{\mathbf{fas}}$ and for all $\rho \leq \max(|T_1|, |T_2|)$ we have:*

$$\mathbf{Cost}^{\mathbf{s}}_{ABP,F}(H_\rho(T_1, T_2)) = \\ 2 \times \mathbf{Cost}^{\mathbf{r}}_{ABP,F}(H_\rho(T_1, T_2)) = 2 \times \left(\frac{(p+q)^2}{q^2} - 1\right) \times \min(\rho, |T_1|, |T_2|)$$

with $0 < p, q < 1$.

Proof.

The proof relies on the observation that correct packets are sent only when the mobile nodes are in the locations l_1 and l_3. Hence, by exploiting the independence between the stochastic processes underlying the node movements, the result follows by standard analysis of absorbing Markov chains.

Note that our sender-based interference metric coincides with the number of lost packets. For the ABP with SIC, we have:

[3]The analysis for the other case is similar.

Proposition 5.7 *For all F in $\mathcal{F}_{\mathbf{fas}}$ and each $\rho \leq \max(|T_1|, |T_2|)$ we have:*

$$\begin{aligned}&\mathbf{Cost}^{\mathbf{s}}_{SIC\_ABP,F}(H_\rho(T_1,T_2)) = 2 \times \mathbf{Cost}^{\mathbf{r}}_{SIC\_ABP,F}(H_\rho(T_1,T_2)) = \\ &2 \times \frac{p}{(p+q)^3}\Big(n(p+q)(p+2q) - \\ &\frac{((1-p-q)^n-1)(p+q-1)(p^2-p(1-p-q)^{n+1}-4q+3pq+2q^2-p)}{p+q-2}\Big) \\ &\times \min(\rho, |T_1|, |T_2|)\end{aligned}$$

with $0 < p, q < 1$.

Proof.

Also in this case the proof is based on standard transient Markov chain analysis and exploits the independence among the processes that regulate the node movements. Indeed, the n-th steps transition probability matrix J^n is:

$$J^n = \begin{vmatrix} \frac{p(1-p-q)^n+q}{p+q} & \frac{p-p(1-p-q)^n}{p+q} \\ \frac{q-q(1-p-q)^n}{p+q} & \frac{p+q(1-p-q)^n}{p+q} \end{vmatrix}$$

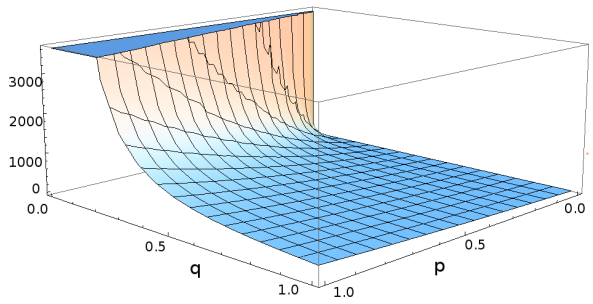
According to the SIC specification, nodes need only to send one packet for a successful packet transmission if they are in the locations l_1 and l_3. All the other location combinations require one of the nodes to send two packets for each successful transmission (while the other sends just one). Starting from states l_1 and l_3, the probability of being still in the same state after $i > 0$ steps is given by $(p(1-p-q)^i + q)^2/(p+q)^2$ (by independence). We derive the expression given by Proposition 5.7 as the closed expression of the following sum which represents the expected number of observed interferences for sending n packets:

$$\sum_{i=1}^{n} \left(1 - \left(\frac{p(1-p-q)^i + q}{p+q}\right)^2\right).$$

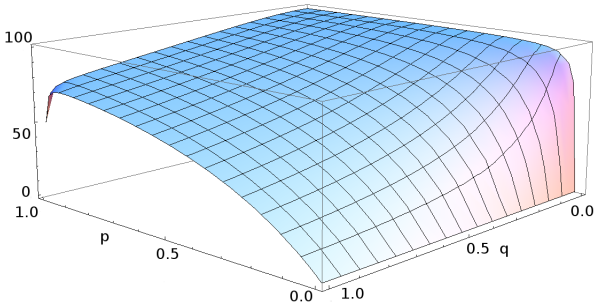
Theorem 5.4 *$SIC\_ABP \sqsubseteq^{\chi}_{\langle \mathcal{F}_{\mathbf{fas}}, \mathcal{H}(T_1,T_2)\rangle} ABP$,*
with $\chi \in \{\mathbf{s}, \mathbf{r}\}$.

Proof.

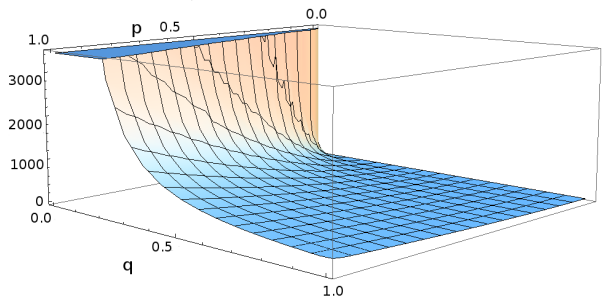
Let us denote by H_ρ the set $H_\rho(T_1, T_2)$. In Fig. 5.3 (a) and (b) we show a plot of $\mathbf{Cost}^{\mathbf{s}}_{ABP,F}(H_\rho)$ and $\mathbf{Cost}^{\mathbf{r}}_{SIC\_ABP,F}(H_\rho)$, respectively, as a function of p and q, for $\min(\rho, |T_1|, |T_2|) = 100$, while Fig. 5.3 (c) shows a plot of $\mathbf{Cost}^{\mathbf{r}}_{ABP,F}(H_\rho) - \mathbf{Cost}^{\mathbf{r}}_{SIC\_ABP,F}(H_\rho)$. Finally, from Propositions 5.5, 5.6, and 5.7, we can conclude that the SIC-based ABP protocol is much more interference efficient than its simple version.



(a) Plotting of $\mathbf{Cost}^{\mathbf{r}}_{ABP,F}(H_\rho)$ given by Proposition 5.6



(b) Plotting of $\mathbf{Cost}^{\mathbf{r}}_{SIC\_ABP,F}(H_\rho)$ given by Proposition 5.7



(c) Plotting of $\mathbf{Cost}^{\mathbf{r}}_{ABP,F}(H_\rho) - \mathbf{Cost}^{\mathbf{r}}_{SIC\_ABP,F}(H_\rho)$

Figure 5.3: Interference levels for ABP and SIC_ABP and their comparison

5.6 Resistance to Jamming and Casual Interception

Following we introduce another interesting case study, showing how this calculus can be use to analyse the resilience to jamming attacks of a wireless network governed by two different routing protocols.

5.6.1 Scenario

Let us consider an in-door MANET operating in a building of 30×20 meters with three floors whose height is 3 meters. For the sake of simplicity, we assume each floor to have the same topology: eight rooms connected by a central corridor as shown in Figure 5.4. A location is denoted by a pair $\langle f, r \rangle$ where $f \in \{1, 2, 3\}$ indicates the floor, while $r \in \{A, B, C, D, E, F, G, H, S\}$ indicates the room. Rooms A, B, C, D, E, F have dimension 5×5 meters, while G, H, S have dimension 10×15 meters.

The MANET consists of seven devices: three static nodes (n_1, n_2, n_3), located

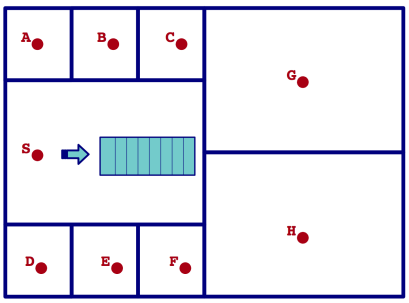


Figure 5.4: Topology of a building floor

respectively in the first, second and third floor, which represent the servers inside the building, and four mobile nodes (n_4, n_5, n_6, n_7).

Nodes can transmit with three transmission radii: $r_1 = 5m$, $r_2 = 10m$ and $r_3 = 15m$. We assume omnidirectional antennas, and the euclidean function to evaluate the distance between two locations. Hence, a node sending a message with radius r_1 can reach only those nodes lying in the adjacent rooms, or in the same room of an adjacent floor, while using radius r_2 the successfulness of a reception depends on the rooms where the sender and the receiver nodes are located. In order to simplify the computation of the distance between sender and receivers nodes, we assume that each transmission always begins exactly from the centre of a room. In practice, if n_1 is located at $l = \langle 1, A \rangle$ and n_2 is located at $k = \langle 2, A \rangle$ then their distance is $d(l, k) = 3m$. Hence, they are able to mutually communicate by using any of the transmission radii in the set $\{r_1, r_2, r_3\}$. If n_1 is located at $l' = \langle 1, C \rangle$, then the Euclidean distance between l' and k is $d(l', k) = \sqrt{109}$. Since $10 < \sqrt{109} < 15$, n_1 and n_2 can communicate only using radius r_3. Note that these assumptions are intended to simplify the model. However, our calculus may also deal with non-euclidean distances allowing us to take into account the effect of walls on the transmissions.

5.6.2 The HWMP protocol.

The Hybrid Wireless Mesh Protocol (HWMP) is the routing protocol used by the standard IEEE 802.11s [39]. It offers a variety of routing strategies, including some optional ones. HWMP can be configured to operate in two modes: on-demand reactive mode and tree-based proactive mode.
The reactive mode used by HWMP is based on the Ad-hoc On-demand Distance Vector Routing protocol (AODV) [79]. It uses three types of control packets whose forwarding allows each node to update its information about the best path to reach a specific destination. It is one of the most popular routing protocols specifically designed for mobile ad-hoc and sensor networks.

Each node maintains information about its neighbours in its *route table*, where each entry contains the following data associated with a given destination d.

- $seq\#_d$: the sequence number of the destination: it is incremented before sending a request, or (only if the node is the destination) before broadcasting the response;
- $\texttt{nexthop}_d$: the next intermediate node towards the destination d;
- $\texttt{hopcount}_d$: the number of hops from the source to the destination d;
- `lifetime`: the life-time of the record in the route table;
- $\texttt{List of precursors}_d$: the list of nodes using the actual node as `nexthop` to reach d.

The protocol exchanges three types of control packets:

- **RREQ** (Route Request): when a node needs to find a path, it broadcasts a RREQ message and then, when it receives the response, it chooses the cheapest path (in terms of energy costs, delays, number of hops, etc.);
- **RREP** (Route Response): When a node receives a RREQ message, it controls if it is the destination. In this case it immediately sends back the response, otherwise it searches a valid path in its *route table* to send back. If there are no valid paths, it propagates the RREQ packet;
- **RERR** (Route Error): the error message informing the network of a link breakage.

The tree-based proactive mode used by the HWMP protocol is based on a proactive protocol in which the network topology is built statically, forming a tree, i.e., a connected and acyclic graph, rooted in a chosen node. As in the previous mode, packet types, apart from root announcements, are RREQ, RREP and RERR, representing requests, responses and errors, respectively. The difference lays in the fact that, in the tree-based approach, the network topology is built statically from the start, and each node selects as the next hop the neighbour node that is nearest to the root according to its hop count, but it also maintains a table that contains the hop count to the root of its neighbours. The final result of these operations is that a spanning tree is created and used for the subsequent message forwarding. Whenever a node detects that the link with the upstream node, i.e., the node to which it would normally forward packets, is broken, it selects a new upstream node from its neighbours using the table of the hop counts, updates its own count and then broadcasts a RRER message to the downstream nodes. Then, these can select their own alternative paths if the hop count of the upstream node is no longer the shortest one. If a new path is chosen, the RERR message is propagated in turn to the downstream nodes. The effect of this action is to build a new spanning subtree.

In what follows we assume that the root node, e.g., a gateway between a wired network and the wireless one, is chosen statically and that it never moves.

5.6.3 Modelling the system.

We model our case study into the framework of Section 5.2. The set of feasible locations is:

$$\mathbf{Loc} = \{\langle f, r\rangle : f \in \{1,2,3\} \wedge r \in \{A,B,C,D,E,F,G,H,S\}\}\,.$$

Each node n_i is statically characterised by a pair $\langle r_{n_i}, \mathbf{J}^{n_i}\rangle$, where r_n is the maximum transmission radius, i.e., in our scenario $r_{n_i} = r_3$ for all n_i, with $i = 1,\ldots,7$, and $\mathbf{J}^{n_i}$ is the transition matrix of a discrete time Markov chain denoting the probability of movements.
We model the HWMP protocol into our probabilistic calculus with the aim of comparing the resilience to jamming of both the proactive and reactive modes used by the protocols.

In its simplest definition, the routing table maintained by each node has the form:

$$\langle d, seq\#_d, \texttt{nexthop}_d, \texttt{hopcount}_d, LP_d\rangle$$

where d is the primary key of the table and identifies the node name of the destination associated with the entry.

The packets used by HWMP are:

- $(\texttt{rreq}, \texttt{Bid}_{s,d}, (s, seq\#_s), (d, seq\#_d), \texttt{hopcount}_{s,d})$ for the request of a path from s to d;
- $(\texttt{rrep}, (s, seq\#_s), (d, seq\#_d), \texttt{hopcount}_{s,d})$ for the response to a path request from s to d;
- $(\texttt{rerr}, s, d, seq\#_d)$ denotes an error message;
- $(\texttt{ack}, d, seq\#_d)$ to confirm the correct reception of a route response.

For the sake of simplicity we abstract out the AODV packet management, and we consider the behaviour of the network assuming that a node needing a path to a given destination simply obtains it by calling the procedure $\texttt{find\_path}(s, d)$ where s and d are the source and destination nodes. The effect of this procedure is to update the node's routing table.

We consider the following network, where nodes n_4 and n_6 want to communicate, and the paths are the results of the routing protocol execution:

$$N = \prod_{i=1}^{7} (n_i[P_i]_{l_i})$$

where $\forall i \in [1-7], l_i \in \mathbf{Loc}$, and:

$$\begin{aligned}
P_4 &= \texttt{find_path}(n_4, n_6).\texttt{out}\langle c_{\texttt{nexthop}_{n_6}, r_2}, (msg, \texttt{nexthop}_{n_6}, n_6)\rangle. \\
&\quad \texttt{in}(c, (x_1, x_2, x_3)).[x_3 = n_4]\texttt{out}\langle \texttt{ok}_{\mathbf{Loc}, r_2}, OK\rangle.P_4, \\
P_6 &= \texttt{in}(c, (x_1, x_2, x_3)).[x_2 = n_6](\texttt{find_path}(n_6, n_4). \\
&\quad \texttt{out}\langle c_{\texttt{nexthop}_{n_4}, r_2}, (ack, \texttt{nexthop}_{n_4}, n_4)\rangle).P_6 \\
&\text{and } \forall i \in \{1, 2, 3, 5, 7\}: \\
P_i &= \texttt{in}(c, (x_1, x_2, x_3)).[x_2 = n_i](\\
&\quad [x_3 = n_4](\texttt{find_path}(n_i, n_4). \\
&\qquad \texttt{out}\langle c_{\texttt{nexthop}_{n_4}, r_2}, (x_1, \texttt{nexthop}_{n_4}, n_4)\rangle).P_i, \\
&\quad [x_3 = n_6](\texttt{find_path}(n_i, n_6). \\
&\qquad \texttt{out}\langle c_{\texttt{nexthop}_{n_6}, r_2}, (x_1, \texttt{nexthop}_{n_6}, n_6)\rangle).P_i).
\end{aligned}$$

The processes depicted above describe the communication between n_4 and n_6. Node n_4 uses the function `find_path` to discover the path to n_6, and then forwards the packet. When it receives the acknowledgment it fires the message OK through the channel `ok`. Node n_6 waits for the packet from n_4 and, when it receives it, it sends back the acknowledgment. Each intermediate node executes a simple process forwarding both the packet sent by n_4 and the ack of n_6.

These processes will be used to study both the *AODV* and the *tree-based* protocols, the only difference is the way the function `find_path` behaves: while in the *AODV* the path is discovered after the RREQ, RREP and RERR packages exchange, in the *tree-based* protocol, in order to find the best path, the process simply follows the predetermined routing spanning tree, built at the moment of the initial network setup through the algorithm informally described above. Thus, HWMP packets are sent only during the network setup or whenever a broken link is detected, causing the corresponding operations to happen.

5.6.4 Resilience to jamming attacks

We focus our attention on proactive jammers, executing the following process: $P = \texttt{out}\langle c_{\emptyset, r_2}, JAM\rangle.P$ which continuously broadcasts the dummy message JAM with radius r_2 on channel c. We consider two malicious nodes: a static jamming attacker m_1 identified by $\langle r_2, \mathbf{I}\rangle$, with $\mathbf{I}$ being the identity matrix, located at $k = \langle 1, H\rangle$, and a mobile node m_2 identified by $\langle r_2, \mathbf{J}^{m_2}\rangle$ whose initial location is $k' = \langle 3, G\rangle$. Notice that the jammer m_1 is a node blocking the activity of each node lying within

$$\mathbf{Loc}_{m_1} = \{\langle 1, H\rangle, \langle 2, H\rangle, \langle 3, H\rangle, \langle 1, F\rangle, \langle 2, F\rangle, \langle 3, F\rangle\}.$$

and its process is a recursive process continuing to send JAM messages through channel c. We study the behaviour of the protocol in a context C_1 consisting of node m_1, and in a context C_2 consisting of both m_1 and m_2. We consider the set of schedulers $\mathcal{F}$ such that:

- during the forwarding of the packets (x_1, x_2, x_3) where $x_1 \in \{msg, ack\}$, movements of nodes n_i, $i \in [4-7]$ occurs at each collision or JAM-message reception,
- the beginning of output actions have priority on the ending of the output actions.

The first constraint allows us to model the fact that mobile nodes react to interference by moving away from their current location, while the second constraint is necessary since we are considering proactive jammers, which continue to send packets to provoke collisions. Let

$C_1[\cdot] = \cdot \mid m_1[P]_k$.

The robustness of the network using the HMWP protocol against the malicious node m_1 can be verified by checking if the observational behaviour of N is independent of the presence of the jammer inside the building. Formally, we have to prove that:

$$(\nu c)N \cong_p^{\mathcal{F}} (\nu c)C_1[N].$$

The restriction operator νc is due to the fact that we want to observe only the correctness of the communication between the nodes n_4 and n_6, without considering the different paths they may choose for the message forwarding. Then, since c is hidden, the only observable action in $(\nu c)N$ and $(\nu c)C_1[N]$ is $\xrightarrow{\texttt{ok!}OK@K \triangleleft K}$ for some $K \subseteq \mathbf{Loc}$.

Moreover, we consider the dynamic context:

$$C_2[\cdot] = \cdot \mid m_1[P]_k \mid m_2[P]_{k'}$$

where m_2 is a mobile node, meaning that the *jamming area* may change in time. We also consider the same restricted set of schedulers $\mathcal{F}$ as above and we aim at observing the successfulness of the communication between n_4 and n_6. Hence, we have to prove that:

$$(\nu c)N \cong_p^{\mathcal{F}} (\nu c)C_2[N].$$

In order to verify resilience of our network case study with respect to the jamming context C_1, we check whether

$$(\nu c)N \cong_p^{\mathcal{F}} (\nu c)C_1[N]. \tag{5.1}$$

By using the proof technique presented in Section 5.3, it is sufficient to find a probabilistic bisimulation containing the pair $((\nu c)N, (\nu c)C_1[N])$. Let us consider the relation

$$\mathcal{R} = \{((\nu c)\bar{N}, (\nu c)\bar{C}_1[\bar{N}]) : \ (\nu c)\bar{C}_1[\bar{N}] \in \bar{\mathcal{M}}\}$$

where $\bar{\mathcal{M}} = \{(\nu c)\bar{C}_1[\bar{N}] : (\nu c)C_1[N] \rightarrow^* (\nu c)\bar{C}_1[\bar{N}]\}$. In order to prove (5.1), it is enough to prove that $\mathcal{R}$ is a bisimulation relative to $\hat{\mathcal{F}}$ with $\mathcal{M} = \{(\nu c)N, (\nu c)C_1[N])\}$.

Since in $\bar{\mathcal{M}}$ channel c is hidden, the only actions that $\bar{N}$ can do are τ actions, or output through the channel `ok`, while the jamming context $\bar{C}_1[\cdot]$ can only make τ actions. Let us consider $(\nu c)\bar{N} \xrightarrow{\tau} [\![(\nu c)\bar{N}']\!]_\theta$ driven by $F \in \hat{\mathcal{F}}_{\mathcal{C}}$. Then, $\forall \mathcal{C} \in \mathcal{N}/\mathcal{R}$, $Prob^F_{(\nu c)\bar{N}}(\xrightarrow{\tau}, \mathcal{C}) =$

$$\sum\nolimits_{(\nu c)\bar{N}'' \in \mathcal{C} \text{ in the support of } [\![(\nu c)\bar{N}']\!]_\theta} [\![(\nu c)\bar{N}']\!]_\theta((\nu c)\bar{N}'').$$

If the τ action is due to the application of the rule (Move), we conclude by applying rule (Par) to m_1 and, by Definitions 5.3 and 5.8 $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that

$$Prob^F_{(\nu c)\bar{N}}(\xrightarrow{\tau}, \mathcal{C}) = Prob_{(\nu c)\bar{C}_1[\bar{N}]}(\Longrightarrow, \mathcal{C}),$$

as required.

If $(\nu c)\bar{N} \xrightarrow{\tau} (\nu c)[\![\bar{N}']\!]_\Delta$, because of an application of rule (Beg-Bcast) then $\bar{N} \xrightarrow{c_L![l,r_2]} [\![\bar{N}']\!]_\Delta$ for some location l and some set L of locations. If $\bar{C}_1[\mathbf{0}] \equiv C_1[\mathbf{0}]$, or $d(l,k) > r_2$ we are done, because it is enough to apply rule (Par), since both $\bar{C}_1[\bar{N}]$ and $\bar{C}_1[\bar{N}']$ are well-formed networks, and $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $Prob^F_{(\nu c)\bar{N}}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F'}_{(\nu c)\bar{C}_1[\bar{N}]}(\Longrightarrow, \mathcal{C})$, as required. If $\bar{C}_1[\mathbf{0}] \equiv C'_1[\mathbf{0}]$ and $d(l,k) \leq r_2$, we can not apply rule (Par) because $C'_1[\bar{N}']$ is not well-formed (i.e., there are two active senders whose distance is smaller of their transmission radius). Now, in order to prove bisimulation, we have to find in $\hat{\mathcal{F}}_{\mathcal{C}}$ a scheduler allowing $C'_1[\bar{N}]$ to reach $\bar{C}_1[\bar{N}']$ with probability 1, where $\bar{C}_1[\cdot] \in \{C_1[\cdot], C'_1[\cdot]\}$. In this case, there exists F' allowing m_1 to finish its communication, i.e., $C_1[\bar{N}] \xrightarrow{\tau} [\![C_1[\bar{N}']]\!]_\Delta$.
The most interesting case is when $d(k,l_h), d(l,l_h) \leq r_2$ for some $h \in [1-7]$, where $\bar{N} \equiv \prod_{i=1}^{7} n_i[Q_i]_{l_i}$ and $Q_h = \ c(\tilde{x}).Q'_h$.
This means that n_h is in the jammer's transmission area, and it receives the correct packet in $\bar{N}$, while it receives a collision in $C'_1[\bar{N}]$ avoiding it to immediately reach $\bar{C}_1[\bar{N}']$ with probability 1. If the node receiving the collision ($\perp$) is an intermediate node in the path from n_4 to n_6, we are done, since, when n_4 does not receive the acknowledge, it searches another path to reach n_6 (by re-executing `find_path`) and it sends again the message. If the node receiving the collision is the source or the destination of the communication (n_4, n_6), the bisimulation depends on the transition matrix modelling its mobility: if the node, with a finite number h of steps, goes far away from the jammer with probability 1, i.e., the probability to end up in an ergodic set in which all the states represents locations inside the jammed area is 0, then the bisimulation is proved, otherwise the probability for $C'_1[\bar{N}]$ to reach a

state in $\mathcal{C}$ is $1-p$, where $p = \sum_{l' \in \mathbf{Loc}_{jam}} \mathbf{J}^{n_i}_{ll'}{}^{(h)}$ and $\mathbf{J}^{n_i}$ being the transition matrix associated with the node n_i.
Moreover, when the protocol uses the tree-based strategy, assuming that root nodes are static (in our example nodes n_1, n_2 and n_3), e.g., when those nodes are bridges between wireless and wired networks, or access points, if the node receiving the collision is the root itself, it can never be possible to detect an alternative route, thus the bisimulation does not hold.

Let us now consider now the dynamic context:

$$C_2[\cdot] = \cdot \mid m_1[P]_k \mid m_2[P]_{k'}.$$

We have to prove that:

$$(\nu c)N \cong_p^{\mathcal{F}} (\nu c)C_2[N]. \tag{5.2}$$

Again, we check whether the relation

$$\mathcal{S} = \{((\nu c)\bar{N}, (\nu c)\bar{C}_2[\bar{N}]) : (\nu c)\bar{C}_2[\bar{N}] \in \mathcal{M}\},$$

where $\mathcal{M} = \{(\nu c)\bar{C}_2[\bar{N}] : (\nu c)C_2[N] \rightarrow^* (\nu c)\bar{C}_2[\bar{N}]\}$, is a bisimulation. The proof proceeds as for the previous context (the behaviours of N and $C_2[N]$ only differ for the collisions caused by the jammers attacks), but we have to take into account that the second jammer is mobile: if in the previous case we had only to verify the positions of n_4 and n_6, while the jamming area was known statically, now we have to analyse all the possible locations occupied by the mobile jammer.

Let the Markov chain $\mathbf{J}^{n_i m_2}$ be the joint process between $\mathbf{J}^{n_i}$ and $\mathbf{J}^{m_2}$, in which states are pairs (l,k) of states of the chains $\mathbf{J}^{n_i}$ and $\mathbf{J}^{m_2}$, which are associated with nodes n_i and m_2 (the mobile jammer), respectively. Hence, each pair (l,k) of $\mathbf{J}^{n_i m_2}$ means that n_i is located at l and m_2 is located at k. The state space of $\mathbf{J}^{n_i m_2}$ is thus the Cartesian product of the state spaces of $\mathbf{J}^{n_i}$ and $\mathbf{J}^{m_2}$. We consider a subset $\mathbf{S}^{n_i m_2}$ of $\mathbf{J}^{n_i m_2}$, such that $(l,k) \in \mathbf{S}^{n_i m_2} \Leftrightarrow d(l,k) \geq r_{m_2}$, i.e., all combinations of states in which the jammer can interfere with the node n_i. n_i can always communicate successfully (and then the bisimulation is proved) only if the probability to end up in an ergodic set in which all states are member of $\mathbf{S}^{n_i m_2}$ is 0.

Again, when using the tree-based strategy, if the root nodes are stationary, even if n_4 and n_6, when jammed, are able to reach a safe location with probability 1, the success of the communication depends on the root nodes, and if the root nodes are jammed, n_4 and n_6 may not be able to find a valid path to complete their communication.

Even if both relations $\mathcal{R}$ and $\mathcal{S}$ are proved to be bisimulations only under particular conditions depending on the mobility of nodes, with our proof technique we have been able to show that the reactive approach used by the HWMP protocol is more robust against jamming than the proactive one, since, when using the tree-based routing, the bisimulations are proved under more restricted conditions, which depend on the mobility behaviour of the root nodes.

5.7 Conclusions

In this chapter we introduce a version of the Probabilistic EBUM calculus, aimed at capturing the collisions caused by multiple nodes transmitting with the same channel.

We showed how this calculus is useful to study the performances of different ad hoc communications protocols, as well as to face the problem of jamming attacks, due to malicious nodes constantly occupying a channel with dummy transmissions.

Due to the non-atomic nature of input and output actions, this calculus is fit for studying ad hoc communication protocols only when we are interested in the analysis of the level of interference, while, for all the other kinds of performance analysis the best choice is to use the probabilistic calculus introduced in chapter 4.

$$\text{(Beg-Snd)}\ \frac{P \xrightarrow{\bar{c}_{L,r}} P'}{n[P]_l \xrightarrow{c_L![l,r]} [\![n[P']_l]\!]_\Delta} \qquad \text{(End-Snd)}\ \frac{P \xrightarrow{\bar{c}_{L,r}\tilde{v}} P'}{n[P]_l \xrightarrow{c_L!\tilde{v}[l,r]} [\![n[P']_l]\!]_\Delta}$$

$$\text{(Beg-Rcv)}\ \frac{P \xrightarrow{c} P'}{n[P]_l \xrightarrow{c?@l} [\![n[P']_l]\!]_\Delta} \qquad \text{(End-Rcv)}\ \frac{P \xrightarrow{c\vartheta} P'}{n[P]_l \xrightarrow{c?\vartheta@l} [\![n[P']_l]\!]_\Delta}$$

$$\text{(Beg-Bcast)}\ \frac{M \xrightarrow{c_L![l,r]} [\![M']\!]_\Delta \quad N \xrightarrow{c?@l'} [\![N']\!]_\Delta \quad d(l,l') \le r \ \wedge\ \mathtt{A}^s_N(c,l) = \mathtt{A}^s_N(c,l') = \emptyset}{M|N \xrightarrow{c_L![l,r]} [\![M'|N']\!]_\Delta}$$

$$\text{(Coll-Bcast)}\ \frac{M \xrightarrow{c_L![l,r]} [\![M']\!]_\Delta \quad N \xrightarrow{c?\perp@l'} [\![N']\!]_\Delta \quad d(l,l') \le r \ \wedge\ \mathtt{A}^s_N(c,l) = \emptyset}{M|N \xrightarrow{c_L![l,r]} [\![M'|N']\!]_\Delta}$$

$$\text{(End-Bcast)}\ \frac{M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta \quad N \xrightarrow{c?\tilde{v}@l'} [\![N']\!]_\Delta \quad d(l,l') \le r}{M|N \xrightarrow{c_L!\tilde{v}[l,r]} [\![M'|N']\!]_\Delta}$$

$$\text{(Lose1)}\ \frac{M \xrightarrow{c_L![l,r]} [\![M']\!]_\Delta}{M \xrightarrow{\tau} [\![M']\!]_\Delta} \qquad \text{(Lose2)}\ \frac{M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta}{M \xrightarrow{\tau} [\![M']\!]_\Delta}$$

$$\text{(Move)}\ \frac{\mathtt{Active}(P) = \mathtt{false}}{n[P]_l \xrightarrow{\tau} [\![n[P]_l]\!]_{\mu^n_l}} \qquad \text{(Res)}\ \frac{M \xrightarrow{\gamma} [\![M']\!]_\theta \quad \mathtt{Chan}(\gamma) \neq c}{(\nu c)M \xrightarrow{\gamma} [\![(\nu c)M']\!]_\theta}$$

$$\text{(Obs)}\ \frac{M \xrightarrow{c_L!\tilde{v}[l,r]} [\![M']\!]_\Delta \quad R \subseteq \{l' :\ d(l,l') \le r \ \wedge\ |\ \mathtt{A}^s_M(c,l')\ |= 1\} \quad K = R \cap L,\ K \neq \emptyset}{M \xrightarrow{c!\tilde{v}@K \lhd R} [\![M']\!]_\Delta}$$

$$\text{(Par)}\ \frac{M \xrightarrow{\gamma} [\![M']\!]_\theta}{M|N \xrightarrow{\gamma} [\![M'|N]\!]_\theta}$$

Table 5.4: LTS rules for Networks

$SND_j\langle b_j, T_j\rangle =$	$[\texttt{empty}(T) = \texttt{false}](\texttt{out}\langle c_{\{k\},r_j}, (b_j, \texttt{head}(T_j), n_j)\rangle.WAIT\_Ack_j\langle b_j, T_j\rangle),$ $\texttt{out}\langle \texttt{ok}_{\{k\},r_j}, (n_j, END)\rangle$
$WAIT\_Ack_j\langle b_j, T_j\rangle =$	$\texttt{in}(c, (x, y, z)).[y = n_j]([(x = b_j) \wedge (z = ACK)]SND_j\langle \neg b_j, \texttt{dequeue}(T_j)\rangle,$ $SND_j\langle b_j, T_j\rangle), WAIT\_Ack_j\langle b_j, T_j\rangle$
$RCV\langle b_1, b_2\rangle =$	$\texttt{in}(c, (x, y, z)).[z = n_1]\big(([x = b_1]\texttt{out}\langle c_{\{l_1,l_2,l_3,l_4\},r}, (b_1, n_1, ACK)\rangle.$ $RCV\langle \neg b_1, b_2\rangle, \texttt{out}\langle c_{\{l_1,l_2,l_3,l_4\},r}, (b_1, n_1, NACK)\rangle.RCV\langle b_1, b_2\rangle),$ $[z = n_2]([x = b_2]\texttt{out}\langle c_{\{l_1,l_2,l_3,l_4\},r}, (b_2, n_2, ACK)\rangle.RCV\langle b_1, \neg b_2\rangle,$ $\texttt{out}\langle c_{\{l_1,l_2,l_3,l_4\},r}, (b_2, n_2, NACK)\rangle.RCV\langle b_1, b_2\rangle)\big),$ $\texttt{out}\langle c_{\{l_1,l_2,l_3,l_4\},r}, (b_1, n_1, NACK)\rangle.\texttt{out}\langle c_{\{l_1,l_2,l_3,l_4\},r}, (b_2, n_2, NACK)\rangle.$ $RCV\langle b_1, b_2\rangle$
$ABP =$	$(\nu c)\big(n_1[SND_1\langle 1, T_1\rangle]_{l_1} \mid n_2[SND_2\langle 1, T_2\rangle]_{l_3}\ m[RCV\langle 1, 1\rangle]_k\big)$

Table 5.5: ABP

$RCV_{SIC}\langle b_1, b_2\rangle =$	$\texttt{in}(c, (x_1, x_2, x_3))[x_3 = n_1]\big(([x_1 = b_1]\texttt{out}\langle c_{\{l_1,l_2,l_3,l_4\},r}, (b_1, n_1, ACK)\rangle.$ $RCV_{SIC}\langle \neg b_1, b_2\rangle, \texttt{out}\langle c_{\{l_1,l_2,l_3,l_4\},r}, (b_1, n_1, NACK)\rangle.RCV_{SIC}\langle b_1, b_2\rangle),$ $[x_3 = n_2]([x_1 = b_2]\texttt{out}\langle c_{\{l_1,l_2,l_3,l_4\},r}, (b_2, n_2, ACK)\rangle.RCV_{SIC}\langle b_1, \neg b_2\rangle,$ $\texttt{out}\langle c_{\{l_1,l_2,l_3,l_4\},r}, (b_2, n_2, NACK)\rangle.RCV_{SIC}\langle b_1, b_2\rangle)\big),$ $\texttt{out}\langle c_{\{l_1,l_2,l_3,l_4\},r}, (b_1, n_1, NACK)\rangle.WAIT\langle \perp_{x_1,x_2,x_3}, b_1, b_2\rangle$
$WAIT\langle \perp_{p_1,p_2,p_3}, b_1, b_2\rangle =$	$\texttt{in}(c, (x_1, x_2, x_3))[x_3 = n_1]([x_1 = b_1](\texttt{out}\langle c_{\{l_1,l_2,l_3,l_4\},r}, (b_1, n_1, ACK)\rangle.$ $[f(x_3, p_3) = n_2][b_2 = f(x_1, p_1)](\texttt{out}\langle c_{\{l_1,l_2,l_3,l_4\},r}, (b_2, n_2, ACK)\rangle.$ $RCV_{SIC}\langle \neg b_1, \neg b_2\rangle), \texttt{out}\langle c_{\{l_1,l_2,l_3,l_4\},r}, (b_2, n_2, NACK)\rangle.RCV_{SIC}\langle \neg b_1, b_2\rangle),$ $\texttt{out}\langle c_{\{l_1,l_2,l_3,l_4\},r}, (x_1, n_1, NACK)\rangle.WAIT\langle \perp_{x_1,x_2,x_3}, b_1, b_2\rangle$
$SIC\_ABP =$	$(\nu c)\big(n_1[SND_1\langle 1, T_1\rangle]_{l_1} \mid n_2[SND_2\langle 1, T_2\rangle]_{l_3} \mid m[RCV_{SIC}\langle 1, 1\rangle]_k\big)$

Table 5.6: SIC_ABP

6

Automatic Performance Analysis

6.1 Introduction

In Chapter 4 we defined a preorder among networks (see Definition 4.9), which allows us to study how different protocols with the same connectivity can have different performances in terms of energy conservation, level of interference or other metrics. The preorder has been defined in two steps: the first step consists in verifying that the networks compared are equivalent with respect to our definition of barbed congruence, while in the second step we prove that one of the systems we are comparing is always able to mimic the behaviour of the other one, with a less cost.

In Chapter 2, a number of tools have been cited for model checking probabilistic systems. Among the existing tools we decided to use the PRISM Model Checker [49], because it supports the modelling of Markov Decision Processes (MDPs), (where non-deterministic and probabilistic aspects coexist), as well as the definition of costs and rewards of the models executions. We exploit the PRISM tool to perform automated, quantitative verification and analysis of wireless networks for a range of performance metrics [50]. Specifically, we develop a parser to translate an EBUM process term, representing a network, into an MDP model expressed in the PRISM language. Then we use the PRISM property specification language to analyse the connectivity of a networks and to compute the costs for a network to reach a particular state.

6.2 Introduction to the PRISM language

PRIMS MDPs are expressed through a simple state-based language, composed of modules. Each module in turn is composed of variables and commands.

Modules. A *module* is specified as

```
module name
...
endmodule
```

The first part of the module is the list of its local variables, describing the different states the module can reach. A *variable declaration* contains its name, type and initial value. As an example, the following declaration

```
x : [1 .. 5] init 3;
```

means that variable x is an integer, which can take a value in the interval [1-5], and its initial value is 3.

Commands describe all the possible behaviours of the modules, i.e. all the possible transitions from a state to another one. They include *guards*, which indicate the states where the transition can occurs, and the *updates*, which modify the variables in order to reach the arrival states. In PRISM it is possible to represent both probabilistic and non-deterministic transitions.

An example of probabilstic command is:

```
[] (x = 1 ) -> 0.6:(x' = 2) + 0.4:(x' = 3);
```

that means that from the state where $x = 1$ we can reach the state where $x = 2$ with probability 0.6 , and the state where $x = 3$ with probability 0.4 .

The non-deterministic version of this command is:

```
[] (x = 1 ) -> (x' = 2);
[] (x = 1 ) -> (x' = 3);
```

meaning that the model will arbitrarily choose one of that commands, assigning to x either the value 2 or the value 3, i.e., from the states where variable x is set to 1, we can arbitrarily reach one of the two possible arrival states. We can also have multiple variables in a module. In this case guards and updates are enriched with the & and || operators in order to combine the modifications of different variables inside the same command.

Commands can also be associated with labels, and this is useful in order to model synchronisations. In PRISM synchronisations can occur only among commands with the same label. Usually only one of the synchronising commands is probabilistic while all the other ones are non-deterministic; if we have multiple probabilistic commands with the same label, the probability of the synchronisation will be the normalized product of the synchronising commands probabilities.

6.2.1 The Property Specification Language of PRISM

PRISM provides a *property specification language* in order to specify properties of Markov decision processes, probabilistic and timed automata, discrete time and continuous time markov chains. It supports several temporal logics, such as PCTL (Probabilistic Computation Tree Logic) and LTL (Linear Temporal Logic).

In particular, when dealing with MDPs, the PRISM property specification language enables us to study a lot of important properties, such as the minimum or

maximum probability to reach a particular state under some conditions. Moreover, PRISM supports also the specification of reward properties, i.e., the expected values of some given rewards (or costs) associated with a model.

The P operator.
The **P** operator is used to reason about the probability of an event's occurrence. Formally, we write:

```
P bound [ pathprop ]
```

which is `true` if the probability that the path property `pathprop` is satisfied by the paths from the initial states respects the bound `bound`. In particular, when dealing with MDPs, PRISM allows only to calculate the minimum or the maximum probability, due to the property of Markov Decision Processes to allow non-determinism. As an example, the property:

```
Pmax > 0.5 [ pathprop]
```

is true if the maximum probability to eventually reach the state satisfying `pathprop` is greater than 0.5, false otherwise.

We can also adopt a quantitative approach, by computing the actual probability a path property is satisfied. An example is:

```
Pmin =? [pathprop]
```

which computes the minimum probability to satisfy `pathprop`.

The PRISM property specification language introduces a set of *temporal operators* in order to express the PCTL path formulas or the LTL formulas which can be verified for a single path of a model.

Among these operators, the most used are:

- **F** (= "eventually") : **F** "`condition`" expresses the property that the condition will be eventually satisfied by the path.
- **U** (= "until") : "`condition1`" **U** "`condition2`" expresses the property that `condition2` is eventually true and that `condition1` is always true until `condition2` becomes true.
- **G** (= "always") : **G** "`condition`" expresses the property that the condition is always true (i.e., it expresses the *invariancy* property).

Costs and Rewards.
Reward properties are based on the possibility of defining rewards associated with a given PRISM model. Rewards are associated with models using the following construct:

```
rewards ‘‘name’’
...
endrewards
```

Rewards can be defined anywhere in the model file, except within a module definition. These constructs contain one or more *reward items*, which assign rewards to particular states or transitions. The syntax for the state reward is:

```
constraints: cost;
```

which assigns the cost `cost` to the states satisfying the list of constrains.

On the other hand, in order to assign a cost to a transition, we have

```
[action] constraints: cost;
```

which assigns the cost `cost` to the transition [`action`], when the list of constraints is satisfied.

With the PRISM property specification language we can compute the expected value of the rewards associated with the model. Again, since MDPs support the non-deterministic choices the syntax will be:

```
Rmin bound [ rewardprop ]
Rmax bound [ rewardprop ]
```

which calculate the minumum (respectively the maximum) expected reward associated with `rewardprop` of the model.

As an example, the following reward property

```
Rmax = ?  [F state]
```

returns the maximum cost for the model, to eventually reach the state `state`.

PRISM supports experiments, which is a way of automating multiple instances of model checking. Experiments can be performed by verification or by simulation.

6.3 A Parser for the Probabilistic EBUM calculus

In the following we introduce a parser to automatically translate Probabilistic EBUM networks into PRISM MDPs, in order to have an automatic and efficient tool to study and compare their performances, in terms of a range of indexes, such as energy consumption and time delay. In particular, we use a translator, implemented in F# [17] with FsYacc and FsLex [54], which compiles EBUM networks into the PRISM language. The result is an MDP, such that there is a one-to-one correspondence between the PRISM transitions and the EBUM reductions.

6.3.1 Input file of the parser

The input of the parser is a text file containing all the information necessary to build the corresponding MDP. In particular we remind that each network node is associated with a Markov Matrix corresponding to the probability distribution which describes its mobility. On the other hand, the reduction semantics needs the definition of a distance function $d(\cdot,\cdot)$ which computes the distance between each pair of locations in the set **Loc**.

Finally we need to use a regular grammar for the EBUM Syntax, in order to make it readable by the Lexer.

Following we describe in detail the contents of the Input text file.

- The PEBUM network (see Table 3.1 in Chapter 3) is written in the following grammar:

	Regular grammar		**PEBUM Syntax**
$M ::=$	$\mathbf{0}$	Empty Network	$\mathbf{0}$
	$\mid M_1 \mid M_2$	Parallel Composition	$M_1 \mid M_2$
	$\mid [c]M$	Restriction	$(\nu c)M$
	$n@l\{P\}$	Node (or device)	$n[P]_l$

where each process in turn is written as follows:

	Regular grammar		**PEBUM Syntax**
$P ::=$	$\mathbf{0}$	Inactive process	$\mathbf{0}$
	$\mid (\tilde{x}) \leftarrow c; P$	Input	$c(\tilde{x}).P$
	$\mid \tilde{v} \rightarrow c@L/r; P$	Output	$\bar{c}_{L,r}\langle\tilde{w}\rangle.P$
	$\mid \texttt{if}\, w_1 = w_2\, \texttt{then}\, P\, \texttt{else}\, Q$	Matching	$[w_1 = w_2]P, Q$
	$\mid \texttt{rec}\, \{P\}$	Recursion	$A\langle\tilde{w}\rangle$

- A transition probability matrix $\mathbf{J}^n$ is associated with each node n as illustrated below, where $\mathtt{L}_1$, ..., $\mathtt{L}_k$ are the locations in the network and $\mathtt{p}_{i,j}$ is the probability for the node n to move from location $\mathtt{L}_i$ to $\mathtt{L}_j$.

$$\mathbf{J}^n = \begin{array}{cccc} \mathtt{L}_1 & \mathtt{L}_2 & \ldots & \mathtt{L}_k \\ \mathtt{p}_{1,1} & \mathtt{p}_{1,2} & \cdots & \mathtt{p}_{1,k} \\ & \ldots & & \\ \mathtt{p}_{k,1} & \mathtt{p}_{k,2} & \cdots & \mathtt{p}_{k,k} \end{array}$$

- A triangular matrix **Dist**, describing the distances between each pair of network locations. The matrix is triangular because we assume the distance to be symmetric.

$$\mathbf{Dist} = \begin{array}{cccc} \mathtt{L}_1 & \mathtt{L}_2 & \ldots & \mathtt{L}_k \\ \mathtt{d}_{1,1} & \mathtt{d}_{1,2} & \ldots & \mathtt{d}_{1,k} \\ & & \ldots & \\ & & & \mathtt{d}_{k,k} \end{array}$$

6.3.2 The parsing task

The parser processes the input text to collect all the information we need to capture the behaviour of a network expressed by the reduction semantic rules (see Table 4.1). The key is to generate a list of records containing information about the possible synchronisations the network can perform (see (R-Bcast) rule in Table 4.1). This list can be generated by analysing all the network processes, with respect to the probability distribution of each node's mobility (expressed by $\mathbf{J}^n$) and the topology of the network (expressed by **Dist**).

As a result of the input elaboration, the parser creates a structure, called *out_entry list*, containing all the possible output reductions the network can perform.

Each out_entry contains:

- The name of the sender node;
- The list of the synchronising nodes;
- The channel associated with the transmission;
- The radius associated with the transmission;
- The list of the messages transmitted;
- The possible locations the sender can occupy in order to produce the same observable action;
- The set of the locations able to receive the message.

This entry list is the key aspect of our parser, because it allows one to deduce all the possible behaviours of the input network.

Notice that we elaborate the network in order to create a model corresponding to the reduction semantics, without taking into account the observable behaviour of the network, since we want to use model checking to study the costs of the protocols, and not to analyse their observable behaviours.

6.3.3 Output of the parser: generation of a PRISM MDP

The output of the parser is an MDP written in the PRISM language [49]. Given a network $M \equiv \prod_{i\in I} n_i[P_i]_{l_i}$, each network node $n_i[P_i]_{l_i}$ corresponds to an MDP module, with name n_i. The variable loc_{n_i} refers to the current location of n_i; the variable $proc_{n_i}$ represents the current process status, which is used to control the order of process execution (since processes are deterministic and sequential). Free variables and locations are defined as constants at the beginning of the file while, if a process contains bound variables, they are defined inside the node module.

Modules may execute commands (or actions) corresponding to the reduction rules (R-Bcast) and (R-Move) in Table 4.1:

- (R-Bcast) $n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid \prod_{i\in I} n_i[c(\tilde{x}_i).P_i]_{l_i} \rightarrow [\![n[P]_l \mid \prod_{i\in I} n_i[P_i\{\tilde{v}_i/\tilde{x}_i\}]_{l_i}]\!]_\Delta$

Since in PRISM synchronisations are modelled by labelling commands with *actions*, we associate the following *action* label with each message transmission:

$$[\mathtt{cn\_n_1...n_{h1}\_v_1...v_{h2}\_R_1..._R_{h3}}]$$

where

$\{\mathtt{c}\}$ is the channel of the transmission;

$\{\mathtt{n}\}$ is the sender node;

$\{\mathtt{n_1}, ..., \mathtt{n_{h1}}\}$ is the set of the nodes n_i $(i \in I)$ receiving the message;

$\{\mathtt{v_1}, ..., \mathtt{v_{h2}}\}$ is the tuple $\tilde{v}$ of messages;

$\{\mathtt{R_1}, ..., \mathtt{R_{h3}}\}$ is the set of locations whose distance from l is less than the radius.

The module of sender n contains the following command in PRISM (for $L_k = l$):

$[\mathtt{cn\_n_1...n_{h1}\_v_1...v_{h2}\_R_1..._R_{h3}}]\,(\mathtt{proc\_n{=}counter})\,\&\,(\mathtt{loc\_n{=}L_k})$
$\rightarrow\ (\mathtt{proc\_n'{=}counter{+}1});$

The module of a receiver node $n_i (i \in I)$ contains the following command in PRISM:

$[\mathtt{cn\_n_1...n_{h1}\_v_1...v_{h2}\_R_1..._R_{h3}}]\,(\mathtt{proc\_n_i = counter})\,\&\,((\mathtt{loc\_n_i = R_1})|\cdots|(\mathtt{loc\_n_i = R_k}))$
$\rightarrow (\mathtt{proc\_n'_i = counter + 1})\&(\mathtt{x'_{i1} = v_1})\&\ ...\ \&(\mathtt{x_{1h2} = v_{h2}});$

If I is the empty set, then it means that no nodes receive the transmission, and the message is lost. We add a command with a loss-message action to the sender's module:

$[\mathtt{cn\_v_1...v_{h2}\_lost}]\,(\mathtt{proc\_n = counter})\,\&\,(\mathtt{loc\_n = L_k}) \rightarrow\ (\mathtt{proc\_n' = counter + 1});$

- (R-Move) $n[P]_l \rightarrow [\![n[P]_l]\!]_{\mu_l^n}$

In PRISM the mobility of nodes is expressed using a different command for each row of the matrix $\mathbf{J}^n$ associated with each node. In particular, if $\mathtt{L_1}, ..., \mathtt{L_j}$ is the set *Loc* of the network locations, each i-th row of the matrix $\mathbf{J}^n$ corresponds to the command:

$$[]\,(\texttt{loc\_n}{=}\texttt{L}_\texttt{i}) \rightarrow \texttt{p}_{\texttt{i,1}}{:}(\texttt{loc\_n}'{=}\texttt{L}_\texttt{1}){+}\cdots{+}\texttt{p}_{\texttt{i,j}}(\texttt{loc\_n}'{=}\texttt{L}_\texttt{j});$$

The internal actions executed by a process P are expressed by a sequence of labelled commands:

- $\texttt{if}-\texttt{then}-\texttt{else}$: $P \equiv [v_1 = v_2]P1, P2$

  Each choice corresponds to the following command:

  $[\texttt{if\_n\_then}](\texttt{proc\_n} = \texttt{counter})\,\&\,(\texttt{v}_\texttt{1} = \texttt{v}_\texttt{2}) \rightarrow (\texttt{proc\_n}' = \texttt{counter} + \texttt{1});$

  $[\texttt{else\_n}](\texttt{proc\_n} = \texttt{counter})\,\&\,!(\texttt{v}_\texttt{1} = \texttt{v}_\texttt{2}) \rightarrow (\texttt{proc\_n}' = \texttt{counter} + \texttt{N});$

  where $\texttt{N}$ is the number of steps in P_1. Notice that the presence of the node's name inside the action label has been added in order to differentiate the action labels for each module's $\texttt{if}-\texttt{then}-\texttt{else}$ commands, since they are not synchronising actions.

- **Recursion** : $P \equiv A\langle w\rangle$

  Each recursion is expressed by the following command

  $[\texttt{rec\_n}](\texttt{proc\_n} = \texttt{counter}) \rightarrow (\texttt{proc\_n}' = \texttt{counter} - \texttt{N});$

  where N is the number of steps inside the recursion.

6.3.4 Correctness of the translation

Let $\mathcal{N}$ be the set of networks, and let $\mathbf{T}$ be the function mapping networks into corresponding MDPs. Let $M \in \mathcal{N}$ be a network, and let $\mathbf{T}(M)$ be its corresponding MDP model.

The translation captures all the behaviours expressed by the reduction semantics, i.e., there is a one-to-one correspondence between the reduction rules of M and the possible actions in $\mathbf{T}(M)$. This correspondence can be proved by induction on the reduction rules.

- (R-Bcast)

  $$\frac{}{n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid \prod_{i\in I} n_i[c(\tilde{x}_i).P_i]_{l_i} \rightarrow [\![n[P]_l \mid \prod_{i\in I} n_i[P_i\{\tilde{v}/\tilde{x}_i\}]_{l_i}]\!]_\Delta}$$

  where $0 < r \leq r_n$, $\forall i \in I.d(l, l_i) \leq r$, $r_i > 0$ and $|\tilde{x}_i| = |\tilde{v}|$

  In this case the module n contains the action

  $[\texttt{cn\_n}_\texttt{1}...\texttt{n}_\texttt{h1}\texttt{\_v}_\texttt{1}...\texttt{v}_\texttt{h2}\texttt{\_R}_\texttt{1}...\texttt{\_R}_\texttt{h3}]\,(\texttt{proc\_n}{=}\texttt{counter})\,\&\,(\texttt{loc\_n}{=}\texttt{L}_\texttt{k})$

  $$\rightarrow (\texttt{proc\_n}'{=}\texttt{counter}{+}\texttt{1});$$

  where $I = \{1, ...h1\}$, $\tilde{v} = v_1, ..v_{h2}$, $\{R_1, ...R_{h3}\} = \{k \in \mathbf{Loc} \mid d(l, k) \leq r\}$, $v_1, ..., v_{h2} = \tilde{v}$ and $l_i \in \{R_1, ...R_{h3}\}$ $\forall i \in I$.

  On the other end, each module n_i, $i \in I$ contains the action

$$[\texttt{cn\_n}_1...\texttt{n}_{h1}\texttt{\_v}_1...\texttt{v}_{h2}\texttt{\_R}_1...\texttt{\_R}_{h3}]\,(\texttt{proc\_n}_i = \texttt{counter})\,\&\,((\texttt{loc\_n}_i = \texttt{R}_1)|\cdots|(\texttt{loc\_n}_i = \texttt{R}_k))$$
$$\rightarrow (\texttt{proc\_n}'_i = \texttt{counter} + 1)\&(\texttt{x}'_{i1} = \texttt{v}_1)\&...\&(\texttt{x}'_{ih2} = \texttt{v}_{h2});$$

- (R-Move)

  $$\frac{}{n[P]_l \rightarrow [\![n[P]_l]\!]_{\mu_l^n}}$$

  Given $\mathbf{Loc} = \{l_1, ..., l_k\}$, the module n contains, $\forall l_i \in \mathbf{Loc}$, the row is:

  $$[]\,(\texttt{loc\_n}{=}\texttt{L}_i) \rightarrow \texttt{p}_{i,1}{:}(\texttt{loc\_n}'{=}\texttt{L}_1) + \cdots + \texttt{p}_{i,j}(\texttt{loc\_n}'{=}\texttt{L}_j);$$

  where $p_{i,j} = \mu_{l_i}^n(l_j)\ \forall j \in [1-k]$.

- (R-Par)

  $$\frac{M \rightarrow [\![M']\!]_\theta}{M|N \rightarrow [\![M'|N]\!]_\theta}$$

  The parallel composition is ensured by the fact that each module has always the possibility to perform non-synchronising actions. In the case of a (R-Bcast) reduction, e.g., there always exists the command labelled with a loose action, which avoids any synchronisation with other nodes.

- (R-Res)

  $$\frac{M \rightarrow [\![M']\!]_\theta}{(\nu\tilde{c})M \rightarrow [\![(\nu\tilde{c})M']\!]_\theta}$$

  When building the list of all possible output reductions, only the nodes sharing the same channels are included in the list of the synchronising nodes.

- (R-Struct)

  $$\frac{N \equiv M\ \ M \rightarrow [\![M']\!]_\theta\ \ M' \equiv N'}{N \rightarrow [\![N']\!]_\theta}$$

  The correctness of the translation with respect to this rule is ensured by the structural congruence closure of the reduction barbed congruence (See Table 3.2 in Chapter 3).

6.3.5 A simple example

In order to show how our networks become PRISM MDPs, following we will show the translation of a simple message exchange case. We use the toy example of a network composed of two mobile nodes trying to communicate.

Formally:
$M \equiv n_1[P_1]_{l_1} \mid n_2[P_2]_{l_2}$
where:

```
mdp

const L1=1;
const L2=2;

const msg1 = 1;
const msg2 = 2;

module n1

loc_n1 : [L1 .. L2] init L1;
x : [0 .. 2] init 0;
proc_n1 : [0 .. 2] init 0;

[] (loc_n1 = L1) -> 0.5:(loc_n1' = L1) + 0.5:(loc_n1' = L2);
[] (loc_n1 = L2) -> 0.5:(loc_n1' = L1) + 0.5:(loc_n1' = L2);

[cn1_n2_msg1_l1] (proc_n1 = 0) & (( loc_n1 = L1) ) -> (proc_n1' = 1);
[cn1_msg1_lost] (proc_n1 = 0) -> (proc_n1' = 1);

[cn2_n1_msg2_l1] (proc_n1 = 1) & ((loc_n1 = L1) ) -> (proc_n1'=2) & (x' = msg2);

[rec_n1] (proc_n1 = 2) -> (proc_n1' =0);
endmodule

module n2

loc_n2 : [L1 .. L2] init L2;
y : [0 .. 2] init 0;
proc_n2 : [0 .. 2] init 0;

[] (loc_n2 = L1) -> 0.5:(loc_n2' = L1) + 0.5:(loc_n2' = L2);
[] (loc_n2 = L2) -> 0.5:(loc_n2' = L1) + 0.5:(loc_n2' = L2);

[cn1_n2_msg1_l1] (proc_n2 = 0) & ((loc_n2 = L1) ) -> (proc_n2'=1) & (y' = msg1);

[cn2_n1_msg2_l1] (proc_n2 = 1) & (( loc_n2 = L1) ) -> (proc_n2' = 2);
[cn2_msg2_lost] (proc_n2 = 1) -> (proc_n2' = 2);

[rec_n2] (proc_n2 =2) -> (proc_n2' = 0);
endmodule
```

Figure 6.1: Sample

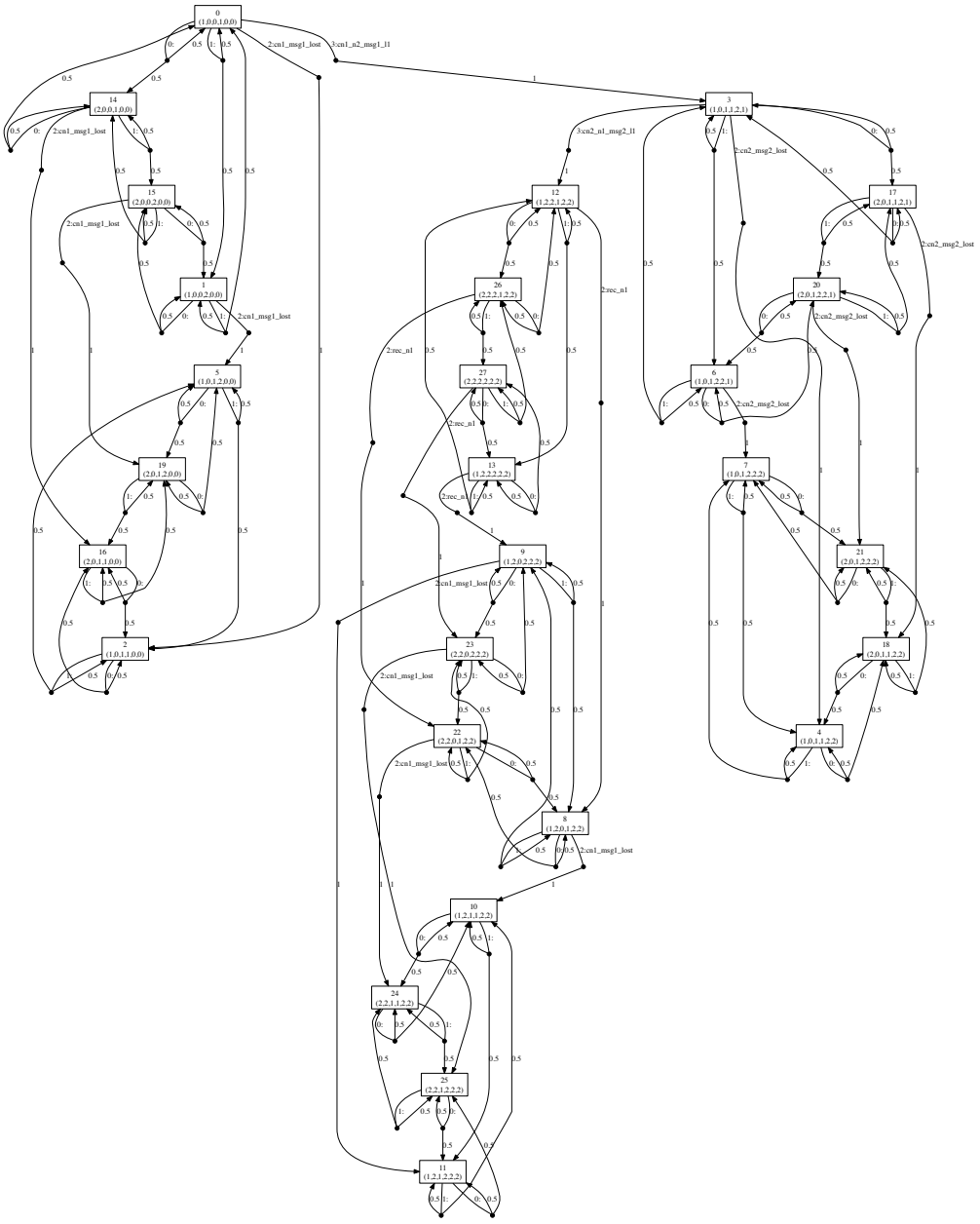


Figure 6.2: Transitions graph

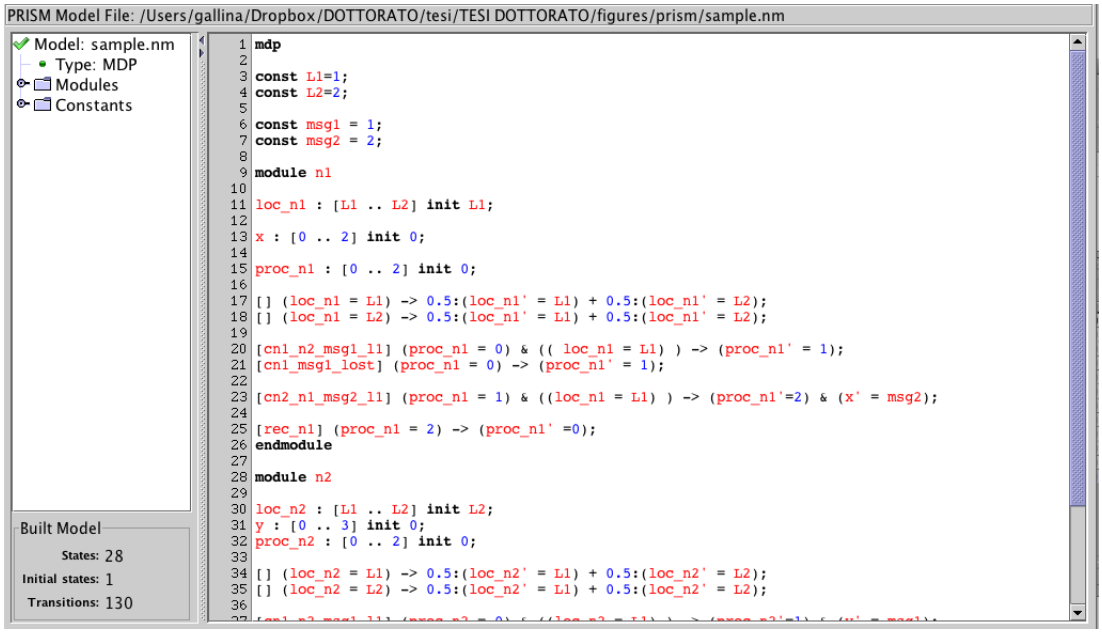


Figure 6.3: PRISM: Building the model

- $P_1 \equiv \bar{c}_{l1,1}\langle \texttt{msg1} \rangle.c(x).P_1$;
- $P_2 \equiv c(y).\bar{c}_{l1,1}\langle \texttt{msg2} \rangle.P_2$;

Following we write the text file corresponding to M, given as input file to the parser:

```
n1@l1{rec {msg1 -> c@l1/1 ;
(x) <- c }}
| n2@l2{ rec {(y) <-c;
msg2 -> c@l1/1}}

J =  l1 l2
[0.5 0.5,
0.5 0.5]

n1 : J
n2 : J

Dist = l1 l2
[ 3.0  ]
```

`J` is the matrix describing the probability distributions associated with the nodes. `Dist` is the triangular matrix describing the function $d(\cdot,\cdot)$: $\texttt{Dist}_{ij}$ indicates the distance between the location i and the location j.

Figure 6.1 shows the resulting PRISM MDP, having 28 states and 130 transitions; Figure 6.2 shows the resulting graph where nodes represent all the possible

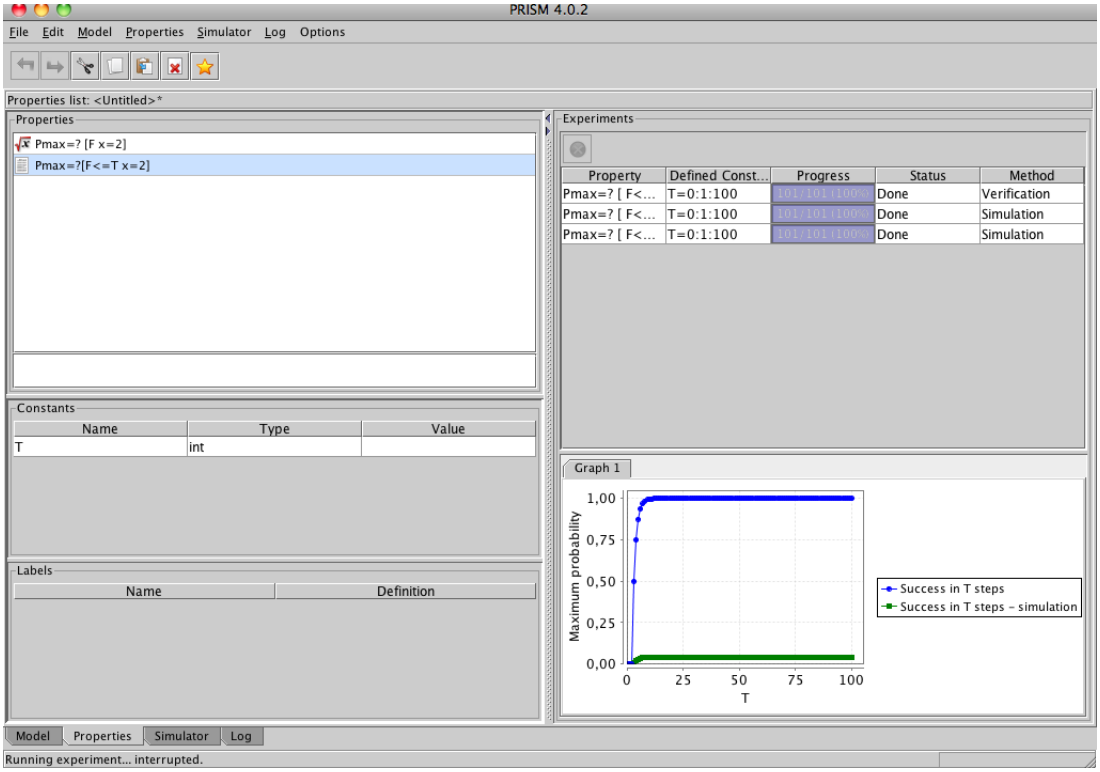


Figure 6.4: PRISM: Verification of properties and rewards

states the model can reach, while the edges represent all the possible transitions the model can perform. Notice that the probabilistic transitions are tagged with the correspondent probability, while the non-deterministic transitions are simply labelled with 1. Figure 6.3 shows a screenshot of the PRISM framework when building the network model.

The model can be used to prove some interesting properties related to the network behaviour. As an example, we may be interested in studying the probability that n_1 receives the message `msg2` from n_2. Hence we can write the following property:

```
Pmax=? [F x=2].
```

We can also study the probability that n_1 receives the message from n_2 in less than a given number of steps. We define a constant T and we write:

```
Pmax=? [F<=T x=2].
```

Figure 6.5 shows an example of simulation, while Figure 6.4 shows the results of the experiment run to compute the probability for n_1 to reveive $msg2$ in less than T steps. The experiment has been run in the verification as well as simulation mode. Notice that the difference between the results is very large: this is due to the choice of using MDPs, which allows to make non-deterministic choices.

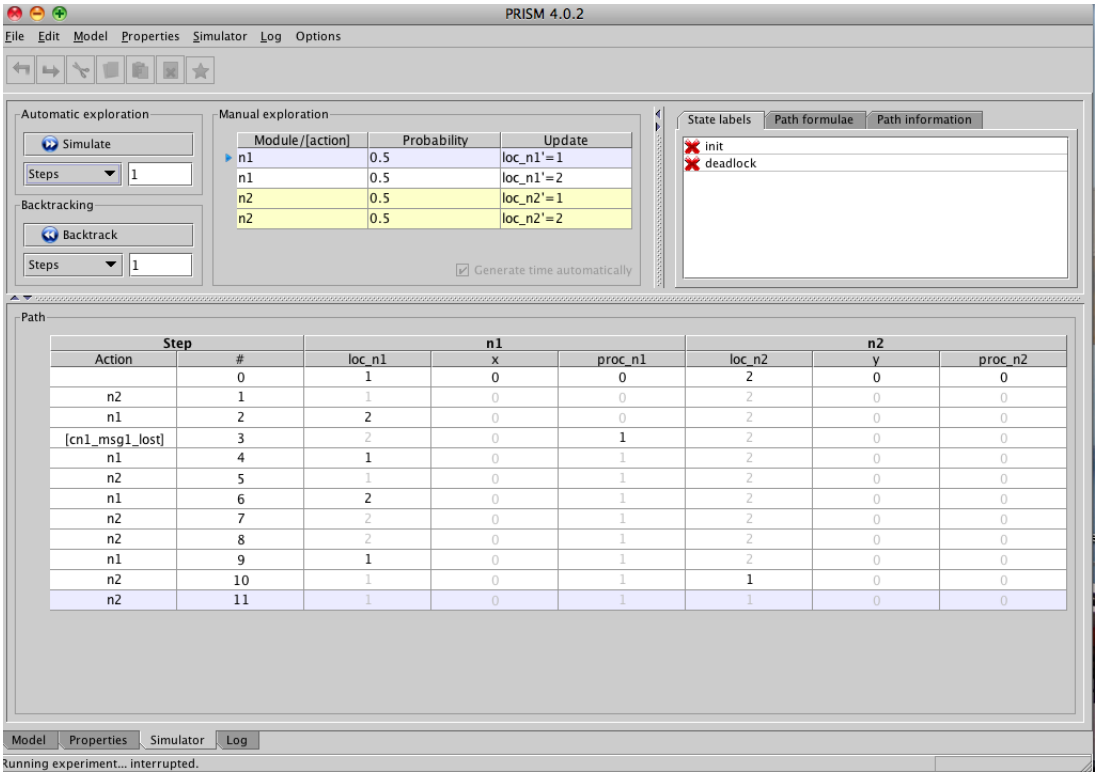
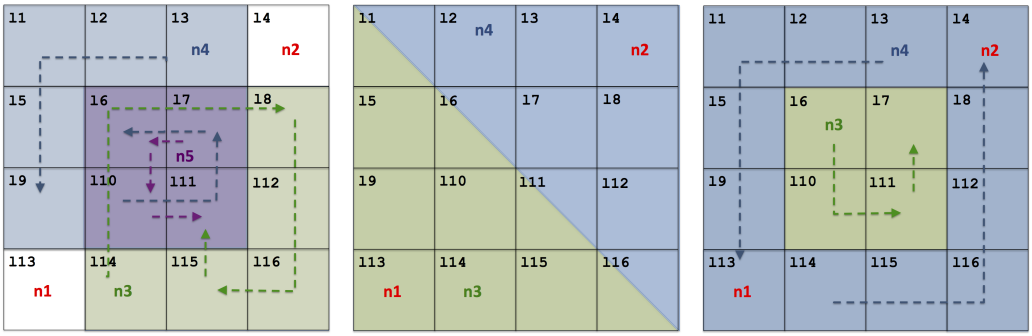


Figure 6.5: PRISM: Simulations



(a) Scheme 1 (b) Scheme 2 (c) Scheme 3

Figure 6.6: The different network schemes

6.4 A case study

We present a case study to show the effectiveness of translating our process algebra into an MDP in order to automatise the performance analysis for a range of metrics using PRISM.

6.4.1 The network.

We consider two static nodes, n_1 and n_2 communicating in a 60×60 metres network area (see Figure 6.6). The distances between cells are determined by considering the centre of each cell and calculating the euclidean distance between each pair of centres (each cell is 15×15 metres). Since the two static nodes are too distant to reach each other (compared to their radius 20), the communication is possible only if there are other intermediate nodes inside the network forwarding the messages between n_1 and n_2. We consider a simple flooding algorithm in which mobile nodes forward the messages in lazy mode, i.e., they store the data to be sent until there is another node ready to receive it. This scenario is typical for gossip protocols.

In the following, we analyse different situations, each with some intermediate nodes that may move in a fixed area of the network (that is formally captured by the probability transition matrix associated with each intermediate node).

Scheme 1 The first network we consider has three mobile nodes able to forward messages inside the network. As shown in Figure 6.6.(a), n_4 can move within the top-left square of nine locations, n_3 moves within the bottom-right square of nine locations, while n_5 can only move within the four central cells of the network area.

Scheme 2 The second scheme we consider is a network with only two mobile nodes where, as shown in Figure 6.6.(b), one of the two intermediate nodes can move in the

triangular area around the position of n_1, while the other one covers the triangular area around n_2.

Scheme 3 In the last scheme, shown in Figure 6.6.(c), one node can only move inside the central area of the network (4 cells), while the second one can only move in the 12 border cells.

We study the performance of these networks by varying the transmission power of the intermediate nodes, in order to find the best power strategy which optimises the performance in terms of energy consumption and throughput.

The three networks only differ in the transition probability matrix that is associated with each mobile node, while their behaviour (when $RAD \in \{20, 30, 40, 50\}$) is expressed in the EBUM calculus as follows :

$M_{RAD} \equiv \prod_{i=1}^{j} n_i[P_i]_{l_i}$, $j = \{4, 5\}$ where:

- $P_1 \equiv \bar{c}_{l4,20}\langle \texttt{msg\_for\_n2} \rangle.P_1'$
  $P_1' \equiv c(x_1).[x_1 = \texttt{ack\_to\_n1}]\bar{d}_{l13,20}\langle \texttt{ok} \rangle, P_1';$
- $P_2 \equiv c(x_2).[x_2 = \texttt{msg\_to\_n2}]\bar{c}_{l13,20}\langle \texttt{ack\_to\_n1} \rangle, P_2$
- $P_k \equiv c(x_k).\bar{c}_{\emptyset,RAD}\langle x_k \rangle.P_k, \quad k \in \{3, 4, 5\}$

with $j = 5$ for *scheme 1*, and $j = 4$ for *schemes 2, 3*.

We consider the restricted set of schedulers $\mathcal{F} \in Sched$ such that the nodes n_i always perform synchronisations where possible and they transmit a message only if at least one node is listening to the channel (i.e., messages are not lost).

First we prove that, by varying the value of *RAD*, among the values in the set $\{20, 30, 40, 50\}$, the probabilistic observational behaviour of M_{RAD} does not change.

Theorem 6.1 *For* $R1, R2 \in \{20, 30, 40, 50\}$,

$$M_{R1} \cong_p^{\mathcal{F}} M_{R2}.$$

Proof.

In order to prove the congruence it is sufficient to find a bisimulation containing the pair (M_{R1}, M_{R2}). Let us consider the relation:

$$\mathcal{R} = \{(\bar{M}, \bar{N}) : M_{R1} \rightarrow^* \bar{M} \text{ and } \bar{N} \equiv \bar{M}\{R2/R1\}\}.$$

We prove that $\mathcal{R}$ is a bisimulation we have to show that, $\forall \alpha$ and $\forall \mathcal{C} \in \mathcal{N}/\mathcal{R}$

- $\alpha = \tau$

  If the τ action has been caused by the application of the Rule (Move) the proof is trivial since $\bar{N} \equiv \bar{M}\{R2/R1\}$, and $\bar{N}$ can perform exactly the same movements as $\bar{M}$. If the τ action has been caused by an application of rule

(Lose) we have $\bar{M} \xrightarrow{c'_L!\tilde{v}[l,r]}_{\Delta} \bar{M}'$ for some channel $c' \in \{c, d\}$, some message $\tilde{v}$ some location l, some set L of locations and some radius r. If the transmission has been performed by n_1 or n_2 which uses a fixed radius 20, $\bar{N}$ is able to perform exactly the same output as $\bar{M}$ and again the bisimulation is proved. If the node transmitting is n_i, $i > 2$, then $r = R1$, meaning $\bar{N} \xrightarrow{c'!\tilde{v}[l,r']}$ with $r' = R2$. If $R2 \geq R1$, we are done, because, $\bar{N}$ can mimic the transmission performed by $\bar{M}$ and we get, by Definition 4.6 $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that

$$Prob^{F}_{\bar{M}}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F'}_{\bar{N}}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F'}_{\bar{N}}(\Longrightarrow, \mathcal{C}),$$

as required. If $R2 \leq R1$, then, since the Markov matrices modelling the mobility of the intermediate nodes are ergodic, the output action can be mimicked by a sequence of movements and outputs by $\bar{N}$ which reach the same receivers that have been able to synchronise with $\bar{M}$, and $\exists F' \in \hat{\mathcal{F}}_{\mathcal{C}}$ such that $\bar{N} \Longrightarrow \xrightarrow{c'!\tilde{v}[k_1,r']} \Longrightarrow \ldots. \Longrightarrow \xrightarrow{c'!\tilde{v}[k_n,r']} \Longrightarrow \bar{N}' \equiv \bar{M}'\{R2/R1\}$, where, $\forall l' \in \mathbf{Loc}.d(l,l') \leq r$, $\exists k_i$, $i \in [1-n]$ such that $d(l', k_i) \leq r'$. By applying rule (Lose) to each output action we get:

$$Prob^{F}_{\bar{M}}(\xrightarrow{\tau}, \mathcal{C}) = Prob^{F'}_{\bar{N}}(\Longrightarrow, \mathcal{C}),$$

as required.

- $\alpha = c'!\tilde{v}@K \triangleleft R$

  We notice that the only observable action is the transmission $d!\mathtt{ok}@\{l13\} \triangleleft \{l9, l13, l14\}$ performed by n_1 with a fixed radius 20. This mean that $\bar{N}$ can perform exactly the same action as $\bar{M}$.

- $\alpha = c'?\tilde{v}@k$

  The proof is simple since $\bar{N}$ only differs from $\bar{M}$ for the radius used in the output actions, while the networks can perform exactly the same input.

For each scheme we build a PRISM MDP, assigning to the variable RAD the values in the set $\{20, 30, 40, 50\}$, in order to investigate how different transmission powers influence the performances of the networks.

We use PRISM to verify the following property which computes the maximum probability (among all schedulers) of eventually reaching the goal states:

```
Pmax=? [F goal_state]
```

From the experiments, we obtain probability 1 for all the different schemes, and for all possible radii. In the next paragraph, time and energy cost information is added to the MDP. We investigate how the cost will be affected by varying the network topology and the transmission radii of the nodes.

6.4.2 Time and Energy Costs.

In order to analyse the performance of our network, we define a cost function in terms of energy consumption and time delay. In our analysis we study the energy spent in the transmissions assuming that the cost for the control packets is negligible.

We assume quasi-linear cost functions of the form

$$\sum_k w_k \cdot \Phi_k(\_) \tag{6.1}$$

that are sums of arbitrary functions $\Phi_k(\_)$, each of which maps one metric (e.g., energy cost and throughput) to some common measure. In general, we give a different weight $w_k \geq 0$ to each $\Phi_k(\_)$ to reflect its importance. We assume that $\sum_k w_k = 1$.

In the following, we first introduce the energy cost $\Phi_{\mathbf{E}}$, then the time cost $\Phi_{\mathbf{T}}$ and finally the global cost function.

Energy Cost. We associate an energy cost with the probabilistic network reductions, as follows:

$$\mathbf{e}(M,N) = \begin{cases} (\mathtt{En}_{elec} \times \mathtt{packet\_len} + \\ \mathtt{En}_{ampl} \times \mathtt{packet\_len} \times r^2) \\ \qquad \text{if } M \rightarrow_\theta N, M \equiv n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid M', \\ \qquad N \equiv n[P]_l \mid N' \text{ for some } c, L, \tilde{v}, l \\ 0 \quad \text{otherwise.} \end{cases}$$

where $\mathtt{En}_{elec}(\ nJ/b)$ is the energy dissipated to run the transmitter circuit, $\mathtt{En}_{ampl}$ $(pJ/b/m^2)$ is the Radio amplifier energy. They are constants given a priori (see [57]).

Time Cost. We associate time cost with the probabilistic network reductions as follows:

$$\mathbf{t}(M,N) = \begin{cases} (\mathtt{packet\_len/bandwith}) \\ \quad \text{if } M \rightarrow_\Delta N, M \equiv n[\bar{c}_{L,r}\langle\tilde{v}\rangle.P]_l \mid M' \\ \quad \text{and } N \equiv n[P]_l \mid N' \text{ for some } c, L, v, l \\ (\mathtt{cell\_size/velocity}) \\ \quad \text{if } M \rightarrow_{\mu_l^n} N, M \equiv n[P]_l \mid M' \\ \quad \text{and } N \equiv n[P]_k \mid M' \end{cases}$$

where $\mathtt{cell\_size}$ (m) is the size of a cell, $\mathtt{velocity}$ (m/s) is the velocity of the devices and $\mathtt{bandwith}$ (MB) is the channel bandwidth of the network. Again they are constants.

RAD [m]	TIME [s]	ENERG [J*1(
20	1269.45	5360.
30	884.04	6880.
40	874.06	1248(
50	874.06	1680(

(a) Scheme 1

RAD [m]	TIME [s]	ENERG [J*1(
20	448.70	5279
30	200.72	6880
40	205.48	5760
50	205.48	7199

(b) Scheme 2

RAD [m]	TIME [s]	ENERGY [J*10^{-7}]
20	1333.33	3039.99
30	34.67	4029.33
40	34.67	4960.00
50	166.67	6400.00

(c) Scheme 3

Figure 6.7: The time-only and energy-only cost

Composition of Cost Functions Now consider Definition 4.8, and let $\Phi_E = \mathbf{Cost}^{\mathbf{e}}$ and $\Phi_T = \mathbf{Cost}^{\mathbf{t}}$, and w be the weight of the energy function. We define the cost of reaching a set H from the network M according to the scheduler F as:

$$\mathbf{Cost}^{(\mathbf{e}\circ\mathbf{t})}{}_M^F(H) = \alpha\, \mathbf{Cost}^{\mathbf{e}}{}_M^F(H) + \beta\, \mathbf{Cost}^{\mathbf{t}}{}_M^F(H), \tag{6.2}$$

where α and β are weighting coefficients expressed in s^{-1} and J^{-1} respectively, and henceforth we simply consider $\alpha = w$ and $\beta = 1 - w$, with $w \in [0, 1]$.

According to the energy and time costs we introduced above, in our PRISM MDP the constants are given as follows:

```
const double cell_size = 10.0;
const double bandwith = 1.0 ;
const double packet_len = 200.0;
const double velocity = 0.3 ;

//ENERGY FOR COMMUNICATION
//50 nJ/b*packet_len+100 pJ/bit/m^2*packet_len*r^2

const double bcast_pow_n1= 800+1.6*RAD1*RAD1;
const double bcast_pow_n2= 800+1.6*RAD2*RAD2;
const double bcast_pow_n3= 800+1.6*RAD4*RAD4;
const double bcast_pow_n4= 800+1.6*RAD3*RAD3;

const double move_power = 0;

const double move_time = cell_size/velocity;
const double bcast_time = packet_len/1000000 ;
```

where `bcast_pow_ni` and `bcast_time` are the energy and time spent for a single transmission performed by n_i, while `move_power` and `move_time` are the energy and time spent for each single movement step. Notice that only the cost of the

transmissions depends on the identity of the sender nodes, i.e. to on the radius each node uses to communicate. The time and energy costs of each movement are fixed because we consider devices having the same physical characteristics, which performs a movements of $15 - 20$ metres for each move step. The time cost of each transmission is fixed because we assume that all the packets have the same size.

The cost can be calculated by defining a reward structure:

```
rewards "cost"
[]       true : (1-w)*move_time + w*(move_power);
[out_ni] true : (1-w)*bcast_time +w*bcast_pow_ni;
...
endrewards
```

Since for all the schemes introduced above we proved that, for each pair $R1, R2 \in \{20, 30, 40, 50\}$, $M_{R1} \cong_p^{\mathcal{F}} M_{R2}$, we are ready to investigate if there exists, for each scheme, a preorder among M_{20}, M_{30}, M_{40} and M_{50}. This means that we have a way of choosing the radius optimising the performance of the network in terms of time and energy cost.

We use the PRISM tool to automatically verify the second point of Def. 4.9, by calculating, for each radius in the set $\{20, 30, 40, 50\}$, the minimum cost among all schedulers under which the goal state is reached with probability 1:

```
R{"cost"}min=? [F goal_state]
```

Fig. 6.7 shows the time and energy costs we calculated for each scheme, while the general cost functions are specified in Fig. 6.8 (time cost is expressed in s while energy is expressed in $J \times 10^{-7}$). Here we summarize our results.

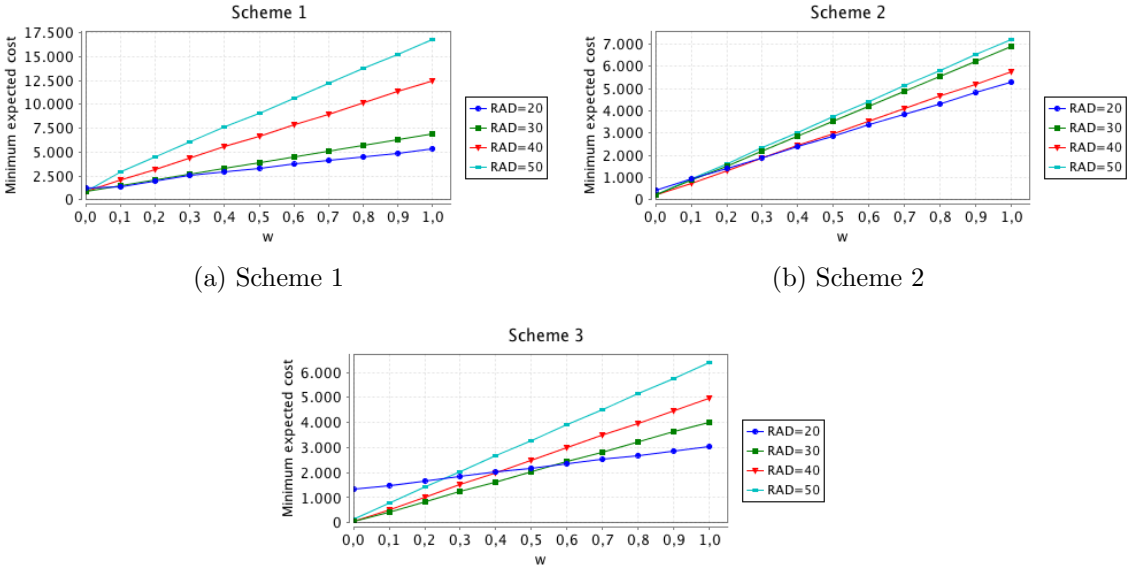
(i) Time cost ($w = 0$). For each scheme, a higher time cost is needed for $RAD = 20$ (see Figure 6.7).

This is due to the fact that a small radius requires more rounds of forwarding and more movements of the nodes: both these factors increase the time cost.

(ii) Energy cost ($w = 1$). In all of the three schemes in Figure 6.7, the smallest radius always has the minimum energy cost. This is because, regardless of time, it is always possible that mobile nodes move closer to each other or move close to n_1 or n_2, and this reduces the energy cost. It is worth pointing out that, due to a scaling representation factor, Figure 6.8 tends to hide the relative differences in the total weighted cost among the radii for $w = 0$.

(iii) For $0 < w < 1$ in general, we investigate for each scheme how the total cost is affected by choosing different power strategies (i.e., transmission radii). The results are shown in Figure 6.8.

Scheme 1 It can be seen in Figure 6.7 that radius 30 is better than 20 in terms of delays (about $\frac{884}{1269} = 70\%$ of time cost for radius 20), and radius 20 is better



(a) Scheme 1

(b) Scheme 2

(c) Scheme 3

Figure 6.8: Cost results for each scheme

than 30-50 in terms of energy (about $\frac{5360}{6880} = 78\%$ of the energy cost for radius 30). So the best radius depends on the w we choose: roughly speaking, if $w \leq 0.6$, then radius 30 is better than 20; otherwise 20 is better. We notice that here large radii consistently deteriorate the energy performance of the network, without a real improvement in terms of transmission delays: this is due to the fact that there is a high density of nodes in the network area, which can cause unnecessary and expensive transmissions.

Scheme 2 In Scheme 2 (see Figure 6.8.(b)), the smallest (largest) radius still seems to give the best (worst) cost. However, using radius 40 is much better than 30 due to the mobility behaviours associated with the intermediate nodes: with radius 30, n_3 is only able to reach n_1, while n_4 can only forward n_2 communications: this situation enlarges the minimum number of transmissions allowing n_1 and n_2 to complete their communications. Conversely, choosing a larger radius (40) enables n_3 (respectively n_4) to eventually reach n_2 (respectively n_1), drastically reducing the number of transmissions. Since the energy cost using radius 40 is not much higher than the cost spent using radius 20, while the time costs are halved, we can conclude that for the second scheme the best power management strategy for time-aware applications is to use radius 40.

Scheme 3 Finally, let us consider the third scheme (see Figure 6.8.(c)). With $w \leq 0.57$ the best solution is radius 30, while with $w > 0.57$ the best strategy is radius 20.

However, if we are interested only in reducing the time spent to communicate (see Fig. 6.7.(c)), radius 40 is the best choice, since the costs are consistently reduced with respect to the other radii, while radius 20 critically increases time cost. Generally speaking, the best solution (in order to reduce both energy and time costs) is to use radius 30. We notice that 30 is the smallest distance that the inner area (4 cells) and the outer area (12 cells) can always directly reach each other, while radius 20 never allows n_3 to reach n_1 and n_2. For these reasons time cost consistently increases for radius 20, while with radius 30 is better (about $\frac{34.67}{1333.33} = 2.5\%$ of the time cost for radius 20). With respect to radii 40 and 50, both the radius 30 and the radius 20 allow one to reduce the energy cost, but only radius 30 do not significantly deteriorate the time delays.

6.5 Conclusions

In this chapter we presented an automatic tool, based on the probabilistic calculus introduced in 4, for comparing different costs of networks having the same probabilistic observable behaviour. The automatic quantitative verification of a network has been implemented by model checking. In particular we provided a translation of the EBUM models into PRISM MDP models, which allows one to automatically verify the performances of the networks in terms of various metrics.

In Chapter 4 we introduced a Preorder with the aim of comparing different networks, having the same connectivity behaviour, but different performances in terms of energy conservation or other metrics. This preorder has been defined in two steps: we first use the bisimulation proof technique to verify that two systems have the same connectivity behaviour, then we compare the average costs of the systems to reach a particular state. While in the literature there are several frameworks to automatically verify the bisimilarity of two systems, in this chapter we provided a way to automatically verify the second step of the preorder: this framework can be used to predict the performances of ad hoc or sensor networks in their planning stage.

Conclusions

Ad hoc networks is an area of mobile communication networks that has attracted significant attention due to its challenging problems. The main goal of our work is to provide a formal model to reason about the problem of limiting the energy consumption of the communications, without compromising the network connectivity.

As a first contribution, we define the EBUM calculus: a process calculus for the analysis of mobile ad hoc and sensor networks, supporting broadcast as well as unicast and multicast communication, the possibility for each node to arbitrarily connect and disconnect from the network and the possibility for the mobile nodes to arbitrarily move within the network area. Another important characteristic of the EBUM calculus is the ability of a node to change its transmission radius (and, as a consequence, the transmission power) in accordance with the protocol it is executing. Our calculus results to be a valid formal model for an accurate analysis of ad hoc and sensor networks, and allows us to evaluate and compare the behaviours of the energy-aware protocols and algorithms used to mange the communications.

The second contribution of this thesis is the development of a probabilistic version of the EBUM calculus, where we use probabilities to model the mobility of the nodes.This extension allows us to give a more realistic representation of the networks, where the movements of the devices are not completely casual, but they follows particular trajectories, caused, e.g., by the physical obstacles of the network area. The introduction of the probabilities in the model allows us to make both a qualitative and a quantitative analysis, and to compare different networks which have the same connectivity but different performances, in terms of various metrics such as throughput and energy consumption. This is important because, during the network implementation, there always exists a trade-off between the energy conservation and the connectivity maintenance.

The third contribution of this thesis is the development of another probabilistic version of the EBUM calculus, where again probabilities are used to model the mobility of nodes, but where the communications among the network nodes are modelled as non-atomic actions. This characteristic allows us to capture the collisions which can occur when the transmission areas of different sender nodes overlap. We can use our model to make a qualitative and quantitative analysis of the performances of the networks, concentrating the attention on the level of interference in terms of the number of collisions occurring during the network communications.

The last contribution of this thesis is the presentation of a tool, based on the PRISM Model Checker, for the automatic verification of mobile ad hoc and sensor networks for a range of performance metrics. Specifically, we develop a parser to translate EBUM terms into models expressed in PRISM language, and we provide

a formulation of the metrics for the computation of the time and the energy costs of the communications in terms of reward structures. The PRISM framework has been already used to analyse the performances of several communication protocols for wireless networks (see, e.g. [18] , [48]). In this thesis we combined the powerful of process algebra to compare the behaviours of different systems, and the possibility to make automatic derivation of some system performances. While in the literature there already exists applications of formal verification techniques to process algebras (e.g., [70], [16]), to the best of our knoledge, as concerns the process algebras specifically developed to analyse mobile ad hoc and sensor networks, no automatic verification has already been implemented.

A

A Parser for the Probabilistic EBUM Calculus

In the following we will give a detailed description of the parser, presented in Chapter 6 which has been implemented in order to translate an EBUM network into a PRISM mdp.

A.1 Development Environment

The Parser has been realized using Microsoft Visual Studio 2010, with the addition of the FSharp PowerPack 2.0.

In particular, we implemented a translator in F# with FsYacc and FsLex.

The resulting project is not simply aimed at building PRISM MDPs, but it can be used to translate EBUM networks in other languages: as an example, we can use one of the existing tools supporting bisimulation in order to automatically verify the behavioural equivalence of different networks.

Indeed the parser here described is an important contribution to our work, since, having a way of interacting with other framework make our calculus useful in practical situations.

A.2 Structure of the Parser

In the following we introduce the source files managing the different steps of the parsing task, and we report some sketch of the source code.

Ver.fs : is a simple file only containing the version of the parser.

Prelude.fs : is a file containing a set of general auxiliary functions. As an example:

```
let identity x = x
```

returns the input element, while:

```
let capitalize (s : string) =
       if s.Length > 1
                  then s.Substring(0, 1).ToUpper() + s.Substring(1)
                  else s.ToUpper()
```

converts the first character of a string with the correspondent upper case character.

Config.fs : is a file containing the functions for the static configuration of the parsing task, e.g., all the auxiliary functions for the creation of the Log File.

Log.fs : is a file containing the functions to write the log.

Env.fs : is a file containing a set of polymorphic functions useful for the information management during the parsing task. As an example:

```
let map f (Env m) = Env (Map.map f m)
```

maps the function `f` on all the elements in the environment `m`, while

```
let exists f (Env m) = Map.exists f m
```

controls the existence of an element in `Env`, satisfying `f`.

Absyn.fs is the file defining the Abstract Syntax Tree (ATS), and it contains the definition of all the data types (channels, nodes, distances, etc.), and the auxiliary functions elaborating and preparing the parsed input file, for the creation of the labelled transition system of the input EBUM network. The types for simple values, names and channels are:

```
type id = string
type channel = id
type location = id
type radius = float
type node = id

type value = Int of int
             | Float of float
             | Char of char
             | String of string
             | Id of id
```

Types for networks are expressed as follows:

```
type proc = PEmpty
        | Receive of id list*channel*proc
        | Send of value list *channel*location list*radius*proc
        | If of value*value*proc*proc
        | Rec of proc
```

where `PEmpty` denotes the empty process, `Receive` and `Send` respectively the process input and output actions, `if` the if-then-else construct, and `Rec` the recursion. The network type definition is:

```
type net = NEmpty
         | Exec of node*location*proc
         | Parallel of net*net
         | Restrict of id*net
```

where `NEmpty` denotes the empty network, `Exec` a network constituted of a single node executing a process, `Parallel` the parallel composition of two network an `Restrict` the channel restriction. The matrices describing the mobility of the nodes, and the matrix describing the distances among the locations are expressed as

```
type matrix = {
              labels  : id list
              values  : float[,] }
```

where `label` is the list of all the locations names of the network topology, while `values` contains all the values of the matrix. Finally a program is expressed as following:

```
type program = {
               net         : net
               matrix_env  : Env.t<id, matrix>
               node_env    : Env.t<id, id>
               dist        : matrix }
```

where `net` is the network (the parallel composition of nodes executing processes), `matrix_env` is a list of pairs, where each pair is made of a name, and the correspondent Markov matrix describing a mobility probability distribution; `node_env` is a list of pairs, where each pair is made of a node name, and the correspondent name of the markovian matrix describing its mobility; finally, `dist` is the triangular matrix describing the distance function of the program. The file presents another type which is used to build the labelled transition tree: in particular the following type describes the element of an output reduction; the definition of the output type is:

```
type out_entry = {
            node : node;
            nodes: node list;
            ch   : channel;
            rad  : radius;
            msg  : value list;
            locs  : location list;
            locsR : location list}
```

where `node` is the sender node, `nodes` is the set of the synchronising receiver nodes, `ch` is the transmission channel, `rad` the transmission radius, `locs` is the list of all the locations the sender node may occupy during the transmission, and `locsR` the set of all the locations inside the network area. The remaining functions in Absyn.fs allow the creation of the reduction tree: the set of all possible behaviours of the input network can be deduced by associating the processes definition with the distance function and with the mobility matrices, in order to infer the set of all the possible synchronisations occurring during the processes executions. As an example, the function:

```
let find_dist = fun loc1 loc2 dist->
    let rec find_index = fun loc locs count->
        match locs with
              [] -> -1
              | x::xs -> if (x = loc)
                         then count
                         else find_index loc xs (count+1)
    in let l1 = find_index loc1 dist.labels 0
      in let l2 = find_index loc2 dist.labels 0
           in if l1 = l2 then 0.0 else
               if l1 <  l2 then (dist.values.[l1,l2])
                           else (dist.values.[l2,l1])
```

computes the distance among two locations (`loc1` and `loc2` are locations, while `dist` is the matrix representing the distance function). The function:

```
let create_output_list net prog
  let rec aux = fun net prog n->
   match n with
     NEmpty -> []
    |Exec (node, loc, proc) ->
        if ((Env.lookup node prog.node_env).Equals("I"))
           then ( create_out_list_proc_sta proc node net loc prog.dist)
           else ( create_out_list_proc proc node net prog.dist)
```

```
    |Parallel (net1, net2) -> List.append (aux net prog net1)
                                           (aux net prog net2)
    |Restrict (chan,net2) ->
              let nodes = (restricted_channel net chan)
                     in aux net prog net2
     in aux net prog net
```

creates the list of all the possible network synchronisations, by exploring the network and analysing each single output action, in order to discover the set of all the possible synchronisations which can occur, depending on the locations covered by the receiver nodes.

Lexer.fsl : is the source file specifying the grammar in the FsLex metalanguage. As an example, the code:

```
// brakets
| '('{ BRA }
| ')'{ KET }
| '['{ SQBRA }
| ']'{ SQKET }
| '{'{ CURBRA }
| '}'{ CURKET }

// punctuation
| ";" { SEMICOLON }
| ":" { COLON }
| "," { COMMA }
| "." { DOT }
| "@" { AT }
| "0" { ZERO }
```

recognizes the symbols of the input network regular grammar, and encodes them with a list of key words that will be used by the parser to translate the input file.

Parser.fsy : is the source file specifying the grammar in the FsYacc metalanguage. As an example, the code:

```
proc:
    proc_atom { $1 PEmpty }
  | proc_atom SEMICOLON proc { $1 $3 }
  | IF value EQ value THEN proc ELSE proc { If ($2, $4, $6, $8) }
  | REC CURBRA proc CURKET { Rec $3 }
```

```
  | ZERO { PEmpty }
  | BRA proc KET { $2 }
```

is the grammar for processes.

Parsing.fs : is a file containing a set of auxiliary functions for the parsing phase.

Main.fs : is the file containing the main code to parse the input file. The main code of the file is:

```
module Test =

let parse_program filename mode=
          ignore (Parsing.load_and_parse_program filename mode)


[<EntryPoint>]
let main (args : string[]) =
    if args.Length <> 2 then usage ()
    else try Test.parse_program args.[0] args.[1]
        with :? Parsing.syntax_error as e ->
                        fatal_error "syntax error: %s" e.Message
           | Env.Report.UnboundSymbolError s    ->
                        fatal_error "error: %s" s
           | Unexpected s ->
                        unexpected_error "%s" s
           | e ->
                       fatal_error "uncaught exception: %s" e.Message
    ignore <| System.Console.ReadKey ()
    0
```

The module `Test` parses the input file and manages the eventual error occurrances.

The code above described constitutes the general parser for a probabilistic EBUM process, and it can be used to translate an EBUM network in any language supporting the representation of labelled transition systems.

As concerns the creation of the PRISM MDPs, the following file have been created on purpose.

Functions.fs : is the file containing general auxiliary functions that will be used for the translation. As and example, the function:

```
let eliminate_duplicates = fun in_list ->
    let rec aux = fun l1 l2 ->
            match l1 with
                 []  -> l2
                |x::xs -> if(is_in x l2) then (aux xs l2)
                                         else (aux xs (x::l2))
       in aux in_list []
```

eliminates the duplicate elements of a list, while the function:

```
let rec add_el = fun e l ->
    match l with
         [] -> []
        |x::xs -> (e::x)::(add_el e xs)

 let rec power_set = fun list l2 ->
    match list with
        [] -> [[]]
        |x::xs -> (List.append (add_el x (power_set xs l2))
                                         (power_set xs l2))
```

creates the power set of a given list. The file contains also some of the functions participating to the output file creation. As an example, the functions:

```
let header = "mdp\n"

let locations (locs : location list) =
      List.fold (fun s (id: string) ->
             List.append s [id.ToUpper()]) [] locs

let locations_constants (d : matrix) =
 let (s, _ ) = List.fold (fun (s,cnt) (id : string) ->
  (s+"const "+id.ToUpper()+"="
  +(cnt+1).ToString()+";\n" , cnt+1))
  ("",0) d.labels
   in s+"\n"
```

create the header and the global constants of the mdp, while

```
 let module_header (node : node) = "module "+ node + "\n"

 let loc_var  (node : node) (loc : location) (d : matrix) =
  let s =
```

```
      "loc_"+node+" : ["+(List.head (locations d.labels))+" .. "+
      (last (locations d.labels))+"] init "+loc.ToUpper()+";\n"
     in s

let print_variables = fun l n p ->
  let print_variable = fun (net : net)  prog (var : id) ->
     let count = (count_members (
                          eliminate_duplicates (
                            create_message_list prog.net (
                              create_variable_list prog.net))))
       in var+" : "+"[0 .. "+count.ToString()+"] init 0;\n"
         in let rec aux = fun list net prog->
           match list with
                [] -> ""
               |x::xs -> (print_variable net prog x)+
                              (aux xs net prog)
            in aux l n p
```

create the header and the variable of each mdp module.

Ebum.fs : is the file creating and writing the output file in the PRISM language: The file created will have the same name of the input file, but with the extension *.nm*, that is the extension for the PRISM mdps. The file is again constituted of a set of functions; as an example the function:

```
let rec print_modules = fun net prog out ->
     match net with
           NEmpty -> "\n"
         | Exec(node,location,proc) ->
                (print_module node location proc prog out)+"\n\n"
         | Parallel (net1,net2) ->
                (print_modules net1 prog out)+
                (print_modules net2 prog out)
         | Restrict(c,net1) ->
                 print_modules net1 prog out
```

create the mdp modules corresponding to each network node, where `net` is the network, `prog` the whole input program, and `out` the list of all possible broadcast actions the network can perform. The functions for the output file creation are:

```
let save_string_to_file filename (s : string) =
  use fstr = new IO.FileStream (filename, IO.FileMode.Create)
```

```
        in use wr = new IO.StreamWriter (fstr)
           in wr.Write s

let create_file prog filename =
let s =
   Functions.header+
  (Functions.locations_constants prog.dist)+
  (Functions.messages_constants prog.net)+
  (print_modules prog.net prog (create_output_list prog.net prog))
    in let writing =
       save_string_to_file (filename.Replace(".mc", "_p.nm")) s
      in s
```

where the first function creates the file, while the second function combine all the strings created for each part of the final mdp network in a single string that will be then written in the output file.

A.3 The parsing task: results and performances

The compilation of he project produces an executable file, which can be run by a Microsoft Windows system. The execution of the application creates the desired mdp, written in the PRISM language, corresponding to the input EBUM network.

The proof of the correctness can be found in Chapter 6, together with a case study showing the practical usefulness of the resulting framework.

The response time of the parsing task is of the order of few seconds, but can vary, depending on the network dimension, and on the number of instructions constituting the nodes processes.

The final Output of the program is a PRISM model, that can be built and used by the PRISM framework.

The model captures all the possible transitions that the reduction semantic infer for the input EBUM network, as well as a set of transitions that have probability 0 to occur (e.g., the parser, for each transmission, consider that the sender may perform the output action from any network location, while it is not sure that the mobility behaviour of that node covers the entire network area).

This problem can be easily solved with a brief pruning operation consisting in manually eliminating all the actions with probability 0 of the final MDP.

These kinds of pruning operations will make the MDP more readable, and reduce the number of states and transitions.

Bibliography

[1] M .Abadi and C. Fournet. Mobile Values, New Names, and Secure Communication. *SIGPLAN Not.*, 36(3):104–115, 2001.

[2] A. Acquaviva, A. Aldini, M. Bernardo, A. Bogliolo, E. Bontà, and E. Lattanzi. A Methodology Based on Formal Methods for Predicting the Impact of Dynamic Power Management. In *Formal Methods for Mobile Computing*, volume 3465 of *lncs*, pages 51–58. Springer Berlin / Heidelberg, 2005.

[3] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless Sensor Networks: A Survey. *Computer Networks*, 38(4):393 – 422, 2002.

[4] A. Aldini, M. Bernardo, and F. Corradini. *A Process Algebraic Approach to Software Architecture Design*. Springer Publishing Company, Incorporated, 1st edition, 2009.

[5] C. Baier, F. Ciesinski, and M. Grosser. PROBMELA: a Modeling Language for Communicating Probabilistic Processes. In *Proc. of the 2nd ACM and IEEE International Conference on Formal Methods and Models for Co-Design (MEMOCODE '04)*, pages 57– 66. ieee, 2004.

[6] C. Baier and J.P. Katoen. *Principles of Model Checking*. The MIT Press, 2008.

[7] J.A. Bergstra and J.W. Klop. Algebra of Communicating Processes. *NASA STI/Recon Technical Report N*, 85:28214, 1984.

[8] M. Bernardo and M. Bravetti. Performance Measure Sensitive Congruences for Markovian Process Algebras. *Theoretical Computer Science*, 290(1):117–160, 2003.

[9] S. Bucchegger and J.Y. Le Boudec. Cooperative Rrouting in Mobile Ad-hoc Networks: Current Efforts Against Malice and Selfishness. In *Proc. of Mobile Internet Workshop*, 2002.

[10] M. Bugliesi, L. Gallina, S. Hamadou, A. Marin, and S. Rossi. Interference-sensitive preorders for manets. Technical Report DAIS-2011-10, University Ca' Foscari Venice, 2011.

[11] M. Bugliesi, L. Gallina, S. Hamaodu, A. Marin, and S. Rossi. Interference-sensitive Preorders for MANETs. In *Proc. 9th International Conference on Quantitative Evaluation of SysTems (QEST '12)*. IEEE, 2012.

[12] M. Burkhart, P. von Rickenbach, R. Wattenhofer, and A. Zollinger. Does Topology Control Reduce Interference? In *Proc. of the 5th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc '04)*, pages 9–19. ACM, 2004.

[13] L. Cardelli and R. Mardare. The Measurable Space of Stochastic Processes. In *Proc. of the 7th International Conference on the Quantitative Evaluation of Systems (QEST'10)*, pages 171–180. ieee, 2010.

[14] A. Cerone and M. Hennessy. Modelling probabilistic wireless networks. In *Formal Techniques for Distributed Systems*, volume 7273 of *lncs*, pages 135–151. Springer Berlin / Heidelberg, 2012.

[15] F. Ciesinski and C. Baier. LiQuor: A tool for Qualitative and Quantitative Linear Time analysis of Reactive Systems. In *Proc. of the 3rd International Conference on Quantitative Evaluation of Systems (QEST'06)*, pages 131–132. IEEE CS Press, 2006.

[16] F. Ciocchetta and J. Hillston. Bio-PEPA: A Framework for the Modelling and Analysis of Biological Systems. *Theoretical Computer Science*, 410(33-34):3065 – 3084, 2009.

[17] F. DeRemer and T. Pennello. Efficient Computation of LALR(1) Look-ahead Sets. *ACM Transactions on Programming Languages and Systems*, 4(4):615–649, 1982.

[18] A. Fehnker and P. Gao. Formal Verification and Simulation for Performance Analysis for Probabilistic Broadcast Protocols. In *Proc. of the 5th International Conference on Ad-Hoc, Mobile, and Wireless Networks (ADHOC-NOW'06)*, volume 4104 of *LNCS*, pages 128–141. Springer, 2006.

[19] A. Fehnker, R. van Glabbeek, P. Höfner, A. McIver, M. Portmann, and W. Tan. A process algebra for wireless mesh networks. In *Programming Languages and Systems*, volume 7211 of *lncs*, pages 295–315. Springer Berlin / Heidelberg, 2012.

[20] P. Hö fner, R. van Glabbeek, W.L. Tan, M. Portmann, A. McIver, and A. Fehnker. A rigorous analysis of aodv and its variants. In *Proc. of the 15th International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM '12)*. acm, 2012.

[21] L. Gallina, S. Hamadou, A. Marin, and S. Rossi. A Framework for Throughput and Energy Efficiency in Mobile Ad hoc Networks. In *Proc. of IFIP Wireless Days 2011 (WD '11)*. IEEE Press, 2011.

[22] L. Gallina, S. Hamadou, A. Marin, and S. Rossi. A Probabilistic Energy-aware Model for Mobile Ad-hoc Networks. In *Proc. of the 18th International Conference on Analytical and Stochastic Modelling Techniques and Applications (ASMTA '11)*, volume 6751 of *LNCS*, pages 316–330. Springer-Verlag, 2011.

[23] L. Gallina, S. Hamadou, A. Marin, and S. Rossi. Connectivity and Energy Aware Preorders for Mobile Ad hoc Networks. Technical Report DAIS-2012-1, University Ca' Foscari Venice, 2012.

[24] L. Gallina, G. Dei Rossi, A. Marin, and S. Rossi. Evaluating Resistance to Jamming and Casual Interception in Mobile Wireless Networks. In *Proc. of the 15th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM '12)*. acm, 2012.

[25] L. Gallina and S. Rossi. A Formal Model for the Analysis of Mobile Ad-hoc Networks. Research Report CS-2009-9, Department of Computer Science, University Ca' Foscari of Venice, 2009.

[26] L. Gallina and S. Rossi. A Calculus for Power-aware Multicast Communications in Ad hoc Networks. In *Proc. of the 6th IFIP International Conference on Theoretical Computer Science (TCS '10)*, pages 20–31. Springer, 2010.

[27] L. Gallina and S. Rossi. A Process Calculus for Energy-aware Multicast Communications of Mobile Ad hoc Networks. *Wireless Communications and Mobile Computing*, 2012.

[28] M. Gerla, X. Hong, and C.C. Chiang. A Wireless Hierarchical Routing Protocol with Group Mobility. In *Proc. Wireless Communications and Networking Conference (WCNC '99)*, volume 3, pages 1538 – 1542. ieee, 1999.

[29] S. Gilmore and J. Hillston. The PEPA Workbench: A Tool to Support a Process Algebra-based Approach to Performance Modelling. In *Computer Performance Evaluation Modelling Techniques and Tools*, volume 794 of *lncs*, pages 353–368. Springer Berlin / Heidelberg, 1994.

[30] J.C. Godskesen. A calculus for mobile ad hoc networks. In *Coordination Models and Languages*, volume 4467 of *Lecture Notes in Computer Science*, pages 132–150. Springer Berlin Heidelberg, 2007.

[31] J. Goubault-Larrecq, C. Palamidessi, and A. Troina. A Probabilistic Applied Pi-Calculus. In *Proc. of the 5th Asian Symposium on Programming Languages and Systems (APLAS '07)*, volume 4807/2009 of *LNCS*, pages 175–190. Springer-Verlag, 2007.

[32] Z.J. Haas, M.R. Pearlman, and P. Samar. The Zone Routing Protocol (ZRP) for Ad hoc Networks. IETF Internet Draft, 2002.

[33] W. Rabiner Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient Communication Protocol for Wireless Microsensor Networks. In *Proc. of the 33rd International Conference on System Sciences (HICSS '00)*, 2000.

[34] M. Hennessy. *A Distributed Pi-Calculus*. Cambridge University Press, 2007.

[35] M. Hennessy and T. Regan. A Process Algebra for Timed Systems. *Information and Computation*, 117(2):221 – 239, 1995.

[36] Matthew Hennessy, Massimo Merro, and Julian Rathke. Towards a behavioural theory of access and mobility control in distributed systems. In *Foundations of Software Science and Computation Structures*, volume 2620 of *Lecture Notes in Computer Science*, pages 282–298. Springer Berlin Heidelberg, 2003.

[37] O. Herescu and C. Palamidessi. Probabilistic Asynchronous Pi-Calculus. In *Foundations of Software Science and Computation Structures*, volume 1784 of *Lecture Notes in Computer Science*, pages 146–160. Springer Berlin / Heidelberg, 2000.

[38] H. Hermanns, V. Mertsiotakis, and M. Rettelbach. A Construction and Analysis Tool based on the Stochastic Process Algebra tipp. In *Tools and Algorithms for the Construction and Analysis of Systems*, volume 1055 of *lncs*, pages 427–430. Springer Berlin / Heidelberg, 1996.

[39] G.R. Hiertz, D. Denteneer, S. Max, R. Taori, J. Cardona, L. Berlemann, and B. Walke. IEEE 802.11s: The WLAN Mesh Standard. *Wireless Communications, IEEE*, 17(1):104–111, 2010.

[40] J. Hillston. *A Compositional Approach to Performance Modelling*. Distinguished Dissertations in Computer Science. Cambridge University Press, 2005.

[41] C. A. R. Hoare. Communicating Sequential Processes. *Commun. ACM*, 21(8):666–677, 1978.

[42] M. Huth and M. Ryan. *Logic in Computer Science*. Cambridge University Press, 2nd edition, 2004.

[43] D. B. Johnson and D.A. Maltz. Dynamic Source Routing in Ad hoc Wireless Networks. In *Mobile Computing*, volume 353 of *The Kluwer International Series in Engineering and Computer Science*, pages 153–181. Springer US, 1996.

[44] C. Jouand and S. Smolka. Equivalences, Congruences, and Complete Axiomatizations for Probabilistic Processes. In *CONCUR '90 Theories of Concurrency: Unification and Extension*, volume 458 of *Lecture Notes in Computer Science*, pages 367–383. Springer Berlin / Heidelberg, 1990.

[45] E.D. Kaplan. *Understanding GPS: Principles and Applications*. Artech House Publishing, 1996.

[46] J.-P. Katoen, E. M. Hahn, H. Hermanns, D. Jansen, and I. Zapreev. The Ins and Outs of the Probabilistic Model Checker MRMC. In *Proc. 6th International Conference on Quantitative Evaluation of Systems (QEST'09)*, pages 167–176. IEEE CS Press, 2009.

[47] Y.B. Ko and N. Vaidya. Locationaided Routing (LAR) in Mobile Ad hoc Networks. *Wireless Networks*, 6:307–321, 2000.

[48] M. Kwiatkowska, G. Norman, and D. Parker. Analysis of a Gossip Protocol in PRISM. *ACM SIGMETRICS Performance Evaluation Review*, 36(3):17–22, 2008.

[49] M. Z. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of Probabilistic Real-time Systems. In *CAV*, pages 585–591, 2011.

[50] T. Han L. Gallina, M. Kwiatkowska, A. Marin, S. Rossi, and A. Spanò. Automatic Energy-aware Performance Analysis of Mobile Ad-hoc Networks. In *Proc. of IFIP Wireless Days Conference (WD '12)*. IEEE Press, 2012.

[51] I. Lanese and D. Sangiorgi. An Operational Semantics for a Calculus for Wireless Systems. *Theoretical Computer Science*, 411(19):1928–1948, 2010.

[52] R. Lanotte and M. Merro. Semantic Analysis of Gossip Protocols for Wireless Sensor Networks. In *CONCUR 2011 Concurrency Theory*, volume 6901 of *lncs*, pages 156–170. Springer Berlin / Heidelberg, 2011.

[53] L.B. Le, E. Hossain, and M. Zorzi. Queueing Analysis for GBN and SR ARQ Protocols under Dynamic Radio Link Adaptation with Non-zero Feedback Delay. *IEEE Transactions on Wireless Communications*, 6(9):3418–3428, 2007.

[54] M.E. Lesk. Lex – A Lexical Analyzer Generator. Computer Science Technical Report 39, Murray Hill, New Jersey: Bell Laboratories, 1975.

[55] J. Li, J. Jannotti, D.S.J. De Couto, D.R. Karger, and R. Morris. A Scalable Location Service for Geographic Ad hoc Routing. In *Proceedings of the 6th annual international conference on Mobile computing and networking (MobiCom'00), Boston, Massachussets, United States*, pages 120–130. ACM, 2000.

[56] D. Macedonio and M. Merro. A Semantic Analysis of Wireless Network Security Protocols. In *NASA Formal Methods*, volume 7226 of *lncs*, pages 403–417. Springer Berlin / Heidelberg, 2012.

[57] T. Venu Madhav and N.V.S.N. Sarma. Maximizing Network Lifetime through Varying Transmission Radii with Energy Efficient Cluster Routing Algorithm in Wireless Sensor Networks. *International Journal of Information and Electronics Engineering*, 2(2):205–209, 2012.

[58] M. Merro. An Observational Theory for Mobile Ad Hoc Networks. *Information and Computation*, 207(2):194–208, 2009.

[59] R. Milner. Calculi for Synchrony and Asynchrony. *Theoretical Computer Science*, 25(3):267–310, 1983.

[60] R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.

[61] R. Milner. *Communicating and Mobile Systems: the Pi-Calculus*. Cambridge University Press, 1999.

[62] R. Milner and D. Sangiorgi. Barbed Bisimulation. In *Proc. of International Colloquium on Automata, Languages and Programming (ICALP '92)*, volume 623 of *LNCS*, pages 685–695. Springer-Verlag, 1992.

[63] F. Moller and C. Tofts. A Temporal Calculus of Communicating Systems. In *CONCUR '90 Theories of Concurrency: Unification and Extension*, volume 458 of *Lecture Notes in Computer Science*, pages 401–415. Springer Berlin / Heidelberg, 1990.

[64] A. Muqattash and M. Krunz. CDMA-based MAC Protocol for Wireless ad Hoc Networks. In *Proc. of the 4th ACM International Symposium on Mobile ad hoc Networking & Computing (MobiHoc '03)*, pages 153–164. ACM, 2003.

[65] S. Murthy and J. Garcia-Luna-Aceves. An Efficient Routing Protocol for Wireless Networks. *Mobile Networks and Applications*, 1:183–197, 1996.

[66] S. Nanz and C. Hankin. A Framework for Security Analysis of Mobile Wireless Networks. *Theoretical Computer Science*, 367(1):203 – 227, 2006.

[67] R. De Nicola, G. Ferrari, R. Pugliese, and B. Venneri. Types for Access Control. *Theoretical Computer Science*, 240(1):215 – 254, 2000.

[68] R. De Nicola, G.L. Ferrari, and R. Pugliese. Klaim: a Kernel Language for Agents Interaction and Mobility. *IEEE Transactions on Software Engineering*, 24(5):315–330, 1998.

[69] X. Nicollin and J. Sifakis. An Overview and Synthesis on Timed Process Algebras. In *Computer Aided Verification*, volume 575 of *Lecture Notes in Computer Science*, pages 376–398. Springer Berlin / Heidelberg, 1992.

[70] G. Norman, C. Palamidessi, D. Parker, and P. Wu. Model Checking Probabilistic and Stochastic Extensions of the π-Calculus. *IEEE Transactions on Software Engineering*, 35(2):209–223, 2009.

[71] G. Norman, D. Parker, M. Kwiatkowska, S. Shukla, and R. Gupta. Using Probabilistic Model Checking for Dynamic Power Management. In M. Leuschel, S. Gruner, and S. Lo Presti, editors, *Proc. 3rd Workshop on Automated Verification of Critical Systems (AVoCS'03)*, Technical Report DSSE-TR-2003-2, University of Southampton, pages 202–215, April 2003.

[72] G. Norman and V. Shmatikov. Analysis of Probabilistic Contract Signing. *Journal of Computer Security*, 14(6):561–589, 2006.

[73] V.D. Park and M.S. Corson. A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks. In *Proc. of the 16th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '97)*, volume 3, pages 1405–1413. ieee, 1997.

[74] C.E. Perkins. *Ad Hoc Networking*. Addison-Wesley, 2001.

[75] C.E. Perkins and P. Bhagwat. Highly dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In *Proc. of the Conference on Communications architectures, protocols and applications*, SIGCOMM '94, pages 234–244, New York, NY, USA, 1994. ACM.

[76] K. V. S. Prasad. A Calculus of Broadcasting Systems. *Science of Computer Programming*, 25(2-3):285–327, 1995.

[77] C. Priami. Application of a Stochastic Name-passing Calculus to Representation and Simulation of Molecular Processes. *Information Processing Letters*, 80(1):25 – 31, 2001.

[78] V. Narasimha Raghavan and Suvitha Kesavan T. Peer Meera Labbai, N. Bhalaji. Extended Dynamic Source Routing Protocol for the Non Co-operating Nodes in Mobile Ad-hoc Networks. *International Journal of Applied Mathematics and Computer Sciences*, 3(1):12–17, 2007.

[79] E. M. Royer and C. E. Perkins. Multicast Operation of the Ad-hoc On-demand Distance Vector Routing Protocol. In *Proc. of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 207–218. ACM, 1999.

[80] Davide Sangiorgi. *Introduction to Bisimulation and Coinduction*. Cambridge University Press, 2011.

[81] P. Santi. Topology Control in Wireless Ad hoc and Sensor Networks. *ACM Computing Surveys (CSUR)*, 37(2):164–194, 2005.

[82] R. Segala. A Compositional Trace-based Semantics for Probabilistic Automata. In *CONCUR '95: Concurrency Theory*, volume 962 of *Lecture Notes in Computer Science*, pages 234–248. Springer Berlin / Heidelberg, 1995.

[83] R. Segala and N.A. Lynch. Probabilistic Simulations for Probabilistic Processes. In *Proc. of the 5th International Conference on Concurrency Theory (CONCUR '94)*, volume 836 of *LNCS*, pages 481–496. Springer-Verlag, 1994.

[84] A. Singh, C.R. Ramakrishnan, and S.A. Smolka. A Process Calculus for Mobile Ad hoc Networks. In *Proc. of the 10th International Conference on Coordination Models and Languages (COORDINATION '08)*, volume 5052 of *LNCS*, pages 296–314. Springer-Verlag, 2008.

[85] L. Song and J. Godskesen. Probabilistic mobility models for mobile and wireless networks. In *Proc. the 6th IFIP TC 1/WG 202 international conference on Theoretical Computer Science (TCS'10)*, volume 323 of *IFIP Advances in Information and Communication Technology*, pages 86–100. Springer Boston, 2010.

[86] L. Song and J. Godskesen. Broadcast abstraction in a stochastic calculus for mobile networks. In *Proc. the 7th IFIP TC 1/WG 202 international conference on Theoretical Computer Science (TCS'12)*, volume 7604 of *lncs*, pages 342–356. Springer-Verlag, 2012.

[87] A. S. Tanenbaum. *Computer Networks*. Prentice-Hall, 2003.

[88] F. Tobagi and L. Kleinrock. Packet Switching in Radio Channels: Part II–The Hidden Terminal Problem in Carrier Sense Multiple-Access and the Busy-Tone Solution. *IEEE Transactions on Communications*, 23(12):1417 – 1433, 1975.

[89] M. Tribastone. The PEPA Plug-in Project. In *Proc. 4th International Conference on Quantitative Evaluation of Systems (QEST'07)*, pages 53–54. IEEE Computer Society, 2007.

[90] T. van Dam and K. Langendoen. An Adaptive Energy-efficient MAC Protocol for Wireless Sensor Networks. In *Proc. of the 1st International Conference on Embedded Networked Sensor Systems*, SenSys '03, pages 171–180. ACM, 2003.

[91] R. Wattenhofer, L. Li, P. Bahl, and Y.M. Wang. Distributed Topology Control for Power Efficient Operation in Multihop Wireless Ad hoc Networks. In *Proc. 20th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '01).*, volume 3, pages 1388– 1397. ieee, 2001.

[92] P. Yang, C.R. Ramakrishnan, and S.A. Smolka. A Logical Encoding of the π-calculus: Model Checking Mobile Processes using Tabled Resolution. *International Journal on Software Tools for Technology Transfer (STTT)*, 6:38–66, 2004.

[93] Wei Ye, J. Heidemann, and D. Estrin. An Energy-efficient MAC Protocol for Wireless Sensor Networks. In *Proc. of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002).*, volume 3, pages 1567–1576. ieee, 2002.

[94] B. Zhang and H.T. Mouftah. Energy-aware On-demand Routing Protocols for Wireless Ad hoc Networks. *Wireless Networks*, 12(4):481–494, 2006.

[95] L. Zhou and Z.J. Haas. Securing Ad Hoc Networks. *IEEE Network Magazine*, 3(6):24–30, 1999.

[96] M. Zorzi and R.R. Rao. Error Control and Energy Consumption in Communications for Nomadic Computing. *IEEE Transactions on Computers*, 46(3):279–289, 1997.