Modelling IP Mobility *

Roberto M. Amadio †
Université de Provence, Marseille

Sanjiva Prasad ‡
Indian Institute of Technology, Delhi

October 10, 1997

Abstract

We study the modelling of *mobile hosts* on a network in a simple name-passing process calculus, with the intention of being able to prove properties about a protocol for supporting mobility. Our model may be considered a highly simplified version of proposals for *mobility support* in the version 6 of the Internet Protocols (IP). Being fairly general, the model may also apply to mobile software architectures. We believe that such simplified models help in prototyping mobility protocols and reasoning about them while abstracting away excessive details.

We concentrate on the issue of ensuring that messages to and from mobile agents are delivered without loss of connectivity. We provide three models of increasingly complex nature of a network of routers and computing agents that are interconnected via the routers: the first is without mobile agents and is treated as a specification for the next two; the second supports mobile agents, and the third additionally allows correspondent agents to *cache* the current location of a mobile agent. Following a detailed analysis of the three models to extract invariant properties, we show that the three models are related by a suitable notion of equivalence based on *barbed bisimulation.*

Keywords: Internet Protocols, Mobility, Protocol Analysis, Formal Methods, Modelling, Verification, Process Calculi, Barbed Bisimulation.

1 Introduction

We study the modelling of *mobile hosts* on a network in a simple name-passing process calculus, with the intention of being able to prove properties about a protocol for supporting mobility.

*This work was supported under the aegis of IFCPAR 1502-1: "Specification and Verification of Distributed Systems: The Finite-state and Higher-order Cases" and partially conducted at CMI, Université de Provence, May-June 1997.

†Corresponding author. CMI (LIM), 39 rue Joliot-Curie, F-13453, Marseille, France. amadio@gyptis.univ-mrs.fr. Partially supported by CTI-CNET 95-1B-182, HCM Express, WG Confer.

‡IIT Delhi, New Delhi 110016, India. sanjiva@cse.iitd.ernet.in. Partially supported by AICTE 1-52/CD/CA(08)/96-97.

Our model may be considered a highly simplified version of proposals for *mobility support* in the version 6 of Internet Protocols (IP) [IDM91, TUSM94, JP96]. Being fairly general, the model may also apply to mobile software architectures. We believe that such simplified models help in prototyping a protocol and reasoning about it while abstracting away excessive details. We expect that such modelling together with reasoning about the model may assist in the formulation and revision of real-world protocols for mobile systems.

The particular issue we explore is that of supporting mobility while ensuring that *messages to and from mobile agents are delivered without loss of connectivity* during and after an agent's move. From the informal description of mobility protocols in the literature, it is difficult to assure oneself of their correctness, without undertaking a detailed analysis. As borne out by our work, the specification and combinatorial analysis of such protocols is too complicated to rely on an informal justification.

Our work may be classified as *protocol analysis*. We take an informal description of a protocol, abstract away aspects that seem irrelevant or which are details for providing a particular functionality, then make a model of the simplified protocol and apply mathematical techniques to discover the system structure and its behavioural properties [1].

The structure of the paper reflects this point of view. After introducing our language for modelling the protocol (§2), we present a model of the protocol (§3), and then look for its essential structure in the form of a big invariant (§4). From this analysis, at the end, we gather some insight on why the protocol works correctly, and suggest some variations. We provide three models of increasingly complex nature of a network of routers and computing agents that are interconnected via the routers. In the first (§3.1), which we call *Stat*, computing agents are not mobile. We then extend the system *Stat* to permit agents to move from a router and attach themselves at another router (§3.2). We call such a system *Mob* (in our terminology the current router of an agent is the same thing as the current location or site of an agent). Finally, to possibly reduce indirection and to avoid excessive centralisation and traffic congestion, we extend the system *Mob* to a system (call it *CMob*), where the current router of an agent may be *cached* by its correspondent agents (§3.3). In §4, we analyse the three different models that we have given, and establish the correspondence between them by showing that the three systems *Stat*, *Mob* and *CMob* are *barbed bisimilar*, with respect to a suitable notion of observation. We conclude in §5, by summarising our contributions, recalling the simplifications we have made, and mentioning future directions of work.

How to read the paper A sequential reading of the paper may be a bit overwhelming, for this reason we suggest some possible shortcuts. By reading §3.1, 3.2 and glancing at §4.1, 4.2 the reader will have a general idea of the basic systems

[1]One may liken this approach to modern pharmaceutical analysis where a natural herb is taken, its essence extracted, and the non-active ingredients filtered away; then physical and mathematical models are used to discover the chemical properties of the active ingredients, and possibly synthesize cheaper substitutes.

Stat and *Mob* (in particular figure 5 should provide a good operational intuition) and of the techniques we apply in their analysis. The more challenging system *CMob*, whose size is about twice the size of *Mob*, is described in §3.3, 4.3. The footnotes comment on the relationship between our model and the IP informal specification. They are addressed mainly to the reader familiar with IP, and they can be skipped at a first reading. Finally, the reader motivated by the formal analysis, will have to take a closer look at the invariants described in figures 11, 13. Understanding the invariants is the demanding part, the related proofs (which are available in appendix A) are basically large case analyses which require little mathematical sophistication.

2 The process calculus

We describe the system in a formal notation, that of a polyadic calculus of communicating processes with asynchronous message passing over channels. That is, tuples may be communicated over a channel in a single action. When compared with, say, the asynchronous π-calculus [HT91] we notice the following differences:

- There is *no* dynamic generation of names (we do not need it).
- There is a form of *internal choice* over an arbitrary (possibly infinite) domain and it is possible to define *processes parametric over functions.* These two extensions allow a more natural description of the protocols: we use infinite internal choice to abstract from the details of the control, and function parameters to model the cache memory.

We assume a collection of basic sorts, and allow functions between basic sorts. We let $x, y, \ldots$ range over channel names, values of basic sorts, and functions from basic sorts to basic sorts. $\vec{x}$ stands for a tuple $x_1, \ldots, x_n$. The expressions $t, t' \ldots$ range over tests for equality between names, $X, X' \ldots$ are process identifiers, and $V, V' \ldots$ stand for value domains. Processes are typically denoted by $p, p' \ldots$ and are specified by the following grammar:

$$p \quad ::= \mathbf{0} \mid \overline{x}\vec{x} \mid x(\vec{x}).p \mid p \mid p \mid [t]p, p \mid X(\vec{x}) \mid \oplus_{x\in V,\ldots,x\in V}\, p$$

The operators have the following interpretation: $\mathbf{0}$ is the terminated process, $x(\vec{x}).p$ is the input prefix, $\overline{x}\vec{x}$ is a message, $\mid$ is the (asynchronous) parallel composition operator, and $[t]p, p'$ is a case statement. $X(\vec{x})$ is a process identifier applied to its actual parameters; as usual for every process identifier X, there is a unique defining equation $X(\vec{x}) = p$ such that all variables free in p are contained in $\{\vec{x}\}$. $\oplus$ is the internal choice operation, where in p we substitute a non-deterministically chosen tuple of values (from appropriate domains) for the specified tuple of variables.

We define a structural equivalence $\equiv$ on processes which is the least equivalence relation that includes: α-renaming of bound names, associativity and commutativity of $\mid$, the equation $p \mid \mathbf{0} \equiv p$, equation unfolding and:

$$[t]p_1, p_2 \equiv \begin{cases} p_1 & \text{if } t \text{ holds} \\ p_2 & \text{otherwise} \end{cases}$$

Reduction is up to structural equivalence and is defined by the following rules:

$$\frac{}{\overline{z}\vec{y} \mid z(\vec{x}).p \to [\vec{y}/\vec{x}]p} \qquad \frac{v_1 \in V_1, \ldots, v_n \in V_n}{\oplus_{x_1 \in V_1, \ldots, x_n \in V_n} p \to [v_1/x_1, \ldots, v_n/x_n]p} \qquad \frac{p \to p'}{p \mid q \to p' \mid q}$$

We also introduce the following abbreviations:

$$\begin{array}{l} \mathsf{if}\ t_1 \quad : p_1 \\ \vdots \\ \mathsf{if}\ t_n \quad : p_n \\ \mathsf{else} \quad : p_{n+1} \end{array} \equiv [t_1]p_1, ([t_2]p_2, (\ldots, [t_n]p_n, p_{n+1})\ldots)$$

$$\mathsf{let}\ x_1 = v_1, \ldots, x_n = v_n\ \mathsf{in}\ p \equiv [v_1/x_1]\cdots[v_n/x_n]p$$

$$p \oplus^c p' \equiv [c = 1]p, p' \quad (\text{where } c \in \{0, 1\})$$

3 The model

We describe the three systems *Stat*, *Mob*, and *CMob*. All three consist of a collection of computational *agents* that may communicate with one another over the network. Each agent is attached to a *router* that is its interface to the rest of the network. Of course, many agents may be attached to the same router. We assume that each entity, agent or router, has a globally unique identifying name (its address).

For simplicity, we assume a very elementary functionality for the agents — they can only communicate with other agents, sending or receiving messages via the routers to which they are attached. The agents cannot communicate directly between themselves, all communication being mediated by the routers. We assume that routers may directly communicate with one another, abstracting away the details of message delivery across the network. The communication mechanism we assume is an asynchronous one, involving unbounded buffers and allowing message overtaking.

3.1 The system without mobility *Stat*

We assume a collection of names $\mathcal{N}$ defined as the union of pairwise disjoint sets as follows:

$$\mathcal{N} = \mathcal{RN} \cup \mathcal{AN} \cup \mathcal{LAN} \cup \mathcal{DN} \cup \mathcal{CN}$$

where:	$r_i \in \mathcal{RN}$	Router Names	$a_i \in \mathcal{AN}$	Agent Names
	$l_i \in \mathcal{LAN}$	Local Agent Addresses	$d_i \in \mathcal{DN}$	Data Items
	$c \in \mathcal{CN}$	Control Directives		

The set $\mathcal{CN}$ of Control Directives has the following elements (the Control Directives that we consider in *Stat* consist of exactly *msg*, to indicate a data message; the directives *fwdd* and *upd* will be used only in *CMob*):

msg	message	*regd*	registered	*infmd*	informed	*fwdd*	forwarded
immig	immigrating	*repat*	repatriating	*mig*	migrating	*upd*	update

$$\text{Tables:} \begin{cases} \mathcal{L}: & \mathcal{RN} \times \mathcal{AN} \to \mathcal{LAN} \quad \text{(injective)} \\ \mathcal{H}: & \mathcal{AN} \to \mathcal{RN} \end{cases}$$

$$[x_1, x_2, x_3, x_4, x_5, x_6] \in \mathcal{CN} \times \mathcal{AN} \times \mathcal{RN} \times \mathcal{AN} \times \mathcal{RN} \times \mathcal{DN}$$

$$\mathsf{obs}([x_1, x_2, x_3, x_4, x_5, x_6]) = \begin{cases} \overline{o}[x_2, x_4, x_6] & \text{if } x_1 = msg \text{ or } x_1 = fwdd \\ \overline{o}\epsilon & \text{otherwise} \end{cases}$$

Figure 1: Tables and atomic observation

The sets $\mathcal{AN}$ and $\mathcal{DN}$ are assumed to be non-empty; we may assume any convenient base sort for these "pure" names. The elements of $\mathcal{RN}$ and $\mathcal{LAN}$ are intended as interface names, and in a process calculus, should be considered channel names that can carry values of the following domain (note that the sort corresponding to the sets $\mathcal{RN}$ and $\mathcal{LAN}$ is recursively defined):

$$\mathcal{CN} \times \mathcal{AN} \times \mathcal{RN} \times \mathcal{AN} \times \mathcal{RN} \times \mathcal{DN}$$

Elements of this domain may be interpreted as:

$$[control - directive, to - agent, at - router, from - agent, from - router, data]$$

We often write $\vec{x}$ to stand for the tuple $[x_1, x_2, x_3, x_4, x_5, x_6]$. An underscore _ indicates that the name is irrelevant ("don't care").

In figure 1, we present the tables used for address translation, $\mathcal{L}$ and $\mathcal{H}$, as well as the notion of *atomic observation* of the system ($\mathsf{obs}(\vec{x})$). If $\overline{z}[\vec{x}]$ is a message, we call the triple $[x_2, x_4, x_6]$ its *observable content*. $\mathcal{L}$ is an *injective* function which gives the local address for an agent at a given router, $\mathcal{H}$ computes the "home router" of an agent. We assume a distinguished channel name o on which we can observe either the reception of a message (as a triple specifying the original sender agent, the addressee, and the data transmitted), or anomalous behaviour represented by a special value ϵ.

In figure 2, we present (formally) the system *Stat*.

Agents An agent $A(a)$ either receives a message from its home router on its local address and observes it, or it generates a message to a correspondent agent, and sends a message to its home router for delivery to the correspondent agent via the latter's home router. $\mathcal{L}(\mathcal{H}(a), a)$ represents the local address of the agent a in its home subnet[2].

Router The router examines an incoming message, and if it is the destination router mentioned in the message, accordingly delivers it to the corresponding

[2]We have modelled some actions from the transport or higher layers corresponding to the processing of a received message, or the generation of a message to send to a correspondent agent. We also have modelled the link level communication between the router and the agent during message delivery.

$[x_1, x_2, x_3, x_4, x_5, x_6] \in \mathcal{CN} \times \mathcal{AN} \times \mathcal{RN} \times \mathcal{AN} \times \mathcal{RN} \times \mathcal{DN}$

$$
\begin{array}{lll}
A(a) & = & \bigoplus_{c \in \{in, out\}, y \in \mathcal{AN}, w \in \mathcal{DN}} \\
 & & \mathsf{if}\ c = in \quad : A_{in}(a) \\
 & & \mathsf{else} \quad\quad\ : \mathsf{let}\ \ z = \mathcal{H}(y),\ r_0 = \mathcal{H}(a) \\
 & & \quad\quad\quad\quad\quad\ \ \mathsf{in}\ \ \overline{r_0}[msg, y, z, a, r_0, w] \mid A(a) \\
 & & \\
A_{in}(a) & = & \mathsf{let}\ \ l = \mathcal{L}(\mathcal{H}(a), a) \\
 & & \ \mathsf{in}\ \ l(\vec{x}).(\mathsf{obs}(\vec{x}) \mid A(a)) \\
 & & \\
Router(r) & = & r(\vec{x}).\ \ \mathsf{if}\ x_3 = r \ \ :\ \mathsf{let}\ \ l = \mathcal{L}(r, x_2) \\
 & & \quad\quad\quad\quad\quad\quad\quad\quad\quad\ \ \mathsf{in}\ \ \bar{l}\vec{x} \mid Router(r) \\
 & & \quad\quad\quad\ \ \mathsf{else} \quad\quad\ \ : \overline{x_3}\vec{x} \mid Router(r) \\
 & & \\
Stat & \equiv & \Pi_{r \in \mathcal{RN}} Router(r) \mid \Pi_{a \in \mathcal{AN}} A(a)
\end{array}
$$

Figure 2: System without mobility

agent. Otherwise it sends it to the appropriate router. $\mathcal{L}(r, x_2)$ is the local address of x_2, the addressee of the message, whereas x_3 is the destination router.

3.2 The system with mobility *Mob*

We now allow agents to migrate from one router (*i.e.*, subnet) to another. We require that the identifying name of an agent be preserved during and after the migration. While moving from one router to another, the agents and routers engage in a *handover protocol* that allows messages to be sent and received by the agent once it has moved (thus ensuring higher layer connectivity is maintained). The handover protocol that we follow is inspired by the one given in [BTP95, JP96]. When an agent moves to another router, a proxy "home agent" at its home router forwards messages intended for the mobile agent to a "care-of address", which maps to the agent's current router. To avoid message loss, the forwarding home agent should have an up-to-date idea of the current router of the mobile agent. Hence when a mobile agent moves, it must inform the home agent of its new coordinates. In the first approximation, we model all messages addressed to a mobile agent being forwarded via the home agent; later we will consider correspondent agents caching the current router of a mobile agent[3]. The router description remains unchanged.

The handover protocol can be modelled in terms of processes. Although we

[3] We do not model various aspects of network protocols for mobility. In particular, we totally ignore security and authentication issues, as well as representational formats and conventions in network packets, *e.g.*, encapsulation/decapsulation of messages, and tunneling.

call these processes agents, we must note that these processes are roles played by a mobile node or a router on behalf of a mobile node. While in the actual Mobile IP protocol, a mobile node "discovers" its new milieu and engages in protocols such as "Neighbour Discovery" to connect itself to the new subnet, and disengage itself from the old subnet, we model agent migration in a more abstract fashion. We observe that the migration of a mobile agent from one router to another can be modelled "statically": *for each router, for each agent*, we have a process that represents the behaviour of a mobile agent either being present there or absent there, or that of a router enacting the role of a *forwarder* for the agent, routing messages addressed to that agent to its current router. Migration may now be described in terms of a *coordinated state change* by processes at each of the locations involved [4].

Although the model involves a matrix of shadow agents running at each router, it has the advantage of being *static*, in terms of processes and channels, requiring neither dynamic name generation nor dynamic process generation. The only aspect of the modelling that brings us outside the realm of *finite control* systems is the fact that channels have an infinite capacity, and there is no bound on the number of messages generated.

States of the agent at home We describe an agent at its home router in figure 3.

Ha The mobile agent is at its home base. It can receive and send messages, as in the definition of $A(a)$ in figure 2, and can also move to another router. When the agent "emigrates", say, to router z, it changes state to $Ham(a)$. We model the migration by the agent intimating its "shadow" at router z that it is "immigrating" there, and to prepare to commence operation.

Ham The mobile agent during emigration. We model the agent during "emigration" by the state $Ham(a)$. During the process of migration, messages addressed to the agent may continue to arrive; eventually, these messages should be received and handled by the mobile agent. The emigration occurs only when the shadow agent at the target site registers (by sending control message *regd*) its new care-of router (x_5) at the home base, thus indicating that it is ready to operate at that foreign subnet once it receives an acknowledgement from the home agent (control message *infmd*). The control messages (*regd* and *infmd*) are required to model the coordinated change of state at the two sites, and may be likened to the "binding update" and "binding acknowledgement" of [JP96]. The home agent *filters* messages while waiting for the *regd* message; this filtration is expressed by having other messages "reput" into the message buffer, and remaining in the current state, *i.e.*, $Ham(a)$.

Haf The home agent as a forwarder. The home agent forwards messages to the mobile agent at its current router (via the routers of course), unless informed

[4] Thus, our formalisation of the migration of an agent, involving the small coordinated state change protocol, may be considered an abstraction (rather than a faithful representation) of some of the actions performed when an agent attaches itself to a new router and disengages itself from an old one.

$$
\begin{array}{lll}
Ha(a) & = & \bigoplus_{c\in\{in,out,mv\},y\in\mathcal{AN},w\in\mathcal{DN},u\in\mathcal{RN}} \\
& & \mathsf{if}\ c = in \quad : Ha_{in}(a) \\
& & \mathsf{if}\ c = out \ \ : \mathsf{let}\ \ z = \mathcal{H}(y),\ \ r_0 = \mathcal{H}(a) \\
& & \qquad\qquad\qquad \mathsf{in}\ \ \overline{r_0}[msg, y, z, a, r_0, w] \mid Ha(a) \\
& & \mathsf{else} \qquad\quad : \mathsf{let}\ \ r_0 = \mathcal{H}(a) \\
& & \qquad\qquad\qquad \mathsf{in}\ \ \mathsf{if}\ u = r_0 \ \ : Ha(a) \\
& & \qquad\qquad\qquad\qquad \mathsf{else} \qquad\ : \overline{r_0}[immig, a, u, a, r_0, \_] \mid Ham(a)
\end{array}
$$

$$
\begin{array}{lll}
Ha_{in}(a) & = & \mathsf{let}\ \ l = \mathcal{L}(\mathcal{H}(a), a) \\
& & \mathsf{in}\ \ l(\vec{x}).(\mathsf{obs}(\vec{x}) \mid Ha(a))
\end{array}
$$

$$
\begin{array}{lll}
Ham(a) & = & \mathsf{let}\ \ r_0 = \mathcal{H}(a), l = \mathcal{L}(r_0, a) \\
& & \mathsf{in}\ \ l(\vec{x}).\ \ \mathsf{if}\ x_1 = regd \ \ : \overline{r_0}[infmd, a, x_5, a, r_0, \_] \mid Haf(a, x_5) \\
& & \qquad\qquad\ \mathsf{if}\ x_1 = msg \ \ : \overline{l}\vec{x} \mid Ham(a) \\
& & \qquad\qquad\ \mathsf{else} \qquad\quad\ : \overline{o}\epsilon \mid Ham(a)
\end{array}
$$

$$
\begin{array}{lll}
Haf(a, r) & = & \mathsf{let}\ \ r_0 = \mathcal{H}(a), l = \mathcal{L}(r_0, a) \\
& & \mathsf{in}\ \ l(\vec{x}).\ \ \mathsf{if}\ x_1 = repat \ \ : Ha(a) \\
& & \qquad\qquad\ \mathsf{if}\ x_1 = mig \quad : \overline{r_0}[infmd, a, x_5, a, r_0, \_] \mid Haf(a, x_5) \\
& & \qquad\qquad\ \mathsf{if}\ x_1 = msg \quad : \overline{r_0}[msg, a, r, x_4, x_5, x_6] \mid Haf(a, r) \\
& & \qquad\qquad\ \mathsf{else} \qquad\quad\ : \overline{o}\epsilon \mid Haf(a, r)
\end{array}
$$

Router(r) (as in figure 2)

Figure 3: States of the agent at home

by the mobile agent that it is moving from that router. There are two cases we consider: either the mobile agent is coming home ("repatriation") or it is migrating elsewhere.

States of the agent away from home We describe the agents at a foreign router in figure 4.

Idle If the agent has never visited. The *Idle* state captures the behaviour of the shadow of an agent at a router it has never visited. If the agent moves to that router, indicated by the control message *immig*, then the shadow agent changes state to $Bma(a, r)$, from where it will take on the behaviour of mobile agent a at the foreign router r. Any other message is ignored, and indeed it should be erroneous to receive any other message in this state.

Fwd If the agent is not at foreign router r, but has been there earlier. This state is similar to *Idle*, except that any delayed messages that had been routed to the agent while it was here previously are re-routed via the home router [5]. This state

[5] Since messages forwarded by the home agent may get arbitrarily delayed in transit, it is

$$
\begin{array}{l}
Idle(a,r) = \\
\ \ \mathsf{let} \ \ l = \mathcal{L}(r,a), r_0 = \mathcal{H}(a) \\
\ \ \ \mathsf{in} \ \ l(\vec{x}). \ \begin{array}[t]{ll}
\mathsf{if}\ x_1 = immig, x_5 \neq r_0 & : \overline{r}[mig, a, r_0, a, r, \_] \mid Bma(a,r) \\
\mathsf{if}\ x_1 = immig, x_5 = r_0 & : \overline{r}[regd, a, r_0, a, r, \_] \mid Bma(a,r) \\
\mathsf{else} & : \overline{o}\epsilon \mid Idle(a,r)
\end{array}
\end{array}
$$

$$
\begin{array}{l}
Fwd(a,r) = \\
\ \ \mathsf{let} \ \ l = \mathcal{L}(r,a), r_0 = \mathcal{H}(a) \\
\ \ \ \mathsf{in} \ \ l(\vec{x}). \ \begin{array}[t]{ll}
\mathsf{if}\ x_1 = immig, x_5 \neq r_0 & : \overline{r}[mig, a, r_0, a, r, \_] \mid Bma(a,r) \\
\mathsf{if}\ x_1 = immig, x_5 = r_0 & : \overline{r}[regd, a, r_0, a, r, \_] \mid Bma(a,r) \\
\mathsf{if}\ x_1 = msg & : \overline{r}[msg, a, r_0, x_4, x_5, x_6] \mid Fwd(a,r) \\
\mathsf{else} & : \overline{o}\epsilon \mid Fwd(a,r)
\end{array}
\end{array}
$$

$$
\begin{array}{l}
Bma(a,r) = \\
\ \ \mathsf{let} \ \ l = \mathcal{L}(r,a) \\
\ \ \ \mathsf{in} \ \ l(\vec{x}). \ \begin{array}[t]{ll}
\mathsf{if}\ x_1 = infmd & : Ma(a,r) \\
\mathsf{if}\ x_1 = msg & : \overline{l}\vec{x} \mid Bma(a,r) \\
\mathsf{else} & : \overline{o}\epsilon \mid Bma(a,r)
\end{array}
\end{array}
$$

$$
\begin{array}{l}
Ma(a,r) = \\
\bigoplus_{c \in \{in, out, mv\}, y \in \mathcal{AN}, w \in \mathcal{DN}, u \in \mathcal{RN}} \\
\ \ \begin{array}{ll}
\mathsf{if}\ c = in & : Ma_{in}(a,r) \\
\mathsf{if}\ c = out & : \ \mathsf{let} \ \ z = \mathcal{H}(y) \\
 & \ \ \ \ \mathsf{in} \ \ \overline{r}[msg, y, z, a, r, w] \mid Ma(a,r) \\
\mathsf{else} & : \ \mathsf{let} \ \ r_0 = \mathcal{H}(a) \\
 & \ \ \ \ \mathsf{in} \ \ \begin{array}[t]{ll}
 \mathsf{if}\ u = r & : Ma(a,r) \\
 \mathsf{if}\ u \neq r, u = r_0 & : \overline{r}[repat, a, r_0, a, r, \_] \mid Fwd(a,r) \\
 \mathsf{else} & : \overline{r}[immig, a, u, a, r, \_] \mid Fwd(a,r)
 \end{array}
\end{array}
\end{array}
$$

$$
\begin{array}{l}
Ma_{in}(a,r) = \\
\ \ \mathsf{let} \ \ l = \mathcal{L}(r,a) \\
\ \ \ \mathsf{in} \ \ l(\vec{x}).(\mathsf{obs}(\vec{x}) \mid Ma(a,r))
\end{array}
$$

$$
Mob \equiv \Pi_{r \in \mathcal{RN}} Router(r) \mid \Pi_{a \in \mathcal{AN}} Ha(a) \mid \Pi_{r \in \mathcal{RN}, a \in \mathcal{AN}, r \neq \mathcal{H}(a)} Idle(a,r)
$$

Figure 4: States of the agent away from home

may be compared to *Haf*, except that it does not have to concern itself with the agent migrating elsewhere.

Bma Becoming a foreign mobile agent. Once the protocol for establishing movement to this router is complete, the agent becomes a foreign mobile agent. Messages are filtered looking for an acknowledgement from the home agent that it is aware of the mobile agent's current router. Once the home agent has acknowledged its noting the new coordinates, the mobile agent may become operational.

Ma The mobile agent at a foreign router. As with the mobile agent at its home base $Ha(a)$, the mobile agent may receive messages, send messages, or move away. The behaviour of the mobile agent in state *Ma* is similar to that of *Ha* except that during movement, different control messages need to be sent to the target site depending on whether it is home or another site. If the target site is the home base, then a *repat* message is sent to home base (via the routers). Otherwise the target site is intimated of the wish to "immigrate". The agent goes into the state *Fwd*.

In the upper part of figure 5 we describe the possible transition which relate to control messages, not including filtering, forwarding, and erroneous situations. We decorate the transitions with the control messages that are received (-) and emitted (+). In the lower part of figure 5 we outline the *three basic movements* of an agent a: leaving the home router, coming back to the home router, and moving between routers different from the home router. We point-out the relevant states and control messages.

3.3 The system with caching *CMob*

The previous approach suffers from being overly centralised. All traffic to an agent is routed through its home router, thus creating inefficiencies as well as poor fault tolerance. Therefore it is desirable that we allow correspondent agents to cache the current router of a mobile agent. This is the approach suggested in [JP96]. We parameterise the definition over a function $f : \mathcal{AN} \rightarrow \mathcal{RN}$ which represents the current cache of the agent. This cache is used to approximate the current router of an agent. Note that this function can be simply implemented by associating an array to every agent [6].

We use the control directives *fwdd* and *upd*; the former indicates that the current data message has been *forwarded* thus pointing out a "cache miss" (in IP jargon one says that the message has been "tunnelled"), the latter *suggests an update* of a cache entry, following a cache miss. An agent may also decide to reset a cache entry to the home router [7]. Note that the protocol does *not* require the

important that the mobile node, in addition to informing its home agent of its current router, arrange for a forwarder at its prior router to handle such delayed messages. This point is not explicitly highlighted in the Mobile IP proposal, although it is known in the folklore regarding implementation of process migration.

[6] When moving to another router, we could deliver the current cache with the message *immig*. In the presented version we always re-start with the cache $\mathcal{H}$.

[7] In IP, the validity of a cache entry may expire. In the informal description of the protocol,

$$\begin{array}{llllll}
(1) & Ha(a) & \xrightarrow{+immig} & Ham(a) & \xrightarrow{-regd\ +infmd} & Haf(a,r) \\
(2.1) & Idle/Fwd(a,r) & \xrightarrow{-immig\ +mig} & Bma(a,r) & \xrightarrow{-infmd} & Ma(a,r) \\
(2.2) & Idle/Fwd(a,r) & \xrightarrow{-immig\ +regd} & Bma(a,r) & \xrightarrow{-infmd} & Ma(a,r) \\
(3.1) & Ma(a,r) & \xrightarrow{+immig} & Fwd(a,r) & & \\
(3.2) & Ma(a,r) & \xrightarrow{+repat} & Fwd(a,r) & & \\
(4.1) & Haf(a,r) & \xrightarrow{-mig\ +infmd} & Haf(a,r') & & \\
(4.2) & Haf(a,r) & \xrightarrow{-repat} & Ha(a) & &
\end{array}$$

I–Leaving home

$Ha(a)$	$Idle/Fwd(a,r)$	–
$Ham(a)$	$Idle/Fwd(a,r)$	$immig$
$Ham(a)$	$Bma(a,r)$	$regd$
$Haf(a,r)$	$Bma(a,r)$	$infmd$
$Haf(a,r)$	$Ma(a,r)$	–

II–Coming home

$Ma(a,r)$	$Haf(a,r)$	–
$Fwd(a,r)$	$Haf(a,r)$	$repat$
$Fwd(a,r)$	$Ha(a)$	–

III–Moving between routers different from the home router

$Ma(a,r)$	$Haf(a,r)$	$Idle/Fwd(a,r')$	–
$Fwd(a,r)$	$Haf(a,r)$	$Idle/Fwd(a,r')$	$immig$
$Fwd(a,r)$	$Haf(a,r)$	$Bma(a,r')$	mig
$Fwd(a,r)$	$Haf(a,r')$	$Bma(a,r')$	$infmd$
$Fwd(a,r)$	$Haf(a,r')$	$Ma(a,r')$	–

Figure 5: Control transitions

coherence of the caches. In case of cache miss, we may forward the message either to the home router (which maintains as in the previous protocol an up-to-date view of the current router) or to the router the agent moved to.

We note the introduction of two extra states: $Fwd_{in}(a,r,r')$, and $Mam(a,r)$. $Fwd_{in}(a,r,r')$ is needed to model the internal choice of a forwarder on whether to reset the current router r' to the home router $\mathcal{H}(a)$. $Mam(a,r)$ is an extra state that we need when an agent moves from a router different from the home router. Before becoming a forwarder to the router to which the agent moved, we have to make sure that the agent has arrived there, otherwise we may forward messages to an $Idle(a,r')$ process, thus producing a run-time error (this situation does not arise in system *Mob* because we always forward to the home router).

the update and reset of a cache entry are sometimes *optional* operations. We model this by using again internal choice. No messages to reset a cache entry (binding deletion updates) are ever sent out. Instead, in the presented version of the protocol, an agent may nondeterministically decide to reset a cache entry, thus abstracting from a particular cache management mechanism. Timing out of cache entries is done using nondeterminism, rather than by explicit representation of time stamps in a message (note that in the Mobile IP protocol no hypothesis is made regarding the coordination of the clocks of the agents, so it seems an overkill to introduce time to speak about these time stamps).

$$
\begin{array}{l}
Ha(a,f) = \\
\bigoplus_{c\in\{in,out,mv,rst\},c_1\in\{0,1\},c_2\in\{0,1\},y\in\mathcal{AN},w\in\mathcal{DN},u\in\mathcal{RN}} \\
\quad \begin{array}{lll}
\textsf{if } c = in & : Ha_{in}(a,f,c_1,c_2) & \\
\textsf{if } c = out & : \textsf{let} & r_0 = \mathcal{H}(a), z = f(y) \\
 & \ \textsf{in} & \overline{r_0}[msg,y,z,a,r_0,w] \mid Ha(a,f) \\
\textsf{if } c = mv & : \textsf{let} & r_0 = \mathcal{H}(a) \\
 & \ \textsf{in} & \begin{array}{ll} \textsf{if } u = r_0 & : Ha(a,f) \\ \textsf{else} & : \overline{r_0}[immig,a,u,a,r_0,\_] \mid Ham(a) \end{array} \\
\textsf{else} & : \textsf{let} & r' = \mathcal{H}(y) \\
 & \ \textsf{in} & Ha(a,f[r'/y])
\end{array}
\end{array}
$$

$$
\begin{array}{l}
Ha_{in}(a,f,c_1,c_2) = \\
\quad \textsf{let}\ \ r_0 = \mathcal{H}(a), l = \mathcal{L}(r_0,a) \\
\quad \textsf{in}\ \ l(\vec{x}).\ \begin{array}{ll}
\textsf{if } x_4 = a, x_1 \in \{msg, fwdd\} & : \textsf{obs}(\vec{x}) \mid Ha(a,f) \\
\textsf{if } x_4 \neq a, x_1 = msg & : \textsf{obs}(\vec{x}) \mid (Ha(a,f[x_5/x_4]) \oplus^{c_1} Ha(a,f)) \\
\textsf{if } x_4 \neq a, x_1 = fwdd & : \ \textsf{obs}(\vec{x}) \mid (\overline{r_0}[upd,x_4,x_5,a,r_0,\_] \oplus^{c_2} \mathbf{0}) \mid \\
 & \quad (Ha(a,f[x_5/x_4]) \oplus^{c_1} Ha(a,f)) \\
\textsf{if } x_4 \neq a, x_1 = upd & : (Ha(a,f[x_5/x_4]) \oplus^{c_1} Ha(a,f)) \\
\textsf{else} & : \overline{o}\epsilon \mid Ha(a,f)
\end{array}
\end{array}
$$

$$
\begin{array}{l}
Ham(a) = \\
\quad \textsf{let}\ \ r_0 = \mathcal{H}(a), l = \mathcal{L}(r_0,a) \\
\quad \textsf{in}\ \ l(\vec{x}).\ \begin{array}{ll}
\textsf{if } x_1 = regd & : \overline{r_0}[infmd,a,x_5,a,r_0,\_] \mid Haf(a,x_5) \\
\textsf{if } x_1 \in \{msg, fwdd, upd\} & : \overline{l}\vec{x} \mid Ham(a) \\
\textsf{else} & : \overline{o}\epsilon \mid Ham(a)
\end{array}
\end{array}
$$

$$
\begin{array}{l}
Haf(a,r) = \\
\quad \textsf{let}\ \ r_0 = \mathcal{H}(a), l = \mathcal{L}(r_0,a) \\
\quad \textsf{in}\ \ l(\vec{x}).\ \begin{array}{ll}
\textsf{if } x_1 = repat & : \overline{r_0}[regd,a,x_5,a,r_0,\_] \mid Ha(a,\mathcal{H}) \\
\textsf{if } x_1 = mig & : \overline{r_0}[infmd,a,x_5,a,r_0,\_] \mid Haf(a,x_5) \\
\textsf{if } x_1 \in \{msg, fwdd\} & : \overline{r_0}[fwdd,a,r,x_4,x_5,x_6] \mid Haf(a,r) \\
\textsf{if } x_1 = upd, x_4 \neq a & : \overline{r_0}[upd,a,r,x_4,x_5,\_] \mid Haf(a,r) \\
\textsf{else} & : \overline{o}\epsilon \mid Haf(a,r)
\end{array}
\end{array}
$$

$Router(r)$ (as in figure 2)

Figure 6: Modified control for agent at home with caching

$$
\begin{array}{l}
\mathit{Idle}(a,r) = \\
\quad \mathsf{let} \quad l = \mathcal{L}(r,a), r_0 = \mathcal{H}(a) \\
\quad \mathsf{in} \quad l(\vec{x}).\ \begin{array}{ll}
\mathsf{if}\ x_1 = \mathit{immig}, x_5 \neq r_0 & : \overline{r}[\mathit{regd}, a, x_5, a, r, \_] \mid \overline{r}[\mathit{mig}, a, r_0, a, r, \_] \mid \mathit{Bma}(a,r) \\
\mathsf{if}\ x_1 = \mathit{immig}, x_5 = r_0 & : \overline{r}[\mathit{regd}, a, r_0, a, r, \_] \mid \mathit{Bma}(a,r) \\
\mathsf{else} & : \overline{o}\epsilon \mid \mathit{Idle}(a,r)
\end{array}
\end{array}
$$

$$
\begin{array}{l}
\mathit{Fwd}(a,r,r') = \\
\bigoplus_{c\in\{0,1\}} \\
\quad \mathsf{let} \quad r_0 = \mathcal{H}(a) \\
\quad \mathsf{in} \quad \mathit{Fwd}(a,r,r_0) \oplus^c \mathit{Fwd}_{in}(a,r,r')
\end{array}
$$

$$
\begin{array}{l}
\mathit{Fwd}_{in}(a,r,r') = \\
\quad \mathsf{let} \quad l = \mathcal{L}(r,a), r_0 = \mathcal{H}(a) \\
\quad \mathsf{in} \quad l(\vec{x}).\ \begin{array}{ll}
\mathsf{if}\ x_1 = \mathit{immig}, x_5 \neq r_0 & : \overline{r}[\mathit{regd}, a, x_5, a, r, \_] \mid \overline{r}[\mathit{mig}, a, r_0, a, r, \_] \mid \mathit{Bma}(a,r) \\
\mathsf{if}\ x_1 = \mathit{immig}, x_5 = r_0 & : \overline{r}[\mathit{regd}, a, r_0, a, r, \_] \mid \mathit{Bma}(a,r) \\
\mathsf{if}\ x_1 = \mathit{upd}, x_4 \neq a & : \overline{r}[\mathit{upd}, a, r', x_4, x_5, \_] \mid \mathit{Fwd}(a,r,r') \\
\mathsf{if}\ x_1 \in \{\mathit{msg}, \mathit{fwdd}\} & : \overline{r}[\mathit{fwdd}, a, r', x_4, x_5, x_6] \mid \mathit{Fwd}(a,r,r') \\
\mathsf{else} & : \overline{o}\epsilon \mid \mathit{Fwd}(a,r,r')
\end{array}
\end{array}
$$

$$
\begin{array}{l}
\mathit{Bma}(a,r) = \\
\quad \mathsf{let} \quad l = \mathcal{L}(r,a) \\
\quad \mathsf{in} \quad l(\vec{x}).\ \begin{array}{ll}
\mathsf{if}\ x_1 = \mathit{infmd} & : \mathit{Ma}(a,r,\mathcal{H}) \\
\mathsf{if}\ x_1 \in \{\mathit{msg}, \mathit{fwdd}, \mathit{upd}\} & : \overline{l}\vec{x} \mid \mathit{Bma}(a,r) \\
\mathsf{else} & : \overline{o}\epsilon \mid \mathit{Bma}(a,r)
\end{array}
\end{array}
$$

Figure 7: Modified control for agent away from home with caching, part I

We present in figure 6, the new definitions of the agent at home. Note the use of the directives *fwdd* and *upd* to update the cache and to suggest cache updates. In figures 7,8 we present the modified definitions for the agent away from home.

4 The analysis

In this section, we analyse the three different systems *Stat*, *Mob* and *CMob*, and show that they are barbed bisimilar with respect to a suitable notion of observation.

4.1 Analysis of *Stat*

In figure 9 we introduce the notion of admissible configuration for *Stat*. We will write $s.Rt$, $s.Ob$, $s.Ag$, and $s.Ms$ to denote the state of the routers, atomic observations, agents, and data messages, respectively, in configuration s. We will abuse notation, and regard products of messages as multi-sets. This is justified by the

$$
\begin{array}{l}
Ma(a,r,f) = \\
\bigoplus_{c\in\{in,out,mv,rst\},c_1\in\{0,1\},c_2\in\{0,1\},y\in\mathcal{AN},w\in\mathcal{DN},u\in\mathcal{RN}} \\
\quad \begin{array}{lll}
\text{if } c = in & : Ma_{in}(a,r,f,c_1,c_2) & \\
\text{if } c = out & : \text{let} & z = f(y) \\
 & \quad\ \text{in} & \overline{r}[msg,y,z,a,r,w] \mid Ma(a,r,f) \\
\text{if } c = mv & : \text{let} & r_0 = \mathcal{H}(a) \\
 & \quad\ \text{in} & \begin{array}{ll} \text{if } u = r & : Ma(a,r,f) \\ \text{if } u = r_0 & : \overline{r}[repat,a,r_0,a,r,\_] \mid Mam(a,r) \\ \text{else} & : \overline{r}[immig,a,u,a,r,\_] \mid Mam(a,r) \end{array} \\
\text{else} & : \text{let} & r' = \mathcal{H}(y) \\
 & \quad\ \text{in} & \begin{array}{ll} \text{if } y \neq a & : Ma(a,r,f[r'/y]) \\ \text{else} & : Ma(a,r,f) \end{array}
\end{array}
\end{array}
$$

$$
\begin{array}{l}
Ma_{in}(a,r,f,c_1,c_2) = \\
\quad \text{let} \quad l = \mathcal{L}(r,a) \\
\quad \text{in} \quad l(\vec{x}).\ \begin{array}{ll}
\text{if } x_4 = a, x_1 \in \{msg, fwdd\} & : \mathsf{obs}(\vec{x}) \mid Ma(a,r,f) \\
\text{if } x_4 \neq a, x_1 = msg & : \mathsf{obs}(\vec{x}) \mid (Ma(a,r,f[x_5/x_4]) \oplus^{c_1} Ma(a,r,f)) \\
\text{if } x_4 \neq a, x_1 = fwdd & : \ \mathsf{obs}(\vec{x}) \mid (\overline{r}[upd,x_4,x_5,a,r,\_] \oplus^{c_2} \mathbf{0}) \mid \\
 & \quad (Ma(a,r,f[x_5/x_4]) \oplus^{c_1} Ma(a,r,f)) \\
\text{if } x_4 \neq a, x_1 = upd & : Ma(a,r,f[x_5/x_4]) \oplus^{c_1} Ma(a,r,f) \\
\text{else} & : \overline{o}\epsilon \mid Ma(a,r,f)
\end{array}
\end{array}
$$

$$
\begin{array}{l}
Mam(a,r) = \\
\quad \text{let} \quad l = \mathcal{L}(r,a) \\
\quad \text{in} \quad l(\vec{x}).\ \begin{array}{ll}
\text{if } x_1 = regd & : Fwd(a,r,x_5) \\
\text{if } x_1 \in \{fwdd, msg, upd, immig\} & : \overline{l}\vec{x} \mid Mam(a,r) \\
\text{else} & : \overline{o}\epsilon \mid Mam(a,r)
\end{array}
\end{array}
$$

$$
CMob \equiv \Pi_{r\in\mathcal{RN}} Router(r) \mid \Pi_{a\in\mathcal{AN}} Ha(a,\mathcal{H}) \mid \Pi_{r\in\mathcal{RN},a\in\mathcal{AN},r\neq\mathcal{H}(a)} Idle(a,r)
$$

Figure 8: Modified control for agent away from home with caching, part II

hypothesis that parallel composition is associative and commutative. When working on multi-sets we will use standard set-theoretic notation, however operations such as union and difference are intended as taking multiplicity of the occurrences into account.

We will assume that $\sharp\mathcal{RN} \geq 3$. In this way we avoid considering degenerate cases when establishing the correspondence between *Stat* and *Mob* (if $\sharp\mathcal{RN} \leq 2$, then the transitions III in figure 5 cannot arise).

Proposition 4.1 *The initial configuration Stat is admissible, and admissible configurations are closed under reduction.*

By the definition of admissible configuration, we can conclude that the error

$$
\begin{array}{lll}
s & \equiv Rt \mid Ob \mid Ag \mid Ms & \\
Rt & \equiv \Pi_{r \in \mathcal{RN}} Router(r) & \\
Ob & \equiv \Pi_{i \in I} \overline{o}[x_i, y_i, z_i] \quad x_i, y_i \in \mathcal{AN}, z_i \in \mathcal{DN}, & \\
Ag & \equiv \Pi_{a \in \mathcal{AN}} B(a) \quad B(a) ::= A(a) \mid A_{in}(a), & \\
Ms & \equiv \Pi_{j \in J} \overline{z_j}[msg, x_{2,j}, x_{3,j}, x_{4,j}, x_{5,j}, x_{6,j}] & \\
& \quad x_{2,j}, x_{4,j} \in \mathcal{AN}, x_{3,j}, x_{5,j} \in \mathcal{RN}, x_{6,j} \in \mathcal{DN}, & \\
& \quad x_{3,j} = \mathcal{H}(x_{2,j}), x_{5,j} = \mathcal{H}(x_{4,j}), z_j \in \{x_{5,j}, x_{3,j}, \mathcal{L}(x_{3,j}, x_{2,j})\} &
\end{array}
$$

Figure 9: Admissible configurations for *Stat*

message $\overline{o}\epsilon$ is never generated (a similar remark can be made for the systems *Mob* and *CMob*, applying theorems 4.5 and 4.13, respectively).

Corollary 4.2 (message integrity) *Messages do not get lost or tampered with. Let s be an admissible configuration for Stat, let $\overline{z}\vec{x} \in s.Ms$ and suppose $s \to s'$. Then either $\overline{z'}\vec{x} \in s'.Ms$, for some z'; or else $\overline{o}[x_2, x_4, x_6] \in s'.Ob$ when the message gets received by its intended addressee.*

Corollary 4.3 (message delivery) *Let s be an admissible configuration from Stat such that $\overline{z}\vec{x} \in s.Ms$. Then the data message can be observed in at most 4 reductions.*

4.2 Analysis of *Mob*

The table in figure 10 lists the situations that can arise during the migration of an agent from a router to another. k is the case number, $Ag(a, k, r, r')$ denotes the shadow agents of a not in a *Fwd* or *Idle* state in situation k, $CMs(a, k, r, r', z)$ the migration protocol control messages at z involving at most the sites $\mathcal{H}(a), r, r'$ for that situation, and $R(a, k, r, r')$ denotes the routers involved in situation k of the protocol at which a's shadow is not in a *Fwd* or *Idle* state.

Relying on this table, we define in figure 11 a notion of *admissible function* γ. Intuitively, the function γ associates to every agent a its current migration control (state and protocol messages), the routers already visited, and the data messages in transit which are addressed to a.

We denote with $\mathcal{P}_{fin}(X)$ and $\mathcal{M}_{fin}(X)$ the finite parts, and finite multi-sets of X, respectively, and with $\gamma(a)_i$ the i-th projection of the tuple $\gamma(a)$. Then $Act(a, \gamma)$ denotes the routers where a has visited, which are not in an *Idle* state. Condition (C_1) states that at most finitely many agents can be on the move ("deranged") at any instant. (C_2) is a hygiene condition on migration control messages, indicating that they may be at exactly one of three positions, and that if an agent is on the move (cases 7 and 9), the home forwarder always points to an active router, where a proxy agent will return delayed data messages messages back to

k	$Ag(a,k,r,r')$	$CMs(a,k,r,r',z)$	$R(a,k,r,r')$
1	$Ha(a)$	$\mathbf{0}$	$\{r_0\}$
2	$Ha_{in}(a)$	$\mathbf{0}$	$\{r_0\}$
3	$Ham(a)$	$\overline{z}[immig, a, r, a, r_0, \_]$	$\{r_0\}$
4	$Ham(a) \mid Bma(a,r)$	$\overline{z}[regd, a, r_0, a, r, \_]$	$\{r_0, r\}$
5	$Haf(a,r) \mid Bma(a,r)$	$\overline{z}[infmd, a, r, a, r_0, \_]$	$\{r_0, r\}$
6	$Haf(a,r) \mid Ma(a,r)$	$\mathbf{0}$	$\{r_0, r\}$
7	$Haf(a,r)$	$\overline{z}[repat, a, r_0, a, r, \_]$	$\{r_0\}$
8	$Haf(a,r) \mid Ma_{in}(a,r)$	$\mathbf{0}$	$\{r_0, r\}$
9	$Haf(a,r)$	$\overline{z}[immig, a, r', a, r, \_]$	$\{r_0\}$
10	$Haf(a,r) \mid Bma(a,r')$	$\overline{z}[mig, a, r_0, a, r', \_]$	$\{r_0, r'\}$

.

Figure 10: Control migration ($r_0 = \mathcal{H}(a)$)

$$K = \{1, \ldots, 10\} \qquad K_s = \{1, 2, 6, 8\} \quad \text{(stable states)}$$

$$\begin{array}{lcl}
\gamma : \mathcal{AN} \to & K \times \mathcal{RN} \times \mathcal{RN} \times (\mathcal{RN} \cup \mathcal{LAN}) \times & \text{(control migration)} \\
& \mathcal{P}_{fin}(\mathcal{RN}) \times & (\mathit{Fwd's}) \\
& \mathcal{M}_{fin}((\mathcal{RN} \cup \mathcal{LAN}) \times \mathcal{RN} \times \mathcal{AN} \times \mathcal{RN} \times \mathcal{DN}) & \text{(data messages)}
\end{array}$$

$$Act(a, \gamma) \qquad = \gamma(a)_5 \cup R(a, \gamma(a)_1, \gamma(a)_2, \gamma(a)_3)$$

Admissibility conditions on γ:

(C_1) $$\{a \in \mathcal{AN} \mid \gamma(a)_1 \notin K_s\} \text{ finite}$$

(C_2) $$\forall a \in \mathcal{AN} (k = \gamma(a)_1,\ r = \gamma(a)_2,\ r' = \gamma(a)_3,\ z = \gamma(a)_4, \text{ and } F = \gamma(a)_5) \Rightarrow$$

$$\begin{array}{c}
(\sharp\{\mathcal{H}(a), r, r'\} = 3, F \cap R(a,k,r,r') = \emptyset, \\
(k \in \{7, 9\} \Rightarrow r \in Act(a, \gamma)), \text{ and} \\
(CMs(a,k,r,r',z) \equiv \overline{z}[cdir, a, r_1, a, r_2, _] \Rightarrow z \in \{r_1, r_2, \mathcal{L}(r_1, a)\}))
\end{array}$$

(C_3) $$\begin{array}{c}
\forall\, a \in \mathcal{AN}\ \ \forall\, (z, r_1, a_2, r_2, d) \in \gamma(a)_6 \\
(r_1 \in Act(a, \gamma), r_2 \in Act(a_2, \gamma), \text{ and} \\
z \in Act(a, \gamma) \cup \{r_2\} \cup \{\mathcal{L}(r'', a) \mid r'' \in Act(a, \gamma)\})
\end{array}$$

Figure 11: Admissible configurations

a's home; after receiving the pending control message, the home forwarder will deliver the data message to the current (correct) location of the mobile agent. Thus, although there may apparently be forwarding cycles, these will always involve the home forwarder and will be broken immediately on receipt of the pending control message. Condition (C_3) explicitly indicates where a control message involving a may be.

Definition 4.4 (admissible configuration) *An admissible configuration m is generated by a pair (γ, Ob) composed of an admissible function and a process as follows:*

$$m \equiv Rt \mid Ob \mid \Pi_{a \in \mathcal{AN}} V(a, \gamma)$$

where Rt,Ob are as in figure 9 and

$$V(a,\gamma) \quad \equiv \begin{cases} Ag(a,k,r,r') \mid CMs(a,k,r,r',z) \mid \Pi_{r \in F} Fwd(a,r) \mid \\ \Pi_{r \in \mathcal{RN} \setminus Act(a,\gamma)} Idle(a,r) \mid \Pi_{(z,r_1,a_2,r_2,d) \in D} \overline{z}[msg, a, r_1, a_2, r_2, d] \end{cases}$$

$$\text{where} \qquad k = \gamma(a)_1, r = \gamma(a)_2, r' = \gamma(a)_3, z = \gamma(a)_4, F = \gamma(a)_5, D = \gamma(a)_6$$

Let m be an admissible configuration from Mob, generated by (γ, Ob). Further, let $m.DMs(a)$ denote $\Pi_{(z,r_1,a_2,r_2,d) \in \gamma(a)_6} \overline{z}[msg, a, r_1, a_2, r_2, d]$, the data messages in state m addressed to a, and let $m.DMs$ denote all data messages in state m. Sometimes, we will abuse notation and consider a product of messages as a multiset. For convenience, we will write $m.CMs$ and $m.Ob$ to denote the state of the control messages, and atomic observations respectively in configuration m.

Theorem 4.5 *The initial configuration Mob is admissible, and admissible configurations are closed under reduction.*

From this result, it is possible to derive an important property of system Mob: it is always possible to bring the system in a *stable* state.

Corollary 4.6 (control stabilization) *Let m be an admissible configuration generated by (γ, Ob) and let $n = \sharp\{a \mid \gamma(a) \notin K_s\}$. Then $m \rightarrow^{\leq (9*n)} m'$ such that m' is determined by (γ', Ob) and $\forall\, a \; (\gamma'(a)_1 \in K_s$ and $\gamma'(a)_6 = \gamma(a)_6)$. In particular, if $\gamma(a)_1 \notin K_s$, then $\gamma'(a)_1 \in \{1, 6\}$.*

The analogous of corollaries 4.2 and 4.3 can be stated as follows.

Corollary 4.7 (message integrity) *Messages do not get lost nor is their observable content tampered with. Let m be an admissible configuration for Mob, generated by (γ, Ob), and suppose $m \rightarrow m'$. Then for all $\overline{z_j}\vec{x}_j \in m.DMs$ either $\overline{o}[x_{2,j}, x_{4,j}, x_{6,j}] \in m'.Ob \backslash m.Ob$ or else there exists a $\overline{z_l'}\vec{x}_l' \in m'.DMs$ such that $\mathsf{obs}(\vec{x}_l') = \mathsf{obs}(\vec{x}_j)$.*

Corollary 4.8 (message delivery) *Let $a \in \mathcal{AN}$ and let m be an admissible configuration generated by (γ, Ob) such that $\gamma(a)_1 \in K_s$ and $(z, r_1, a_2, r_2, w) \in \gamma(a)_6$. Then the data message can be observed in at most 10 reductions.*

We will now introduce a notion of *observable* of a process, and a related notion of barbed bisimulation (cf. [Par81, Pnu85]).

Definition 4.9 *Let p be a process. Then $\mathcal{O}(p)$ is the following* multi-set *($\vec{y}$ can be ϵ):*

$$\mathcal{O}(p) = \{\overline{o}\vec{y} \mid \exists\, p' \;\; p \equiv (p' \mid \overline{o}\vec{y})\}$$

We note that on an admissible configuration s, $\mathcal{O}(s) = s.Ob$. A similar remark can be applied to an admissible configuration m or c (cf. following definition 4.12).

Definition 4.10 *A binary relation on processes $\mathcal{R}$ is a* barbed bisimulation *if whenever $p\,\mathcal{R}\,q$ then the following conditions hold:*
(1) $\exists\, q' \;\; (q \overset{*}{\to} q'$ *and* $\mathcal{O}(p) = \mathcal{O}(q'))$,
(2) $p \to p'$ *implies* $\exists\, q' \;\; (q \overset{*}{\to} q'$ *and* $p'\,\mathcal{R}\,q')$,
and symmetrically. Two processes p, q are barbed bisimilar*, written $p \overset{\bullet}{\approx} q$, if they are related by a barbed bisimulation.*

We use the notion of barbed bisimulation to relate the simple system *Stat* (viewed as a specification) to the more complex systems *Mob* and *CMob*. Note that each process p has a *unique* commitment $\mathcal{O}(p)$. Taking as commitments the atomic observations would lead to a stricly weaker equivalence.

Theorem 4.11 *Stat* $\overset{\bullet}{\approx}$ *Mob.*

PROOFHINT. We define the *relevant observable content of data messages* in an admissible configuration s from *Stat* and m from *Mob* as the following *multi-sets*, respectively:

$$\begin{array}{ll} \mathcal{O}'(s) & = \{[x_{2,j}, x_{4,j}, x_{6,j}] \mid \overline{z_j}\vec{x}_j \in s.Ms\} \\ \mathcal{O}'(m) & = \{[x_{2,j}, x_{4,j}, x_{6,j}] \mid \overline{z_j}\vec{x}_j \in m.DMs\} \end{array}$$

Next, we introduce a relation $\mathcal{S}$ between admissible configurations from *Stat* and *Mob* as:

$$\begin{array}{ll} \mathcal{S} = \{(s,m) \mid & \mathcal{O}(s) = \mathcal{O}(m), \mathcal{O}'(s) = \mathcal{O}'(m), \text{ and} \\ & (\gamma, Ob) \text{ generates } m \;\;\Rightarrow\;\; \forall\, a \;\; (A_{in}(a) \in s.Ag \;\text{ iff }\; \gamma(a)_1 \in \{2,8\})\} \end{array}$$

We show that $\mathcal{S}$ is a barbed bisimulation. □

4.3 Analysis of *CMob*

Next we turn to the analysis of the protocol with cache. The table in figure 12 lists the situations that can arise during the migration of an agent from a router to another. Relying on this table, we define in figure 13 a notion of admissible function γ. Intuitively, the function γ associates to every agent a its current migration control (state and protocol messages), the routers already visited (either *Fwd's* or *Mam's*), and the data messages and update messages in transit which are addressed to a.

Again, $Act(a, \gamma)$ denotes the routers where a has visited, which are not in an *Idle* state. Condition (C_1), as before, states that at most finitely many agents can

k	$Ag(a,k,r,r',f,c_1,c_2)$	$CMs(a,k,r,r',z)$	$R(a,k,r,r')$
1	$Ha(a,f)$	$\mathbf{0}$	$\{r_0\}$
2	$Ha_{in}(a,f,c_1,c_2)$	$\mathbf{0}$	$\{r_0\}$
3	$Ham(a)$	$\overline{z}[immig,a,r,a,r_0,\_]$	$\{r_0\}$
4	$Ham(a) \mid Bma(a,r)$	$\overline{z}[regd,a,r_0,a,r,\_]$	$\{r_0,r\}$
5	$Haf(a,r) \mid Bma(a,r)$	$\overline{z}[infmd,a,r,a,r_0,\_]$	$\{r_0,r\}$
6	$Haf(a,r) \mid Ma(a,r,f)$	$\mathbf{0}$	$\{r_0,r\}$
7	$Haf(a,r) \mid Ma_{in}(a,r,f,c_1,c_2)$	$\mathbf{0}$	$\{r_0,r\}$
8	$Haf(a,r) \mid Mam(a,r)$	$\overline{z}[repat,a,r_0,a,r,\_]$	$\{r_0,r\}$
9	$Haf(a,r) \mid Mam(a,r)$	$\overline{z}[immig,a,r',a,r,\_]$	$\{r_0,r\}$
10	$Haf(a,r) \mid Bma(a,r')$	$\overline{z}[mig,a,r_0,a,r',\_]$	$\{r_0,r'\}$

Figure 12: Control migration with caching ($r_0 = \mathcal{H}(a)$)

be on the move ("deranged") at any instant. (C_2) is, as before, an invariant on control messages and the forwarder caches, indicating that there are no forwarding cycles, and the cached entries for each agent a always point to routers where the mobile agent has visited. M serves to indicates the router at which there is a $Mam(a)$ when there is a pending *regd* message whose currently location is indicated using Z. (C_3) is an invariant dealing with data messages or forwarded data messages, which indicates that such messages may never arise from, be addressed to, or be present at agents at routers not yet visited. (C_4) is a condition on update messages, stating that such messages are only sent between shadow agents of two different agents, and that they may only originate, be present at and be targetted to routers where the tow agents have been active.

Definition 4.12 (admissible configuration with caching) *An admissible configuration cm is generated by a pair* (γ, Ob)*, composed of an admissible function with caching and a process, as follows:*

$$cm \equiv Rt \mid Ob \mid \Pi_{a\in\mathcal{AN}} V(a,\gamma)$$

where Rt and Ob are as in figure 9, and

$$V(a,\gamma) \equiv \begin{cases} Ag(a,k,r,r',f,c_1,c_2) \mid CMs(a,k,r,r',z) \mid \\ \Pi_{r\in dom(F),F'(r)=nin} Fwd(a,r,F(r)) \mid \Pi_{r\in dom(F),F'(r)=in} Fwd_{in}(a,r,F(r)) \mid \\ \Pi_{r\in dom(M)}(Mam(a,r) \mid \overline{Z(r)}[regd,a,r,a,M(r),\_]) \mid \\ \Pi_{r\in \mathcal{RN}\setminus Act(a,\gamma)} Idle(a,r) \mid \\ \Pi_{(ddir,z,r_1,a_2,r_2,d)\in D}\overline{z}[ddir,a,r_1,a_2,r_2,d] \mid \\ \Pi_{(z,r_1,a_2,r_2)\in UMs}\overline{z}[upd,a,r_1,a_2,r_2,\_] \end{cases}$$

where $k=\gamma(a)_1, r=\gamma(a)_2, r'=\gamma(a)_3, f=\gamma(a)_4, z=\gamma(a)_5, F=\pi_1\circ\gamma(a)_8,$
$F'=\pi_2\circ\gamma(a)_8, M=\pi_1\circ\gamma(a)_9, Z=\pi_2\circ\gamma(a)_9, D=\gamma(a)_{10}, UMs=\gamma(a)_{11}$

$$K = \{1, \ldots, 10\} \qquad K_s = \{1, 2, 6, 7\} \quad \text{(stable states)}$$

$\gamma : \mathcal{AN} \rightarrow$

$(K \times \mathcal{RN} \times \mathcal{RN} \times (\mathcal{AN} \rightarrow \mathcal{RN}) \times (\mathcal{RN} \cup \mathcal{LAN}) \times \{0,1\}^2$	(control migration)
$(\mathcal{RN} \rightharpoonup (\mathcal{RN} \times \{in, nin\}))\times$	(*Fwd's*)
$(\mathcal{RN} \rightharpoonup (\mathcal{RN} \times (\mathcal{RN} \cup \mathcal{LAN})))\times$	(*Mam's*)
$\mathcal{M}_{fin}(\{msg, fwdd\} \times (\mathcal{RN} \cup \mathcal{LAN}) \times \mathcal{RN} \times \mathcal{AN} \times \mathcal{RN} \times \mathcal{DN})\times$	(data messages)
$\mathcal{M}_{fin}((\mathcal{RN} \cup \mathcal{LAN}) \times \mathcal{RN} \times \mathcal{AN} \times \mathcal{RN}))$	(update messages)

$$Act(a, \gamma) = R(a, \gamma(a)_1, \gamma(a)_2, \gamma(a)_3) \cup dom(\gamma(a)_8) \cup dom(\gamma(a)_9)$$

Admissibility conditions on γ:

(C_1)
$$\{a \in \mathcal{AN} \mid \gamma(a)_1 \notin K_s\} \text{ finite}$$

(C_2)
$$\begin{gathered}
\forall a \in \mathcal{AN} \;\; (k = \gamma(a)_1, r = \gamma(a)_2, r' = \gamma(a)_3, f = \gamma(a)_4, z = \gamma(a)_5, \\
F = \pi_1 \circ \gamma(a)_8, M = \pi_1 \circ \gamma(a)_9, \text{ and } Z = \pi_2 \circ \gamma(a)_9) \Rightarrow \\[1em]
(\sharp\{\mathcal{H}(a), r, r'\} = 3, \\
Act(a, \gamma) \text{ finite}, \\
\forall\, a' \in \mathcal{AN} \;\; (f(a') \in Act(a', \gamma)) \\
(CMs(a, k, r, r', z) \equiv \overline{z}[cdir, a, r_1, a, r_2, _] \Rightarrow z \in \{r_1, r_2, \mathcal{L}(r_1, a)\}), \\
dom(F), dom(M), R(a, k, r, r') \text{ pairwise disjoint}, \\
cod(F), cod(M) \subseteq Act(a, \gamma), \\
Acyclic(F, M), \text{ and} \\
(Z(r) \text{ defined } \;\; \Rightarrow \;\; Z(r) \in \{r, \mathcal{L}(r, a), M(r)\}) \;)
\end{gathered}$$

where $Acyclic(F, M)$ means:
$$\nexists r_1, \ldots, r_n, X_1, \ldots X_n \; (n \geq 1, (X_1(r_1) = r_2, \cdots, X_n(r_n) = r_1) \text{ and } X_i \in \{F, M\})$$

(C_3)
$$\begin{gathered}
\forall\, a \in \mathcal{AN} \;\; \forall\, (ddir, z, r_1, a_2, r_2, d) \in \gamma(a)_{10} \\
(r_1 \in Act(a, \gamma), r_2 \in Act(a_2, \gamma), \text{ and} \\
z \in Act(a, \gamma) \cup \{r_2\} \cup \{\mathcal{L}(r'', a) \mid r'' \in Act(a, \gamma)\})
\end{gathered}$$

(C_4)
$$\begin{gathered}
\forall\, a \in \mathcal{AN} \;\; \forall\, (z, r_1, a_2, r_2) \in \gamma(a)_{11} \\
(a_2 \neq a, r_1 \in Act(a, \gamma), r_2 \in Act(a_2, \gamma), \text{ and} \\
z \in Act(a, \gamma) \cup \{r_2\} \cup \{\mathcal{L}(r'', a) \mid r'' \in Act(a, \gamma)\})
\end{gathered}$$

Figure 13: Admissible configurations with caching

Let c be an admissible configuration with caching from *CMob*, generated by (γ, Ob). Further, let $c.DMs(a)$ denote $\Pi_{(ddir,z,r_1,a_2,r_2,d)\in\gamma(a)_{10}}\overline{z}[ddir, a, r_1, a_2, r_2, d]$, the data messages addressed to a in configuration c, and let $c.DMs$ stand for all data messages in configuration c. For convenience, we will write $c.CMs$ and $c.Ob$ to denote the state of the control messages, and atomic observations respectively in configuration c.

Theorem 4.13 *The initial configuration CMob is admissible, and admissible configurations with caching are closed under reduction.*

As in *Mob*, it is possible to bring *CMob* in a stable state.

Corollary 4.14 (control stabilization) *Let c be an admissible configuration generated by (γ, Ob) and let $n = \sharp\{a \mid \gamma(a) \notin K_s\}$. Then $c \rightarrow^{\leq(10*n)} c'$ such that c' is generated by (γ', Ob) and $\forall\, a \;\; (\gamma'(a)_1 \in K_s$ and $\gamma'(a)_{10} = \gamma(a)_{10})$. In particular, if $\gamma(a)_1 \notin K_s$, then $\gamma'(a)_1 \in \{1, 6\}$.*

As in *Stat* and *Mob*, it is easy to derive corollaries concerning message integrity and message delivery. Moreover, the analysis of the invariant allows us to extract some *general principles* for the correct definition of the protocol (note that these principles are an *output* of the analysis of our protocol model, they are not explicitly stated in the informal description of the protocol).

- Cache entries and *Fwd*'s always point to routers which have been visited by the agent.
- Any messages from agent a to agent a' come from a router r, and are directed to a router r', which has been visited by agent a and a', respectively.
- Agent a never sends update messages to a shadow agent itself.
- The protocol for moving an agent a from a router to another terminates in a fixed number steps.
- Given an agent a, the forwarding proxy agents never form forwarding cycles. This ensures that once the agent a has settled in one router, data messages and update messages in transit *can* reach it in a number of steps which is proportional to the length of the longest chain of *Fwd*'s.

We conclude our analysis with the analogous of theorem 4.11 for system *CMob*.

Theorem 4.15 *Stat* $\stackrel{\bullet}{\approx}$ *CMob.*

PROOFHINT. We define the *relevant observable content of data messages* in an admissible configuration with caching c from *CMob*, as the following *multi-set*:

$$\mathcal{O}'(c) = \{[x_{2,j}, x_{4,j}, x_{6,j}] \mid \overline{z_j}\vec{x}_j \in c.Ms\}$$

We define a relation $\mathcal{S}$ between admissible configurations from *Stat* and *CMob* as:

$$\begin{array}{rl} \mathcal{S} = \{(s,c) \mid & \mathcal{O}(s) = \mathcal{O}(c), \mathcal{O}'(s) = \mathcal{O}'(c) \text{ and} \\ & (\gamma, Ob) \text{ generates } c \;\; \Rightarrow \;\; \forall\, a \;\; (A_{in}(a) \in s.Ag \;\text{ iff }\; \gamma(a)_1 \in \{2,7\})\} \end{array}$$

We show that $\mathcal{S}$ is a barbed bisimulation. □

5 Conclusion

We have described in an elementary process calculus notation a simplified version of the Mobile IP protocol. We believe that a precise yet abstract model is useful in establishing the correctness of the protocol, as well as providing a basis for simulation and experimentation. Our modelling uses nondeterminism, and asynchronous communication (with unbounded and unordered buffers). Nondeterminism serves as a powerful abstraction mechanism, assuring us of the correctness of the protocol for arbitrary behaviours of the processes, even if we try different instances of particular management policies (*e.g.*, routing and cache management policies) provided they maintain the same invariants as in the nondeterministic model. Asynchronous communication makes minimal assumptions on the properties of the communication channels. All we require is that messages are not lost and in particular we assume there is a mechanism for avoiding *store-and-forward* deadlocks. An alternative framework for the description of the protocol which is based on Unity is proposed in [MR97] (the formal analysis of the protocol using this framework is not available yet).

In our modelling, we have greatly simplified various details. On the one hand, this simplification is useful, since it again serves as a way of abstracting from particular protocols for establishing connections (*e.g.*, Neighbour Discovery, etc.). On the other hand, we have assumed that our so-called "control messages" eventually reach their destination without getting lost or corrupted. A future direction of work may be to model protocols that cope with failures, or to model security and authentication issues.

By concentrating on an abstract and simple model we have been able to specify the protocol and by a process of analysis to discover and explicate some of its organising principles. The specification and combinatorial analysis of the protocol is sufficiently complicated to preclude leaving it "implicit" in the informal protocol description. By a careful analysis we have been able to carry out a hand proof. A direction for further research is the formal development of the proof using a proof assistant (a so-called theorem prover). Another problem is the analysis of the protocol by some automatic verification tool (typically a model checker). For such verification, the protocol has to be adapted to fit into the finite state framework. Note that even if we assume the sets $\mathcal{RN}, \mathcal{AN}, \mathcal{DN}$ finite we cannot bound the number of messages in transit. One strategy to be explored is to introduce bounded buffers along with a discipline for avoiding deadlocks.

References

[BTP95] P. Bhagwat, S. Tripathi, and C. Perkins. Network layer mobility: an architecture and survey. Technical Report TR 3570, CS Dept. University of Maryland, 1995.

[HT91] K. Honda and M. Tokoro. An object calculus for asynchronous communication. *Proc. ECOOP 91, Geneve*, 1991.

[IDM91] J. Ioannidis, D. Duchamp, and G. Maguire. IP-based protocols for mobile internetworking. In *Proc. ACM SIGCOMM*, 1991.

[JP96] D. Johnson and C. Perkins. Mobility support in IPv6 (RFC). Version expiring May 97, 1996.

[MR97] P. McCann and G.-C. Roman. Mobile Unity coordination constructs applied to packet forwarding. In *Proc. Coordination, Springer Lect. Notes in Comp. Sci. 1282*, 1997.

[Par81] D. Park. Concurrency and automata on infinite sequences. In *Springer Lect. Notes in Comp. Sci. 104*, 1981.

[Pnu85] A. Pnueli. Linear and branching systems in the semantics and logics of reactive systems. In *Springer Lect. Notes in Comp. Sci. 194*, 1985.

[TUSM94] F. Teraoka, K. Uehara, H. Sunahara, and J. Murai. Vip: a protocol providing host mobility. *Comm. ACM*, 37(8), 1994.

A Proofs

In the following, we present the proofs related to the analysis carried on in §4.

A.1 Proofs for system *Stat*

Proof of proposition 4.1 *Stat* trivially satisfies the admissibility criteria. Closure under reduction is established by examination of the possible transitions. The only possible transitions are:

• The reduction follows from an internal choice. This may be performed only by some $A(a)$, resulting in either $A_{in}(a)$, or else in a message origination. In the latter case, the resulting configuration is $A(a)$ together with a *new* data message that satisfies the admissibility criterion.

• $A_{in}(a)$ receives a data message $\vec{x}$ addressed to a. By the side condition on z_j and the injectivity of $\mathcal{L}$ $A_{in}(a)$ cannot receive a message addressed to another agent. In this case, the data message is removed and replaced with a new observation $\mathsf{obs}(\vec{x}) = \overline{o}[x_2, x_4, x_6]$. The observation cannot be an error message $\overline{o}\epsilon$, since only data messages are present. Moreover, this transition is the only possible way of introducing new elements in the Ob component of an admissible configuration.

• A router receives a data message addressed to a. The message is forwarded to either $\mathcal{H}(a)$, or to $\mathcal{L}(\mathcal{H}(a), a)$, and its observable content is left unchanged. □

Proof of corollary 4.2 The analysis in the above proof. □

Proof of corollary 4.3 At most two routers are involved in moving the message from its source router to channel $\mathcal{L}(\mathcal{H}(a), a)$. For consumption, an internal choice by $A(a)$ may be necessary before consuming the message. □

A.2 Proofs for system *Mob*

Proof of theorem 4.5 The initial configuration *Mob* is generated by the pair $(\gamma_o, \emptyset)$, where $\gamma_o(a) = (1, r_a, r'_a, z_a, \emptyset, \emptyset)$, and r_a, r'_a are chosen so that $\sharp\{\mathcal{H}(a), r_a, r'_a\} = 3$, which is always possible by the hypothesis that $\sharp\mathcal{RN} \geq 3$, and where z_a is suitably chosen ($\mathcal{H}(a)$ may be a good choice). Let m be an admissible configuration generated by (γ, Ob), and suppose $m \rightarrow m'$. By case analysis, we build a pair

(γ', Ob') which generates m' and such that γ' is admissible. In the following, we only indicate the components that need to be altered and leave the verification of admissibility to the reader.

• The reduction follows from an internal choice. This reduction can be performed only by the agents $Ha(a)$ $(\gamma(a)_1 = 1)$ and $Ma(a,r)$ $(\gamma(a)_1 = 6, r \neq r_0)$.

$$
\begin{array}{ll}
Ha(a) \to Ha_{in}(a) & \gamma'(a)_1 = 2 \\
Ha(a) \to Ha(a) \mid \overline{r_0}[msg, y, z, a, r_0, w] & \gamma'(y)_6 = \gamma(y)_6 \cup \{(r_0, z, a, r_0, w)\} \\
Ha(a) \to Ha(a) & \_ \\
Ha(a) \to Ham(a) \mid \overline{r_0}[immig, a, u, a, r_0, \_] & \gamma'(a)_1 = 3, \\
 & \gamma'(a)_2 = u, \gamma'(a)_4 = r_0 \\
Ma(a,r) \to Ma_{in}(a,r) & \gamma'(a)_1 = 8 \\
Ma(a,r) \to Ma(a,r) \mid \overline{r}[msg, y, z, a, r, w] & \gamma'(y)_6 = \gamma(y)_6 \cup \{(r, z, a, r, w)\} \\
Ma(a,r) \to Ma(a,r) & \_ \\
Ma(a,r) \to Fwd(a,r) \mid \overline{r}[repat, a, r_0, a, r, \_] & \gamma'(a)_1 = 7, \gamma'(a)_4 = r, \\
 & \gamma'(a)_5 = \gamma(a)_5 \cup \{r\} \\
Ma(a,r) \to Fwd(a,r) \mid \overline{r}[immig, a, u, a, r, \_] & \gamma'(a)_1 = 9, \gamma'(a)_3 = u, \\
 & \gamma'(a)_4 = r, \gamma'(a)_5 = \gamma(a)_5 \cup \{r\}
\end{array}
$$

In no case does Ob change, and DMs changes only in the second and sixth situations, by the addition of a new data message as specified by $\gamma'(y)_6$.

• A router receives a data message addressed to a. We have:

$$
\begin{array}{l}
Router(r) \mid \overline{r}[msg, a, r_1, a_2, r_2, w] \to \\
Router(r) \mid \left\{ \begin{array}{ll} \overline{l}[msg, a, r_1, a_2, r_2, w] & \text{if } r = r_1, l = \mathcal{L}(r, a) \\ \overline{r_1}[msg, a, r_1, a_2, r_2, w] & \text{otherwise} \end{array} \right.
\end{array}
$$

By (C_3) we know $r_1 \in Act(a,\gamma)$, $r_2 \in Act(a_2,\gamma)$, and $r \in \{r_2\} \cup Act(a,\gamma)$. In both cases we alter $\gamma'(a)_6$. Ob does not change, and for the only change we make in going to $\gamma'(a)_6$, namely changing the data message to some $\overline{z}\vec{x}$, the observable content of the message $(\vec{x})$ remains unchanged.

• A router receives a control message addressed to a. The analysis is similar to the previous case, but in this case we use condition (C_2) and alter $\gamma'(a)_4$. Ob and DMs do not change.

• An agent receives a data message. Agents $Ha(a)$ and $Ma(a)$ cannot receive. Because of condition (C_3) an agent $Idle(a,r)$ cannot receive a data message. If $Ham(a)$ or $Bma(a,r)$ receive then nothing changes. If $Ha_{in}(a)$ or $Ma_{in}(a,r)$ receive then we change $\gamma'(a)_1$ (move to $Ha(a)$ or $Ma(a,r)$), $\gamma'(a)_6$ (remove message received), and Ob' (observe message received). If $Fwd(a,r)$ receives then the message is forwarded to the home router and we emend $\gamma'(a)_6$. Finally, if $Haf(a,r)$ receives we forward to r (which maintains the invariant by (C_2)) and we alter $\gamma'(a)_6$. In the last two subcases, Ob does not change, and for the only change we make to $\gamma'(a)_6$, namely to some $\overline{z}\vec{x}$, the observable content of the message remains unchanged.

• An agent receives a control message. We have to consider the situations $\gamma(a)_1 \in K \backslash K_s$. We have the following transitions: $3 \to 4 \to 5 \to 6$, $7 \to 1$, $9 \to 10 \to$

5($\to$ 6) which correspond to the control transitions outlined in Figure 5. The details of the alterations to γ are left to the reader. Note that at most three control messages may be involved in any migration. In no case do either Ob or DMs change. $\Box$

Proof of corollary 4.6 It takes at most three communications to deliver a control message. If $\gamma(a)_1 \notin K_s$ then we can bring the system in a configuration in K_s by delivering at most three control messages. Ob remains unchanged as indicated in the proof of theorem 4.5. $\Box$

Proof of corollary 4.7 The proof follows from the case analysis of theorem 4.5, as is indicated at each stage. $\Box$

Proof of corollary 4.8 In the worst case, the message has been forwarded by the home forwarder to the shadow at a router r from where the agent has moved to router r', but the message has got delayed *en route*. In 2 reductions, the message is delivered to the forwarder at r, which sends it back to the home router in 1+2 additional steps. The home forwarder now sends the message to the agent at r' in 3 further steps. With 2 more reductions the message is observed. $\Box$

Proof of theorem 4.11 We show that $\mathcal{S}$ is a barbed bisimulation. By definition of $\mathcal{S}$, we have $\mathcal{O}(s) = \mathcal{O}(m)$.

Simulation of s by m. Let $(s, m) \in \mathcal{S}$, and suppose $s \to s'$. We show that there exists a m' such that $m \overset{*}{\to} m'$ and $(s', m') \in \mathcal{S}$. Let m be generated by some (γ, Ob).

- $s \to s'$ by an internal choice creating a new data message dm, from a to some y. Thus $A(a) \in s.Ag$, whence $\gamma(a)_1 \notin \{2, 8\}$. If $\gamma(a)_1 \notin K_s$, then by corollary 4.6, there exists a m'' generated by (γ'', Ob) such that $\gamma''(a)_1 \in \{1, 6\}$. Thus $m'' \to m'$ is possible for some m' by $Ha(a)$ or $Ma(a)$ performing an internal choice generating the same data message. For m', generated by (γ', Ob), we have $\gamma'(y)_6 = \gamma''(y)_6 \cup \{dm\}$, Ob unchanged, and $\gamma'(a)_1 \in \{1, 6\}$. Since $\gamma''(y)_6 = \gamma(y)_6$, we have $\mathcal{O}'(s') = \mathcal{O}'(m')$. Since $A(a) \in s'.Ag$, we have $(s', m') \in \mathcal{S}$.

- $s \to s'$ by some $A(a) \in s.Ag$ performing $A(a) \to A_{in}(a)$. Hence $\gamma(a)_1 \notin \{2, 8\}$. If $\gamma(a)_1 \notin K_s$, then by corollary 4.6, there exists a m'' generated by (γ'', Ob) such that $\gamma''(a)_1 \in \{1, 6\}$. Thus $m'' \to m'$ is possible for some m' by performing an internal choice going to state 2 or 8. For m', generated by (γ', Ob), we have $\gamma''(y)_6 = \gamma'(y)_6$, Ob unchanged, and $\gamma'(a)_1 \in \{2, 8\}$. Since $\gamma''(y)_6 = \gamma(y)_6$, we have $\mathcal{O}'(s') = \mathcal{O}'(m')$. Since $A_{in}(a) \in s'.Ag$, we have $(s', m') \in \mathcal{S}$.

- $s \to s'$ by some $A_{in}(a) \in s.Ag$ receiving a message $\overline{l}\vec{x} \in s.Ms$, resulting in state s' having $s'.Ob = s.Ob \cup \mathsf{obs}(\vec{x})$, $s'.Ms = s.Ms \backslash \{\overline{l}\vec{x}\}$ and $A(a) \in s'.Ag$. By assumption on m, $\gamma(a)_1 \in \{2, 8\} \subset K_s$. $\mathcal{O}'(s) = \mathcal{O}'(m)$ implies there is a $\overline{z}\vec{x}' \in m.DMs$, where $\vec{x}$ and $\vec{x}'$ have the same observable content. If $\overline{l}\vec{x}' \in m.DMs$, then $m \to m'$ by receiving this message, for some m' generated by $\gamma', m.Ob \cup \mathsf{obs}(\vec{x})$. By corollary 4.7, we have $\gamma'(a)_1 \in \{1, 6\}$, and $\mathcal{O}'(s') = \mathcal{O}'(m')$. If $\overline{l}\vec{x}' \notin m.DMs$, then by corollaries 4.7 and 4.8, in at most 8 steps, $m \xrightarrow{*} m''$ corresponding to the previous case, with $m''.Ob = m.Ob$ and $\mathcal{O}'(m'') = \mathcal{O}'(m)$. Then $m'' \to m'$, as before. Thus we have $(s', m') \in \mathcal{S}$.

• $s \to s'$ by a router's reduction. m does nothing, and we have, by corollary 4.7, $(s', m) \in \mathcal{S}$.

Simulation of m by s.

• $m \to m'$ by an internal choice such that for some a, $\gamma(a)_1 \in \{1,6\}$ and $\gamma'(a)_1 \in \{2,8\}$. By assumption $A(a) \in s.Ag$, so by $A(a) \to A_{in}(a)$, $s \to s'$. Since the data messages and observations remain unchanged, $(s', m') \in \mathcal{S}$.

• $m \to m'$ by an internal choice to initiate a migration. That is, for some a, $\gamma(a)_1 \in \{1,6\}$. By assumption $A(a) \in s.Ag$, s remains in the same configuration. Since the data messages and observations remain unchanged, $(s, m') \in \mathcal{S}$.

• $m \to m'$ by an internal choice generating a new data message $dm = \overline{z}\vec{x}$. That is, for some a, $\gamma(a)_1 \in \{1,6\}$. By assumption $A(a) \in s.Ag$. $s \to s'$ is possible by $A(a)$ generating a data message $\overline{z'}\vec{x}'$ such that $\vec{x}, \vec{x}'$ have the same observable content, and $A(a) \in s'.Ag$. Clearly, $(s', m') \in \mathcal{S}$.

• $m \to m'$ by receiving a data message $\overline{z}\vec{x}$. Thus, for some a, $\gamma(a)_1 \in \{2,8\}$, and by corollary 4.7, $\mathcal{O}(m') = \mathcal{O}(m) \cup \{\overline{o}[x_2, x_4, x_6]\}$, and $m'.DMs = m.DMs \backslash \{\overline{z}\vec{x}\}$. By assumption, $A_{in}(a) \in s.Ag$, and there is a data message $\overline{z'}\vec{x}' \in s.Ms$ such that $\vec{x}, \vec{x}'$ have the same observable content. By corollaries 4.3, and 4.2, this message can be delivered in at most 2 reductions and $s \to s'$ is possible by $A_{in}(a)$ consuming that message. Clearly, $(s', m') \in \mathcal{S}$.

• $m \to m'$ by performing a control (delivery / consumption / production) transition involving an agent a. By assumption, $\gamma(a)_1 \notin \{2,8\}$, otherwise this move would not have been possible. By corollary 4.6, we are guaranteed that $\gamma'(a)_1 \notin \{2,8\}$. In response, s does nothing. Since, $m'.DMs = m.DMs$, $(s, m') \in \mathcal{S}$.

• $m \to m'$ by a router's step. s does nothing. Since by corollary 4.7, $\mathcal{O}'(m') = \mathcal{O}'(m)$, $(s, m') \in \mathcal{S}$.

• $m \to m'$ by a message forwarding step, in which some $Fwd(a, r)$ or $Haf(a, r)$ redirects a message. s does nothing. Since by corollary 4.7, $\mathcal{O}'(m') = \mathcal{O}'(m)$, $(s, m') \in \mathcal{S}$. □

A.3 Proofs for system *CMob*

Proof of theorem 4.13 The initial configuration *Cmob* is generated by the pair $(\gamma_o, \emptyset)$ where $\gamma_o(a) = (1, r_a, r'_a, \mathcal{H}, z_a, c_1, c_2, \emptyset, \emptyset, \emptyset, \emptyset)$ and r_a, r'_a, z_a, c_1, c_2 are suitably chosen.

Suppose cm is generated by (γ, Ob) and $cm \to cm'$. We can then define a pair (γ', Ob') which determines cm' and such that γ' is admissible. We only indicate the components of (γ', Ob') that need to be altered, but for those which just need to be chosen.

• The reduction is an internal choice. This step can be taken by the processes $Ha(a,f)$, $Ma(a,r,f)$, and $Fwd(a,r,r')$.

$Ha(a,f) \to Ha_{in}(a,f,c_1,c_2)$	$\gamma'(a)_1 = 2$
$Ha(a,f) \to Ha(a,f) \mid \overline{r_0}[msg,y,z,a,r_0,w]$	$\gamma'(y)_{10} = \gamma(y)_{10} \cup \{(msg,r_0,z,a,r_0,w)\}$
$Ha(a,f) \to Ha(a,f)$	-
$Ha(a,f) \to Ham(a) \mid \overline{r_0}[immig,a,u,a,r_0,\_]$	$\gamma'(a)_1 = 3, \gamma'(a)_5 = r_0$
$Ha(a,f) \to Ha(a,f[r'/y])$	$\gamma'(a)_4 = \gamma(a)_4[r'/y]$
$Ma(a,r,f) \to Ma_{in}(a,r,f,c_1,c_2)$	$\gamma'(a)_1 = 7$
$Ma(a,r,f) \to Ma(a,r,f) \mid \overline{r}[msg,y,z,a,r,w]$	$\gamma'(y)_{10} = \gamma(y)_{10} \cup \{(msg,r,z,a,r,w)\}$
$Ma(a,r,f) \to Ma(a,r,f)$	-
$Ma(a,r,f) \to Mam(a,r) \mid \overline{r}[repat,a,r_0,a,r,\_]$	$\gamma'(a)_1 = 8, \gamma'(a)_5 = r$
$Ma(a,r,f) \to Mam(a,r) \mid \overline{r}[immig,a,u,a,r,\_]$	$\gamma'(a)_1 = 9, \gamma'(a)_5 = r$
$Ma(a,r,f) \to Ma(a,r,f[r'/y])$	$\gamma'(a)_4 = \gamma(a)_4[r'/y] \quad (y \neq a)$
$Fwd(a,r,r') \to Fwd(a,r,r_0)$	$\gamma'(a)_8$ (first component)
$Fwd(a,r,r') \to Fwd_{in}(a,r,r')$	$\gamma'(a)_8$ (second component)

About the transitions in line 5 and 11, we note that (C_2) is invariant, since in the cache update $r' = \mathcal{H}(y)$, and by (C_4) with respect to agent y, $\mathcal{H}(y) \in R(y,k,r'',r''') \subseteq Act(y,\gamma')$ for some r'', r'''.

• A router receives a message addressed to a. From the definition of $Router$, we have:

$$Router(r) \mid \overline{r}[dir,a,r_1,a_2,r_2,w] \to$$
$$Router(r) \mid \begin{cases} \overline{l}[dir,a,r_1,a_2,r_2,w] & \text{if } r = r_1, l = \mathcal{L}(r,a) \\ \overline{r_1}[dir,a,r_1,a_2,r_2,w] & \text{otherwise} \end{cases}$$

We distinguish three cases according to the value of dir:

data message By (C_3), we have $r \in Act(a,\gamma) \cup \{r_2\}$. We update $\gamma(a)_{10}$, and so (C_3) is invariant. Ob does not change, and for the only change we make in getting to $\gamma'(a)_{10}$, namely changing the data message to some $\overline{z}\vec{x}$, the observable content of the message $(\vec{x})$ remains unchanged.

update message By (C_4), we have $r \in Act(a,\gamma) \cup \{r_2\}$. We update $\gamma(a)_{11}$, and so (C_4) is invariant. Ob and DMs remain unchanged.

control message Again similar to the previous subcase, with Ob and DMs remaining unchanged. Note that the conditions on $\gamma(a)_5$ in (C_2) are satisfied.

• An agent receives a message addressed to a. We observe that Ha, Ma, Fwd cannot receive.

data message $Idle$ cannot receive a data message (by (C_3), which states that data messages originate and are routed only through active routers). If $Ha_{in}(a,f,c_1,c_2)$ receives a data message from y, we update Ob, we may change cache and remove the data message from (DMs), and we move to $Ha(a,f')$. We may also send an update message. That is $\gamma(a)_4, \gamma(a)_{10}$ and $\gamma(y)_{11}$ may change. We note that (C_1) is invariant trivially, (C_3) is invariant, since a data message is only removed from the configuration, and

(C_4) is invariant even in case we add a data message. (C_2) is invariant even when the cache is changed. $f'(y) = r_y \in Act(y, \gamma')$, since the message from y originated at $r_y \in Act(y, \gamma)$ by (C_3). The analysis for Ma_{in} is similar. *Ham*, *Bma* and *Mam* filter data messages, leaving *Ob* and *DMs* unchanged. *Haf* and Fwd_{in} forward data messages, leaving *Ob* unchanged, altering *DMs* without changing the observable content $([x_2, x_4, x_6])$ of the data message, and also keeping (C_3) invariant.

update message If $Ha_{in}(a, f)$ receives an update message from agent y, we may update cache, and we move to $Ha(a, f')$. (C_4) is invariant since we are only deleting a message from $\gamma(a)_{11}$. (C_2) is invariant even when the cache is changed since $f'(y) = r_y \in Act(y, \gamma')$, since the message from y originated at $r_y \in Act(y, \gamma)$ by (C_4). The analysis for Ma_{in} is similar. *Ham* and *Mam* filter update messages. *Haf* and Fwd_{in} forward the update message. In the last two cases, the invariants, (C_4) in particular, are maintained, and *Ob* and *DMs* are unchanged.

control message We have to consider the situations for $\gamma(a)_1 \in K \backslash K_s$. We get the following transitions $3 \to 4 \to 5 \to 6$, $8 \to 1$, and $9 \to 10 \to 5 (\to 6)$. *Ob* and *DMs* do not change.

We consider the cases in the last sequence of transitions (which are the most difficult), and show that (C_2) is invariant. We only consider the receipt of a control message, and not delivery steps. By assumption, r, r', r_0 are all distinct.

$9 \to 10$: $Fwd_{in}(a, r', r'')$ or $Idle(a, r')$ receives $\overline{\mathcal{L}(r', a)}[immig, a, r', a, r, \_]$, and goes to $Bma(a, r')$, producing control messages $\overline{r'}[regd, a, r, a, r', \_]$ and $\overline{r'}[mig, a, r_0, a, r', \_]$. The changes made are: $\gamma'(a)_1 = 10, \gamma'(a)_5 = r', \gamma'(a)_8 = \gamma(a)_8 \backslash \{r'\}$, $\gamma'(a)_9 = \gamma(a)_9[(r', r')/r]$. $Act(a, \gamma)$ remains invariant. The new control messages are at r'. $dom(F), dom(M), R(a, k, r, r')$ remain pairwise disjoint, since $r' \notin dom(F)$ and $R(a, k, r, r') = \{r_0, r'\}$ in $\gamma'(a)$. It is easily verified that $cod(F)$, $cod(M) \subseteq Act(a, \gamma')$. Finally, $Acyclic(F, M)$ is maintained: If by γ, the shadow at r' was $Idle(a, r)$, then r' is now added to the path, where it was not previously present. If by γ the shadow at r' was $Fwd(a, r, r'')$, now in γ', $r' \notin dom(F) \cup dom(M)$.

$10 \to 5$: $Haf(a, r)$ receives $\overline{\mathcal{L}(r_0, a)}[mig, a, r_0, a, r', \_]$, and goes to $Haf(a, r')$, producing control message $\overline{r_0}[infmd, a, r', a, r_0, \_]$. Only the state and *CMs* undergo any change. Thus (C_2) remains invariant.

$5 \to 6$: $Bma(a, r')$ receives $\overline{\mathcal{L}(r', a)}[infmd, a, r', a, r_0, \_]$ and becomes $Ma(a, r, f)$ where $f = \mathcal{H}$ in our modelling. The changes are to the state, to *CMs*, and $\gamma'(a)_4 = f$. Since $clH(y) \in Act(y, \gamma')$ for all $y \in \mathcal{AN}$, (C_2) is maintained invariant. □

Proof of corollary 4.14 It takes at most four reductions to deliver and process a control message (two delivery steps plus one internal choice step, plus one step to consume the message; otherwise three reductions suffice). If $\gamma(a)_1 \notin K_s$ then we can bring the system in a configuration in K_s by delivering at most three control messages (*regd* need not be delivered to reach a state in K_s). *Ob* remains unchanged as indicated in the proof of theorem 4.13. □

We explicitly state and prove the corollaries concerning message integrity and message delivery for *CMob*.

Corollary A.1 (message integrity) *Messages do not get lost nor is their observable content tampered with. Let c be an admissible configuration for CMob, generated by* (γ, Ob)*, and suppose* $c \to c'$*. Then for all* $\overline{z_j}\vec{x}_j \in c.DMs$ *either* $\overline{o}[x_{2,j}, x_{4,j}, x_{6,j}] \in c'.Ob \backslash c.Ob$*, or else there exists a* $\overline{z_l'}\vec{x}_l' \in c'.DMs$ *such that* $\mathsf{obs}(\vec{x}_l') = \mathsf{obs}(\vec{x}_j)$.

Proof. The proof follows from the case analysis of theorem 4.13, as is indicated at each stage. □

Corollary A.2 (message delivery) *Let* $a \in \mathcal{AN}$ *and let c be an admissible configuration generated by* (γ, Ob) *such that* $(ddir, z, r_1, a_2, r_2, w) \in \gamma(a)_{10}$ *and* $\gamma(a)_1 \in K_s$*. Then the data message can be "delivered" in a number of reductions proportional to the length of the longest forwarding chain.*

Proof. By (C_2) (the acyclicity condition and conditions on $cod(F)$, $cod(M)$), for all *forwarding chains* of the form:

$$X_1(r_1) = r_2, \ldots, X_i(r_i) = r_{i+1}$$

where $X_j \in \{F, M\}$, there is an n such that $r_{n+1} \notin dom(M) \cup dom(F)$. Additionally, since all routers on a forwarding chain for a are in $Act(a, \gamma)$, $r_{n+1} \in R(a, k, r, r')$ by (C_2). Thus for $k \in K_s$, the forwarding chain ends either in $\mathcal{H}(a)$ or at the actual location of the agent. Now, in at most 3 reductions, (2 for delivering a *regd* message and 1 for consuming it), per $Mam(a, r_i)$ process, it can become $Fwd(a, r_i, r_{i+1})$ while maintaining the *same* forwarding chain. $\sharp dom(M)$ is finite since $Act(a, \gamma)$ is finite. Now in at most 4 reductions per hop, the data message can be forwarded to a forwarder at the next router in the forwarding chain. Thus in finitely many steps it can reach the agent at the end of the forwarding chain. If this is $Haf(a, r')$, then in another hop, the message can be delivered to the mobile agent. □

Proof of theorem 4.15 We show that $\mathcal{S}$ is a barbed bisimulation. By definition of $\mathcal{S}$, we have $\mathcal{O}(s) = \mathcal{O}(m)$.

Simulation of s by c. The proof is almost identical to that of theorem 4.11. The only case that we will describe is:

- $s \to s'$ by some $A_{in}(a) \in s.Ag$ receiving a message $\overline{l}\vec{x} \in s.Ms$, resulting in state s' having $s'.Ob = s.Ob \cup \mathsf{obs}(\vec{x})$, $s'.Ms = s.Ms \backslash \{\overline{l}\vec{x}\}$ and $A(a) \in s'.Ag$. By assumption on c, $\gamma(a)_1 \in \{2, 7\} \subset K_s$. $\mathcal{O}'(s) = \mathcal{O}'(c)$ implies there is a $\overline{z}\vec{x}' \in c.DMs$ such that $\mathsf{obs}(\vec{x}) = \mathsf{obs}(\vec{x}')$. If $\overline{l}\vec{x}' \in c.DMs$, then $c \to c'$ by receiving this message, for some c' generated by $\gamma', c.Ob \cup \mathsf{obs}(\vec{x})$. By corollary A.1, we have

$\gamma'(a)_1 \in \{1,6\}$, and $\mathcal{O}'(s') = \mathcal{O}'(c')$. If $\overline{l}\vec{x}' \notin c.DMs$, then by corollaries A.1 and A.2, in a bounded number of steps, $c \xrightarrow{*} c''$ corresponding to the previous case, with $c''.Ob = c.Ob$ and $\mathcal{O}'(c'') = \mathcal{O}'(c)$. Then $c'' \to c'$, as before. Thus we have $(s', c') \in \mathcal{S}$.

Simulation of c by s The proof is again almost identical to that of theorem 4.11, except for the following new cases:

- $c \to c'$ by an internal choice such that for some a, $\gamma(a)_4$ is changed (cache is reset). s performs no reductions and $(s, c') \in \mathcal{S}$.

- $c \to c'$ by an internal choice causing a cache update. s performs no move and $(s, c') \in \mathcal{S}$.

- $c \to c'$ by an internal choice such that for some a, r, $Fwd(a, r, r') \to Fwd_{in}(a, r, r')$. Again, $(s, c') \in \mathcal{S}$, with s making no move.

- $c \to c'$ by delivering or forwarding an update message. Clearly, $(s, c') \in \mathcal{S}$, with s making no move.

- $c \to c'$ by consuming an update message, and possibly altering the cache. Since $c.DMs$ is unchanged, $(s, c') \in \mathcal{S}$, with s making no move.

- We present the case of $c \to c'$ by consuming a data message, perhaps creating an update message and possibly altering the cache. These last two effects are irrelevant to the relation $\mathcal{S}$, and so we only need to consider the effects of consuming the data message. $c \to c'$ by receiving a data message $\overline{z}\vec{x}$. Thus, for some a, $\gamma(a)_1 \in \{2,8\}$, and by corollary A.1, $\mathcal{O}(c') = \mathcal{O}(c) \cup \{\overline{o}[x_2, x_4, x_6]\}$, and $\gamma'(a)_{10} = \gamma(a)_{10} \backslash \{\overline{z}\vec{x}\}$. By assumption, $A_{in}(a) \in s.Ag$, and there is a data message $\overline{z'}\vec{x}' \in s.Ms$ such that $\vec{x}, \vec{x}'$ have the same observable content. By corollaries 4.3, and 4.2, this message can be delivered in at most 2 reductions and $s \to s'$ is possible by $A_{in}(a)$ consuming that message. Clearly, $(s', c') \in \mathcal{S}$.

- Again the case of $c \to c'$ by performing a control (delivery / consumption / production) transition involving an agent a is no different that in the proof of the theorem 4.11. By assumption $\gamma(a)_1 \notin \{2,7\}$, otherwise this move would not have been possible. By corollary 4.14, we are guaranteed that $\gamma'(a)_1 \notin \{2,7\}$. s does nothing. Since, $c'.DMs = c.DMs$, $(s, c') \in \mathcal{S}$.

The reader may verify that the other cases are as in the proof of theorem 4.11. □