UNIVERSITÄT AUGSBURG
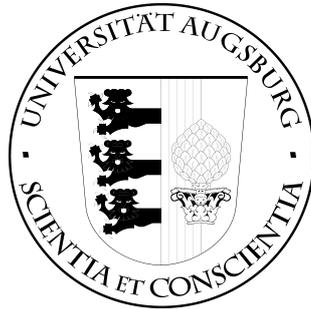
Fairness of Actions
in System Computations

F. Corradini, M.R. Di Berardini, W. Vogler

institut für informatik
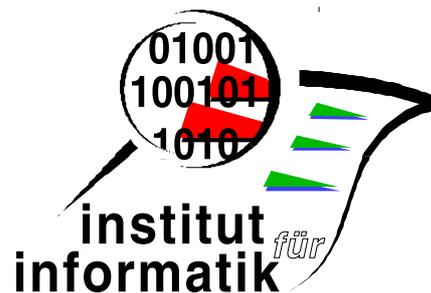
INSTITUT FÜR INFORMATIK
D-86135 AUGSBURG

Fairness of Actions in System Computations[*]

F. Corradini,M.R. Di Berardini [†]
Dipartimento di Matematica e Informatica
Università di Camerino

W. Vogler [‡]
Institut für Informatik
Universität Augsburg

Abstract

This paper contrasts two important features of parallel system computations: *fairness* and *timing*. The study is carried out at specification system level by resorting to a well-known process description language. The language is extended with labels which allow to filter out those process executions that are not (weakly) fair (as in [7, 8]), and with upper time bounds for the process activities (as in [6]).

We show that fairness and timing are closely related. Two main results are stated. First, we show that each everlasting (or non-Zeno) timed process execution is fair. Second, we provide a characterization for fair executions of untimed processes in terms of timed process executions. This results in a finite representation of fair executions using regular expressions.

1   Introduction

In the theory and practice of parallel systems, fairness and timing play an important role when describing the system dynamics. Fairness requires that a system activity which is continuously enabled along a computation will eventually proceed; this is usually a necessary requirement for proving liveness properties of the system. Timing gives information on when actions are performed and can serve as a basis for considering efficiency.

We will show that fairness and timing are somehow related - although they are used in different contexts. Our comparison is conducted at system specification level by resorting to a standard (CCS-like) process description language. We consider two extensions of this basic language. The first extension permits to isolate the fair system executions and follows the approach of Costa and Stirling [7, 8]. The second one adds upper time bounds for the execution time of system activities and follows the approach taken in [6].

Costa and Stirling distinguish between fairness of actions (also called events) and fairness of components; these coincide in [7] for a CCS-like language without restriction, while fairness of components is studied in [8] for full CCS. In both cases, Costa and Stirling distinguish between weak and strong fairness. Weak fairness requires that if an action (a component, resp.) can *almost always* proceed then it must eventually do so, and in fact it must proceed

---

[†]{flavio.corradini,mariarita.diberardini}@unicam.it
[‡]vogler@informatik.uni-augsburg.de

infinitely often, while strong fairness requires that if an action (a component) can proceed *infinitely often* then it must proceed infinitely often. Differences between fairness of actions and fairness of components and between weak and strong fairness are detailed in [8]; for the purpose of this paper, we are interested in weak fairness of actions. An important and useful result stated in [7, 8] characterizes fair computations as the concatenation of certain finite sequences, called LP-steps in [8]. This characterization permits to think of fairness in terms of a localizable property and not as a property of complete (maximal) executions; but even for a finite-state process, LP-steps usually give rise to a transition system with infinitely many transitions.

Regarding timing, we follow the approach taken in the timed process algebra PAFAS (Process Algebra for Faster Asynchronous Systems). Based on ideas first studied for Petri nets e.g. in [12, 1], this new process description language has been proposed as a useful tool for comparing the worst-case efficiency of asynchronous systems (see [11, 6] for the general theory and [3] for an application). PAFAS is a CCS-like process description language [14] where basic actions are atomic and instantaneous but have an associated time bound (which is 1 or 0 for simplicity) as a maximal time delay for their execution.[1] When, for an action with time bound 1, this idle-time of 1 has elapsed, the action becomes *urgent* (i.e. its time bound becomes 0) and it must be performed (or be deactivated) before time may pass further – unless it has to wait for synchronization with another component, which either does not offer synchronization on this action at all or at least can still delay the synchronization. Consequently, a synchronization is urgent when all the partners are urgent. We assume that time is discrete, since this is simple and since continuous time does not make a difference for the (testing-based) observational preorder defined on top of the operational models (see [6] for this and many other results on PAFAS).

We prove two main results relating timed computations of PAFAS processes and weak fairness of actions. First, we prove that all everlasting (or non-Zeno) computations [2] are fair. This result shows that timing with upper time bounds imposes fairness among the different system activities. Intuitively, this is easy to see: when one time unit passes, the active actions become urgent and must be performed (or be deactivated) before time may pass further; this clearly ensures that an activated action does not wait forever in a computation with infinitely many time steps.

As a second main result we show that LP-steps – defined for untimed processes – coincide in the timed setting with sequences of basic actions between two consecutive time steps. As a consequence of this lemma we have that non-Zeno process computations fully characterize fair computations.

Besides providing a formal comparison between fairness and timing, our timed characterization of fair executions results in a representation with technical advantages compared to the approach of [7, 8]. In order to keep track of the different instances of system activities along a system execution, Costa and Stirling associate labels to actions, and the labels are essential in the definition of fair computations. New labels are created dynamically during the system evolution with the immediate effect of changing the syntax of process terms; thus, cycles in the transition system of a process are impossible and even finite-state processes (according to the ordinary operational semantics) usually become infinite-state.

---

[1]As discussed in [6], due to these upper time bounds time can be used to evaluate efficiency, but it does not influence functionality (which actions are performed); so compared to CCS, also PAFAS treats the full functionality of asynchronous systems.

[2]A process computation is a Zeno computation when infinitely many actions happen in finite time.

From the maximal runs of such a transition system, Costa and Stirling filter out the unfair computations by a criterion that considers the processes and their labels on a maximal run. Our timed semantics also provides such a two-level description: we also change the syntax of processes – in our case by adding timing information –, but this is much simpler than the labels of [7, 8], and it leaves finite-state processes finite-state. Then we apply a simpler filter, which does not consider the processes: we simply require that infinitely many time steps occur in a run. As a small price, we have to project away these time steps in the end.

As mentioned above, Costa and Stirling give a one-level characterization of fair computations with an SOS-semantics defining so-called LP-steps; these are (finite, though usually unbounded) sequences of actions leading from ordinary processes to ordinary processes, with the effect that even finite-state transition systems for LP-steps usually have infinitely many transitions – although they are at least finite-state. In contrast, our time-based operational semantics defines steps with single actions (or unit time steps), and consequently a finite-state transition system is really finite.

Finally, using standard automata-theoretic techniques, we can get rid of the time steps in such a finite-state transition system by constructing another finite-state transition system with regular expressions as arc labels; maximal runs in this transition system are exactly the fair runs. This way we also arrive at a one-level description, and ours is truly finite. Compared to the conference version of this paper [4], the one-level description is improved.

For this purpose, we concentrate on the so-called initial processes, which are standard CCS-like processes; we present a slight variation of the PAFAS operational semantics, which for initial processes very closely corresponds to the original one, but has the following advantage: in the one-level description of the fair runs of an initial process P, only initial processes appear that are reachable from P according to a standard semantics.

The rest of the paper is organized as follows. The next section recalls PAFAS and presents the new operational semantics. Section 3 describes the theory of fairness we consider. Section 4 is the core of the paper; it relates fairness and timing, and presents the one-level description of fair runs. Section 5 compares the operational semantics for PAFAS used in this paper with the one of [6]. We conclude with Section 6. Many proofs have been moved to appendices to improve readability.

2 PAFAS - A Process Algebra for Faster Asynchronous Systems

In this section we give a brief description of PAFAS, a process algebra introduced in [6] to consider the functional behaviour and the temporal efficiency of asynchronous systems. The PAFAS transitional semantics is given by two sets of SOS-rules. One describes the functional behaviour and is very similar to the SOS-rules for standard CCS [14]. The other describes the temporal behaviour and is based on a notion of refusal sets. On top of this transitional semantics, a preorder relation is defined which is naturally a (worst-case) efficiency preorder. Since we contrast the notions of timing and fairness at operational level, we do not present the preorder in this paper and refer the reader to [6] for more details and results on PAFAS.

2.1 PAFAS Process

PAFAS is a CCS-like process description language [14] (with *TCSP*-like parallel composition), where basic actions are atomic and instantaneous but have associated a time bound interpreted as a maximal time delay for their execution. As explained in [6], these upper time bounds (which are either 0 or 1, for simplicity) are suitable for evaluating the performance of asynchronous systems. Moreover, time bounds do not influence functionality (which actions are performed); so compared to CCS, also PAFAS treats the full functionality of asynchronous systems.

We use the following notation: $\mathbb{A}$ is an infinite set of basic actions. An additional action τ is used to represent internal activity, which is unobservable for other components. We define $\mathbb{A}_\tau = \mathbb{A} \cup \{\tau\}$. Elements of $\mathbb{A}$ are denoted by $a, b, c, \ldots$ and those of $\mathbb{A}_\tau$ are denoted by $\alpha, \beta, \ldots$. Actions in $\mathbb{A}_\tau$ can let time 1 pass before their execution, i.e. 1 is their maximal delay. After that time, they become *urgent* actions written $\underline{a}$ or $\underline{\tau}$; these have maximal delay 0. The set of urgent actions is denoted by $\underline{\mathbb{A}}_\tau = \{\underline{a} \mid a \in \mathbb{A}\} \cup \{\underline{\tau}\}$ and is ranged over by $\underline{\alpha}, \underline{\beta}, \ldots$. Elements of $\mathbb{A}_\tau \cup \underline{\mathbb{A}}_\tau$ are ranged over by μ.

$\mathcal{X}$ is the set of process variables, used for recursive definitions. Elements of $\mathcal{X}$ are denoted by $x, y, z, \ldots$.

$\Phi : \mathbb{A}_\tau \to \mathbb{A}_\tau$ is a *general relabelling function* if the set $\{\alpha \in \mathbb{A}_\tau \mid \emptyset \neq \Phi^{-1}(\alpha) \neq \{\alpha\}\}$ is finite and $\Phi(\tau) = \tau$. Such a function can also be used to define *hiding*: P/A, where the actions in A are made internal, is the same as $P[\Phi_A]$, where the relabelling function Φ_A is defined by $\Phi_A(\alpha) = \tau$ if $\alpha \in A$ and $\Phi_A(\alpha) = \alpha$ if $\alpha \notin A$.

We assume that time elapses in a discrete way. (PAFAS is not time domain dependent, meaning that the choice of discrete or continuous time makes no difference for the testing-based semantics of asynchronous systems, see [6] for more details.) Thus, an action prefixed process $a.P$ can either do action a and become process P (as usual in CCS) or can let one time step pass and become $\underline{a}.P$; $\underline{a}$ is called *urgent a*, and $\underline{a}.P$ as a stand-alone process cannot let time pass, but can only do a to become P.

Definition 2.1 (*timed process terms*)

The set $\tilde{\mathbb{P}}_1$ of *initial (timed) process terms* is generated by the following grammar

$$P ::= \mathsf{nil} \mid x \mid \alpha.P \mid \ \mid P + P \mid P\|_A P \mid P[\Phi] \mid \mathsf{rec}\ x.P$$

where $x \in \mathcal{X}$, $\alpha \in \mathbb{A}_\tau$, Φ is a general relabelling function and $A \subseteq \mathbb{A}$ possibly infinite. Elements in $\tilde{\mathbb{P}}_1$ correspond to ordinary CCS-like terms

The set $\tilde{\mathbb{P}}$ of the (general) *(timed) process terms* is generated by the following grammar:

$$Q ::= P \mid \underline{\alpha}.P \mid Q + Q \mid Q \|_A Q \mid Q[\Phi] \mid \mathsf{rec}\ x.Q$$

where $P \in \tilde{\mathbb{P}}_1$, $x \in \mathcal{X}$, $\alpha \in \mathbb{A}_\tau$, Φ is a general relabelling function and $A \subseteq \mathbb{A}$ possibly infinite. We assume that the recursion is *guarded*, i.e. for $\mathsf{rec}\ x.Q$ variable x only appears in Q within the scope of a prefix $\mu.()$ with $\mu \in \mathbb{A}_\tau \cup \underline{\mathbb{A}}_\tau$. A term Q is *guarded* if each occurrence of a variable is guarded in this sense.

The set of closed (i.e., every variable x in a timed process term Q is bound by the corresponding $\mathsf{rec}\ x$-operator) timed process terms in $\tilde{\mathbb{P}}$ and $\tilde{\mathbb{P}}_1$, simply called *processes* and *initial processes* resp., is denoted by $\mathbb{P}$ and $\mathbb{P}_1$ resp.[3]

---

[3]As shown in [3], $\mathbb{P}_1$ processes do not have time-stops; i.e. every finite process run can be extended such that time grows unboundedly.

A brief description of the (PAFAS) operators now follows. nil is the Nil-process; it cannot perform any action, but may let time pass without limit. A trailing nil will often be omitted, so e.g. $a.b + c$ abbreviates $a.b.$nil$ + c.$nil. $Q_1 + Q_2$ models the choice between two conflicting processes Q_1 and Q_2. $Q_1\|_A Q_2$ is the parallel composition of two processes Q_1 and Q_2 that run in parallel and have to synchronize on all actions from A; this synchronization discipline is inspired from TCSP. $Q[\Phi]$ behaves as Q but with the actions changed according to Φ. rec $x.Q$ models a recursive definition.

Initial processes are just standard processes of a standard process algebra. General processes are defined here such that they include all processes reachable from the initial ones according to the operational semantics to be defined below. In contrast to the more general PAFAS processes in [6] and the conference version of the present paper, our processes have urgent actions as 'top-prefixes' only.

We can now define the set of active actions in a process term. Given a process term Q, $\mathcal{A}(Q, A)$ denotes the set of the *active* (or enabled) actions of Q when the environment prevents the actions in A. For technical convenience, we allow A to be a subset of $\mathbb{A}_\tau$ here and in some similar definitions that follow.

Definition 2.2 (*activated basic actions*)
Let $Q \in \tilde{\mathbb{P}}$ and $A \subseteq \mathbb{A}_\tau$. The set $\mathcal{A}(Q, A)$ is defined by induction on Q.

Nil, Var: $\mathcal{A}($nil$, A) = \mathcal{A}(x, A) = \emptyset$

Pref: $\mathcal{A}(\mu.P, A) = \begin{cases} \{\alpha\} & \text{if } \mu = \alpha \text{ or } \mu = \underline{\alpha} \text{ and } \alpha \notin A \\ \emptyset & \text{otherwise} \end{cases}$

Sum: $\mathcal{A}(Q_1 + Q_2, A) = \mathcal{A}(Q_1, A) \cup \mathcal{A}(Q_2, A)$

Par: $\mathcal{A}(Q_1 \|_B Q_2, A) = \mathcal{A}(Q_1, A \cup B) \cup \mathcal{A}(Q_2, A \cup B) \cup$
$\mathcal{A}(Q_1, \mathbb{A}_\tau \backslash (B \backslash A)) \cap \mathcal{A}(Q_2, \mathbb{A}_\tau \backslash (B \backslash A))$

Rel: $\mathcal{A}(Q[\Phi], A) = \Phi(\mathcal{A}(Q, \Phi^{-1}(A)))$
Rec: $\mathcal{A}($rec $x.Q, A) = \mathcal{A}(Q, A)$

The *activated actions* of Q are defined as $\mathcal{A}(Q, \emptyset)$ which we abbreviate to $\mathcal{A}(Q)$.

The set A represents the actions restricted upon. This is the reason why $\mathcal{A}(\alpha.P, A) = \mathcal{A}(\underline{\alpha}.P, A) = \emptyset$ if $\alpha \in A$ and $\mathcal{A}(\alpha.P, A) = \mathcal{A}(\underline{\alpha}.P, A) = \{\alpha\}$, if $\alpha \notin A$. A nondeterministic process can perform all the actions that its alternative components can perform minus the restricted ones. Parallel composition increases the prevented set. $\mathcal{A}(Q_1 \|_B Q_1, A)$ includes the actions that Q_1 and Q_2 can perform when the actions in A and the actions in B (the synchronizing ones) are prevented. $\mathcal{A}(Q_1 \|_B Q_2, A)$ also includes the actions in B, but not in A, that both Q_1 and Q_2 can perform. The other rules are as expected.

A significant subset of the activated actions is the set of urgent ones. These are activated actions that cannot let time pass.

Definition 2.3 (*urgent activated action*)
Let $Q \in \tilde{\mathbb{P}}$ and $A \subseteq \mathbb{A}_\tau$. The set $\mathcal{U}(Q, A)$ is defined as in Definition 2.2 when $\mathcal{A}(\_)$ is replaced by $\mathcal{U}(\_)$ and the Pref-rule is replaced by the following one:

Pref:
$$\mathcal{U}(\mu.P, A) = \begin{cases} \{\alpha\} & \text{if } \mu = \underline{\alpha} \text{ and } \alpha \notin A \\ \emptyset & \text{otherwise} \end{cases}$$

The *urgent activated actions* of Q are defined as $\mathcal{U}(Q, \emptyset)$ which we abbreviate to $\mathcal{U}(Q)$.

The operational semantic exploits two functions on process terms: $\mathsf{clean}(\_)$ and $\mathsf{unmark}(\_)$. Function $\mathsf{clean}(\_)$ removes *all inactive urgencies* in a process term $Q \in \tilde{\mathbb{P}}$. When a process evolves and a synchronized action is no longer urgent or enabled in some synchronization partner, then it should also lose its urgency in the others; the corresponding change of markings is performed by clean, where again set A in $\mathsf{clean}(Q, A)$ denotes the set of actions that are not enabled or urgent due to restrictions of the environment. Function $\mathsf{unmark}(\_)$ simply removes all urgencies (inactive or not) in a process term $Q \in \tilde{\mathbb{P}}$ and can be defined, as follows, by induction on the process structure.

Definition 2.4 (*cleaning inactive urgencies*)

Given a process term $Q \in \tilde{\mathbb{P}}$ we define $\mathsf{clean}(Q)$ as $\mathsf{clean}(Q, \emptyset)$ where, for a set $A \subseteq \mathbb{A}$, $\mathsf{clean}(Q, A)$ is defined as follows:

Nil, Var: $\mathsf{clean}(\mathsf{nil}, A) = \mathsf{nil}, \qquad \mathsf{clean}(x, A) = x$

Pref:
$$\mathsf{clean}(\underline{\alpha}.P, A) = \begin{cases} \alpha.P & \text{if } \alpha \in A \\ \underline{\alpha}.P & \text{otherwise} \end{cases}$$

$$\mathsf{clean}(\alpha.P, A) = \alpha.P$$

Sum: $\mathsf{clean}(Q_1 + Q_2, A) = \mathsf{clean}(Q_1, A) + \mathsf{clean}(Q_2, A)$

Par: $\mathsf{clean}(Q_1 \|_B Q_2, A) = \mathsf{clean}(Q_1, A \cup A') \|_B \mathsf{clean}(Q_2, A \cup A'')$
        where $A' = (\mathcal{U}(Q_1) \backslash \mathcal{U}(Q_2)) \cap B$ and $A'' = (\mathcal{U}(Q_2) \backslash \mathcal{U}(Q_1)) \cap B$

Rel: $\mathsf{clean}(Q[\Phi], A) = \mathsf{clean}(Q, \Phi^{-1}(A))[\Phi]$

Rec: $\mathsf{clean}(\mathsf{rec}\ x.Q, A) = \mathsf{rec}\ x.\,\mathsf{clean}(Q, A)$

Definition 2.5 (*cleaning all urgencies*)

Let Q be a $\tilde{\mathbb{P}}$ term. Then $\mathsf{unmark}(Q)$ is defined by induction on Q as follows:

Nil, Var: $\mathsf{unmark}(\mathsf{nil}) = \mathsf{nil}, \qquad \mathsf{unmark}(x) = x$

Pref: $\mathsf{unmark}(\alpha.P) = \mathsf{unmark}(\underline{\alpha}.P) = \alpha.P$

Sum: $\mathsf{unmark}(Q_1 + Q_2) = \mathsf{unmark}(Q_1) + \mathsf{unmark}(Q_2)$

Par: $\mathsf{unmark}(Q_1 \|_B Q_2) = \mathsf{unmark}(Q_1) \|_B \mathsf{unmark}(Q_2)$

Rel: $\quad\quad\quad$ unmark$(Q[\Phi])$ = unmark$(Q)[\Phi]$

Rec: $\quad\quad\quad$ unmark$(\mathsf{rec}\ x.Q)$ = $\mathsf{rec}\ x.$ unmark(Q)

Some useful properties of functions clean and unmark are proved in Appendix 7.1.

2.2 The functional behaviour of PAFAS process

The transitional semantics describing the functional behaviour of PAFAS processes indicates which basic actions they can perform; timing information can be disregarded, since we only have *upper* time bounds.

Definition 2.6 (*Functional operational semantics*) The following SOS-rules define the transition relations $\xrightarrow{\alpha}\subseteq(\tilde{\mathbb{P}}\times\tilde{\mathbb{P}})$ for $\alpha\in\mathbb{A}_\tau$, the *action transitions*.

As usual, we write $Q\xrightarrow{\alpha}Q'$ if $(Q,Q')\in\xrightarrow{\alpha}$ and $Q\xrightarrow{\alpha}$ if there exists a $Q'\in\tilde{\mathbb{P}}$ such that $(Q,Q')\in\xrightarrow{\alpha}$, and similar conventions will apply later on.

$$\text{PREF}_{a1}\frac{}{\alpha.P\xrightarrow{\alpha}P}\quad\quad\quad\text{PREF}_{a2}\frac{}{\underline{\alpha}.P\xrightarrow{\alpha}P}$$

$$\text{SUM}_a\frac{Q_1\xrightarrow{\alpha}Q'}{Q_1+Q_2\xrightarrow{\alpha}Q'}$$

$$\text{PAR}_{a1}\frac{\alpha\notin A,\ Q_1\xrightarrow{\alpha}Q_1'}{Q_1\|_AQ_2\xrightarrow{\alpha}\mathsf{clean}(Q_1'\|_AQ_2)}\quad\quad\text{PAR}_{a2}\frac{\alpha\in A,\ Q_1\xrightarrow{\alpha}Q_1',\ Q_2\xrightarrow{\alpha}Q_2'}{Q_1\|_AQ_2\xrightarrow{\alpha}\mathsf{clean}(Q_1'\|_AQ_2')}$$

$$\text{REL}_a\frac{Q\xrightarrow{\alpha}Q'}{Q[\Phi]\xrightarrow{\Phi(\alpha)}Q'[\Phi]}\quad\quad\text{REC}_a\frac{Q\{\mathsf{rec}\ x.\mathsf{unmark}(Q)/x\}\xrightarrow{\alpha}Q'}{\mathsf{rec}\ x.Q\xrightarrow{\alpha}Q'}$$

Additionally, there are symmetric rules for Par_{a1} and Sum_a for actions of Q_2.

The use of unmark in rule REC_a has to be contrasted with the temporal behaviour defined next. Consider an initial process Q; after a time-step, the recursive term $\mathsf{rec}\ x.Q$ evolves to $\mathsf{rec}\ x.Q'$ (see the rule Rec_r below) where Q' is Q after the passage of time; in general, Q' will contain some urgent actions. Since occurrences of x in Q are guarded, each x stands for a process which is not enabled yet and cannot have urgent actions; thus, these recursive calls in $\mathsf{rec}\ x.Q'$ refer to Q and not to Q', which explains the substitution in rule REC_a of Definition 2.6 based on unmark; cf. the example at the end of the section.

2.3 The temporal behaviour of PAFAS process

We are now ready to define the refusal traces of a term $Q\in\tilde{\mathbb{P}}$. Intuitively a refusal trace records, along a computation, which actions process Q can perform ($Q\xrightarrow{\alpha}Q'$, $\alpha\in\mathbb{A}_\tau$) and which actions Q can refuse to perform when time elapses ($Q\xrightarrow{X}_rQ'$, $X\subseteq\mathbb{A}$).

A transition like $Q\xrightarrow{X}_rQ'$ is called a (partial) *time-step*. The actions listed in X are not urgent; hence Q is justified in not performing them, but performing a time step instead.

This time step is partial because it can occur only in contexts that can refuse the actions not in X. If $X = \mathbb{A}$ then Q is fully justified in performing this time-step; i.e., Q can perform it independently of the environment. If $Q \xrightarrow{\mathbb{A}}_r Q'$ we write $Q \xrightarrow{1} Q'$ and say that P performs a *full time-step*. In [6], it is shown that inclusion of refusal traces characterizes an efficiency preorder which is intuitively justified by a testing scenario. In the present paper, we need partial time steps only to set up the following SOS-semantics; our real interest is in runs where all time steps are full. We let λ range over $\mathbb{A}_\tau \cup \{1\}$.

Definition 2.7 (*Refusal transitional semantics*)

The following inference rules define $\xrightarrow{X}_r \subseteq (\tilde{\mathbb{P}} \times \tilde{\mathbb{P}})$, where $X \subseteq \mathbb{A}$.

$$\text{NIL}_r \frac{}{\mathsf{nil} \xrightarrow{X}_r \mathsf{nil}}$$

$$\text{PREF}_{r1} \frac{}{\alpha.P \xrightarrow{X}_r \underline{\alpha}.P} \qquad \text{PREF}_{r2} \frac{\alpha \notin X \cup \{\tau\}}{\underline{\alpha}.P \xrightarrow{X}_r \underline{\alpha}.P}$$

$$\text{PAR}_r \frac{Q_i \xrightarrow{X_i}_r Q_i' \text{ for } i = 1, 2, \ X \subseteq (A \cap (X_1 \cup X_2)) \cup (X_1 \cap X_2) \backslash A}{Q_1 \|_A Q_2 \xrightarrow{X}_r \mathsf{clean}(Q_1' \|_A Q_2')}$$

$$\text{SUM}_r \frac{\forall \, i = 1, 2 \ Q_i \xrightarrow{X}_r Q_i'}{Q_1 + Q_2 \xrightarrow{X}_r Q_1' + Q_2'}$$

$$\text{REL}_r \frac{Q \xrightarrow{\Phi^{-1}(X \cup \{\tau\}) \backslash \{\tau\}}_r Q'}{Q[\Phi] \xrightarrow{X}_r Q'[\Phi]} \qquad \text{REC}_r \frac{Q \xrightarrow{X}_r Q'}{\mathsf{rec}\ x.Q \xrightarrow{X}_r \mathsf{rec}\ x.Q'}$$

The rules in Definition 2.7 explain the refusal operational semantics of a PAFAS term. Rule PREF_{r1} says that a process $\alpha.Q$ can let time pass and refuse to perform any action while rule PREF_{r2} says that a process Q prefixed by an urgent action $\underline{\alpha}$, can let time pass but action α cannot be refused. Process $\underline{\tau}.Q$ cannot let time pass and cannot refuse any action; also in any context, $\underline{\tau}.Q$ has to perform τ as explained by Rule PREF_{a2} in Definition 2.6 before time can pass further.

Another rule worth noting is PAR_r which defines which actions a parallel composition can refuse during a time-step. The intuition is that $Q_1 \|_A Q_2$ can refuse an action α if either $\alpha \notin A$ (Q_1 and Q_2 can perform α independently) and both Q_1 and Q_2 can refuse α, or $\alpha \in A$ (Q_1 and Q_2 are forced to synchronize on α) and at least one of Q_1 and Q_2 can refuse α, i.e. can delay it. Thus, an action in a parallel composition is urgent (cannot be further delayed) only when all synchronizing 'local' actions are urgent. Also in this case we unmark the inactive urgencies. The other rules are as expected.

For the use of the various definitions consider the following example.

Example 2.8 Let $P = \mathsf{rec}\ x.a.\ \|_{\{a\}} \mathsf{rec}\ x.\ (a.\mathsf{nil} + b.c.x)$ and consider the following behavior:

$P \xrightarrow{1} \mathsf{rec}\ x.\underline{a}.x \ \|_{\{a\}} \mathsf{rec}\ x.\ (\underline{a}.\mathsf{nil} + \underline{b}.c.x) \xrightarrow{b}$

$P' = \mathsf{rec}\ x.a.x \ \|_{\{a\}} c.\mathsf{rec}\ x.\ (a.\mathsf{nil} + b.c.x) \xrightarrow{1}$

$\mathsf{rec}\ x.a.x \ \|_{\{a\}} \underline{c}.\mathsf{rec}\ x.\ (a.\mathsf{nil} + b.c.x) \xrightarrow{c}$

$\mathsf{rec}\ x.a.x \ \|_{\{a\}} \mathsf{rec}\ x.\ (a.\mathsf{nil} + b.c.x)$

P can either perform b or synchronize on a. Thus both these actions become urgent after the first time-step. Now, if the right-hand component performs b, the action a on the left-hand side is no longer enabled and it has to lose its urgency. At the same time, on the right-hand side x is replaced by the right-hand component of P using unmark. Finally, after the second time-step c as the only activated action of P' is marked as urgent.

3   Fairness and PAFAS

In this section we briefly describe our theory of fairness. It closely follows Costa and Stirling's theory of (weak) fairness. The main ingredients of the theory are:

- *A labelling for process terms.* This allows to detect during a transition which action is actually performed; e.g., for process $P = \mathsf{rec}\ x.\alpha.x$, we need additional information to detect whether the left-hand side instance of action α or the right-hand one is performed in the transition $P\|_\emptyset P \xrightarrow{\alpha} P\|_\emptyset P$. When an action is performed, we speak of an *event*, which corresponds to a label – or actually, different from [7, 8], a tuple of labels as we will see.

- *Live events.* An action of a process term is live if it can currently be performed. In a term like $a.b.\mathsf{nil}\|_{\{b\}}\ b.\mathsf{nil}$ only action a can be performed while b cannot, momentarily. Such a live action corresponds to a possible event, i.e. to a label.

- *Fair sequences.* A maximal sequence is fair when no event in a process term becomes live and then remains live throughout.

These items sketch the general methodology used by Costa and Stirling to define and isolate fair computations in [7, 8]. It has to be noted, however, that in [8] Costa and Stirling concentrate on fairness of process components; i.e., along a fair computation, there cannot exist any subprocess that could always contribute some action but never does so. In contrast, we will require fairness for actions. In the setting of [7], i.e. with CCS-composition but without restriction, these two views coincide.

To demonstrate the difference, consider $a\|_{\{a\}}\mathsf{rec}\ x.(a.x + b.x)$ and a run consisting of infinitely many b's. This run is not fair to the component a, since this component is enabled at every stage, but never performs its a. In our view, this run is fair for the synchronized *action a*, since the second component offers always a fresh a for synchronization. (Another intuitive explanation is that action a is not possible *while* b is performed.) Correspondingly, the label for such a synchronization (called an event label) is a pair of labels, each stemming from one of the components; such a pair is a live event, and it changes with each transition. In [5], we have shown how to change this framework in order to capture the fairness of components.

We now describe the three items in more detail. Most of the definitions in the rest of this section are taken from [8] with the obvious slight variations due to the different language we are using (the timed process algebra PAFAS with TCSP parallel composition instead of CCS). We also take from [8] those results that are language independent. The others will be proven.

3.1 A labelling for process terms

In order to determine the fairness of a transition sequence, Costa and Stirling use a labelling method. Labels are associated with basic actions and operators inside a process. Along a computation, labels are unique and, once a label disappears, it will not reappear in the process anymore.

The set of *labels* is $\mathsf{LAB} = \{1,2\}^*$ with ε as the empty label and $u, v, w, \ldots$ as typical elements; $\leq$ is the *prefix preorder* on LAB. We have that $u \leq v$ if there is $u' \in \mathsf{LAB}$ such that $v = uu'$ (and $u < v$ if $u' \in \{1,2\}^+$). We also use the following notation:

- (Set of tuples) $\mathcal{N} = \{\langle v_1, \ldots, v_n \rangle \mid n \geq 1, \ v_1, \ldots, v_n \in \mathsf{LAB}\}$;

- (Composition of tuples) $s_1 \times s_2 = \langle v_1, \ldots, v_n, w_1, \ldots, w_m \rangle$, where $s_1, s_2 \in \mathcal{N}$ and $s_1 = \langle v_1, \ldots, v_n \rangle$, $s_2 = \langle w_1, \ldots, w_m \rangle$;

- (Composition of sets of tuples) $N \times M = \{s_1 \times s_2 \mid s_1 \in N \text{ and } s_2 \in M\}$, where $N, M \subseteq \mathcal{N}$. Note that $N = \emptyset$ or $M = \emptyset$ implies $N \times M = \emptyset$.

All PAFAS operators and variables will now be labelled in such a way that no label occurs more than once in an expression. We call this property *unicity of labels*. As indicated above, an action being performed might correspond to a pair or more generally to a tuple of labels, cf. the definition of live events below (3.7); therefore, we call tuples of labels *event labels*.

Labels (i.e. elements of LAB) are assigned systematically following the structure of PAFAS terms usually as indexes and in case of parallel composition as upper indexes. Due to recursion the labelling is dynamic: the rule for rec generates new labels.

Definition 3.1 (*labelled process algebra*)

The labelled process algebra $\mathsf{L}(\tilde{\mathbb{P}})$ (and similarly $\mathsf{L}(\tilde{\mathbb{P}}_1)$ etc.) is defined as $\bigcup_{u \in \mathsf{LAB}} \mathsf{L}_u(\tilde{\mathbb{P}})$, where $\mathsf{L}_u(\tilde{\mathbb{P}}) = \bigcup_{Q \in \tilde{\mathbb{P}}} \mathsf{L}_u(Q)$ and $\mathsf{L}_u(Q)$ is defined inductively as follows:

Nil, Var: $\mathsf{L}_u(\mathsf{nil}) = \{\mathsf{nil}_u\}$, $\mathsf{L}_u(x) = \{x_u\}$
                In examples, we will often write nil for nil_u, if the label u is not relevant.

Pref: $\mathsf{L}_u(\mu.P) = \{\mu_u.P' \mid P' \in \mathsf{L}_{u1}(P)\}$

Sum: $\mathsf{L}_u(Q_1 + Q_2) = \{Q_1' +_u Q_2' \mid Q_1' \in \mathsf{L}_{u1}(Q_1), \ Q_2' \in \mathsf{L}_{u2}(Q_2)\}$

Par: $\mathsf{L}_u(Q_1 \parallel_A Q_2) = \{Q_1' \parallel_A^u Q_2' \mid Q_1' \in \mathsf{L}_{u1v}(Q_1), \ Q_2' \in \mathsf{L}_{u2v'}(Q_2)$
               where $v, v' \in \mathsf{LAB}\}$

Rel: $\mathsf{L}_u(Q[\Phi]) = \{Q'[\Phi_u] \mid \ Q' \in \mathsf{L}_{u1v}(Q) \text{ where } v \in \mathsf{LAB}\}$

Rec: $\mathsf{L}_u(\mathsf{rec}\ x.Q) = \{\mathsf{rec}\ x_u.Q' \mid Q' \in \mathsf{L}_{u1}(Q)\}$

We assume that, in $\mathsf{rec}\ x_u.Q$, $\mathsf{rec}\ x_u$ binds all free occurrences of a labelled x. We let $\mathsf{L}(Q) = \bigcup_{u \in \mathsf{LAB}} \mathsf{L}_u(Q)$ and $\mathsf{LAB}(Q)$ is the set of labels occurring in Q.

The unicity of labels must be preserved under derivation. For this reason in the rec rule the standard substitution must be replaced by a substitution operation which also changes the labels of the substituted expression.

Definition 3.2 (*a new substitution operator*)

The new substitution operation, denoted by $\{\!| \, \_ \, |\!\}$, is defined on $\mathsf{L}(\tilde{\mathbb{P}})$ using the following operators:

i. $()^{+v}$ If $Q \in \mathsf{L}_u(\tilde{\mathbb{P}})$, then $(Q)^{+v}$ is the term in $\mathsf{L}_{vu}(\tilde{\mathbb{P}})$ obtained by prefixing v to all labels in Q.

ii. $()_\varepsilon$ If $Q \in \mathsf{L}_u(\tilde{\mathbb{P}})$, then $(Q)_\varepsilon$ is the term in $\mathsf{L}_\varepsilon(\tilde{\mathbb{P}})$ obtained by removing the prefix u from all labels in Q. (Note that u is the unique prefix-minimal label in Q.)

Suppose $Q, Q' \in \mathsf{L}(\tilde{\mathbb{P}})$ and $x_u, \ldots, x_v$ are all free occurrences of a labelled x in Q then $Q\{\!| Q'/x |\!\} = Q\{((Q')_\varepsilon)^{+u}/x_u, \ldots, ((Q')_\varepsilon)^{+v}/x_v\}$. The motivation of this definition is that in $Q\{\!|Q'/x|\!\}$ each substituted Q' inherits the label of the x it replaces.

Easy but important are the relationships between activated and urgent actions of PAFAS and of labelled PAFAS processes. Since labels are just annotations used to distinguish different instances of basic actions, they do not interfere with these notions and we can define $\mathcal{A}(Q, A)$ and $\mathcal{U}(Q, A)$ for a labelled PAFAS process Q just as in Definitions 2.2 and 2.3, resp.

Similarly, the operation of removing urgencies, inactive or not, does not depend on labels. They are performed in the same way both in the unlabelled and labelled setting and we can define $\mathsf{clean}(Q, A)$ and $\mathsf{unmark}(Q)$ for a labelled PAFAS process P just as in Definitions 2.4 and 2.5, resp.

Finally, the behavioural operational semantics of the labelled PAFAS is obtained by replacing the rule Rec_a in Definition 2.6 with the rule:

$$\mathrm{Rec}_a \frac{Q\{\!| \, \mathsf{rec}\ x_u.\mathsf{unmark}(Q)/x \, |\!\} \xrightarrow{\alpha} Q'}{\mathsf{rec}\ x_u.Q \xrightarrow{\alpha} Q'}$$

and the rules Pref_{a1} and Pref_{a2} in Definition 2.6 with the rules:

$$\mathrm{Pref}_{a1} \frac{}{\alpha_u.P \xrightarrow{\alpha} P} \qquad\qquad \mathrm{Pref}_{a2} \frac{}{\underline{\alpha}_u.P \xrightarrow{\alpha} P}$$

because we assume that labels are not observable when actions are performed. The other rules are unchanged.

As a consequence, a labelled term Q and its unlabelled version, that we we denote with $\mathsf{R}(Q)$, can perform exactly the same transitions, as stated by the following proposition.

Proposition 3.3 Let $Q \in \mathsf{L}_u(\tilde{\mathbb{P}})$ and $A \subseteq \mathbb{A}_\tau$. Then:

i. $Q \xrightarrow{\alpha} Q'$ $(Q \xrightarrow{X}_r Q')$ implies $\mathsf{R}(Q) \xrightarrow{\alpha} \mathsf{R}(Q')$ $(\mathsf{R}(Q) \xrightarrow{X}_r \mathsf{R}(Q'))$ in unlabelled PAFAS;

ii. if $\mathsf{R}(Q) \xrightarrow{\alpha} R$ $(\mathsf{R}(Q) \xrightarrow{X}_r R)$ in unlabelled PAFAS then for some Q' with $R = \mathsf{R}(Q')$, we have $Q \xrightarrow{\alpha} Q'$ $(Q \xrightarrow{X}_r Q')$;

iii. $\mathcal{A}(Q, A) = \mathcal{A}(\mathsf{R}(Q), A)$ and $\mathcal{U}(Q, A) = \mathcal{U}(\mathsf{R}(Q), A)$.

As an example for 3.3 ii), observe that for $a.\mathsf{nil}\|_\emptyset \mathsf{nil} \xrightarrow{a} \mathsf{nil}\|_\emptyset \mathsf{nil}$ and $\mathsf{R}(a_{u1}.\mathsf{nil}_{u11}\|_\emptyset^u \mathsf{nil}_{u2}) = a.\mathsf{nil}\|_\emptyset \mathsf{nil}$ we indeed have $a_{u1}.\mathsf{nil}_{u11}\|_\emptyset^u \mathsf{nil}_{u2} \xrightarrow{a} \mathsf{nil}_{u11}\|_\emptyset^u \mathsf{nil}_{u2}$; the latter term is a labelled process since we allow $P' \in \mathsf{L}_{u1v}(P)$ in case Par of Definition 3.1, while e.g. in case Pref we require $P' \in \mathsf{L}_{u1}(P)$.

An immediate consequence of the labelling are the following facts that have been proven in [8].

Fact 3.4 Let $Q \in \mathsf{L}_u(\tilde{\mathbb{P}})$. Then

1. no label occurs more than once in Q,

2. $w \in \mathsf{LAB}(Q)$ implies $u \leq w$.

Central to labelling is the persistence and disappearance of labels under derivation. In particular, once a label disappears it can never reappear. It is these features which allow us to recognize when a component contributes to the performance of an action.

Fact 3.5 Let $Q \in \mathsf{L}_u(\tilde{\mathbb{P}})$ and $\alpha, \alpha_1, \ldots \alpha_n \in \mathbb{A}_\tau$.

1. $Q \xrightarrow{\alpha} Q'$ implies $Q' \in \mathsf{L}_v(\tilde{\mathbb{P}})$ with $u \leq v$.

2. $Q \xrightarrow{\alpha_1} Q_1 \xrightarrow{\alpha_2} \ldots \xrightarrow{\alpha_n} Q_n$ implies $Q_i \in \mathsf{L}_{v_i}(\tilde{\mathbb{P}})$ with $u \leq v_i$. Moreover, if $w \in \mathsf{LAB}$ such that $w < u$ then $w \notin \mathsf{LAB}(Q_i)$.

Fact 3.6 Let $Q_0 \in \mathsf{L}(\tilde{\mathbb{P}})$. If $Q_0 \xrightarrow{\alpha_1} Q_1 \xrightarrow{\alpha_2} \ldots \xrightarrow{\alpha_i} Q_i \xrightarrow{\alpha_{i+1}} \ldots \xrightarrow{\alpha_n} Q_n$ and $v \in \mathsf{LAB}(Q_0) \cap \mathsf{LAB}(Q_n)$ then $v \in \mathsf{LAB}(Q_i)$, for every $i \in [0, n]$.

Throughout the rest of this section we assume the labelled calculus. However, whenever possible, labels will be left implicit to keep the notation simple (as, for instance, in proofs of statements that do not explicitly deal with labels in processes) – and the same applies for the treatment of labelled processes in the next section.

3.2 Live events

To capture the fairness constraint for execution sequences, we need to define the *live events*. For a process like $\alpha_u.\mathsf{nil}\|_{\{\alpha\}} \alpha_v.\mathsf{nil}$ (with labels u and v), there is only one live action. This is action α; it can be performed in only one way, i.e. there is only one α-event, which we will identify with the tuple $\langle u, v \rangle$, i.e. with the tuple of labels of 'local' α's that synchronize when the process performs α; recall that we call such tuples *event labels*.[4] In a similar way, there is only one live action in $\alpha_u.\beta_v.\mathsf{nil} \|_{\{\beta\}} \beta_y.\mathsf{nil}$ (action α corresponding to tuple $\langle u \rangle$) because the parallel composition prevents the instance of β labelled by $\langle y \rangle$ from contributing an action. However, note that $\langle v, y \rangle$ becomes live, once action α is performed.

We now define $\mathsf{LE}(Q, A)$ as the set of live events of Q (when the execution of actions in A are prevented by the environment). Again for technical reasons, we allow τ to be part of the set A.

Definition 3.7 (*live events*)
Let $Q \in \mathsf{L}(\tilde{\mathbb{P}})$ and $A \subseteq \mathbb{A}_\tau$. The set $\mathsf{LE}(Q, A)$ is defined by induction on Q.

---

[4]Since Costa and Stirling deal with fairness of components, they have no need for tuples.

Nil, Var:	$\mathsf{LE}(\mathsf{nil}_u, A) = \mathsf{LE}(x_u, A) = \emptyset$

Pref: $\mathsf{LE}(\mu_u.P, A) = \begin{cases} \{\langle u \rangle\} & \text{if } \mu = \alpha \text{ or } \mu = \underline{\alpha} \text{ and } \alpha \notin A \\ \emptyset & \text{otherwise} \end{cases}$

Sum: $\mathsf{LE}(Q_1 +_u Q_2, A) = \mathsf{LE}(Q_1, A) \cup \mathsf{LE}(Q_2, A)$

Par: $\mathsf{LE}(Q \parallel_B^u Q_1, A) = \mathsf{LE}(Q_1, A \cup B) \cup \mathsf{LE}(Q_1, A \cup B) \cup$
$\bigcup_{\alpha \in B \setminus A} (\, \mathsf{LE}(Q_1, \mathbb{A}_\tau \setminus \{\alpha\}) \times \mathsf{LE}(Q_2, \mathbb{A}_\tau \setminus \{\alpha\}) \,)$

Rel: $\mathsf{LE}(Q[\Phi_u], A) = \mathsf{LE}(Q, \Phi^{-1}(A))$

Rec: $\mathsf{LE}(\mathsf{rec}\ x_u.Q, A) = \mathsf{LE}(Q, A)$

The *set of live events* in Q is defined as $\mathsf{LE}(Q, \emptyset)$ which we abbreviate to $\mathsf{LE}(Q)$.

As for the definition of activated actions, the set A represents the restricted actions. Then, $\mathsf{LE}(a_u.P, \{a\})$ must be empty because the action a is prevented. Note that, in the Par-case, $\mathsf{LE}(Q_1, A \cup B) \cup \mathsf{LE}(Q_2, A \cup B)$ is the set of the labels of the live actions of Q_1 and Q_2, when the environment prevents actions from A and from the synchronization set B – corresponding to those actions that Q_1 and Q_2 can perform independently. To properly deal with synchronization, for all $\alpha \in B \setminus A$ we combine each live event of Q_1 corresponding to α with each live event of Q_2 corresponding to α, getting tuples of labels.

An important subset of the live events of a process Q is the subset of urgent live events, those that cannot be delayed anymore.

Definition 3.8 (*urgent live events*)
Let $Q \in \mathsf{L}(\tilde{\mathbb{P}})$ and $A \subseteq \mathbb{A}_\tau$. The set $\mathsf{UE}(Q, A)$ is defined as in Definition 3.7 when $\mathsf{LE}(\_)$ is replaced by $\mathsf{UE}(\_)$ and rule Pref is replaced by the following one:

Pref: $\mathsf{UE}(\mu_u.P, A) = \begin{cases} \{\langle u \rangle\} & \text{if } \mu = \underline{\alpha} \text{ and } \alpha \notin A \\ \emptyset & \text{otherwise} \end{cases}$

Again, define $\mathsf{UE}(Q) = \mathsf{UE}(Q, \emptyset)$.

An easy observation is the following lemma.

Lemma 3.9 Let Q be a labelled process term. Then:

1. $\mathsf{UE}(Q, A) \subseteq \mathsf{LE}(Q, A)$, for every $A \subseteq \mathbb{A}_\tau$.

2. $\langle v_1, \ldots, v_n \rangle \in \mathsf{LE}(Q)$ implies $v_i \in \mathsf{LAB}(Q)$, for every $i \in [1, n]$.

3. $Q \in \mathsf{L}(\tilde{\mathbb{P}}_1)$ implies $\mathsf{UE}(Q, A) = \emptyset$, for every $A \subseteq \mathbb{A}_\tau$.

Example 3.10 Let $P = (Q \parallel_{\{b\}}^1 b_{12}.\mathsf{nil}) \parallel_{\{b\}}^\varepsilon b_2.\mathsf{nil}$, where $Q = \mathsf{rec}\ x_{11}.(a_{v1}.x +_v b_{v2}.\mathsf{nil})$ and $v = 111$. Let $A_\alpha = \mathsf{LE}(Q \parallel_{\{b\}}^1 b_{12}.\mathsf{nil}, \mathbb{A}_\tau \setminus \{\alpha\}) \times \mathsf{LE}(b_2.\mathsf{nil}, \mathbb{A}_\tau \setminus \{\alpha\})$. Then, by definition,

$$\mathsf{LE}(P) = \mathsf{LE}(Q \parallel_{\{b\}}^1 b_{12}.\mathsf{nil}, \{b\}) \cup \mathsf{LE}(b_2.\mathsf{nil}, \{b\}) \cup \bigcup_{\alpha \in \{b\}} A_\alpha.$$

We now determine these three subsets of $\mathsf{LE}(P)$.

$\mathsf{LE}(Q \parallel^1_{\{b\}} b_{12}.\mathsf{nil}, \{b\}) =$
$\mathsf{LE}(Q, \{b\}) \cup \mathsf{LE}(b_{12}.\mathsf{nil}, \{b\}) \cup \bigcup_{\alpha \in \emptyset}(\mathsf{LE}(Q, \mathbb{A}_\tau \backslash \{\alpha\}) \times \mathsf{LE}(b_{12}.\mathsf{nil}, \mathbb{A}_\tau \backslash \{\alpha\}) =$
$\mathsf{LE}(Q, \{b\}) = \mathsf{LE}(a_{v1}.x +_v b_{v2}.\mathsf{nil}, \{b\}) = \{\langle v1 \rangle\}.$

$\mathsf{LE}(b_2.\mathsf{nil}, \{b\}) = \emptyset.$

Since

$\mathsf{LE}(Q\parallel^1_{\{b\}}b_{12}.\mathsf{nil}, \mathbb{A}_\tau\backslash\{b\}) =$
$\mathsf{LE}(Q, \mathbb{A}_\tau) \cup \mathsf{LE}(b_{12}.\mathsf{nil}, \mathbb{A}_\tau) \cup (\mathsf{LE}(Q, \mathbb{A}_\tau\backslash\{b\}) \times \mathsf{LE}(b_{12}.\mathsf{nil}, \mathbb{A}_\tau\backslash\{b\})) =$
$\mathsf{LE}(a_{v1}.x +_v b_{v2}.\mathsf{nil}, \mathbb{A}_\tau\backslash\{b\}) \times \{\langle 12 \rangle\} =$
$\{\langle v2 \rangle\} \times \{\langle 12 \rangle\} = \{\langle v2, 12 \rangle\}$ and

$\mathsf{LE}(b_2.\mathsf{nil}, \mathbb{A}\backslash\{b\}) = \{\langle 2 \rangle\}$ we have that

$\bigcup_{\alpha \in \{b\}} A_\alpha = A_b = \mathsf{LE}(Q\parallel^1_{\{b\}}b_{12}.\mathsf{nil}, \mathbb{A}_\tau\backslash\{b\}) \times \mathsf{LE}(b_2.\mathsf{nil}, \mathbb{A}_\tau\backslash\{b\}) =$
$\{\langle v2, 12 \rangle\} \times \{\langle 2 \rangle\} = \{\langle v2, 12, 2 \rangle\}$

Finally $\mathsf{LE}(P) = \{\langle v1 \rangle\} \cup \{\langle v2, 12, 2 \rangle\} = \{\langle v1 \rangle, \langle v2, 12, 2 \rangle\}.$

In the rest of this section we just state some properties that will be useful to prove our main correspondence results. Detailed proofs have been moved to sections in the appendix. We start with a proposition relating labels and (functional and temporal) transitions. In the case of functional transitions, if an event label is urgent and live in the source process, then either the label preserves its status in the target one or one of its constituents disappears (a similar statement would hold for live events in place of urgent ones). In the case of temporal transitions the set of live events of the source state coincides with the set of live events of the target one. In addition, after the temporal move all live events become urgent. This statement will be used to prove Proposition 4.1.

Proposition 3.11 Let $Q, Q' \in \mathsf{L}(\tilde{\mathbb{P}})$ and $A \subseteq \mathbb{A}_\tau$. Then:

1. $Q \xrightarrow{\alpha} Q'$ and $s = \langle v_1, \ldots, v_n \rangle \in \mathsf{UE}(Q, A)$ implies either $s \in \mathsf{UE}(Q', A)$ or there exists some $j \in [1, n]$ such that $v_j \notin \mathsf{LAB}(Q')$.

2. $Q \xrightarrow{X}_r Q'$ implies $\mathsf{LE}(Q, A) = \mathsf{LE}(Q', A) = \mathsf{UE}(Q', A)$.

The next proposition relates full time-steps and urgent activated actions. A process term can perform a full time-step only if it does not have any pending urgent actions, and vice versa for a guarded process term. Moreover, it shows how urgent activated actions and urgent live events are strictly related. This statement will be used to prove Proposition 4.2 and 4.5.

Proposition 3.12 Let $Q \in \mathsf{L}(\tilde{\mathbb{P}})$ and $A \subseteq \mathbb{A}_\tau$. Then:

1. $Q \xrightarrow{1}$ implies $\mathcal{U}(Q) = \emptyset$;

2. Q guarded and $\mathcal{U}(Q) = \emptyset$ implies $Q \xrightarrow{1}$;

14

3. $\mathcal{U}(Q, A) = \emptyset$ if and only if $\mathsf{UE}(Q, A) = \emptyset$.

The following proposition states that process terms that are able to perform two subsequent time steps cannot exhibit any functional behaviour. Moreover, if a term cannot make any functional move (and is guarded), then it can let two time steps pass. Intuitively, this captures the functional deadlock of terms. Terms that cannot exhibit any functional behaviour can let any amount of time pass. The following proposition formalizes this intuition. It will be used to prove Proposition 4.6.

Proposition 3.13 Let $Q, Q', Q'' \in \mathsf{L}(\tilde{\mathbb{P}})$.

1. $Q \xrightarrow{1} Q' \xrightarrow{1} Q''$ implies $Q \not\xrightarrow{\alpha}$ and $Q' \not\xrightarrow{\alpha}$ for any $\alpha \in \mathbb{A}_\tau$. Moreover $Q' = Q''$;

2. Q guarded and $Q \not\xrightarrow{\alpha}$ for any $\alpha \in \mathbb{A}_\tau$ implies $Q \xrightarrow{1} Q' \xrightarrow{1} Q'$

3.3 Fair execution sequences

We can now define the (weak) fairness constraint. The following definitions and results are essentially borrowed from [8], and just adapted to our notions of fairness and labelling. First of all, for a *process* P_0, we say that a sequence of transitions $\gamma = P_0 \xrightarrow{\lambda_0} P_1 \xrightarrow{\lambda_1} \ldots$ with $\lambda_i \in \mathbb{A}_\tau \cup \{1\}$ is a *timed execution sequence* if it is an infinite sequence of action transitions and full time-steps; note that a maximal sequence of such transitions/steps is never finite, since for $\gamma = P_0 \xrightarrow{\lambda_0} P_1 \xrightarrow{\lambda_1} \ldots \xrightarrow{\lambda_{n-1}} P_n$, we have $P_n \xrightarrow{\alpha}$ or $P_n \xrightarrow{1}$ by Proposition 3.13. The second part of this proposition is applicable, since processes are always guarded.

Note that a timed execution sequence is *everlasting* in the sense of having infinitely many time steps if and only if it is *non-Zeno*; a Zeno run would have infinitely many actions in a finite amount of time, which in a setting with discrete time means exactly that it ends with infinitely many action transitions without a time step.

For an *initial* process P_0, we say that a sequence of transitions $\gamma = P_0 \xrightarrow{\alpha_0} P_1 \xrightarrow{\alpha_1} \ldots$ with $\alpha_i \in \mathbb{A}_\tau$ is an *execution sequence* if it is a maximal sequence of action transitions; i.e. it is infinite or ends with a process P_n such that $P_n \not\xrightarrow{\alpha}$ for any action α.

Now we formalize fairness by calling a (timed) execution sequence *fair*, if no event becomes live and then remains live throughout.

Definition 3.14 (*fair execution sequences*)

Let $\gamma = P_0 \xrightarrow{\lambda_0} P_1 \xrightarrow{\lambda_1} \ldots$ be an execution sequence or a timed execution sequence; we will write '(timed) execution sequence' for such a sequence. We say that γ is *fair* if

$$\neg(\exists\, s\, \exists\, i\, .\, \forall\, k \geq i\, :\, s \in \mathsf{LE}(P_k))$$

Following [8], we now present an alternative, more local, definition of fair computations which will be useful to prove our main statements. In the following, we use $|\gamma|$ to denote the length – i.e. the number of processes – of a (timed) execution sequence γ, which is ∞ if γ is an infinite computation.

Definition 3.15 Let $\gamma = P_0 \xrightarrow{\lambda_0} P_1 \xrightarrow{\lambda_1} \ldots$ be a (timed) execution sequence. We say that

i. γ is *l-fair* at i if there exists $j \geq i$ such that

$$\mathsf{LE}(P_i) \cap \mathsf{LE}(P_{i+1}) \cap \ldots \cap \mathsf{LE}(P_j) = \emptyset$$

ii. γ is *l-fair* if for all $i < |\gamma|$ we have that γ is l-fair at i.

γ is *l-fair* at i when every live event in P_i loses its liveness. The following theorem states that a (timed) execution sequence is l-fair at every i if and only if it is fair.

Theorem 3.16 A (timed) execution sequence $\gamma = P_0 \xrightarrow{\lambda_0} P_1 \xrightarrow{\lambda_1} \ldots$ is l-fair if and only if it is fair.

Proof: We prove that γ is not l-fair if and only if it is not fair.

1. Assume γ not l-fair. Then there exists $i < |\gamma|$ such that for all $j \geq i$, $\mathsf{LE}(P_i) \cap \mathsf{LE}(P_{i+1}) \cap \ldots \cap \mathsf{LE}(P_j) \neq \emptyset$.[5] Then, there is some s such that for all $j \geq i$, $s \in \mathsf{LE}(P_j)$.

2. Vice versa, if γ is not fair, then there are i and s such that, for all $k \geq i$, $s \in \mathsf{LE}(P_k)$, i.e. γ is not l-fair at i and, hence, γ is not l-fair.

$\square$

As remarked in [8], this alternative definition of fairness allows us to think of fairness in terms of a localizable property and not just as a property of (timed) execution sequences as a whole. Starting from P_0 we can generate a derivation $P_0 \xrightarrow{\lambda_0} P_1 \ldots \xrightarrow{\lambda_{n-1}} P_n$ which satisfies l-fairness at 0, i.e. such that $\mathsf{LE}(P_0) \cap \mathsf{LE}(P_1) \cap \ldots \cap \mathsf{LE}(P_n) = \emptyset$. One then continues by generating a derivation $P_n \xrightarrow{\lambda_n} P_{n+1} \ldots \xrightarrow{\lambda_{m-1}} P_m$ which satisfies l-fairness at n. The concatenation of these two derivations guarantees l-fairness at any $j \leq n$. In this way we can generate only fair sequences. The following definition formalizes this strategy.

Definition 3.17 (*B-step*)

For an process P_0, we say that $P_0 \xrightarrow{\lambda_0} P_1 \xrightarrow{\lambda_1} \ldots \xrightarrow{\lambda_{n-1}} P_n$ with $n > 0$ is a *timed B-step* when

i. B is a finite set of event labels,

ii. $B \cap \mathsf{LE}(P_0) \cap \ldots \cap \mathsf{LE}(P_n) = \emptyset$.

If $\lambda_i \in \mathbb{A}_\tau$, $i = 0, \ldots, n-1$, then the sequence is a *B-step*.
If $P_0 \xrightarrow{\lambda_0} P_1 \xrightarrow{\lambda_1} \ldots \xrightarrow{\lambda_{n-1}} P_n$ is a (timed) *B*-step and $v = \lambda_0 \ldots \lambda_{n-1}$ we write $P_0 \xrightarrow{v}_B P_{n+1}$.

In particular, a (timed) $\mathsf{LE}(P)$-step from P is "locally" fair: all live events of P lose their liveness at some point in the step.

Definition 3.18 (*fair-step sequences*)

A *(timed) fair-step sequence* from P_0 is any maximal sequence of (timed) steps of the form

$$P_0 \xrightarrow{v_0}_{\mathsf{LE}(P_0)} P_1 \xrightarrow{v_1}_{\mathsf{LE}(P_1)} \cdots$$

---

[5]Note that $\mathsf{LE}(P_i)$ is finite.

A fair-step sequence is simply a concatenation of locally fair steps. If δ is a (timed) fair-step sequence, then its *associated* (timed) execution sequence is the sequence which drops all references to the sets $\mathsf{LE}(P_i)$.

Theorem 3.19 A (timed) execution sequence is l-fair if and only if it is the sequence associated with a (timed) fair-step sequence.

Proof: Assume $\gamma = P_0 \xrightarrow{\lambda_0} P_1 \xrightarrow{\lambda_1} \ldots$ l-fair. By definition, γ is l-fair at 0 and, hence, there exists $j \geq 0$ such that $\mathsf{LE}(P_0) \cap \ldots \cap \mathsf{LE}(P_j) = \emptyset$. Then, for $v_0 = \lambda_0 \ldots \lambda_{j-1}$, we have that $P_0 \xrightarrow{v_0}_{\mathsf{LE}(P_0)} P_j$. Since γ is l-fair at j for any $j < |\gamma|$, we can iterate this strategy and generate a fair-step sequence $\gamma' = P_0 \xrightarrow{v_0}_{\mathsf{LE}(P_0)} P_j \xrightarrow{v_1}_{\mathsf{LE}(P_j)} \ldots$ Clearly, γ is the execution sequence associated with γ'.

Vice versa, assume that $\gamma' = P_0 \xrightarrow{v_0}_{\mathsf{LE}(P_0)} P_1 \xrightarrow{v_1}_{\mathsf{LE}(P_1)} \ldots$ is a fair-step sequence. Let $v_0 = \lambda_0 \ldots \lambda_{j-1}$, $v_1 = \lambda_j \ldots \lambda_{k-1}$ and so on. The execution sequence associated with γ' is
$$\gamma = P_0 \xrightarrow{\lambda_0} P'_1 \xrightarrow{\lambda_1} \ldots \xrightarrow{\lambda_{j-1}} P'_j = P_1 \xrightarrow{\lambda_j} P'_{j+1} \xrightarrow{\lambda_{j+1}} \ldots \xrightarrow{\lambda_{k-1}} P'_k = P_2 \ldots$$
The definition of a B-step implies that γ is l-fair at j and k etc. and, thus, fair at any $i \leq j$ and $i \leq k$ etc.; therefore, γ is l-fair for any $i < |\gamma|$ and, hence, it is l-fair. $\square$

Now we have the obvious corollary that combines Theorems 3.16 and 3.19 to show that fair execution sequences and fair-step sequences are essentially the same.

Corollary 3.20 A (timed) execution sequence is fair if and only if it is the sequence associated with a (timed) fair-step sequence.

4   Fairness and Timing

This section is the core of the paper. It relates fairness and timing in a process algebraic setting, and it contains three main contributions:

(i) We prove that all everlasting (i.e. non-Zeno) sequences of PAFAS processes are fair.

(ii) We provide a characterization of fair execution sequences of *initial* PAFAS processes (PAFAS processes evolving only via functional operational semantics) in terms of *timed* execution sequences.

(iii) For the case of a finite state process, we derive from this a finite representation of the fair runs with a transition system that has arcs labelled by regular expressions.

4.1   Fairness of everlasting sequences

The following proposition is a key statement for proving that everlasting timed execution sequences of PAFAS processes are fair. It relates time steps, urgent live events and live events.

Proposition 4.1 Let Q be a labelled process term. Then: $Q \xrightarrow{X}_r Q_1 \xrightarrow{\alpha_1} \ldots \xrightarrow{\alpha_{n-1}} Q_n$, where $X \subseteq \mathbb{A}$ and $\alpha_1, \ldots, \alpha_{n-1} \in \mathbb{A}_\tau$, implies $\mathsf{LE}(Q_1) \cap (\mathsf{LE}(Q_n) \backslash \mathsf{UE}(Q_n)) = \emptyset$ (and by Proposition 3.11 also $\mathsf{LE}(Q) \cap (\mathsf{LE}(Q_n) \backslash \mathsf{UE}(Q_n)) = \emptyset$).

Proof: Assume, by contradiction, that there exists a tuple of labels $s = \langle v_1, \ldots, v_m \rangle$ such that $s \in \mathsf{LE}(Q_1) = \mathsf{UE}(Q_1)$ (by Proposition 3.11-2) and $s \in \mathsf{LE}(Q_n)\backslash\mathsf{UE}(Q_n)$. Lemma 3.9-2 and $s \in \mathsf{LE}(Q_1)$ imply $v_1, \ldots, v_m \in \mathsf{LAB}(Q_1)$. On the other hand, since $s \in \mathsf{UE}(Q_1)$ but $s \notin \mathsf{UE}(Q_n)$, we can find j, $1 \le j < n$, such that $s \in \mathsf{UE}(Q_j)$ and $s \notin \mathsf{UE}(Q_{j+1})$. Then, by Proposition 3.11-1, there exists some $k \in [1, m]$ such that $v_k \notin \mathsf{LAB}(Q_{j+1})$. By Fact 3.6, $v_k \in \mathsf{LAB}(Q_1)$ and $v_k \notin \mathsf{LAB}(Q_{j+1})$ implies $v_k \notin \mathsf{LAB}(Q_i)$, for every $i \in [j+1, n]$ and, again by Lemma 3.9, $s \notin \mathsf{LE}(Q_i)$, for every $i \in [j+1, n]$. In particular, $s \notin \mathsf{LE}(Q_n)$ which contradicts the assumption $s \in \mathsf{LE}(Q_n)\backslash\mathsf{UE}(Q_n)$. $\qquad\square$

To prove the main statement of this section, another preliminary proposition is needed.

Proposition 4.2 Let $Q \in \mathsf{L}(\tilde{\mathbb{P}})$ and $v, w \in (\mathbb{A}_\tau)^*$.

1. If $Q \xrightarrow{1} Q_1 \xrightarrow{v} Q_2 \xrightarrow{1}$ then $Q \xrightarrow{1v}_{\mathsf{LE}(Q)} Q_2$;

2. If $Q \xrightarrow{v} Q' \xrightarrow{1} Q_1' \xrightarrow{w} Q_2' \xrightarrow{1}$ then $Q \xrightarrow{v1w}_{\mathsf{LE}(Q)} Q_2'$.

Proof:

1. Assume that $Q \xrightarrow{1} Q_1 \xrightarrow{v} Q_2$. Proposition 4.1 implies $\mathsf{LE}(Q) \cap (\mathsf{LE}(Q_2)\backslash\mathsf{UE}(Q_2)) = \emptyset$. Moreover $Q_2 \xrightarrow{1}$ and Proposition 3.12 imply that $\mathcal{U}(Q_2) = \emptyset$ and $\mathsf{UE}(Q_2) = \emptyset$. Thus $\mathsf{LE}(Q) \cap \mathsf{LE}(Q_1) \cap \ldots \cap \mathsf{LE}(Q_2) \subseteq \mathsf{LE}(Q) \cap \mathsf{LE}(Q_2) = \mathsf{LE}(Q) \cap (\mathsf{LE}(Q_2)\backslash\mathsf{UE}(Q_2)) = \emptyset$. By the definition of a timed B-step, $Q \xrightarrow{1v}_{\mathsf{LE}(P)} Q_2$.

2. This follows immediately from 1. and the definition of a timed B-step.

$\qquad\square$

Theorem 4.3 Each everlasting timed execution sequence, i.e. each timed execution sequence of the form

$$\gamma = P_0 \xrightarrow{v_0} P_1 \xrightarrow{1} P_2 \xrightarrow{v_1} P_3 \xrightarrow{1} P_4 \xrightarrow{v_2} P_5 \xrightarrow{1} \ldots$$

with $v_0, v_1, v_2 \ldots \in (\mathbb{A}_\tau)^*$ is fair.

Proof: By Proposition 4.2 we have that $P_0 \xrightarrow{v_0 1 v_1}_{\mathsf{LE}(P_0)} P_3$, $P_3 \xrightarrow{1v_2}_{\mathsf{LE}(P_3)} P_5$ and so on. Then γ is a sequence associated with a timed fair-step sequence and is fair by Corollary 3.20. $\quad\square$

Observe that an everlasting timed execution sequence, by its definition, does not depend on the labelling, i.e. it is a notion of the *unlabelled* PAFAS calculus.

4.2 Relating Timed Executions and Fair Executions

While in the previous section we have shown that every everlasting timed execution is fair, we show in this section that everlasting timed execution sequences of initial PAFAS processes in fact *characterize* the fair untimed executions of these processes. Observe that the latter is a notion of an ordinary *labelled untimed* process algebra (like CCS or TCSP), while the former is a notion of our *unlabelled timed* process algebra.

The key statement for proving this relates B-steps and action sequences performed between two full time-steps. More in detail, we prove that whenever an initial process P can perform a sequence v of basic actions and this execution turns out to be an $\mathsf{LE}(P)$-step, then P can alternatively let time pass (perform a 1-time step) and then perform the sequence of basic actions v, and vice versa.

The following proposition relates live events and transitional properties of terms in $\tilde{\mathbb{P}}_1$ and their "marked" version. The proof and related results are moved in Appendix E.

Proposition 4.4 Let $Q \in \mathsf{L}(\tilde{\mathbb{P}})$ and $P \in \mathsf{L}(\tilde{\mathbb{P}}_1)$ such that $P = \mathsf{unmark}(Q)$. Then:

1. $\mathsf{LE}(Q, A) = \mathsf{LE}(P, A)$ for every A;

2. $Q \xrightarrow{\alpha} Q'$ implies $P \xrightarrow{\alpha} P'$ and $P' = \mathsf{unmark}(Q')$. Moreover $\mathsf{UE}(Q', A) \subseteq \mathsf{UE}(Q, A)$ and $\mathsf{UE}(Q') = \emptyset$ implies $Q' = P'$;

3. $P \xrightarrow{\mu} P'$ implies $Q \xrightarrow{\mu} Q'$ and $P' = \mathsf{unmark}(Q')$.

Now we are ready to present our key proposition relating LE-step and temporal transitions.

Proposition 4.5 Let $P_0 \in \mathsf{L}(\mathbb{P}_1)$ and $v \in (\mathbb{A}_\tau)^+$. Then:

1. $P_0 \xrightarrow{v}_{\mathsf{LE}(P_0)} P_n$ implies $P_0 \xrightarrow{1} Q_0 \xrightarrow{v} P_n$;

2. $P_0 \xrightarrow{1} Q_0 \xrightarrow{v} Q_n \xrightarrow{1}$ implies $Q_n = P_n \in \mathsf{L}(\mathbb{P}_1)$ and $P_0 \xrightarrow{v}_{\mathsf{LE}(P_0)} P_n$.

Proof: Let $v = \alpha_1 \ldots \alpha_n$, and let us prove Item 1. By definition $P_0 \xrightarrow{v}_{\mathsf{LE}(P_0)} P_n$ implies $P_0 \xrightarrow{\alpha_1} \ldots \xrightarrow{\alpha_n} P_n$ and $\mathsf{LE}(P_0) \cap \ldots \cap \mathsf{LE}(P_n) = \emptyset$. Since $P_0 \in \mathsf{L}(\mathbb{P}_1)$, we have that $P_0 \xrightarrow{1} Q_0$ with $P_0 = \mathsf{unmark}(Q_0)$ (see Proposition 7.8-3) and, by Proposition 4.4-3 and 4.4-1, $Q_0 \xrightarrow{\alpha_0} \ldots \xrightarrow{\alpha_n} Q_n$ with $P_i = \mathsf{unmark}(Q_i)$ and $\mathsf{LE}(P_i) = \mathsf{LE}(Q_i)$ for every $i \in [0, n]$. Then $\mathsf{LE}(Q_0) \cap \ldots \cap \mathsf{LE}(Q_n) = \emptyset$ and, since $\mathsf{UE}(S) \subseteq \mathsf{LE}(S)$ for a generic S (Lemma 3.9), also $\mathsf{UE}(Q_0) \cap \ldots \cap \mathsf{UE}(Q_n) = \emptyset$. Moreover, by Proposition 4.4-2, $\mathsf{UE}(Q_{i+1}) \subseteq \mathsf{UE}(Q_i)$ for $i \in [0, n-1]$. Thus $\mathsf{UE}(Q_n) = \emptyset$ and, again by Proposition 4.4-2, $Q_n = P_n$. We get $P_0 \xrightarrow{1} Q_0 \xrightarrow{v} P_n$.

Now we prove Item 2. Assume $P_0 \xrightarrow{1} Q_0 \xrightarrow{v} Q_n \xrightarrow{1}$. By Proposition 4.1 we have that $\mathsf{LE}(Q_0) \cap (\mathsf{LE}(Q_n) \backslash \mathsf{UE}(Q_n)) = \emptyset$. Moreover $Q_n \xrightarrow{1}$ and Propositions 3.12-1 and 3.12-3 imply $\mathsf{UE}(Q_n) = \emptyset$ and, hence, $\mathsf{LE}(Q_0) \cap \mathsf{LE}(Q_n) = \emptyset = \mathsf{LE}(Q_0) \cap \ldots \cap \mathsf{LE}(Q_n)$. Now, by Propositions 7.8-3, 4.4-2 and 4.4-3 we have that $P_0 \xrightarrow{\alpha_0} \ldots \xrightarrow{\alpha_n} P_n$ with $P_i = \mathsf{unmark}(Q_i)$, $\mathsf{LE}(P_i) = \mathsf{LE}(Q_i)$ for every $i \in [0, n]$ and $Q_n = P_n$. Thus, by definition, $P_0 \xrightarrow{v}_{\mathsf{LE}(P_0)} P_n$. $\square$

Iterative applications of Proposition 4.5 prove the main theorems of this section. To present our characterization results we distinguish between finite and infinite sequences of untimed processes.

Proposition 4.6 Let $P \in \mathsf{L}(\mathbb{P}_1)$ and $v_0, v_1, \ldots \in (\mathbb{A}_\tau)^+$. Then:

1. For any finite fair-step sequence from P

$$P = P_0 \xrightarrow{v_0}_{\mathsf{LE}(P_0)} P_1 \xrightarrow{v_1}_{\mathsf{LE}(P_1)} P_2 \ldots P_{n-1} \xrightarrow{v_{n-1}}_{\mathsf{LE}(P_{n-1})} P_n$$

there exists a timed execution sequence

$$P = P_0 \xrightarrow{1} Q_0 \xrightarrow{v_0} P_1 \xrightarrow{1} Q_1 \xrightarrow{v_1} P_2 \ldots P_{n-1} \xrightarrow{1} Q_{n-1} \xrightarrow{v_{n-1}} P_n \xrightarrow{1} Q_n \xrightarrow{1} Q_n \ldots$$

19

2. For any timed execution sequences

$$P = P_0 \xrightarrow{1} Q_0 \xrightarrow{v_0} P_1 \xrightarrow{1} Q_1 \xrightarrow{v_1} P_2 \ldots P_{n-1} \xrightarrow{1} Q_{n-1} \xrightarrow{v_{n-1}} P_n \xrightarrow{1} Q_n \xrightarrow{1} Q_n \ldots$$

there exists a finite fair-step sequence from P

$$P = P_0 \xrightarrow{v_0}_{\mathsf{LE}(P_0)} P_1 \xrightarrow{v_1}_{\mathsf{LE}(P_1)} P_2 \ldots P_{n-1} \xrightarrow{v_{n-1}}_{\mathsf{LE}(P_{n-1})} P_n$$

Proof:

1. Assume $P_0 \xrightarrow{v_0}_{\mathsf{LE}(P_0)} P_1 \xrightarrow{v_1}_{\mathsf{LE}(P_1)} P_2 \ldots P_{n-1} \xrightarrow{v_{n-1}}_{\mathsf{LE}(P_{n-1})} P_n$ and $P_n \xrightarrow{\alpha}\!\!\!\!\!/\;$ for any α. By iterative applications of Proposition 4.5-1 we can prove that $P_i \xrightarrow{1} Q_i \xrightarrow{v_i} P_{i+1}$ for $i \in [0, n-1]$. Moreover $P_n \in \mathsf{L}(\mathbb{P}_1)$ (and hence P_n guarded) and $P_n \xrightarrow{\alpha}\!\!\!\!\!/\;$ for any $\alpha \in \mathbb{A}_\tau$ imply, by Proposition 3.13-2, $P_n \xrightarrow{1} Q_n \xrightarrow{1} Q_n$.

2. Assume $P = P_0 \xrightarrow{1} Q_0 \xrightarrow{v_0} P_1 \xrightarrow{1} Q_1 \xrightarrow{v_1} P_2 \ldots P_{n-1} \xrightarrow{1} Q_{n-1} \xrightarrow{v_{n-1}} P_n \xrightarrow{1} Q_n \xrightarrow{1} Q_n \ldots$ Then, by iterative applications of Proposition 4.5-2, we can prove that $P = P_0 \xrightarrow{v_0}_{\mathsf{LE}(P_0)} P_1 \xrightarrow{v_1}_{\mathsf{LE}(P_1)} P_2 \ldots P_{n-1} \xrightarrow{v_{n-1}}_{\mathsf{LE}(P_{n-1})} P_n$. Moreover $P_n \xrightarrow{1} Q_n \xrightarrow{1} Q_n$ and Proposition 3.13-1 imply $P_n \xrightarrow{\alpha}\!\!\!\!\!/\;$ for any $\alpha \in \mathbb{A}_\tau$.

$\square$

Similarly we can prove an analogous result for infinite sequences.

Proposition 4.7 Let $P \in \mathsf{L}(\mathbb{P}_1)$ and $v_0, v_1, \ldots \in (\mathbb{A}_\tau)^+$. Then:

1. For any infinite fair-step sequence from P

$$P = P_0 \xrightarrow{v_0}_{\mathsf{LE}(P_0)} P_1 \xrightarrow{v_1}_{\mathsf{LE}(P_1)} P_2 \ldots P_i \xrightarrow{v_i}_{\mathsf{LE}(P_i)} P_{i+1} \ldots$$

there exists a timed execution sequence

$$P = P_0 \xrightarrow{1} Q_0 \xrightarrow{v_0} P_1 \xrightarrow{1} Q_1 \xrightarrow{v_1} P_2 \ldots P_i \xrightarrow{1} Q_i \xrightarrow{v_i} P_{i+1} \ldots$$

2. For any timed execution sequences

$$P = P_0 \xrightarrow{1} Q_0 \xrightarrow{v_0} P_1 \xrightarrow{1} Q_1 \xrightarrow{v_1} P_2 \ldots P_i \xrightarrow{1} Q_i \xrightarrow{v_i} P_{i+1} \ldots$$

there exists a finite fair-step sequence from P

$$P = P_0 \xrightarrow{v_0}_{\mathsf{LE}(P_0)} P_1 \xrightarrow{v_1}_{\mathsf{LE}(P_1)} P_2 \ldots P_i \xrightarrow{v_i}_{\mathsf{LE}(P_i)} P_{i+1} \ldots$$

By Proposition 3.3 we can also remove the labels from processes P_i, Q_i in the timed computation, and by Corollary 3.20 we can replace fair-step sequences by fair execution sequences. This way, we obtain a similar correspondence result between fair executions of labelled PAFAS and timed executions of unlabelled PAFAS. This will actually be our first main result, which we now derive in two stages.

As above and for each stage, we first state the correspondence result for finite fair-step sequences and then we vary it for the infinite ones.

Theorem 4.8 (*Characterization of finite fair-step sequences*)
  Let $P \in \mathsf{L}(\mathbb{P}_1)$ and $v_0, v_1, v_2 \ldots \in (\mathbb{A}_\tau)^+$. Then:

1. For any finite fair-step sequence from P

$$P = P_0 \xrightarrow[\mathsf{LE}(P_0)]{v_0} P_1 \xrightarrow[\mathsf{LE}(P_1)]{v_1} P_2 \ldots P_{n-1} \xrightarrow[\mathsf{LE}(P_{n-1})]{v_{n-1}} P_n$$

   there exists a timed execution sequence in unlabelled PAFAS

$$\mathsf{R}(P) = S_0 \xrightarrow{1} S_0' \xrightarrow{v_0} S_1 \xrightarrow{1} S_1' \xrightarrow{v_1} S_2 \ldots S_n \xrightarrow{1} S_n' \xrightarrow{1} S_n' \xrightarrow{1} \ldots$$

   where $S_i = \mathsf{R}(P_i)$, for every $i \in [0, n]$.

2. For a timed execution sequence from $\mathsf{R}(P)$ in unlabelled PAFAS

$$\mathsf{R}(P) = S_0 \xrightarrow{1} S_0' \xrightarrow{v_0} S_1 \xrightarrow{1} S_1' \xrightarrow{v_1} S_2 \ldots S_n \xrightarrow{1} S_n' \xrightarrow{1} S_n' \xrightarrow{1} \ldots$$

   there exists a finite fair-step sequence

$$P = P_0 \xrightarrow[\mathsf{LE}(P_0)]{v_0} P_1 \xrightarrow[\mathsf{LE}(P_1)]{v_1} P_2 \ldots P_{n-1} \xrightarrow[\mathsf{LE}(P_{n-1})]{v_{n-1}} P_n$$

   where $S_i = \mathsf{R}(P_i)$, for every $i \in [0, n]$.

Proof: We only prove Item 1. The other one is similar. Assume $P = P_0 \xrightarrow[\mathsf{LE}(P_0)]{v_0} P_1 \xrightarrow[\mathsf{LE}(P_1)]{v_1}$ $P_2 \ldots P_{n-1} \xrightarrow[\mathsf{LE}(P_{n-1})]{v_{n-1}} P_n$ and $P_n \not\xrightarrow{\alpha}$ for any $\alpha \in \mathbb{A}_\tau$. By Proposition 4.6 there exists a timed execution sequence $P = P_0 \xrightarrow{1} Q_0 \xrightarrow{v_0} P_1 \xrightarrow{1} Q_1 \xrightarrow{v_1} P_2 \ldots P_n \xrightarrow{1} Q_n \xrightarrow{1} Q_n \ldots$ Then, by Proposition 3.3, there exists a timed execution sequence $\mathsf{R}(P) = S_0 \xrightarrow{1} S_0' \xrightarrow{v_0} S_1 \xrightarrow{1}$ $S_1' \xrightarrow{v_1} S_2 \ldots S_n \xrightarrow{1} S_n' \xrightarrow{1} S_n' \xrightarrow{1} \ldots$ such that $S_i = \mathsf{R}(P_i)$, for every $i \in [0, n]$, which proves the statement. $\square$

Theorem 4.9 (*Characterization of infinite fair-step sequences*)
  Let $P \in \mathsf{L}(\mathbb{P}_1)$ and $v_0, v_1, v_2 \ldots \in (\mathbb{A}_\tau)^+$. Then:

1. For any infinite fair-step sequence from P,

$$P = P_0 \xrightarrow[\mathsf{LE}(P_0)]{v_0} P_1 \xrightarrow[\mathsf{LE}(P_1)]{v_1} P_2 \ldots P_i \xrightarrow[\mathsf{LE}(P_i)]{v_i} P_{i+1} \ldots$$

   there exists a timed execution sequence in unlabelled PAFAS

$$\mathsf{R}(P) = S_0 \xrightarrow{1} S_0' \xrightarrow{v_0} S_1 \xrightarrow{1} S_1' \xrightarrow{v_1} S_2 \ldots S_i \xrightarrow{1} S_i' \xrightarrow{v_i} S_{i+1} \ldots$$

   where $S_i = \mathsf{R}(P_i)$, for every $i \geq 0$.

2. For any timed execution sequence in unlabelled PAFAS

$$\mathsf{R}(P) = S_0 \xrightarrow{1} S'_0 \xrightarrow{v_0} S_1 \xrightarrow{1} S'_1 \xrightarrow{v_1} S_2 \ldots S_i \xrightarrow{1} S'_i \xrightarrow{v_i} S_{i+1} \ldots$$

there exists a fair-step sequence

$$P = P_0 \xrightarrow{v_0}_{\mathsf{LE}(P_0)} P_1 \xrightarrow{v_1}_{\mathsf{LE}(P_1)} P_2 \ldots P_i \xrightarrow{v_i}_{\mathsf{LE}(P_i)} P_{i+1} \ldots$$

where $S_i = \mathsf{R}(P_i)$, for every $i \geq 0$.

Finally, Corollary 3.20 and Theorems 4.8, 4.9 provide a characterization of (finite and infinite) fair execution sequences.

Theorem 4.10 (*Characterization of finite fair timed execution sequences*)
Let $P \in \mathsf{L}(\mathbb{P}_1)$ and $\alpha_0, \alpha_1, \alpha_2 \ldots \in \mathbb{A}_\tau$. Then:

1. For any finite fair execution sequence from P

$$P = P_0 \xrightarrow{\alpha_0} P_1 \xrightarrow{\alpha_1} P_2 \ldots P_{n-1} \xrightarrow{\alpha_{n-1}} P_n$$

there exists a timed execution sequence in unlabelled PAFAS

$$\mathsf{R}(P) = S_{i_0} \xrightarrow{1} S'_{i_0} \xrightarrow{v_{i_0}} S_{i_1} \xrightarrow{1} S'_{i_1} \xrightarrow{v_{i_1}} S_{i_2} \ldots S_{i_m} \xrightarrow{1} S'_{i_m} \xrightarrow{1} S'_{i_m} \xrightarrow{1} \ldots$$

where $i_0 = 0$, $i_m = n$, $v_{i_j} = \alpha_{i_j} \alpha_{i_j+1} \ldots \alpha_{i_{j+1}-1}$ and $S_{i_j} = \mathsf{R}(P_{i_j})$, for every $j \in [0, m]$.

2. For any timed execution sequence from $\mathsf{R}(P)$ in unlabelled PAFAS

$$\mathsf{R}(P) = S_{i_0} \xrightarrow{1} S'_{i_0} \xrightarrow{v_{i_0}} S_{i_1} \xrightarrow{1} S'_{i_1} \xrightarrow{v_{i_1}} S_{i_2} \ldots S_{i_m} \xrightarrow{1} S'_{i_m} \xrightarrow{1} S'_{i_m} \xrightarrow{1} \ldots$$

where $i_0 = 0$, $v_{i_j} = \alpha_{i_j} \alpha_{i_j+1} \ldots \alpha_{i_{j+1}-1}$, for every $j \in [0, m]$, there exists a finite fair execution sequence

$$P = P_0 \xrightarrow{\alpha_0} P_1 \xrightarrow{\alpha_1} P_2 \ldots P_{n-1} \xrightarrow{\alpha_{n-1}} P_n$$

where $i_m = n$ and $S_{i_j} = \mathsf{R}(P_{i_j})$, for every $j \in [0, m]$.

Theorem 4.11 (*Characterization of infinite fair timed execution sequences*)
Let $P \in \mathsf{L}(\mathbb{P}_1)$ and $\alpha_0, \alpha_1, \alpha_2 \ldots \in \mathbb{A}_\tau$. Then:

1. For any infinite fair execution sequence from P

$$P = P_0 \xrightarrow{\alpha_0} P_1 \xrightarrow{\alpha_1} P_2 \ldots P_i \xrightarrow{\alpha_i} P_{i+1} \ldots$$

there exists a timed execution sequence in unlabelled PAFAS

$$\mathsf{R}(P) = S_{i_0} \xrightarrow{1} S'_{i_0} \xrightarrow{v_{i_0}} S_{i_1} \xrightarrow{1} S'_{i_1} \xrightarrow{v_{i_1}} S_{i_2} \ldots S_{i_j} \xrightarrow{1} S'_{i_j} \xrightarrow{v_{i_j}} S'_{i_{j+1}} \ldots$$

where $i_0 = 0$, $v_{i_j} = \alpha_{i_j} \alpha_{i_j+1} \ldots \alpha_{i_{j+1}-1}$ and $S_{i_j} = \mathsf{R}(P_{i_j})$, for every $j \geq 0$.

2. For any timed execution sequence from $\mathsf{R}(P)$ in unlabelled PAFAS

$$\mathsf{R}(P) = S_{i_0} \xrightarrow{1} S'_{i_0} \xrightarrow{v_{i_0}} S_{i_1} \xrightarrow{1} S'_{i_1} \xrightarrow{v_{i_1}} S_{i_2} \ldots S_{i_j} \xrightarrow{1} S'_{i_j} \xrightarrow{v_{i_j}} S'_{i_{j+1}} \ldots$$

where $i_0 = 0$, $v_{i_j} = \alpha_{i_j} \alpha_{i_j+1} \ldots \alpha_{i_{j+1}-1}$, for every $j \geq 0$, there exists an infinite fair execution sequence

$$P = P_0 \xrightarrow{\alpha_0} P_1 \xrightarrow{\alpha_1} P_2 \ldots P_i \xrightarrow{\alpha_i} P_{i+1} \ldots$$

where $S_{i_j} = \mathsf{R}(P_{i_j})$, for every $j \geq 0$.

4.3 Transition systems for fair execution sequences and finite state processes

We call an initial process $P \in \mathsf{L}(\mathbb{P}_1)$ (i.e. a standard untimed process) *finite state*, if only finitely many processes are *action-reachable*, i.e. can be reached according to the standard functional operational semantics, i.e. with transitions $\xrightarrow{\alpha}$.

For the definition of fair executions, we followed Costa and Stirling and introduced two semantic levels: one level (the positive) prescribes the finite and infinite execution sequences of labelled processes disregarding their fairness, while the other (the negative) filters out the unfair ones. The labels are notationally heavy, and keeping track of them is pretty involved. Since the labels evolve dynamically along computations, the transition system defined for the first level is in general infinite state even for finite state processes (as long as they have at least one infinite computation). Also the filtering mechanism is rather involved, since we have to check repeatedly what happens to live events along the computation, and for this we have to consider the processes passed in the computation.

With the characterization results of the previous subsection, we have not only shown a conceptional relationship between timing (as used in the PAFAS approach to the efficiency of asynchronous processes) and fairness. We have also given a much lighter description of the fair execution sequences of a process $P \in \mathsf{L}(\mathbb{P}_1)$ via the transition system of processes *timed-reachable* (i.e. with transitions $\xrightarrow{\alpha}$ and $\xrightarrow{1}$) from P, which we will denote by $\mathcal{TT}(P)$: the marking of some actions with underlines is easier than the labelling mechanism, and the filtering simply requires infinitely many time steps, i.e. non-Zeno behaviour; hence, for filtering one does not have to consider the processes passed. Furthermore, the transition system $\mathcal{TT}(P)$ is finite for finite state processes.

Theorem 4.12 If $P \in \mathsf{L}(\mathbb{P}_1)$ is finite state, then $\mathcal{TT}(P)$ is finite.

Proof: It is easy to prove that for any process $P' \in \mathsf{L}(\mathbb{P}_1)$, there are only finitely many processes $Q \in \mathsf{L}(\mathbb{P})$ with $\mathsf{unmark}(Q) = P'$; the intuitive reason is that Q only differs from P', since some prefixes are marked as urgent.

We will argue that, if Q is time-reachable from P, we have $\mathsf{unmark}(Q) = P'$ for some P' action-reachable from P. Then we are done, since by assumption of the theorem, there are only finitely many such P'. There are also only finitely many arcs in $\mathcal{TT}(P)$; note that, in particular due to the restriction on relabelling functions, our processes are sort-finite.

Assume that Q is time-reachable from P. Then we have to consider two possible cases:

- $P \xrightarrow{v} Q$ with $v \in (\mathbb{A}_\tau)^*$. In this case $Q \in \mathsf{L}(\mathbb{P}_1)$ action-reachable from P such that $\mathsf{unmark}(Q) = Q$ (by Proposition 7.5-2). We can choose $P' = Q$.

- $P \xrightarrow{v} P_1 \xrightarrow{1} Q_1 \xrightarrow{v_1} P_2 \ldots \xrightarrow{v_{k-1}} P_k \xrightarrow{1} Q_k \xrightarrow{v} Q$, where $v_i \in \mathbb{A}_\tau^*$ for all $1 \leq i \leq k$ and $k \geq 1$ (here k is the number of time steps). Obviously, P_1 is action-reachable from P. By Propositions 7.8-3, 4.4-2 and 4.5-2, $P_i \xrightarrow{1} Q_i \xrightarrow{v_i} P_{i+1}$ implies that $P_i \xrightarrow{v_i} P_{i+1}$ for every $i \in [1, k-1]$ and, hence, P_k is action-reachable from P. Again by Propositions 7.8-3 and 4.4-2, $P_k \xrightarrow{1} Q_k \xrightarrow{v} Q$ implies $P_k \xrightarrow{v} P'$ with $P' = \mathsf{unmark}(Q)$. Thus, P' is action-reachable from P_k and, hence, action-reachable from P.

$\square$

The main result in [7, 8] is a characterization of fair execution sequences with only one (positive) level: SOS-rules are given that describe all transitions $P \xrightarrow{v} Q$ with $v \in (\mathbb{A}_\tau)^*$ such that $P \xrightarrow{v}_{\mathsf{LE}(P)} Q$. This is conceptionally very simple, since there is only one level and there is no labelling or marking of processes: the corresponding transition system for P only contains processes reachable from P. In particular, the transition system is finite-state if P is finite-state. The drawback is that, in general, P has infinitely many $\mathsf{LE}(P)$-steps (namely, if it has an infinite computation), and therefore the transition system is infinitely branching and has infinitely many arcs. (Observe that this drawback is not shared by our transition system of timed-reachable processes.)

As a second main result, we will now derive from $\mathcal{TT}(P)$ for a finite-state process P a finite transition system with finitely many arcs that describes the fair execution sequences in one level: the essential idea is that the arcs are inscribed with regular expressions (and not just with sequences as in [7, 8]). The states of this transition system are the states Q of $\mathcal{TT}(P)$ such that $Q \xrightarrow{1} Q'$; if R is another such state, we have an arc from Q to R labelled with a regular expression e. This expression is obtained by taking $\mathcal{TT}(P)$ with Q' as initial state and R as the only final state, deleting all transitions $\xrightarrow{1}$ and applying the well-known Kleene construction to get an (equivalent) regular expression from a nondeterministic automaton. (The arc can be omitted, if e describes the empty set.) Clearly, such an arc corresponds to a set of B-steps which are also present in the one-level characterization of Costa and Stirling, but there is one exception: if $Q' \xrightarrow{1}$, then by Proposition 3.13 Q and Q' cannot perform any action; hence, there will only be an ε-labelled arc from Q' to itself and, if $Q \neq Q'$, from Q to Q'.

Thus, we can obtain exactly the action sequences performed in fair execution sequences of P by taking the infinite paths from P in the new transition system and replacing each regular expression e by a sequence in the language of e.

5   Comparison of old and new operational semantics

In this section we prove that, if we concentrate on initial processes and their descendants, the operational semantics described in Section 2 is equivalent to the standard PAFAS operational semantics (defined in [6]).

Definition 5.1 (*The old timed operational semantics*) The SOS-rules defining the transition relations $\xmapsto{\alpha} \subseteq (\tilde{\mathbb{P}} \times \tilde{\mathbb{P}})$, for $\alpha \in \mathbb{A}_\tau$, and $\xmapsto{X} \subseteq (\tilde{\mathbb{P}} \times \tilde{\mathbb{P}})$, for $X \subseteq \mathbb{A}$, are the same as

in Definitions 2.6 and 2.7 except for the rules Par_{a1}, Par_{a2}, Rec_a, Rec_r and Par_r that are replaced as follows:

$$\text{PAR}_{a1} \ \frac{\alpha \notin A, \ Q_1 \stackrel{\alpha}{\longmapsto} Q_1'}{Q_1\|_A Q_2 \stackrel{\alpha}{\longmapsto} Q_1'\|_A Q_2} \qquad \text{PAR}_{a2} \ \frac{\alpha \in A, \ Q_1 \stackrel{\alpha}{\longmapsto} Q_1', \ Q_2 \stackrel{\alpha}{\longmapsto} Q_2'}{Q_1\|_A Q_2 \stackrel{\alpha}{\longmapsto} Q_1'\|_A Q_2'}$$

$$\text{REC}_a \ \frac{Q\{\text{rec } x.Q/x\} \stackrel{\alpha}{\longmapsto} Q'}{\text{rec } x.Q \stackrel{\alpha}{\longmapsto} Q'} \qquad \text{REC}_r \ \frac{Q\{\text{rec } x.Q/x\} \stackrel{X}{\longmapsto}_r Q'}{\text{rec } x.Q \stackrel{X}{\longmapsto}_r Q'}$$

$$\text{PAR}_r \frac{Q_i \stackrel{X_i}{\longmapsto}_r Q_i' \text{ for } i = 1,2, \ X \subseteq (A \cap (X_1 \cup X_2)) \cup (X_1 \cap X_2)\backslash A}{Q_1\|_A Q_2 \stackrel{X}{\longmapsto}_r Q_1'\|_A Q_2'}$$

The differences between the semantics mainly deal with treatment of recursive terms and inactive urgencies. The new operational semantics does not unfold recursive terms after a time-step (as the old one does), and it unmarks all inactive urgencies after the execution of each action and time-step (which the old one does not do). In the following example we apply the old operational semantics to the process already considered in the Example 2.8.

Example 5.2 Let $P = \text{rec}x.a.x\|_{\{a\}}\text{rec}x.(a.\text{nil}+b.c.x)$. Then, according to the old semantics,

$P \stackrel{1}{\longmapsto} \underline{a}.\text{rec } x.a.x \ \|_{\{a\}} (\underline{a}.\text{nil} + \underline{b}.c.\text{rec } x.(b.c.x + a.\text{nil})) \stackrel{b}{\longmapsto}$

$\underline{a}.\text{rec } x.a.x \ \|_{\{a\}} c.\text{rec } x.(b.c.x + a.\text{nil}) \stackrel{1}{\longmapsto}$

$\underline{a}.\text{rec } x.a.x \ \|_{\{a\}} \underline{c}.\text{rec } x.(b.c.x + a.\text{nil}) \stackrel{c}{\longmapsto}$

$\underline{a}.\text{rec } x.a.x \ \|_{\{a\}} \text{rec } x.(b.c.x + a.\text{nil})$

When an initial process evolves to Q according to our new semantics and to R according to the old semantics, performing the same sequence in both cases, then Q and R should be related somehow; to capture this relation, we introduce $\mathcal{UU}$, which has the property that $R \in \mathcal{UU}(Q)$. Again, $\mathcal{UU}$ has an action set as second parameter; for all $Q \in \tilde{\mathbb{P}}$, $A \subseteq \mathbb{A}$ and $R \in \mathcal{UU}(Q,A)$, R is the same as Q except for the unfolding of recursive terms and some actions in A that are urgent in R and have been unmarked in Q. The formal definition of $\mathcal{UU}$ is given in Appendix F.

The next proposition is a preliminary result to state a correspondence between the new and the old operational semantics.

Proposition 5.3 Let $Q, R \in \tilde{\mathbb{P}}$ and $A \subseteq \mathbb{A}$ such that $R \in \mathcal{UU}(Q,A)$. Then:

1. $Q \stackrel{\alpha}{\rightarrow} Q'$ implies $R \stackrel{\alpha}{\longmapsto} R'$ for some $R' \in \mathcal{UU}(Q',A)$. Moreover $\mathcal{U}(R') \subseteq \mathcal{U}(R)$;

2. $Q \stackrel{X}{\rightarrow}_r Q'$ implies $R \stackrel{X\backslash A}{\longmapsto}_r R'$ for some $R' \in \mathcal{UU}(Q')$;

3. $R \stackrel{\alpha}{\longmapsto} R'$ implies $Q \stackrel{\alpha}{\rightarrow} Q'$ for some Q' such that $R' \in \mathcal{UU}(Q',A)$;

4. $R \stackrel{X}{\longmapsto}_r R'$ implies $Q \stackrel{X}{\rightarrow}_r Q'$ for some Q' such that $R' \in \mathcal{UU}(Q')$.

The previous proposition immediately implies the main result of this section.

Theorem 5.4 Let $Q, R \in \tilde{\mathbb{P}}$ such that $R \in \mathcal{UU}(Q)$. Then:

25

1. $Q \xrightarrow{\alpha} Q'$ implies $R \xmapsto{\alpha} R'$ for some $R' \in \mathcal{UU}(Q')$;

2. $Q \xrightarrow{X}_r Q'$ implies $R \xmapsto{X}_r R'$ for some $R' \in \mathcal{UU}(Q')$;

3. $R \xmapsto{\alpha} R'$ implies $Q \xrightarrow{\alpha} Q'$ for some Q' such that $R' \in \mathcal{UU}(Q')$;

4. $R \xmapsto{X}_r R'$ implies $Q \xrightarrow{X}_r Q'$ for some Q' such that $R' \in \mathcal{UU}(Q')$.

This theorem shows that the transition systems generated by the old and our new semantics are bisimilar. In particular, the bisimulation relates each initial process to itself by Lemma 12.2; hence, for initial processes the old and our new semantics coincide in a strong sense. Most importantly, the timed execution sequences, which we used to characterize fair behaviour, are the same in both versions.

6   Conclusions and Related Work

In this paper, we have presented a characterization of Costa and Stirling's (weak) fairness of actions in terms of a timed operational semantics. This characterization also provides a finite representation of fair runs for finite state processes.

As discussed in Section 5, the timed operational semantics is a slight modification of the original PAFAS timed operational semantics [6]. We have proven in Theorem 5.4 that the two semantics are closely related for initial processes. The new one, however, allows us to provide a more direct correspondence result between timed behaviours and fair runs (see Theorems 4.10 and 4.11) when compared to the one presented in the conference paper [4] (see Theorem 14): when states reached by an LE-step are compared to states reached by an execution between two consecutive time steps, then here we simply remove the labelling from the former, while [4] additional uses a kind of $\mathcal{UU}$ function.

In another paper we have obtained characterization results similar to the ones presented here for fairness of components in place of fairness of actions; see [5]. Though the stated main results are quite similar, the technicalities (see, for instance, the notion of live components - tuples of labels here, single labels in the component setting) and proof techniques are quite different. In particular, a different operational semantics had to be considered that corresponds to a different notion of timing. The differences between fairness of actions and fairness of components have been studied in detail in [7, 8]. For a comparison between the approach of this paper and that of [5], we refer the reader to the latter paper.

A very recent paper shows some similarities with our work. In [2], Stephen Brookes gives a denotational trace semantics for CSP processes to describe a weak notion of fairness close to ours. In his setting, there are values and expressions for them (from which we will abstract in our discussion); furthermore, synchronization is on complementary actions $a?$ and $a!$, which are combined to the internal action. The achievement is a fairly simple notion of traces to describe fair behavior, which can be defined denotationally and operationally, such that the same notion of trace can be used both for synchronous and asynchronous communicating processes. The latter result is particularly significant since denotational trace semantics for CSP processes originally followed different developments for the cases of synchronous and asynchronous communication, thus obscuring the underlying similarities between the different paradigms.

The essential idea for these simple traces is that processes explicitly declare to the external environment the actions that are waiting for a synchronization on the current state. Thus,

besides input actions $a?$ and output actions $a!$, processes can perform transitions like $P \xrightarrow{\delta_X} P$, where X is a set of actions which – in our terms – are live. Now a computation from P is fair ([2], pag. 472) if it contains a complete transition sequence for each syntactic sub-process of P, and no pair of sub-processes is permanently blocked (i.e. declaring a pending synchronization action) yet attempting to synchronize. For example, computation

$$a? \mid a! \xrightarrow{\delta_{a?}} a? \mid a! \xrightarrow{\delta_{a!}} a? \mid a! \xrightarrow{\delta_{a?}} a? \mid a! \xrightarrow{\delta_{a!}} \dots$$

is not fair (and it would neither be in our case) while

$$a! \mid b! \xrightarrow{a!} \mathsf{nil} \mid b! \xrightarrow{\delta_{b!}} \mathsf{nil} \mid b! \xrightarrow{\delta_{b!}} \mathsf{nil} \mid b! \xrightarrow{\delta_{b!}} \dots$$

is fair because no pair of sub-processes is blocked for synchronization. In Costa and Stirling's as in our setting the computation would not be fair. So one difference is that Brookes only cares about fairness of internal actions, which allows his traces to give a compositional semantics. A result in [2] states full abstractness, i.e. that the traces are observable; but it seems that observations are exactly the traces themselves, so that this simply says that traces give a compositional semantics.

Instead, one could take Costa and Stirling's fair traces as observations and look for the coarsest compositional semantics refining the respective equivalence. It would be very interesting how such a semantics differs from the one of [2]. Costa and Stirling's work is mentioned in [2], but no comparison is given.

The approach in the present paper is quite different from [2], since we start with a notion of timed traces that has a meaning of its own, and show how these traces can be used for easier descriptions of Costa and Stirling's fair traces. Timed traces with only full time steps are not compositional, but refusal traces that have refusal sets as partial time steps are fully abstract w.r.t. timed traces. Thus, an analogy to the approach in [2] is that we have letters in the refusal traces showing that some actions can be refused in some sense, while in [2] there are letters showing that some actions could be performed.

In fact, Brookes concentrates on actions that are permanently blocked (in his words), i.e. permanently ready; so his traces describe some sort of ready sets in the infinity – but they presumably give additional information about the stages where actions become ready. In analogy, one could extract from our timed refusal traces those actions that are permanently refused, i.e. a refusal set in the infinity; this would give a kind of failure semantics where elements are (w, X) with a possibly infinite w.

In fact, such an extraction is presented in [15, 16] in a Petri net setting, where it is shown that the resulting failure semantics is the coarsest congruence for parallel composition refining a fair trace semantics very similar to the one by Costa and Stirling and, thus, to the one we have studied here. This failure semantics has already been presented in [17].

Note that, at least in the standard setting, ready sets (i.e. ready semantics) give more information than refusal sets (i.e. failure semantics). It should also be noted that there are subtleties in [2] how the readiness of actions is declared to the environment, with the result that e.g. $a! \mid b!$ and $a!b!\square b!a!$ (or $a!b! + b!a!$ with our operator) do not have the same traces; it is not clear to us whether this helps to deal with fairness.

27

Acknowledgments

We would like to thank Colin Stirling for useful discussions during a preliminary stage of this work.

References

[1] E. Bihler and W. Vogler. Efficiency of token-passing MUTEX-solutions – some experiments. In J. Desel et al., editors, *Applications and Theory of Petri Nets 1998*, Lect. Notes Comp. Sci. 1420, pp. 185-204, Springer, 1998.

[2] S. Brookes. Traces, Pomsets, Fairness and Full Abstractions for Communicating Processes. In *Concur'02*, Lect. Notes Comp. Sci. 2421, pp. 466-482, Springer, 2002.

[3] F. Corradini, M.R. Di Berardini and W. Vogler. PAFAS at work: Comparing the Worst-Case Efficiency of Three Buffer Implementations. In Proc. of 2nd Asia-Pacific Conference on Quality Software, APAQS 2001, pp. 231-240, IEEE, 2001.

[4] F. Corradini, M.R. Di Berardini and W. Vogler. Relating Fairness and Timing in Process Algebras. In *Concur'03*, Lect. Notes Comp. Sci. 2761, pp. 446-460, Springer, 2003.

[5] F. Corradini, M.R. Di Berardini and W. Vogler. Fairness of Components in System Computations. In Proc. of 11th International Workshop on Expressiveness in Concurrency, Express'04, London, August 2004. A full version of this paper with the same title appeared as Technical Report 2005-3, 2005, at http://www.informatik.uni-augsburg.de/skripts/techreports/

[6] F. Corradini, W. Vogler and L. Jenner. Comparing the Worst-Case Efficiency of Asynchronous Systems with PAFAS. *Acta Informatica* **38**, pp. 735-792, 2002.

[7] G. Costa, C. Stirling. A Fair Calculus of Communicating Systems. *Acta Informatica* **21**, pp. 417-441, 1984.

[8] G. Costa, C. Stirling. Weak and Strong Fairness in CCS. *Information and Computation* **73**, pp. 207-244, 1987.

[9] R. De Nicola and M.C.B. Hennessy. Testing equivalence for processes. *Theoret. Comput. Sci.* **34**, pp. 83-133, 1984.

[10] C.A.R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.

[11] L. Jenner, W. Vogler. Comparing the Efficiency of Asynchronous Systems. In Proc. of AMAST Workshop on Real-Time and Probabilistic Systems, LNCS 1601, pp. 172-191, 1999. Modified full version as [6].

[12] L. Jenner and W. Vogler. Fast asynchronous systems in dense time. *Theoret. Comput. Sci.*, 254:379–422, 2001.

[13] N. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers, 1996.

[14] R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.

[15] W. Vogler. Efficiency of Asynchronous Systems and Read Arcs in Petri Nets. In *Icalp'97*, Lect. Notes Comp. Sci. 1256, pp. 538-548, Springer, 1997.

[16] W. Vogler. Efficiency of asynchronous systems, read arcs, and the MUTEX-problem. *Theoret. Comput. Sci.* 275, pp. 589-631, 2002.

[17] W. Vogler. Modular Construction and Partial Order Semantics of Petri Nets. Lect. Notes Comp. Sci. 625, Springer, 1992.

7 Appendix A: Useful Properties

In this appendix section we state and prove some useful properties relating some of the notions in the main body of the paper. They do not relate each other but are useful to prove the main statements.

Proposition 7.1 Let $Q \in \tilde{\mathbb{P}}$, $A, A' \subseteq \mathbb{A}_\tau$ and $\mu \in \mathbb{A}_\tau$.

1. $\mu \in \mathcal{U}(Q, A)$ implies $\mu \notin A$;

2. $\mu \in \mathcal{U}(Q, A)$ and $\mu \notin A'$ implies $\mu \in \mathcal{U}(Q, A')$;

3. $A \subseteq A'$ implies $\mathcal{U}(Q, A') \subseteq \mathcal{U}(Q, A)$;

4. $\mathcal{U}(Q, A) = \mathcal{U}(Q) \backslash A$.

Proof: First we prove Items 1 and 2 by induction hypothesis on $Q \in \tilde{\mathbb{P}}$.

Nil, Var: $Q = \mathsf{nil}$, $Q = x$. These cases are not possible since $\mathcal{U}(Q, A) = \emptyset$.

Pref: $Q = \alpha.P_1$ or $Q = \underline{\alpha}.P_1$ with $P_1 \in \tilde{\mathbb{P}}_1$. Consider only the latter case.

    1. $\mu \in \mathcal{U}(Q, A) \neq \emptyset$ implies $\alpha \notin A$ and $\mu = \alpha$.

    2. Again $\mu \in \mathcal{U}(Q, A)$ implies $\mu = \alpha$. Thus, $\mu = \alpha \notin A'$ implies $\mathcal{U}(Q, A') = \{\alpha\}$ and, hence, $\mu \in \mathcal{U}(Q, A')$.

Sum: $Q = Q_1 + Q_2$

    1. $\mu \in \mathcal{U}(Q, A)$ implies $\mu \in \mathcal{U}(Q_1, A)$ or $\mu \in \mathcal{U}(Q_2, A)$. In both cases, by induction hypothesis, $\mu \notin A$.

    2. $\mu \in \mathcal{U}(Q, A)$ implies (i) $\mu \in \mathcal{U}(Q_1, A)$ or (ii) $\mu \in \mathcal{U}(Q_2, A)$. Consider the case (i) (the other case is similar). $\mu \in \mathcal{U}(Q_1, A)$ and $\mu \notin A'$ implies, by induction hypothesis, $\mu \in \mathcal{U}(Q_1, A') \subseteq \mathcal{U}(Q, A')$.

Par: $Q = Q_1 \|_B Q_2$.

    1. Assume $\mu \in \mathcal{U}(Q, A)$ and consider the following possible subcases.

      - $\mu \in \mathcal{U}(Q_1, A \cup B)$. By induction hypothesis $\mu \notin A \cup B$ and, hence, $\mu \notin A$.
      - $\mu \in \mathcal{U}(Q_2, A \cup B)$. Similar to the previous case.
      - $\mu \in \mathcal{U}(Q_1, \mathbb{A}_\tau \backslash (B \backslash A)) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau \backslash (B \backslash A))$. By induction hypothesis $\mu \notin \mathbb{A}_\tau \backslash (B \backslash A)$. Thus $\mu \in B \backslash A$ and, trivially, $\mu \notin A$.

    2. Assume $\mu \in \mathcal{U}(Q, A)$, $\mu \notin A'$ and consider the following possible subcases.

      - $\mu \in \mathcal{U}(Q_1, A \cup B)$. By Item 1, we have that $\mu \notin A \cup B$ and, hence, $\mu \notin B$. By induction hypothesis $\mu \in \mathcal{U}(Q_1, A \cup B)$ and $\mu \notin A' \cup B$ implies $\mu \in \mathcal{U}(Q_1, A' \cup B)$.
      - $\mu \in \mathcal{U}(Q_2, A \cup B)$. Similar to the previous case.
      - $\mu \in \mathcal{U}(Q_1, \mathbb{A}_\tau \backslash (B \backslash A)) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau \backslash (B \backslash A))$. Then $\mu \notin \mathbb{A}_\tau \backslash (B \backslash A)$ (see Item 1) implies $\mu \in B \backslash A$ and, hence, $\mu \in B$. Thus $\mu \in B \backslash A'$ and $\mu \notin \mathbb{A}_\tau \backslash (B \backslash A')$. By induction hypothesis $\mu \in \mathcal{U}(Q_1, \mathbb{A}_\tau \backslash (B \backslash A)) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau \backslash (B \backslash A))$ and $\mu \notin \mathbb{A}_\tau \backslash (B \backslash A')$ implies $\mu \in \mathcal{U}(Q_1, \mathbb{A}_\tau \backslash (B \backslash A')) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau \backslash (B \backslash A')) \subseteq \mathcal{U}(Q, A')$.

Rel: $Q = Q_1[\Phi]$.

1. If $\mu \in \mathcal{U}(Q, A) = \Phi(\mathcal{U}(Q_1, \Phi^{-1}(A)))$ then $\mu = \Phi(\mu')$ for some $\mu' \in \mathcal{U}(Q_1, \Phi^{-1}(A))$. By induction hypothesis $\mu' \notin \Phi^{-1}(A) = \{\alpha \mid \Phi(\alpha) \in A\}$ and, hence, $\mu \notin A$.

2. Again $\mu \in \mathcal{U}(Q, A)$ implies $\mu = \Phi(\mu')$ for some $\mu' \in \mathcal{U}(Q_1, \Phi^{-1}(A))$. Then $\mu \notin A'$ implies $\mu' \notin \Phi^{-1}(A')$ and, by induction hypothesis, $\mu' \in \mathcal{U}(Q_1, \Phi^{-1}(A'))$. Thus $\mu = \Phi(\mu') \in \Phi(\mathcal{U}(Q_1, \Phi^{-1}(A'))) = \mathcal{U}(Q, A')$.

Rec: $Q = \mathsf{rec}\ x.Q_1$.

1. $\mu \in \mathcal{U}(Q, A) = \mathcal{U}(Q_1, A)$ implies, by induction hypothesis, $\mu \notin A$.

2. $\mu \in \mathcal{U}(Q, A) = \mathcal{U}(Q_1, A)$ and $\mu \notin A'$ implies, by induction hypothesis, $\mu \in \mathcal{U}(Q_1, A') = \mathcal{U}(Q, A')$.

Now we can prove Items 3. and 4.

3. Assume $\mu \in \mathcal{U}(Q, A')$. Then, by Item 1, $\mu \notin A'$ and, since $A \subseteq A'$, $\mu \notin A$. Thus, $\mu \in \mathcal{U}(Q, A')$, $\mu \notin A$ and Item 2. imply $\mu \in \mathcal{U}(Q, A)$.

4. By Item 2, $\mu \in \mathcal{U}(Q) \backslash A$, that is $\mu \in \mathcal{U}(Q)$ and $\mu \notin A$, implies $\mu \in \mathcal{U}(Q, A)$. Moreover, by Items 3. and 1., $\mathcal{U}(Q, A) \subseteq \mathcal{U}(Q)$ and $\mu \in \mathcal{U}(Q, A)$ implies $\mu \notin A$. Thus we can conclude that $\mathcal{U}(Q, A) \subseteq \mathcal{U}(Q) \backslash A$.

$\square$

Proposition 7.2 Let $Q, R \in \tilde{\mathbb{P}}$, $x \in \mathcal{X}$ guarded in Q and $A \subseteq \mathbb{A}_\tau$. Then $\mathcal{U}(Q\{R/x\}, A) = \mathcal{U}(Q, A)$.

Proof: We proceed by induction on $Q \in \tilde{\mathbb{P}}$.

Nil: $Q = \mathsf{nil}$. In this case x is guarded in Q and $Q\{R/x\} = \mathsf{nil}$. Moreover $\mathcal{U}(Q\{R/x\}, A) = \mathcal{U}(Q, A) = \emptyset$.

Var: $Q = y$. x guarded in Q implies $x \neq y$ and $Q\{R/x\} = y$. Similar to the Nil-case.

Pref: $Q = \alpha.P_1$ or $Q = \underline{\alpha}.P_1$. We prove only the latter case (the formes is simpler). In this case x is guarded in Q and $Q\{R/x\} = \underline{\alpha}.(P_1\{R/x\})$. If $\alpha \notin A$ then $\mathcal{U}(Q\{R/x\}, A) = \mathcal{U}(Q, A) = \{\alpha\}$. Otherwise, $\mathcal{U}(Q\{R/x\}, A) = \mathcal{U}(Q, A) = \emptyset$.

Sum: $Q = Q_1 + Q_2$. In this case x guarded in Q implies x guarded in Q_1 and in Q_2. Moreover $Q\{R/x\} = Q_1\{R/x\} + Q_2\{R/x\}$. By induction hypothesis, $\mathcal{U}(Q\{R/x\}, A) = \mathcal{U}(Q_1\{R/x\}, A) \cup \mathcal{U}(Q_2\{R/x\}, A) = \mathcal{U}(Q_1, A) \cup \mathcal{U}(Q_2, A) = \mathcal{U}(Q, A)$.

Par: $Q = Q_1 \parallel_B Q_2$. Again, x guarded in Q implies x guarded in Q_1 and in Q_2. Moreover $Q\{R/x\} = Q_1\{R/x\} \parallel_B Q_2\{R/x\}$. By induction hypothesis $\mathcal{U}(Q\{R/x\}, A) =$

$\mathcal{U}(Q_1\{R/x\}, A \cup B) \cup \mathcal{U}(Q_2\{R/x\}, A \cup B) \cup$

$(\mathcal{U}(Q_1\{R/x\}, \mathbb{A}_\tau \backslash (B \backslash A)) \cap \mathcal{U}(Q_2\{R/x\}, \mathbb{A}_\tau \backslash (B \backslash A))) =$

$\mathcal{U}(Q_1, A \cup B) \cup \mathcal{U}(Q_2, A \cup B) \cup (\mathcal{U}(Q_1, \mathbb{A}_\tau \backslash (B \backslash A)) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau \backslash (B \backslash A))) = \mathcal{U}(Q, A)$.

Rel: $Q = Q_1[\Phi]$. In this case x guarded in Q implies x guarded in Q_1 and $Q\{R/x\} = (Q_1\{R/x\})[\Phi]$. By induction hypothesis we have that

$$\mathcal{U}(Q\{R/x\}, A) = \Phi(\mathcal{U}(Q_1\{R/x\}, \Phi^{-1}(A))) = \Phi(\mathcal{U}(Q_1, \Phi^{-1}(A))) = \mathcal{U}(Q, A).$$

Rec: $Q = \mathsf{rec}\, y.Q_1$. If $x = y$ then $Q\{R/x\} = Q$ and the statement follows easily. Assume $x \neq y$. Then x guarded in Q implies x guarded in Q_1 and $Q\{R/x\} = \mathsf{rec}\, y.(Q_1\{R/x\})$. By induction hypothesis $\mathcal{U}(Q\{R/x\}, A) = \mathcal{U}(Q_1\{R/x\}, A) = \mathcal{U}(Q_1, A) = \mathcal{U}(Q, A)$.

$\square$

By Proposition 3.3 both Proposition 7.1 and 7.2 hold also for labelled terms.

Proposition 7.3 Let $Q, R \in \mathsf{L}(\tilde{\mathbb{P}})$ and $A \subseteq \mathbb{A}_\tau$.

1. $\mathsf{LE}(Q, A) \subseteq \mathsf{LE}(Q\{|R/x|\}, A)$. If $x \in \mathcal{X}$ guarded in Q then $\mathsf{LE}(Q\{|R/x|\}, A) \subseteq \mathsf{LE}(Q, A)$.

2. $\mathsf{UE}(Q, A) \subseteq \mathsf{UE}(Q\{|R/x|\}, A)$. If $x \in \mathcal{X}$ guarded in Q then $\mathsf{UE}(Q\{|R/x|\}, A) \subseteq \mathsf{UE}(Q, A)$.

3. x guarded in Q implies that $Q \xrightarrow{\mu}$ if only if $Q\{|R/x|\} \xrightarrow{\mu}$.

Proof: We only prove Item 1 and Item 3 (Item 2 is similar to Item 1). To prove Item 1 we proceed by induction on Q while to prove Item 3 we proceed by induction on the derivations $Q \xrightarrow{\mu}$ and $Q\{|R/x|\} \xrightarrow{\mu}_r$.

Nil: $Q = \mathsf{nil}_u$. In this case x is guarded in Q and $Q\{|R/x|\} = \mathsf{nil}$.

    1. $\mathsf{LE}(Q, A) = \emptyset = \mathsf{LE}(Q\{|R/x|\}, A)$.

    3. $\mathsf{nil}_u \xrightarrow{\mu}\!\!\!\!\!/\,$ and $\mathsf{nil}_u\{|Q/x|\} = \mathsf{nil}_u \xrightarrow{\mu}\!\!\!\!\!/\,$.

Var: $Q = y_u$.

    1. $\mathsf{LE}(Q, A) = \emptyset \subseteq \mathsf{LE}(Q\{|R/x|\})$. Assume x guarded in Q and, hence, $x \neq y$. Then $Q\{|R/x|\} = y_u$ and $\mathsf{LE}(Q\{|R/x|\}, A) = \mathsf{LE}(Q, A) = \emptyset$

    3. As in the previous case x guarded in Q implies $x \neq y$ and $Q\{|R/x|\} = y_u$. By operational rules we have both $Q \xrightarrow{\mu}\!\!\!\!\!/\,$ and $Q\{|R/x|\} \xrightarrow{\mu}\!\!\!\!\!/\,$.

Pref: $Q = \alpha_u.P_1$ or $Q = \underline{\alpha}_u.P_1$. Either $Q\{|R/x|\} = \alpha_u.(P_1\{|R/x|\})$ or $Q\{|R/x|\} = \underline{\alpha}_u.(P_1\{|R/x|\})$ and x is guarded in Q.

    1. If $\alpha \notin A$ then $\mathsf{LE}(Q, A) = \{\langle u \rangle\} = \mathsf{LE}(Q\{|R/x|\})$. Otherwise, $\mathsf{LE}(Q, A) = \emptyset = \mathsf{LE}(Q\{|R/x|\})$.

    3. $Q \xrightarrow{\mu}$ if and only if $\mu = \alpha$ if and only if, by operational rules, $Q\{|R/x|\} \xrightarrow{\mu}$.

Sum: $Q = Q_1 +_u Q_2$. In this case $Q\{|R/x|\} = Q_1\{|R/x|\} +_u Q_2\{|R/x|\}$.

1. By induction hypothesis
$\mathsf{LE}(Q,A) = \mathsf{LE}(Q_1, A) \cup \mathsf{LE}(Q_2, A) \subseteq \mathsf{LE}(Q_1\{\!|R/x|\!\}, A) \cup \mathsf{LE}(Q_2\{\!|R/x|\!\}, A) = \mathsf{LE}(Q\{\!|R/x|\!\}, A)$. Moreover x guarded in Q implies x guarded in Q_1 and in Q_2 and, by induction hypothesis,
$\mathsf{LE}(Q\{\!|R/x|\!\}, A) = \mathsf{LE}(Q_1\{\!|R/x|\!\}, A) \cup \mathsf{LE}(Q_2\{\!|R/x|\!\}, A) \subseteq \mathsf{LE}(Q_1, A) \cup \mathsf{LE}(Q_2, A) = \mathsf{LE}(Q, A)$.

3. Assume x guarded in Q and, hence, in Q_1 and in Q_2. By operational rules we have that $Q \xrightarrow{\mu}$ if and only if either $Q_1 \xrightarrow{\mu}$ or $Q_2 \xrightarrow{\mu}$ if only if, by induction hypothesis, either $Q_1\{\!|R/x|\!\} \xrightarrow{\mu}$ or $Q_2\{\!|R/x|\!\} \xrightarrow{\mu}$ if only if, again by operational rules, $Q\{\!|R/x|\!\} \xrightarrow{\mu}$.

Par: $Q = Q_1 \parallel_B^u Q_2$. In this case $Q\{\!|R/x|\!\} = Q_1\{\!|R/x|\!\} \parallel_B^u Q_2\{\!|R/x|\!\}$.

1. By inductive reasoning as the previous case.

3. Assume x guarded in Q (and, hence, x guarded in Q_1 and in Q_2). Consider the following possible subcases:

   - $\mu \notin B$. By operational rules $Q \xrightarrow{\mu}$ if and only if either $Q_1 \xrightarrow{\mu}$ or $Q_2 \xrightarrow{\mu}$ if only if, by induction hypothesis, either $Q_1\{\!|R/x|\!\} \xrightarrow{\mu}$ or $Q_2\{\!|R/x|\!\} \xrightarrow{\mu}$ if only if, again by operational rules, $Q\{\!|R/x|\!\} \xrightarrow{\mu}$.
   - $\mu \in B$. By operational rules $Q \xrightarrow{\mu}$ if and only if $Q_1 \xrightarrow{\mu}$ and $Q_2 \xrightarrow{\mu}$ if only if, by induction hypothesis, $Q_1\{\!|R/x|\!\} \xrightarrow{\mu}$ and $Q_2\{\!|R/x|\!\} \xrightarrow{\mu}$ if only if, again by operational rules, $Q\{\!|R/x|\!\} \xrightarrow{\mu}$.

Rel: $Q = Q_1[\Phi_u]$. In this case $Q\{\!|R/x|\!\} = (Q_1[\Phi_u])\{\!|R/x|\!\} = (Q_1\{\!|R/x|\!\})[\Phi_u]$

1. By induction hypothesis $\mathsf{LE}(Q, A) = \mathsf{LE}(Q_1, \Phi^{-1}(A)) \subseteq \mathsf{LE}(Q_1\{\!|R/x|\!\}, \Phi^{-1}(A)) = \mathsf{LE}(Q\{\!|R/x|\!\}, A)$. Moreover, if x is guarded in Q and, hence in Q_1, again by induction hypothesis, we have that $\mathsf{LE}(Q\{\!|R/x|\!\}, A) = \mathsf{LE}(Q_1\{\!|R/x|\!\}, \Phi^{-1}(A)) \subseteq \mathsf{LE}(Q_1, \Phi^{-1}(A)) = \mathsf{LE}(Q, A)$.

3. Assume x guarded in Q and, hence, in Q_1. $Q \xrightarrow{\mu}$ if only if there exists $\mu' \in \Phi^{-1}(\mu)$ such that $Q_1 \xrightarrow{\mu'}$. By induction hypothesis $Q_1 \xrightarrow{\mu'}$ if and only if $Q_1\{\!|R/x|\!\} \xrightarrow{\mu'}$ if only if, again by operational rules, $Q\{\!|R/x|\!\} \xrightarrow{\mu}$.

Rec: $Q = \mathsf{rec}\ y_u.Q_1$. If $x = y$ then x is guarded in Q and $Q\{\!|R/x|\!\} = Q$. Both statements follow easily. We can assume $x \neq y$ and $Q\{\!|R/x|\!\} = \mathsf{rec}\ y_u.(Q_1\{\!|R/x|\!\})$. By induction hypothesis:

1. $\mathsf{LE}(Q, A) = \mathsf{LE}(Q_1, A) \subseteq \mathsf{LE}(Q_1\{\!|R/x|\!\}, A) = \mathsf{LE}(Q\{\!|R/x|\!\}, A)$. Now assume x guarded in Q and, hence, in Q_1. By induction hypothesis $\mathsf{LE}(Q\{\!|R/x|\!\}, A) = \mathsf{LE}(Q_1\{\!|R/x|\!\}, A) \subseteq \mathsf{LE}(Q_1, A) = \mathsf{LE}(Q, A)$

3. Let $R_1 = \mathsf{unmark}(Q_1)$ and $S = Q_1\{\!|\mathsf{rec}\ y_u.R_1/y|\!\}$. In this case x guarded in Q implies x guarded in Q_1 and, hence, in $R_1 = \mathsf{unmark}(Q_1)$. Thus, x is also guarded in $S = Q_1\{\!|\mathsf{rec}\ y_u.R_1/y|\!\}$. Moreover, x guarded in Q_1 and Proposition 7.7-2 imply $\mathsf{unmark}(Q_1\{\!|R/x|\!\}) = \mathsf{unmark}(Q_1)\{\!|R/x|\!\} = R_1\{\!|R/x|\!\}$. Thus, $S\{\!|R/x|\!\} = (Q_1\{\!|\mathsf{rec}\ y_u.R_1/y|\!\})\{\!|R/x|\!\} = (Q_1\{\!|R/x|\!\})\{\!|\mathsf{rec}\ y_u.(R_1\{\!|R/x|\!\})/y|\!\} =$

$(Q_1\{\!|R/x|\!\})\{\!|\mathsf{rec}\ y_u.\mathsf{unmark}(Q_1\{\!|R/x|\!\})/y|\!\}$. By operational rules, $Q \xrightarrow{\mu}$ if only if $S \xrightarrow{\mu}$ if only if, by induction hypothesis, $S\{\!|R/x|\!\} \xrightarrow{\mu}$ if only if, again by operational rules, $Q\{\!|R/x|\!\} \xrightarrow{\mu}$.

$\square$

Proposition 7.4 Let $Q \in \mathsf{L}_u(\tilde{\mathbb{P}})$, A and $A' \subseteq \mathbb{A}_\tau$ with $A \subseteq A'$. Then

1. $\mathsf{LE}(Q, A') \subseteq \mathsf{LE}(Q, A)$;

2. $\mathsf{UE}(Q, A') \subseteq \mathsf{UE}(Q, A)$;

Proof: We only prove $\mathsf{LE}(Q, A') \subseteq \mathsf{LE}(Q, A)$ by induction on Q. The proof for the urgent live processes is similar.

Nil, Var: $Q = \mathsf{nil}_u$, $Q = x_u$. In these cases $\mathsf{LE}(Q, A') = \mathsf{LE}(Q, A) = \emptyset$.

Pref: $Q = \alpha_u.P_1$ or $Q = \underline{\alpha}_u.P_1$. In both cases $\alpha \notin A'$ and $A \subseteq A'$ imply $\alpha \notin A$ and hence $\mathsf{LE}(Q, A') = \mathsf{LE}(Q, A) = \{u\}$. Otherwise, if $\alpha \in A'$, $\mathsf{LE}(Q, A') = \emptyset \subseteq \mathsf{LE}(Q, A)$.

Sum: $Q = Q_1 +_u Q_2$. By induction hypothesis we have that $\mathsf{LE}(Q_i, A') \subseteq \mathsf{LE}(Q_i, A)$ for $i = 1, 2$. Thus, $\mathsf{LE}(Q_1 +_u Q_2, A') = \mathsf{LE}(Q_1, A') \cup \mathsf{LE}(Q_2, A') \subseteq \mathsf{LE}(Q_1, A) \cup \mathsf{LE}(Q_2, A) = \mathsf{LE}(Q_1 +_u Q_2, A)$.

Par: $Q = Q_1 \|_B^u Q_2$. $A \subseteq A'$ implies $A \cup B \subseteq A' \cup B$ and $B \backslash A' \subseteq B \backslash A$. Thus, by induction hypothesis, $\mathsf{LE}(Q_1, A' \cup B) \subseteq \mathsf{LE}(Q_1, A \cup B)$, $\mathsf{LE}(Q_2, A' \cup B) \subseteq \mathsf{LE}(Q_2, A \cup B)$ and, let $A_\alpha = \mathsf{LE}(Q_1, \mathbb{A}_\tau \backslash \{\alpha\}) \cap \mathsf{LE}(Q_2, \mathbb{A}_\tau \backslash \{\alpha\})$, $\bigcup_{\alpha \in B \backslash A'} A_\alpha \subseteq \bigcup_{\alpha \in B \backslash A} A_\alpha$. Hence $\mathsf{LE}(Q, A') \subseteq \mathsf{LE}(Q, A)$.

Rel, Rec: Similar to the previous cases.

$\square$

7.1 clean and unmark Properties

In this appendix section we prove some useful properties of functions clean and unmark. Most of them are stated for terms in $\tilde{\mathbb{P}}$ but, since the "action" of removing urgencies does not depend from labels we can easily prove that they also hold for terms in $\mathsf{L}(\tilde{\mathbb{P}})$.

Proposition 7.5 Let $Q \in \tilde{\mathbb{P}}$ and $A \subseteq \mathbb{A}$. Then:

1. $\mathsf{unmark}(\mathsf{clean}(Q, A)) = \mathsf{unmark}(Q) \in \tilde{\mathbb{P}}_1$;

2. $Q \in \tilde{\mathbb{P}}_1$ implies $\mathsf{clean}(Q, A) = \mathsf{unmark}(Q) = Q$;

3. $\mathcal{U}(Q, A) = \emptyset$ imply $\mathsf{clean}(Q, A) = \mathsf{unmark}(Q)$.

Proof: We prove, by induction on $Q \in \tilde{\mathbb{P}}$, only the latter item. Items 1 and 2 follow directly from Definitions 2.4 and 2.5.

Nil, Var: $Q = \mathsf{nil}$, $Q = x$. In these cases $\mathcal{U}(Q, A) = \emptyset$ and $\mathsf{clean}(Q, A) = \mathsf{unmark}(Q) = Q$ for any A.

Pref: $Q = \alpha.P_1$ or $Q = \underline{\alpha}.P_1$ with $P_1 \in \tilde{\mathbb{P}}_1$. We prove only the latter case (the former case is simpler). Assume $\mathcal{U}(Q, A) = \emptyset$. By Definition 2.3, $\alpha \in A$ and $\mathsf{clean}(Q, A) = \alpha.P_1 = \mathsf{unmark}(Q)$.

Sum: $Q = Q_1 + Q_2$. Assume $\mathcal{U}(Q, A) = \mathcal{U}(Q_1, A) \cup \mathcal{U}(Q_2, A) = \emptyset$. By induction hypothesis $\mathsf{clean}(Q, A) = \mathsf{clean}(Q_1, A) + \mathsf{clean}(Q_2, A) = \mathsf{unmark}(Q_1) + \mathsf{unmark}(Q_2) = \mathsf{unmark}(Q)$.

Par: $Q = Q_1 \|_B Q_2$. Let $A_1 = (\mathcal{U}(Q_1) \backslash \mathcal{U}(Q_2)) \cap B$ and $A_2 = (\mathcal{U}(Q_2) \backslash \mathcal{U}(Q_1)) \cap B$. Assume $\mathcal{U}(Q, A) = \emptyset$ and, by contradiction, $\alpha \in \mathcal{U}(Q_1, A \cup A_1) \neq \emptyset$. By Propositions 7.1-3 and 7.1-1 we have that $\alpha \in \mathcal{U}(Q_1)$ such that $\alpha \notin A$ and $\alpha \notin A_1$. Now, consider the following possible subcases:

- $\alpha \in B$. In this case $\alpha \in \mathcal{U}(Q_1) \cap B$ and $\alpha \notin A_1$ imply also $\alpha \in \mathcal{U}(Q_2)$. Moreover, $\alpha \in B \backslash A$ implies $\alpha \notin \mathbb{A}_\tau \backslash (B \backslash A)$ and, by Proposition 7.1-2, $\alpha \in \mathcal{U}(Q_1, \mathbb{A}_\tau \backslash (B \backslash A)) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau \backslash (B \backslash A)) \subseteq \mathcal{U}(Q, A) = \emptyset$. Clearly, this case is not possible.

- $\alpha \notin B$. In this case $\alpha \in \mathcal{U}(Q_1)$, $\alpha \notin A \cup B$ implies, again by Proposition 7.1-2, $\alpha \in \mathcal{U}(Q_1, A \cup B) \subseteq \mathcal{U}(Q, A) = \emptyset$. Also this case is not possible.

Thus, $\mathcal{U}(Q, A) = \emptyset$ implies $\mathcal{U}(Q_1, A \cup A_1) = \emptyset$ and, similarly, $\mathcal{U}(Q_2, A \cup A_2) = \emptyset$. By induction hypothesis $\mathsf{clean}(Q, A) = \mathsf{clean}(Q_1, A \cup A_1) \|_B \mathsf{clean}(Q_2, A \cup A_2) = \mathsf{unmark}(Q_1) \|_B \mathsf{unmark}(Q_2) = \mathsf{unmark}(Q)$.

Rel: $Q = Q_1[\Phi]$. $\mathcal{U}(Q, A) = \Phi^{-1}(\mathcal{U}(Q_1, \Phi^{-1}(A))) = \emptyset$ implies $\mathcal{U}(Q_1, \Phi^{-1}(A)) = \emptyset$ and, by induction hypothesis, $\mathsf{clean}(Q, A) = \mathsf{clean}(Q_1, \Phi^{-1}(A))[\Phi] = \mathsf{unmark}(Q_1)[\Phi] = \mathsf{unmark}(Q)$.

Rec: $Q = \mathsf{rec}\ x.Q_1$. By induction hypothesis $\mathcal{U}(Q, A) = \mathcal{U}(Q_1, A) = \emptyset$ implies $\mathsf{clean}(Q, A) = \mathsf{rec}\ x.\mathsf{clean}(Q_1, A) = \mathsf{rec}\ x.\mathsf{unmark}(Q_1) = \mathsf{unmark}(Q)$.

$\square$

Proposition 7.6 Let $Q \in \tilde{\mathbb{P}}$, $A, A' \subseteq \mathbb{A}$ and $A'' \subseteq \mathbb{A}_\tau$. Then:

1. $A' \cap \mathcal{U}(Q, A'') = \emptyset$ implies $\mathsf{clean}(Q, A) = \mathsf{clean}(Q, A \cup (A' \backslash A''))$;

2. $\mathcal{U}(\mathsf{clean}(Q, A), A'') = \mathcal{U}(Q, A \cup A'')$;

3. $\mathcal{A}(\mathsf{clean}(Q, A), A'') = \mathcal{A}(Q, A'')$

Proof: We prove Item 1. and 2. by induction on Q. Item 3. follows directly from Definitions 2.2 and 2.4.

Nil, Var: $Q = \mathsf{nil}$, $Q = x$. In these cases we have that

1. $\mathsf{clean}(Q, A) = \mathsf{clean}(Q, A \cup (A' \backslash A'')) = Q$.

2. $\mathcal{U}(\mathsf{clean}(Q, A), A'') = \mathcal{U}(Q, A \cup A'') = \emptyset$.

Pref: $Q = \alpha.P_1$ or $Q = \underline{\alpha}.P_1$. We prove only the latter case (the former one is simpler).

1. If $\alpha \in A''$ then, trivially, $\alpha \notin A'\backslash A''$. If $\alpha \notin A''$, $A' \cap \mathcal{U}(Q, A'') = A' \cap \{\alpha\} = \emptyset$ implies $\alpha \notin A'$ and, again, $\alpha \notin A'\backslash A''$. In both cases, $\alpha \in A$ if and only if $\alpha \in A \cup (A'\backslash A'')$ and, by Definition 2.4, $\mathsf{clean}(Q, A) = \mathsf{clean}(Q, A \cup (A'\backslash A''))$.

2. We have to consider two possible subcases:
   - $\alpha \in A \subseteq A \cup A''$. $\mathcal{U}(\mathsf{clean}(Q, A), A') = \mathcal{U}(\alpha.Q_1, A') = \emptyset = \mathcal{U}(Q, A \cup A'')$.
   - $\alpha \notin A$. In this case $\mathsf{clean}(Q, A) = Q$. Moreover, $\alpha \notin A$ implies $\alpha \in A''$ if and only if $\alpha \in A \cup A''$ and, by Definition 2.3, $\mathcal{U}(Q, A'') = \mathcal{U}(Q, A \cup A'')$. Thus $\mathcal{U}(\mathsf{clean}(Q, A), A'') = \mathcal{U}(Q, A'') = \mathcal{U}(Q, A \cup A'')$.

Sum: $Q = Q_1 + Q_2$.

1. Assume $A' \cap \mathcal{U}(Q, A'') = \emptyset$. Then, since $\mathcal{U}(Q, A'') = \mathcal{U}(Q_1, A'') \cup \mathcal{U}(Q_2, A'')$, we also have $A' \cap \mathcal{U}(Q_1, A'') = A' \cap \mathcal{U}(Q_2, A'') = \emptyset$. By induction hypothesis $\mathsf{clean}(Q_i, A) = \mathsf{clean}(Q_i, A \cup (A'\backslash A''))$, for $i = 1, 2$, and, hence,
   $\mathsf{clean}(Q, A) = \mathsf{clean}(Q_1, A) + \mathsf{clean}(Q_2, A) =$
   $\mathsf{clean}(Q_1, A \cup (A'\backslash A'')) + \mathsf{clean}(Q_2, A \cup (A'\backslash A'')) = \mathsf{clean}(Q, A \cup (A'\backslash A''))$.

2. By induction hypothesis we have that
   $\mathcal{U}(\mathsf{clean}(Q, A), A'') = \mathcal{U}(\mathsf{clean}(Q_1, A) + \mathsf{clean}(Q_2, A), A'') =$
   $\mathcal{U}(\mathsf{clean}(Q_1, A), A'') \cup \mathcal{U}(\mathsf{clean}(Q_2, A), A'') = \mathcal{U}(Q_1, A \cup A'') \cup \mathcal{U}(Q_2, A \cup A'') = \mathcal{U}(Q, A \cup A'')$.

Par: $Q = Q_1 \parallel_B Q_2$. Let $A_1 = (\mathcal{U}(Q_1)\backslash\mathcal{U}(Q_2)) \cap B$ and $A_2 = (\mathcal{U}(Q_2)\backslash\mathcal{U}(Q_1)) \cap B$.

1. First of all, we prove that $\mathcal{U}(Q_1, A'')\backslash A_1 \subseteq \mathcal{U}(Q, A'')$. Let $\alpha \in \mathcal{U}(Q_1, A'')\backslash A_1$. Then, $\alpha \in \mathcal{U}(Q_1, A'')$ and Proposition 7.1-1 imply $\alpha \in \mathcal{U}(Q_1)$ and $\alpha \notin A''$. Moreover $\alpha \notin A_1$. We have to consider two possible subcases:
   - $\alpha \notin B$. Then $\alpha \in \mathcal{U}(Q_1)$, $\alpha \notin A'' \cup B$ and Proposition 7.1-2 imply $\alpha \in \mathcal{U}(Q_1, A'' \cup B) \subseteq \mathcal{U}(Q, A'')$
   - $\alpha \in B$. $\alpha \in \mathcal{U}(Q_1)$ and $\alpha \notin A_1$ imply also $\alpha \in \mathcal{U}(Q_2)$. Moreover, $\alpha \in B$ and $\alpha \notin A''$ imply $\alpha \in B\backslash A''$ and, hence, $\alpha \notin \mathbb{A}_\tau\backslash(B\backslash A'')$. Then $\alpha \in \mathcal{U}(Q_1)$, $\alpha \in \mathcal{U}(Q_2)$ and $\alpha \notin \mathbb{A}_\tau\backslash(B\backslash A'')$ imply, again by Proposition 7.1-2, $\alpha \in \mathcal{U}(Q_1, \mathbb{A}_\tau\backslash(B\backslash A'')) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau\backslash(B\backslash A'')) \subseteq \mathcal{U}(Q, A'')$.

   Now, $A' \cap \mathcal{U}(Q, A'') = \emptyset$ and $(A'\backslash A_1) \cap \mathcal{U}(Q_1, A'') = A' \cap (\mathcal{U}(Q_1, A'')\backslash A_1) \subseteq A' \cap \mathcal{U}(Q, A'')$ implies $(A'\backslash A_1) \cap \mathcal{U}(Q_1, A'') = \emptyset$. Moreover $(A'\backslash A_1)\backslash A'' = (A'\backslash A'')\backslash A_1$. By induction hypothesis $\mathsf{clean}(Q_1, A \cup A_1) = \mathsf{clean}(Q_1, (A \cup A_1) \cup ((A'\backslash A_1)\backslash A'')) = \mathsf{clean}(Q_1, (A \cup A_1) \cup ((A'\backslash A'')\backslash A_1)) = \mathsf{clean}(Q_1, (A \cup (A'\backslash A'')) \cup A_1)$. Similarly we can prove that $A' \cap \mathcal{U}(Q, A'') = \emptyset$ implies $(A'\backslash A_2) \cap \mathcal{U}(Q_2, A'') = \emptyset$ and $\mathsf{clean}(Q_2, A \cup A_2) = \mathsf{clean}(Q_2, (A \cup (A'\backslash A'')) \cup A_2)$. Finally, we can conclude that $\mathsf{clean}(Q, A) = \mathsf{clean}(Q, A \cup (A'\backslash A''))$.

2. Let us denote with $R_i = \mathsf{clean}(Q_i, A \cup A_i)$, for $i = 1, 2$. Then, by Definition 2.4 $\mathsf{clean}(Q, A) = R_1 \parallel_B R_2 = R$. By induction hypothesis $\mathcal{U}(R_1, A'' \cup B) = \mathcal{U}(Q_1, (A \cup A_1) \cup (A'' \cup B)) = \mathcal{U}(Q_1, (A \cup A'') \cup B)$ (since $A_1 \subseteq B$). Similarly we can prove that $\mathcal{U}(R_2, A'' \cup B) = \mathcal{U}(Q_2, (A \cup A'') \cup B)$. Moreover, again by

induction hypothesis, $\mathcal{U}(R_i, \mathbb{A}_\tau \backslash (B \backslash A'')) = \mathcal{U}(Q_i, (A \cup A_i) \cup (\mathbb{A}_\tau \backslash (B \backslash A'')))$, for $i = 1, 2$.

Let $X = \mathcal{U}(Q_1, (A \cup A_1) \cup (\mathbb{A}_\tau \backslash (B \backslash A''))) \cap \mathcal{U}(Q_2, (A \cup A_2) \cup (\mathbb{A}_\tau \backslash (B \backslash A'')))$ and $Y = \mathcal{U}(Q_1, \mathbb{A}_\tau \backslash (B \backslash (A \cup A''))) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau \backslash (B \backslash (A \cup A'')))$. By Definition 2.3, it remains to prove that $X = Y$.

"$\subseteq$". If $\alpha \in X$ then, by Proposition 7.1, $\alpha \in \mathcal{U}(Q_1) \cap \mathcal{U}(Q_2)$ such that $\alpha \notin \mathbb{A}_\tau \backslash (B \backslash A'')$, $\alpha \notin A$ and $\alpha \notin A_i$ for $i = 1, 2$. Now, $\alpha \notin \mathbb{A}_\tau \backslash (B \backslash A'')$, i.e. $\alpha \in B \backslash A''$, and $\alpha \notin A$ imply $\alpha \in B \backslash (A \cup A'')$ and, hence, $\alpha \notin \mathbb{A}_\tau \backslash (B \backslash (A \cup A''))$. By Proposition 7.1-2, $\alpha \in \mathcal{U}(Q_1) \cap \mathcal{U}(Q_2)$ and $\alpha \notin \mathbb{A}_\tau \backslash (B \backslash (A \cup A''))$ imply $\alpha \in Y$.

"$\supseteq$". Again by Proposition 7.1, $\alpha \in Y$ implies $\alpha \in \mathcal{U}(Q_1) \cap \mathcal{U}(Q_2)$ such that $\alpha \notin \mathbb{A}_\tau \backslash (B \backslash (A \cup A''))$. Now, $\alpha \notin \mathbb{A}_\tau \backslash (B \backslash (A \cup A''))$ implies $\alpha \in B \backslash (A \cup A'')$ and, hence, $\alpha \in B$, $\alpha \notin A''$ and $\alpha \notin A$. Thus $\alpha \notin A \cup (\mathbb{A}_\tau \backslash (B \backslash A''))$. Moreover, $\alpha \in \mathcal{U}(Q_1)$ and $\alpha \in \mathcal{U}(Q_2)$ imply, trivially, $\alpha \notin A_1$ and $\alpha \notin A_2$. Again by Proposition 7.1-2, $\alpha \in \mathcal{U}(Q_i)$, $\alpha \notin \mathbb{A}_\tau \backslash (B \backslash A'')$ and $\alpha \notin A \cup A_i$ imply $\alpha \in \mathcal{U}(Q_i, (A \cup A_i) \cup (\mathbb{A}_\tau \backslash (B \backslash A'')))$, for $i = 1, 2$, and, hence, $\alpha \in X$.

Rel: $Q = Q_1[\Phi]$.

1. Assume $A' \cap \mathcal{U}(Q, A'') = A' \cap \Phi(\mathcal{U}(Q_1, \Phi^{-1}(A''))) = \emptyset$. Then we have also that $\Phi^{-1}(A') \cap \mathcal{U}(Q_1, \Phi^{-1}(A'')) = \emptyset$ and, by induction hypothesis, $\mathsf{clean}(Q, A) = \mathsf{clean}(Q_1, \Phi^{-1}(A))[\Phi] = \mathsf{clean}(Q_1, \Phi^{-1}(A) \cup (\Phi^{-1}(A' \backslash A'')))[\Phi] = \mathsf{clean}(Q_1, \Phi^{-1}(A \cup (A' \backslash A'')))[\Phi] = \mathsf{clean}(Q, A \cup (A' \backslash A''))$

2. By induction hypothesis $\mathcal{U}(\mathsf{clean}(Q, A), A'') = \mathcal{U}(\mathsf{clean}(Q_1, \Phi^{-1}(A))[\Phi], A'') = \Phi(\mathcal{U}(\mathsf{clean}(Q_1, \Phi^{-1}(A)), \Phi^{-1}(A''))) = \Phi(\mathcal{U}(Q_1, \Phi^{-1}(A) \cup \Phi^{-1}(A''))) = \Phi(\mathcal{U}(Q_1, \Phi^{-1}(A \cup A''))) = \mathcal{U}(Q, A \cup A'')$.

Rec: $Q = \mathsf{rec}\, x.Q_1$. By induction hypothesis:

1. $A' \cap \mathcal{U}(Q, A'') = A' \cap \mathcal{U}(Q_1, A'') = \emptyset$ implies $\mathsf{clean}(Q, A) = \mathsf{rec}\, x.\mathsf{clean}(Q_1, A) = \mathsf{rec}\, x.\mathsf{clean}(Q_1, A \cup (A' \backslash A'')) = \mathsf{clean}(Q, A \cup (A' \backslash A''))$.

2. $\mathcal{U}(\mathsf{clean}(Q, A), A'') = \mathcal{U}(\mathsf{rec}\, x.\mathsf{clean}(Q_1, A), A'') = \mathcal{U}(\mathsf{clean}(Q_1, A), A'') = \mathcal{U}(Q_1, A \cup A'') = \mathcal{U}(Q, A \cup A'')$.

$\square$

Proposition 7.7 Let $Q, R \in \tilde{\mathbb{P}}$, $x \in \mathcal{X}$ guarded in Q and $A \subseteq \mathbb{A}$. Then:

1. $\mathsf{clean}(Q\{R/x\}, A) = \mathsf{clean}(Q, A)\{R/x\}$;

2. $\mathsf{unmark}(Q\{R/x\}) = \mathsf{unmark}(Q)\{R/x\}$.

Proof: We prove only Item 1 by induction on Q (Item 2 can be proved similarly).

Nil: $Q = \mathsf{nil}$. Trivially $\mathsf{clean}(\mathsf{nil}\{R/x\}, A) = \mathsf{clean}(\mathsf{nil}, A) = \mathsf{nil}$

Var: $Q = y$. In this case x guarded in Q implies $x \neq y$ and $Q\{R/x\} = y$. Similar to the previous case.

Pref: $Q = \alpha P_1$ or $Q = \underline{\alpha}.P_1$. We prove only the latter case (the former is simpler). In this case we have that x is guarded in Q and $Q\{R/x\} = \underline{\alpha}.(P_1\{R/x\})$. Assume $\alpha \in A$. Then $\mathsf{clean}(Q\{R/x\}, A) = \alpha.(P_1\{R/x\}) = (\alpha.P_1)\{R/x\} = \mathsf{clean}(Q, A)\{R/x\}$. Similarly if $\alpha \notin A$ then $\mathsf{clean}(Q\{R/x\}, A) = \underline{\alpha}.(P_1\{R/x\}) = (\underline{\alpha}.P_1)\{R/x\} = \mathsf{clean}(Q, A)\{R/x\}$.

Sum: $Q = Q_1 + Q_2$. Assume x guarded in Q and, hence, in Q_1 and in Q_2. By induction hypothesis $\mathsf{clean}(Q\{R/x\}, A) =$

$\mathsf{clean}(Q_1\{R/x\} + Q_2\{R/x\}, A) =$

$\mathsf{clean}(Q_1\{R/x\}, A) + \mathsf{clean}(Q_2\{R/x\}, A) =$

$(\mathsf{clean}(Q_1, A)\{R/x\}) + (\mathsf{clean}(Q_2, A)\{R/x\}) =$

$(\mathsf{clean}(Q_1, A) + \mathsf{clean}(Q_2, A))\{R/x\} = \mathsf{clean}(Q, A)\{R/x\}$

Par: $Q = Q_1 \|_B Q_2$. Assume x guarded in Q and, hence, in Q_1 and in Q_2. Let us denote with $A_1 = (\mathcal{U}(Q_1\{R/x\})\backslash\mathcal{U}(Q_2\{R/x\}))\cap B$ and with $A_2 = (\mathcal{U}(Q_2\{R/x\})\backslash\mathcal{U}(Q_1\{R/x\}))\cap B$. x guarded in Q_1, Q_2 and Proposition 7.2 imply $A_1 = (\mathcal{U}(Q_1)\backslash\mathcal{U}(Q_2)) \cap B$ and $A_2 = (\mathcal{U}(Q_2)\backslash\mathcal{U}(Q_1)) \cap B$. By induction hypothesis we have that $\mathsf{clean}(Q\{R/x\}, A) =$

$\mathsf{clean}(Q_1\{R/x\}, A \cup A_1) \|_B \mathsf{clean}(Q_2\{R/x\}, A \cup A_2) =$

$(\mathsf{clean}(Q_1, A \cup A_1)\{R/x\}) \|_B (\mathsf{clean}(Q_2, A \cup A_2)\{R/x\}) =$

$(\mathsf{clean}(Q_1, A \cup A_1) \|_B (\mathsf{clean}(Q_2, A \cup A_2))\{R/x\} = \mathsf{clean}(Q, A)\{R/x\}$

Rel: $Q = Q_1[\Phi_u]$. Assume x guarded in Q and, hence, in Q_1. By induction hypothesis $\mathsf{clean}(Q\{R/x\}, A) = \mathsf{clean}(Q_1\{R/x\}, \Phi^{-1}(A))[\Phi] = (\mathsf{clean}(Q_1, \Phi^{-1}(A))\{R/x\})[\Phi] = (\mathsf{clean}(Q_1, \Phi^{-1}(A))[\Phi])\{R/x\} = \mathsf{clean}(Q, A)\{R/x\}$.

Rec: $Q = \mathsf{rec}\,y.Q_1$. If $x = y$ then $Q\{R/x\} = R$ and the statement follows easily. Now assume $x \neq y$. Then x guarded in Q implies x guarded in Q_1 and $Q\{R/x\} = \mathsf{rec}\,y.(Q_1\{R/x\})$. Finally, $\mathsf{clean}(Q\{R/x\}, A) = \mathsf{rec}\,y.\mathsf{clean}(Q_1\{R/x\}, A) = \mathsf{rec}\,y.(\mathsf{clean}(Q_1, A)\{R/x\}) = (\mathsf{rec}\,y.\mathsf{clean}(Q_1, A))\{R/x\} = \mathsf{clean}(Q, A)\{R/x\}$, by induction hypothesis.

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad$ $\square$

Proposition 7.8 Let $Q, Q' \in \tilde{\mathbb{P}}$ and $X \subseteq \mathbb{A}$ such that $Q \xrightarrow{X}_r Q'$. Then:

1. $\mathcal{A}(Q', A) = \mathcal{U}(Q', A) = \mathcal{A}(Q, A)$ for every $A \subseteq \mathbb{A}_\tau$;

2. $\mathsf{unmark}(Q) = \mathsf{unmark}(Q')$;

3. $Q = P \in \tilde{\mathbb{P}}_1$ implies $\mathsf{unmark}(Q') = P$.

Proof: First, we prove, by induction on $Q \in \tilde{\mathbb{P}}$, Items 1 and 2.

Var: $Q = x$. This case is not possible since $Q \xcancel{\xrightarrow{X}}_r$.

Nil: $Q = \mathsf{nil}$. $Q \xrightarrow{X}_r \mathsf{nil} = Q'$. In this case:

    1. $\mathcal{A}(Q', A) = \mathcal{U}(Q', A) = \mathcal{A}(Q, A) = \emptyset$;

    2. $\mathsf{unmark}(Q) = \mathsf{unmark}(Q') = \mathsf{nil}$.

Pref: $Q = \alpha.P_1$ or $Q = \underline{\alpha}.P_1$. In both cases $Q \xrightarrow{X}_r \underline{\alpha}.P_1 = Q'$.

1. $\alpha \notin A$ implies $\mathcal{A}(Q', A) = \mathcal{U}(Q', A) = \mathcal{A}(Q, A) = \{\alpha\}$. Otherwise, $\mathcal{A}(Q', A) = \mathcal{U}(Q', A) = \mathcal{A}(Q, A) = \emptyset$.

2. $\mathsf{unmark}(Q) = \mathsf{unmark}(Q') = \alpha.P_1$.

Sum: $Q = Q_1 + Q_2$. By operational semantics $Q \xrightarrow{X}_r Q'$ implies $Q_1 \xrightarrow{X}_r Q'_1$, $Q_2 \xrightarrow{X}_r Q'_2$ and $Q' = Q'_1 + Q'_2$. By induction hypothesis:

1. $\mathcal{A}(Q', A) = \mathcal{A}(Q'_1, A) \cup \mathcal{A}(Q'_2, A) = \mathcal{A}(Q_1, A) \cup \mathcal{A}(Q_2, A) = \mathcal{A}(Q, A)$. Similarly we can prove that $\mathcal{U}(Q', A) = \mathcal{A}(Q, A)$.

2. By induction hypothesis, we have that $\mathsf{unmark}(Q) = \mathsf{unmark}(Q_1) + \mathsf{unmark}(Q_2) = \mathsf{unmark}(Q'_1) + \mathsf{unmark}(Q'_2) = \mathsf{unmark}(Q')$.

Par: $Q = Q_1 \parallel_B Q_2$. Assume $Q \xrightarrow{X}_r Q'$. Then, by operational rules, $Q_i \xrightarrow{X_i}_r Q'_i$ for $i = 1, 2$, $X \subseteq (B \cap (X_1 \cup X_2)) \cup ((X_1 \cap X_2) \backslash B)$ and $Q' = \mathsf{clean}(Q'_1 \parallel_B Q'_2)$.

1. By induction hypothesis and Proposition 7.6-3 $\mathcal{A}(Q', A) = \mathcal{A}(Q'_1 \parallel_B Q'_2, A) = \mathcal{A}(Q'_1, A \cup B) \cup \mathcal{A}(Q'_2, A \cup B) \cup (\mathcal{A}(Q'_1, \mathbb{A}_\tau \backslash (B \backslash A)) \cap \mathcal{A}(Q'_2, \mathbb{A}_\tau \backslash (B \backslash A))) = \mathcal{A}(Q_1, A \cup B) \cup \mathcal{A}(Q_2, A \cup B) \cup (\mathcal{A}(Q_1, \mathbb{A}_\tau \backslash (B \backslash A)) \cap \mathcal{A}(Q_2, \mathbb{A}_\tau \backslash (B \backslash A))) = \mathcal{A}(Q, A)$. By Proposition 7.6-2, we have $\mathcal{U}(Q', A) = \mathcal{U}(Q'_1 \parallel_B Q'_2, A)$ and, again by induction hypothesis, we can prove that $\mathcal{U}(Q'_1 \parallel_B Q'_2, A) = \mathcal{A}(Q, A)$.

2. By induction hypothesis, we have that $\mathsf{unmark}(Q) = \mathsf{unmark}(Q_1) \parallel_B \mathsf{unmark}(Q_2) = \mathsf{unmark}(Q'_1) \parallel_B \mathsf{unmark}(Q'_2) = \mathsf{unmark}(Q'_1 \parallel_B Q'_2) = \mathsf{unmark}(\mathsf{clean}(Q'_1 \parallel_B Q'_2)) = \mathsf{unmark}(Q')$ by Proposition 7.5-1.

Rel: $Q = Q_1[\Phi]$. By operational semantics $Q \xrightarrow{X}_r Q'$ implies $Q_1 \xrightarrow{X'}_r Q'_1$, with $X' = \Phi^{-1}(X \cup \{\tau\}) \backslash \{\tau\}$, and $Q' = Q'_1[\Phi]$.

1. $\mathcal{A}(Q', A) = \Phi(\mathcal{A}(Q'_1, \Phi^{-1}(A)) = \Phi(\mathcal{A}(Q_1, \Phi^{-1}(A)) = \mathcal{A}(Q, A)$. Similarly we can prove that $\mathcal{U}(Q', A) = \mathcal{A}(Q, A)$.

2. By induction hypothesis, $\mathsf{unmark}(Q) = \mathsf{unmark}(Q_1)[\Phi] = \mathsf{unmark}(Q'_1)[\Phi] = \mathsf{unmark}(Q')$.

Rec: $Q = \mathsf{rec}\,x.Q_1$. By operational semantics $Q \xrightarrow{X}_r Q'$ implies $Q_1 \xrightarrow{X}_r Q'_1$ and $Q' = \mathsf{rec}\,x.Q'_1$.

1. $\mathcal{A}(Q', A) = \mathcal{A}(Q'_1, A) = \mathcal{A}(Q_1, A) = \mathcal{A}(Q, A)$. Similarly we can prove that $\mathcal{U}(Q', A) = \mathcal{A}(Q, A)$.

2. By induction hypothesis, $\mathsf{unmark}(Q) = \mathsf{rec}\,x.\mathsf{unmark}(Q_1) = \mathsf{rec}\,x.\mathsf{unmark}(Q'_1) = \mathsf{unmark}(Q')$.

Now we can prove Item 3.

Assume $Q = P \in \tilde{\mathbb{P}}_1$ and that $Q \xrightarrow{X}_r Q'$. Then Item 2 and Proposition 7.5-2 imply $\mathsf{unmark}(Q') = \mathsf{unmark}(Q) = Q = P$. $\qquad\square$

Proposition 7.9 Let $Q \in \mathsf{L}(\tilde{\mathbb{P}})$, $A \subseteq \mathbb{A}$ and $A' \subseteq \mathbb{A}_\tau$. Then:

1. $\mathsf{LE}(\mathsf{clean}(Q, A), A') = \mathsf{LE}(Q, A')$;

2. $A \subseteq A'$ implies $\mathsf{UE}(\mathsf{clean}(Q, A), A') = \mathsf{UE}(Q, A')$.

Proof: We prove, by induction on $Q \in \mathsf{L}(\tilde{\mathbb{P}})$, only the latter item. Item 1 follows directly from Definitions 2.4 and 3.7.

Nil, Var: $Q = \mathsf{nil}_u$, $Q = x_u$. In these cases $\mathsf{UE}(\mathsf{clean}(Q, A), A') = \mathsf{UE}(Q, A') = \emptyset$.

Pref: $Q = \alpha_u.P_1$ or $Q = \underline{\alpha}_u.P_1$. We prove only the latter case (the former is similar to the previous cases). Consider the following cases:

- $\alpha \in A \subseteq A'$. $\mathsf{UE}(\mathsf{clean}(Q, A), A') = \mathsf{UE}(\alpha_u.P_1, A') = \emptyset = \mathsf{UE}(Q, A')$.

- $\alpha \notin A$. In this case $\mathsf{clean}(Q, A) = Q$ and the statement follows easily.

Sum: $Q = Q_1 +_u Q_2$. Assume $A \subseteq A'$. Then, by induction hypothesis, $\mathsf{UE}(\mathsf{clean}(Q, A), A') = \mathsf{UE}(\mathsf{clean}(Q_1, A) +_u \mathsf{clean}(Q_2, A), A') = \mathsf{UE}(\mathsf{clean}(Q_1, A), A') \cup \mathsf{UE}(\mathsf{clean}(Q_2, A), A') = \mathsf{UE}(Q_1, A') \cup \mathsf{UE}(Q_2, A') = \mathsf{UE}(Q, A')$.

Par: $Q = Q_1 \parallel_B^u Q_2$. Let $A_1 = (\mathcal{U}(Q_1) \backslash \mathcal{U}(Q_2)) \cap B$, $A_2 = (\mathcal{U}(Q_2) \backslash \mathcal{U}(Q_1)) \cap B$ and $R_i = \mathsf{clean}(Q_i, A \cup A_i)$ for $i = 1, 2$. By Definitions 2.4 and 3.8, $\mathsf{clean}(Q, A) = R_1 \parallel_B^u R_2 = R$ and $\mathsf{UE}(R, A') = \mathsf{UE}(R_1, A' \cup B) \cup \mathsf{UE}(R_2, A' \cup B) \cup \bigcup_{\alpha \in B \backslash A'} A_\alpha$, where $A_\alpha = \mathsf{UE}(R_1, \mathbb{A}_\tau \backslash \{\alpha\}) \times \mathsf{UE}(R_2, \mathbb{A}_\tau \backslash \{\alpha\})$. $A \cup A_i \subseteq A' \cup B$ implies, by induction hypothesis, $\mathsf{UE}(R_i, A' \cup B) = \mathsf{UE}(\mathsf{clean}(Q_i, A \cup A_i), A' \cup B) = \mathsf{UE}(Q_i, A' \cup B)$, for $i = 1, 2$. Now let $\alpha \in B \backslash A'$ and consider the following cases:

- $\alpha \in A_1$. In this case $\alpha \notin \mathcal{U}(Q_2)$ and $\mathcal{U}(Q_2, \mathbb{A}_\tau \backslash \{\alpha\}) = \mathcal{U}(Q_2) \backslash (\mathbb{A}_\tau \backslash \{\alpha\}) = \mathcal{U}(Q_2) \cap \{\alpha\}$ (see Proposition 7.1-4) imply $\mathcal{U}(Q_2, \mathbb{A}_\tau \backslash \{\alpha\}) = \emptyset$. Then, by Proposition 3.12-3, $\mathsf{UE}(Q_2, \mathbb{A}_\tau \backslash \{\alpha\}) = \emptyset$ and $\mathsf{UE}(Q_1, \mathbb{A}_\tau \backslash \{\alpha\}) \times \mathsf{UE}(Q_2, \mathbb{A}_\tau \backslash \{\alpha\}) = \emptyset$. Moreover, $\mathcal{U}(R_2) = \mathcal{U}(\mathsf{clean}(Q_2, A \cup A_2)) = \mathcal{U}(Q_2, A \cup A_2) \subseteq \mathcal{U}(Q_2)$ (see Propositions 7.6-2 and 7.1-3) and $\alpha \notin \mathcal{U}(Q_2)$ also imply $\alpha \notin \mathcal{U}(R_2)$. Again we have $\mathcal{U}(R_2, \mathbb{A}_\tau \backslash \{\alpha\}) = \emptyset$, $\mathsf{UE}(R_2, \mathbb{A}_\tau \backslash \{\alpha\}) = \emptyset$ and $A_\alpha = \mathsf{UE}(R_1, \mathbb{A}_\tau \backslash \{\alpha\}) \times \mathsf{UE}(R_2, \mathbb{A}_\tau \backslash \{\alpha\}) = \emptyset$.

- $\alpha \in A_2$. Similarly, we can prove that $A_\alpha = \mathsf{UE}(R_1, \mathbb{A}_\tau \backslash \{\alpha\}) \times \mathsf{UE}(R_2, \mathbb{A}_\tau \backslash \{\alpha\}) = \mathsf{UE}(Q_1, \mathbb{A}_\tau \backslash \{\alpha\}) \times \mathsf{UE}(Q_2, \mathbb{A}_\tau \backslash \{\alpha\}) = \emptyset$.

- $\alpha \notin A_1$ and $\alpha \notin A_2$. In this case $\alpha \in B \backslash A'$ implies $\alpha \notin A \subseteq A'$ and, hence, $\alpha \notin A \cup A_i$ for $i = 1, 2$. Thus $A \cup A_i \subseteq \mathbb{A}_\tau \backslash \{\alpha\}$ and, by induction hypothesis, $\mathsf{UE}(R_i, \mathbb{A}_\tau \backslash \{\alpha\}) = \mathsf{UE}(\mathsf{clean}(Q_i, A \cup A_i), \mathbb{A}_\tau \backslash \{\alpha\}) = \mathsf{UE}(Q_i, \mathbb{A}_\tau \backslash \{\alpha\})$, for $i = 1, 2$, and $A_\alpha = \mathsf{UE}(R_1, \mathbb{A}_\tau \backslash \{\alpha\}) \times \mathsf{UE}(R_2, \mathbb{A}_\tau \backslash \{\alpha\}) = \mathsf{UE}(Q_1, \mathbb{A}_\tau \backslash \{\alpha\}) \times \mathsf{UE}(Q_2, \mathbb{A}_\tau \backslash \{\alpha\})$.

Thus $\bigcup_{\alpha \in B \backslash A'} A_\alpha = \bigcup_{\alpha \in B \backslash A'} (\mathsf{UE}(Q_1, \mathbb{A}_\tau \backslash \{\alpha\}) \times \mathsf{UE}(Q_2, \mathbb{A}_\tau \backslash \{\alpha\}))$ and, again by Definition 3.8, we can conclude that $\mathsf{UE}(\mathsf{clean}(Q, A), A') = \mathsf{UE}(Q, A')$.

Rel: $Q = Q_1[\Phi_u]$. Assume $A \subseteq A'$ and, hence, $\Phi^{-1}(A) \subseteq \Phi^{-1}(A')$. By induction hypothesis $\mathsf{UE}(\mathsf{clean}(Q, A), A') = \mathsf{UE}(\mathsf{clean}(Q_1, \Phi^{-1}(A))[\Phi_u], A') = \mathsf{UE}(\mathsf{clean}(Q_1, \Phi^{-1}(A)), \Phi^{-1}(A'))$

$= \mathsf{UE}(Q_1, \Phi^{-1}(A')) = \mathsf{UE}(Q, A')$

Let $R_1 = \mathsf{clean}(Q_1, \Phi^{-1}(A))$ and $R = R_1[\Phi_u]$ By Definitions 2.4 and 3.8 we have that $\mathsf{clean}(Q, A) = R_1[\Phi_u] = R$ and $\mathsf{UE}(\mathsf{clean}(Q, A), A') = \mathsf{UE}(R, A) = \mathsf{UE}(R_1, \Phi^{-1}(A'))$. $A \subseteq A'$ implies $\Phi^{-1}(A) \subseteq \Phi^{-1}(A')$ and, by induction hypothesis, $\mathsf{UE}(R_1, \Phi^{-1}(A')) = \mathsf{UE}(\mathsf{clean}(Q_1, \Phi^{-1}(A)), \Phi^{-1}(A')) = \mathsf{UE}(Q_1, \Phi^{-1}(A')) = \mathsf{UE}(Q, A')$. We can conclude that $\mathsf{UE}(\mathsf{clean}(Q, A), A') = \mathsf{UE}(Q, A')$.

Rec: $Q = \mathsf{rec}\ x_u.Q_1$. By induction hypothesis we have $\mathsf{UE}(\mathsf{clean}(Q, A), A') =$

$\mathsf{UE}(\mathsf{rec}\ x_u.\mathsf{clean}(Q_1, A), A') = \mathsf{UE}(\mathsf{clean}(Q_1, A), A') = \mathsf{UE}(Q_1, A') = \mathsf{UE}(Q, A')$.

<div align="right">□</div>

Lemma 7.10 Let $Q \in \mathsf{L}(\tilde{\mathbb{P}})$. Then $\mathsf{UE}(\mathsf{clean}(Q), A) = \mathsf{UE}(Q, A)$ for every $A \subseteq \mathbb{A}_\tau$.

Proof: This follows directly from Proposition 7.9-2 <div align="right" style="display:inline">□</div>

8 Appendix B: A Proof of Proposition 3.11

This section is devoted to proving Proposition 3.11.

Proposition 3.11 Let $Q, Q' \in \mathsf{L}(\tilde{\mathbb{P}})$ and $A \subseteq \mathbb{A}_\tau$. Then:

1. $Q \xrightarrow{\alpha} Q'$ and $s = \langle v_1, \ldots, v_n \rangle \in \mathsf{UE}(Q, A)$ implies either $s \in \mathsf{UE}(Q', A)$ or there exists some $j \in [1, n]$ such that $v_j \notin \mathsf{LAB}(Q')$.

2. $Q \xrightarrow{X}_r Q'$ implies $\mathsf{LE}(Q, A) = \mathsf{LE}(Q', A) = \mathsf{UE}(Q', A)$.

Proof:
   We start proving Item 1. by induction on derivation $Q \xrightarrow{\alpha} Q'$.

Nil, Var: $Q = \mathsf{nil}_u$, $Q = x_u$. This case is not possible since $Q \not\xrightarrow{\alpha}$.

Pref: $Q = \alpha_u.P_1$ or $Q = \underline{\alpha}_u.P_1$ with $P \in \mathsf{L}_{u1}(\tilde{\mathbb{P}}_1)$. Of course only the latter case is possible. Then $Q \xrightarrow{\alpha} P_1$, $s \in \mathsf{UE}(Q, A)$ implies $\alpha \notin A$ and $s = \langle u \rangle$. By Fact 3.5-2, $u \notin \mathsf{LAB}(P_1)$.

Sum: $Q = Q_1 +_u Q_2$. By the operational semantics $Q_1 +_u Q_2 \xrightarrow{\alpha} Q'$ if either $Q_1 \xrightarrow{\alpha} Q'$ or $Q_2 \xrightarrow{\alpha} Q'$. Assume the former case. The other one is similar. If $s \in \mathsf{UE}(Q_1, A)$ then by induction hypothesis either $s \in \mathsf{UE}(Q', A)$ or $v_j \notin \mathsf{LAB}(Q')$ for some $j \in [1, n]$. Then assume $s \in \mathsf{UE}(Q_2, A)$. Moreover, by the labelling function, each $v_j \in \mathsf{LAB}(Q_1)$ is of the form $v_j = u1u_j$ while each $v_j \in \mathsf{LAB}(Q_2)$ is of the form $v_j = u2u_j$, for some label u_j. Then by Fact 3.5-2, we have $v_j \notin \mathsf{LAB}(Q_1)$ and $v_j \notin \mathsf{LAB}(Q')$, for any $j \in [1, n]$.

Par: $Q = Q_1 \|_B^u Q_2$. Assume $Q \xrightarrow{\alpha} Q'$ and $s = \langle v_1, \ldots, v_n \rangle \in \mathsf{UE}(Q, A)$.

   By $s \in \mathsf{UE}(Q_1\|_B^u Q_2, A)$ we have either (i) $s \in \mathsf{UE}(Q_1, A \cup B)$, (ii) $s \in \mathsf{UE}(Q_2, A \cup B)$ or (iii) $s \in \mathsf{UE}(Q_1, \mathbb{A}_\tau\backslash\{\alpha\}) \times \mathsf{UE}(Q_2, \mathbb{A}_\tau\backslash\{\alpha\})$, for some $\alpha \in B\backslash A$. Now consider the following three possible cases:

   - $\alpha \notin B$, $Q_1 \xrightarrow{\alpha} Q_1'$ and $Q' = \mathsf{clean}(Q_1'\|_B^u Q_2)$. Consider case (i). By induction hypothesis either $s \in \mathsf{UE}(Q_1', A \cup B)$ or $v_j \notin \mathsf{LAB}(Q_1')$ for some $j \in [1, n]$. If $s \in \mathsf{UE}(Q_1', A\cup B)$ then $s \in \mathsf{UE}(Q_1'\|_B^u Q_2, A) = \mathsf{UE}(\mathsf{clean}(Q_1'\|_B^u Q_2), A) = \mathsf{UE}(Q', A)$ by Lemma 7.10. Assume that, for some $j \in [1, n]$, $v_j \notin \mathsf{LAB}(Q_1')$. Since $s \in \mathsf{UE}(Q_1, A \cup B)$ implies also $v_j \neq u$ and $v_j \notin \mathsf{LAB}(Q_2)$ (Fact 3.4), we have that $v_j \notin \mathsf{LAB}(Q_1'\|_B^u Q_2) = \mathsf{LAB}(\mathsf{clean}(Q_1'\|_B^u Q_2)) = \mathsf{LAB}(Q')$.
   Consider case (ii). Then, again by Lemma 7.10, we have that $s \in \mathsf{UE}(Q_2, A\cup B) \subseteq \mathsf{UE}(Q_1'\|_B^u Q_2, A) = \mathsf{UE}(\mathsf{clean}(Q_1'\|_B^u Q_2), A) = \mathsf{UE}(Q', A)$.
   Then consider case (iii). By definition of $\times$, $s \in \mathsf{UE}(Q_1, \mathbb{A}_\tau\backslash\{\alpha\})\times\mathsf{UE}(Q_2, \mathbb{A}_\tau\backslash\{\alpha\})$ implies there exist s_1 and s_2 process labels such that $s_1 \in \mathsf{UE}(Q_1, \mathbb{A}_\tau\backslash\{\alpha\})$ and $s_2 \in \mathsf{UE}(Q_2, \mathbb{A}_\tau\backslash\{\alpha\})$. By induction hypothesis, either $s_1 \in \mathsf{UE}(Q_1', \mathbb{A}_\tau\backslash\{\alpha\})$ or $v_j \notin \mathsf{LAB}(Q_1')$ for some $j \in [1, n]$. If $s_1 \in \mathsf{UE}(Q_1', \mathbb{A}_\tau\backslash\{\alpha\})$ then $s_1 \times s_2 \in \mathsf{UE}(Q_1', \mathbb{A}_\tau\backslash\{\alpha\})\times\mathsf{UE}(Q_2, \mathbb{A}_\tau\backslash\{\alpha\}) \subseteq \mathsf{UE}(Q_1'\|_B^u Q_2, A) = \mathsf{UE}(\mathsf{clean}(Q_1'\|_B^u Q_2), A) = \mathsf{UE}(Q', A)$, again by Lemma 7.10. If $v_j \notin \mathsf{LAB}(Q_1')$ as in the previous cases we have that $v_j \notin \mathsf{LAB}(Q_1'\|_B^u Q_2) = \mathsf{LAB}(Q')$.

   - $\alpha \notin B$, $Q_2 \xrightarrow{\alpha} Q_2'$ and $Q' = \mathsf{clean}(Q_1 \|_B^u Q_2')$. This case is similar to the previous one.

- $\alpha \in B$, $Q_i \xrightarrow{\alpha} Q_i'$ for $i = 1, 2$ and $Q' = \mathsf{clean}(Q_1' \|_B^u Q_2')$. Consider case (i). By induction hypothesis either $s \in \mathsf{UE}(Q_1', A \cup B)$ or $v_j \notin \mathsf{LAB}(Q_1')$ for some $j \in [1, n]$. As in the previous items, we can prove that $s \in \mathsf{UE}(Q_1', A \cup B)$ implies $s \in \mathsf{UE}(Q_1' \|_B Q_2', A) = \mathsf{UE}(Q', A)$ and $v_j \notin \mathsf{LAB}(Q_1')$ implies $v_j \notin \mathsf{LAB}(Q_1' \|_B^u Q_2') = \mathsf{LAB}(Q')$. Case (ii) can be proven similarly. Then consider case (iii). By definition of $\times$, $s \in \mathsf{UE}(Q_1, \mathbb{A}_\tau \backslash \{\alpha\}) \times \mathsf{UE}(Q_2, \mathbb{A}_\tau \backslash \{\alpha\})$ implies there exist s_1 and s_2 process labels such that $s_i = \langle v_{i1}, \dots, v_{in_i} \rangle \in \mathsf{UE}(Q_i, \mathbb{A}_\tau \backslash \{\alpha\})$. By induction hypothesis, if both $s_1 \in \mathsf{UE}(Q_1', \mathbb{A}_\tau \backslash \{\alpha\})$ and $s_2 \in \mathsf{UE}(Q_2', \mathbb{A}_\tau \backslash \{\alpha\})$ then $s_1 \times s_2 \in \mathsf{UE}(Q_1', \mathbb{A}_\tau \backslash \{\alpha\}) \times \mathsf{UE}(Q_2', \mathbb{A}_\tau \backslash \{\alpha\}) \subseteq \mathsf{UE}(Q_1' \|_B^u Q_2', A) = \mathsf{UE}(Q', A)$. If $v_{1j} \notin \mathsf{LAB}(Q_1')$ (similarly for $v_{2j} \notin \mathsf{LAB}(Q_2')$) then, as in the Sum case, we have that $v_{1j} \notin \mathsf{LAB}(Q_2')$ and hence also $v_{1j} \notin \mathsf{LAB}(Q')$.

Rel: $Q = Q_1[\Phi_u]$. By the operational semantics, $Q \xrightarrow{\alpha} Q_1'[\Phi_u]$ if there exists $\beta \in \Phi^{-1}(\alpha)$ such that $Q_1 \xrightarrow{\beta} Q_1'$. $s = \langle v_1, \dots v_n \rangle \in \mathsf{UE}(Q_1[\Phi_u], A)$ if $s \in \mathsf{UE}(Q_1, \Phi^{-1}(A))$. By induction hypothesis either $s \in \mathsf{UE}(Q_1', \Phi^{-1}(A))$ or $v_j \notin \mathsf{LAB}(Q_1')$ for some $j \in [1, n]$. In the former case $s \in \mathsf{UE}(Q_1'[\Phi_u], A)$. In the latter one, $v_j \in \mathsf{LAB}(Q_1)$ implies $v \neq u$ and hence also $v_j \notin \mathsf{LAB}(Q_1'[\Phi_u])$.

Rec: $Q = \mathsf{rec}\ x_u.Q_1$. Let $S_1 = \mathsf{unmark}(Q_1)$ and $S = Q_1\{\!|\mathsf{rec}\ x_u.S_1/x|\!\}$. By operational rules $Q \xrightarrow{\alpha} Q'$ implies $S \xrightarrow{\alpha} Q'$. Now assume $s \in \mathsf{UE}(Q, A) = \mathsf{UE}(Q_1, A) = \mathsf{UE}(S, A)$ by Proposition 7.3-2. By induction hypothesis we have either $s \in \mathsf{UE}(Q', A)$ or $v_j \notin \mathsf{LAB}(Q')$ for some $j \in [1, n]$.

Then prove Item 2. The two statements are proven by induction on Q.

Var, Stop: $Q = x_u$. This case is not possible since $Q \xnrightarrow{X}$.

Nil: $Q = \mathsf{nil}_u$. In this case $Q \xrightarrow{X} \mathsf{nil} = Q'$ and $\mathsf{LE}(Q, A) = \mathsf{LE}(Q', A) = \mathsf{UE}(Q', A) = \emptyset$ for any $A \subseteq \mathbb{A}_\tau$.

Pref: $Q = \alpha_u.P_1$ or $Q = \underline{\alpha}_u.P_1$. We prove only the latter case (the former case can be proven similarly). Assume that $Q \xrightarrow{X} \underline{\alpha}_u.P_1 = Q'$. If $\alpha \notin A$ then $\mathsf{LE}(Q, A) = \mathsf{LE}(Q', A) = \mathsf{UE}(Q', A) = \{\langle u \rangle\}$, otherwise $\mathsf{LE}(Q, A) = \mathsf{LE}(Q', A) = \mathsf{UE}(Q', A) = \emptyset$

Sum: $Q = Q_1 +_u Q_2$. By the operational semantics $Q \xrightarrow{X} Q_1' + Q_2'$ implies $Q_1 \xrightarrow{X} Q_1'$ and $Q_2 \xrightarrow{X} Q_2'$. By induction hypothesis
$\mathsf{LE}(Q, A) = \mathsf{LE}(Q_1, A) \cup \mathsf{LE}(Q_2, A) = \mathsf{LE}(Q_1', A) \cup \mathsf{LE}(Q_2', A) = \mathsf{LE}(Q', A)$. Similarly $\mathsf{LE}(Q', A) = \mathsf{LE}(Q_1', A) \cup \mathsf{LE}(Q_2', A) = \mathsf{UE}(Q_1', A) \cup \mathsf{UE}(Q_2', A) = \mathsf{UE}(Q', A)$.

Par: $Q = Q_1 \|_A^u Q_2$. In this case $Q \xrightarrow{X} Q'$ if $X \subseteq (A \cap (X_1 \cup X_2)) \cup ((X_1 \cap X_2) \backslash A)$, $Q_1 \xrightarrow{X_1} Q_1'$ and $Q_2 \xrightarrow{X_2} Q_2'$ and $Q' = Q_1' \|_A^u Q_2'$. By induction hypothesis we have $\mathsf{LE}(Q_i, A) = \mathsf{LE}(Q_i', A) = \mathsf{UE}(Q_i', A)$, for $i = 1, 2$. Then $\mathsf{LE}(Q, A) =$
$\mathsf{LE}(Q_1, A \cup B) \cup \mathsf{LE}(Q_2, A \cup B) \cup \bigcup_{\alpha \in B \backslash A} \mathsf{LE}(Q_1, \mathbb{A}_\tau \backslash \{\alpha\}) \times \mathsf{LE}(Q_2, \mathbb{A}_\tau \backslash \{\alpha\}) =$
$\mathsf{LE}(Q_1', A \cup B) \cup \mathsf{LE}(Q_2', A \cup B) \cup \bigcup_{\alpha \in B \backslash A} \mathsf{LE}(Q_1', \mathbb{A}_\tau \backslash \{\alpha\}) \times \mathsf{LE}(Q_2', \mathbb{A}_\tau \backslash \{\alpha\}) =$
$\mathsf{LE}(Q_1' \|_A Q_2', A) = \mathsf{LE}(\mathsf{clean}(Q_1' \|_A Q_2'), A) = \mathsf{LE}(Q', A)$, by Proposition 7.9-1. Similarly we can prove that $\mathsf{LE}(Q', A) = \mathsf{UE}(Q_1' \|_A Q_2', A) = \mathsf{UE}(\mathsf{clean}(Q_1' \|_A Q_2'), A) = \mathsf{UE}(Q', A)$, by Lemma 7.10.

Rel: $Q = Q_1[\Phi_u]$. $Q \xrightarrow{X} Q'_1[\Phi_u]$ implies $Q_1 \xrightarrow{\Phi^{-1}(X \cup \{\tau\}) \setminus \{\tau\}} Q'_1$. By induction hypothesis $\mathsf{LE}(Q_1, A) = \mathsf{LE}(Q'_1, A) = \mathsf{UE}(Q'_1, A)$. Thus $\mathsf{LE}(Q, A) = \mathsf{LE}(Q_1, \Phi^{-1}(A)) = \mathsf{LE}(Q'_1, \Phi^{-1}(A)) = \mathsf{LE}(Q', A)$. Similarly, $\mathsf{LE}(Q'_1, \Phi^{-1}(A)) = \mathsf{UE}(Q'_1, \Phi^{-1}(A)) = \mathsf{UE}(Q', A)$.

Rec: $Q = \mathsf{rec}\, x_u.Q_1$. $Q \xrightarrow{X}_r Q'$ implies $Q_1 \xrightarrow{X}_r Q'_1$ and $Q' = \mathsf{rec}\, x_u.Q'_1$. By induction hypothesis $\mathsf{LE}(Q', A) = \mathsf{LE}(Q'_1, A) = \mathsf{LE}(Q_1, A) = \mathsf{LE}(Q, A)$ and $\mathsf{UE}(Q', A) = \mathsf{UE}(Q'_1, A) = \mathsf{LE}(Q_1, A) = \mathsf{LE}(Q, A)$.

$\square$

9   Appendix C: A Proof of Proposition 3.12

This section is devoted to proving Proposition 3.12. A preliminary lemma is needed.

Lemma 9.1 Let $Q \in \mathsf{L}(\tilde{\mathbb{P}})$, X and $Y \subseteq \mathbb{A}$.

1. $Q \xrightarrow{X}_r Q'$ and $Y \cap \mathcal{U}(Q, A) = \emptyset$ imply $Q \xrightarrow{X \cup (Y \backslash A)}_r Q'$;

2. $Q \xrightarrow{X}_r$ implies $\mathcal{U}(Q, \mathbb{A} \backslash X) = \emptyset$;

3. Q guarded and $\mathcal{U}(Q, \mathbb{A} \backslash X) = \emptyset$ implies $Q \xrightarrow{X}_r$.

Proof: We prove these items together by induction on Q.

Var: $Q = x_u$. These cases are not possible since $Q \not\xrightarrow{X}_r$ and Q is not guarded.

Nil: $Q = \mathsf{nil}_u$. In this case:

1. $Q \xrightarrow{X}_r \mathsf{nil}$, $Y \cap \mathcal{U}(Q, A) = \emptyset$ and $Q \xrightarrow{X \cup (Y \backslash A)}_r \mathsf{nil}$.
2. $Q \xrightarrow{X}_r$ and $\mathcal{U}(Q, \mathbb{A} \backslash X) = \emptyset$.
3. Q is guarded, $\mathcal{U}(Q, \mathbb{A} \backslash X) = \emptyset$ and $Q \xrightarrow{X}_r$.

Pref: $Q = \alpha_u.P_1$ or $Q = \underline{\alpha}_u.P_1$. Consider only the second case (the first case is similar to the Nil-case).

1. $Q \xrightarrow{X}_r Q'$ implies, by operational semantics, $\alpha \notin X \cup \{\tau\}$ and $Q' = \underline{\alpha}_u.P_1$. If $\alpha \in A$ then, trivially, $\alpha \notin Y \backslash A$. Otherwise, $\alpha \notin A$ and $Y \cap \mathcal{U}(Q, A) = Y \cap \{\alpha\} = \emptyset$ imply $\alpha \notin Y$ and, again, $\alpha \notin Y \backslash A$. Thus, $\alpha \notin (X \cup (Y \backslash A)) \cup \{\tau\}$ and, by operational rules, $Q \xrightarrow{X \cup (Y \backslash A)}_r \underline{\alpha}_u.P_1 = Q'$

2. $Q \xrightarrow{X}_r$ implies $\alpha \notin X \cup \{\tau\}$ and, hence, $\alpha \in \mathbb{A} \backslash X$. Then $\mathcal{U}(Q, \mathbb{A} \backslash X) = \emptyset$.

3. Q is guarded. Moreover, $\mathcal{U}(Q, \mathbb{A} \backslash X) = \emptyset$ implies $\alpha \in \mathbb{A} \backslash X$. Since $\tau \neq \alpha \in \mathbb{A}$ and $\alpha \notin X$, we have that $\alpha \notin X \cup \{\tau\}$ and, hence, $Q \xrightarrow{X}_r$.

Sum: $Q = Q_1 +_u Q_2$

1. $Q \xrightarrow{X}_r Q'$ and $Y \cap \mathcal{U}(Q, A) = Y \cap (\mathcal{U}(Q_1, A) \cup \mathcal{U}(Q_2, A)) = \emptyset$ imply $Q_i \xrightarrow{X}_r Q'_i$, $Y \cap \mathcal{U}(Q_i, A) = \emptyset$ for $i = 1, 2$ and $Q' = Q'_1 +_u Q'_2$. By induction hypothesis $Q_1 \xrightarrow{X \cup (Y \backslash A)}_r Q'_1$, $Q_2 \xrightarrow{X \cup (Y \backslash A)}_r Q'_2$. Thus, by operational rules, $Q \xrightarrow{X \cup (Y \backslash A)}_r Q'_1 + Q'_2 = Q'$.

2. If $Q \xrightarrow{X}_r$ then $Q_1 \xrightarrow{X}_r$ and $Q_2 \xrightarrow{X}_r$. By induction hypothesis $\mathcal{U}(Q_1, \mathbb{A} \backslash X) = \emptyset$, $\mathcal{U}(Q_2, \mathbb{A} \backslash X) = \emptyset$ and, hence, $\mathcal{U}(Q, \mathbb{A} \backslash X) = \emptyset$.

3. In this case Q guarded implies both Q_1 and Q_2 guarded. Moreover if $\mathcal{U}(Q, \mathbb{A} \backslash X) = \emptyset$ then $\mathcal{U}(Q_1, \mathbb{A} \backslash X) = \mathcal{U}(Q_2, \mathbb{A} \backslash X) = \emptyset$. By induction hypothesis $Q_1 \xrightarrow{X}_r$, $Q_2 \xrightarrow{X}_r$ and, hence, $Q \xrightarrow{X}_r$.

Par: $Q = Q_1 \|_B^u Q_2$.

1. If $Q \xrightarrow{X}_r Q'$ then there exist X_1, X_2 such that $Q_1 \xrightarrow{X_1}_r Q'_1$, $Q_2 \xrightarrow{X_2}_r Q'_2$, $X \subseteq (B \cap (X_1 \cup X_2)) \cup ((X_1 \cap X_2) \backslash B)$ and $Q' = \mathsf{clean}(Q'_1 \parallel^u_B Q'_2)$. Now, let $A_1 = (\mathcal{U}(Q_1) \backslash \mathcal{U}(Q_2)) \cap B$ and $A_2 = (\mathcal{U}(Q_2) \backslash \mathcal{U}(Q_1)) \cap B$. Now we want to prove that $Y \cap \mathcal{U}(Q, A) = \emptyset$ implies $Y \cap \mathcal{U}(Q_1, A \cup A_1) = Y \cap \mathcal{U}(Q_2, A \cup A_2) = \emptyset$.

   Assume $Y \cap \mathcal{U}(Q, A) = \emptyset$ and, by contradiction, $\mu \in Y \cap \mathcal{U}(Q_1, A \cup A_1) \neq \emptyset$. Then, by Proposition 7.1-1, $\mu \in Y$ such that $\mu \in \mathcal{U}(Q_1)$, $\mu \notin A$ and $\mu \notin A_1$. We have two possible subcases. If $\mu \notin B$ then $\mu \in \mathcal{U}(Q_1)$ and $\mu \notin A \cup B$ implies $\mu \in \mathcal{U}(Q_1, A \cup B) \subseteq \mathcal{U}(Q, A)$ (see Proposition 7.1-2). Otherwise, if $\mu \in B$ then $\mu \in \mathcal{U}(Q_1)$ and $\mu \notin A_1$ implies also $\mu \in \mathcal{U}(Q_2)$. Moreover $\mu \in B \backslash A$ implies $\mu \notin \mathbb{A}_\tau \backslash (B \backslash A)$. As above, $\mu \in \mathcal{U}(Q_1)$, $\mu \in \mathcal{U}(Q_2)$ and $\mu \notin \mathbb{A}_\tau \backslash (B \backslash A)$ imply $\mu \in \mathcal{U}(Q_1, \mathbb{A}_\tau \backslash (B \backslash A) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau \backslash (B \backslash A) \subseteq \mathcal{U}(Q, A)$. In both cases $\mu \in Y \cap \mathcal{U}(Q, A)$ which contradicts the assumption $Y \cap \mathcal{U}(Q, A) = \emptyset$.

   This prove that $Y \cap \mathcal{U}(Q, A) = \emptyset$ implies $Y \cap \mathcal{U}(Q_1, A \cup A_1) = \emptyset$. Similarly we can prove that $Y \cap \mathcal{U}(Q, A) = \emptyset$ implies also $Y \cap \mathcal{U}(Q_2, A \cup A_2) = \emptyset$. By induction hypothesis $Q_1 \xrightarrow{X_1 \cup (Y \backslash (A \cup A_1))}_r Q'_1$ and $Q_2 \xrightarrow{X_2 \cup (Y \backslash (A \cup A_2))}_r Q'_2$.

   Let $X'_1 = X_1 \cup (Y \backslash (A \cup A_1))$ and $X'_2 = X_2 \cup (Y \backslash (A \cup A_2))$. By operational rules it remains to prove that $X \cup (Y \backslash A) \subseteq (B \cap (X'_1 \cup X'_2)) \cup ((X'_1 \cap X'_2) \backslash B)$. First of all, $X_1 \subseteq X'_1$ and $X_2 \subseteq X'_2$ imply $X \subseteq (B \cap (X_1 \cup X_2)) \cup ((X_1 \cap X_2) \backslash B) \subseteq (B \cap (X'_1 \cup X'_2)) \cup ((X'_1 \cap X'_2) \backslash B)$. Now assume $\mu \in Y \backslash A$ and consider the following possible subcases:

   - $\mu \notin B$. Then $\mu \notin A_1, A_2$ implies $\mu \in Y \backslash (A \cup A_1) \subseteq X'_1$ and $\mu \in Y \backslash (A \cup A_2) \subseteq X'_2$. Thus, $\mu \in (X'_1 \cap X'_2) \backslash B$.
   - $\mu \in B$. In this case $\mu \notin A_1$ implies $\mu \in Y \backslash (A \cup A_1) \subseteq X'_1$. If $\mu \in A_1$, then $\mu \notin \mathcal{U}(Q_2)$ implies $\mu \notin A_2$ and, hence, $\mu \in Y \backslash (A \cup A_2) \subseteq X'_2$. In both cases $\mu \in B \cap (X'_1 \cup X'_2)$.

   We can conclude that $Q \xrightarrow{X \cup (Y \backslash A)}_r Q'$.

2. If $Q \xrightarrow{X}_r$ then $Q_1 \xrightarrow{X_1}_r$, $Q_2 \xrightarrow{X_2}_r$ with $X \subseteq (B \cap (X_1 \cup X_2)) \cup ((X_1 \cap X_2) \backslash B)$. By induction hypothesis we have that $\mathcal{U}(Q_1, \mathbb{A} \backslash X_1) = \mathcal{U}(Q_2, \mathbb{A} \backslash X_2) = \emptyset$. Moreover, for generic sets A, B and C we have that $A \backslash (B \backslash C) = (A \backslash B) \cup (A \cap C)$. Thus:

   - $X \backslash B \subseteq (X_1 \cup X_2) \backslash B = X_1 \cup X_2 \subseteq X_i$ for $i = 1, 2$. Thus $\mathbb{A} \backslash X_i \subseteq \mathbb{A} \backslash (X \backslash B) = (\mathbb{A} \backslash X) \cup (\mathbb{A} \cap B) = (\mathbb{A} \backslash X) \cup B$. Hence, by Proposition 7.1-3, we have that $\mathcal{U}(Q_i, (\mathbb{A} \backslash X) \cup B) \subseteq \mathcal{U}(Q_i, \mathbb{A} \backslash X_i) = \emptyset$ for $i = 1, 2$.
   - $B \backslash (\mathbb{A} \backslash X) = (B \backslash \mathbb{A}) \cup (B \cap X) = B \cap X \subseteq B \cap (X_1 \cup X_2) \subseteq X_1 \cup X_2$ and, hence, $\mathbb{A}_\tau \backslash (X_1 \cup X_2) \subseteq \mathbb{A}_\tau \backslash (B \backslash (\mathbb{A} \backslash X))$. Assume, by contradiction, $\mu \in \mathcal{U}(Q_1, \mathbb{A}_\tau \backslash (B \backslash (\mathbb{A} \backslash X))) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau \backslash (B \backslash (\mathbb{A} \backslash X))) \neq \emptyset$.
     Then, $\mu \notin \mathbb{A}_\tau \backslash (B \backslash (\mathbb{A} \backslash X))$ (see Proposition 7.1-1) and $\mathbb{A}_\tau \backslash (X_1 \cup X_2) \subseteq \mathbb{A}_\tau \backslash (B \backslash (\mathbb{A} \backslash X))$ imply $\mu \in X_1 \cup X_2$. If $\mu \in X_1$ then $\mu \in \mathcal{U}(Q_1, A)$, $\mu \notin \mathbb{A} \backslash X_1$ and Proposition 7.1-2 imply $\mu \in \mathcal{U}(Q_1, \mathbb{A} \backslash X_1) = \emptyset$. Similarly, $\mu \in X_2$ implies $\mu \in \mathcal{U}(Q_2, \mathbb{A} \backslash X_2) = \emptyset$. Both cases are not possible and thus we can conclude that $\mathcal{U}(Q_1, \mathbb{A}_\tau \backslash (B \backslash (\mathbb{A} \backslash X))) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau \backslash (B \backslash (\mathbb{A} \backslash X))) = \emptyset$.

   By Definition 2.3, we have that $\mathcal{U}(Q, \mathbb{A} \backslash X) = \emptyset$.

3. Assume Q guarded (and, hence, both Q_1 and Q_2 guarded) and $\mathcal{U}(Q, \mathbb{A} \backslash X) = \emptyset$. By Definition 2.3 we have $\mathcal{U}(Q_1, (\mathbb{A} \backslash X) \cup B) = \mathcal{U}(Q_2, (\mathbb{A} \backslash X) \cup B) = \emptyset$ and

$\mathcal{U}(Q_1, \mathbb{A}_\tau \backslash (B \backslash (\mathbb{A} \backslash X))) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau \backslash (B \backslash (\mathbb{A} \backslash X))) = \emptyset$. Moreover (see the proof of the previous case) $(\mathbb{A} \backslash X) \cup B = \mathbb{A} \backslash (X \backslash B)$ and $\mathbb{A}_\tau \backslash (B \backslash (\mathbb{A} \backslash X)) = \mathbb{A}_\tau \backslash (B \cap X)$.

Q_i guarded and $\mathcal{U}(Q_i, \mathbb{A} \backslash (X \backslash B)) = \emptyset$ imply, by induction hypothesis, $Q_i \xrightarrow{X \backslash B}_r$ for $i = 1, 2$. Now, let $X_i = B \backslash \mathcal{U}(Q_i) \subseteq B$ and $X_i' = (X \backslash B) \cup X_i$, for $i = 1, 2$. Then $Q_i \xrightarrow{X \backslash B}_r$ and $X_i \cap \mathcal{U}(Q_i) = \emptyset$ imply, by Item 1, $Q_i \xrightarrow{X_i'}_r$. Moreover:

- $B \cap (X_1' \cup X_2') = B \cap ((X \backslash B) \cup X_1 \cup X_2) = X_1 \cup X_2$;
- $(X_1' \cap X_2') \backslash B = ((X_1' \backslash B) \cap (X_2' \backslash B)) = ((X \backslash B) \cap (X \backslash B)) = X \backslash B$;
- Assume, by contradiction, $\mu \in B \cap X$ and $\mu \notin X_1 \cup X_2$. Then, $\mu \in B$ and $\mu \notin X_1 \cup X_2$ imply $\mu \in \mathcal{U}(Q_1)$ and $\mu \in \mathcal{U}(Q_2)$ and, since $\mu \in B \cap X$ implies $\mu \notin \mathbb{A}_\tau \backslash (B \cap X)$, we have $\mu \in \mathcal{U}(Q_1, \mathbb{A}_\tau \backslash (B \cap X)) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau \backslash (B \cap X)) = \emptyset$ (by Proposition 7.1-2). This is not possible and we can conclude that $B \cap X \subseteq X_1 \cup X_2$.

Finally: $X = (B \cap X) \cup (X \backslash B) \subseteq (X_1 \cup X_2) \cup (X \backslash B) =$
$(B \cap (X_1' \cup X_2')) \cup ((X_1' \cap X_2') \backslash B)$ and, by operational semantics, $Q \xrightarrow{X}_r$.

Rel: $Q = Q_1[\Phi_u]$. Let $X' = \Phi^{-1}(X \cup \{\tau\}) \backslash \{\tau\}$. First, we prove that $\Phi^{-1}(\mathbb{A} \backslash X) = \mathbb{A} \backslash X'$.

Let $\mu \in \Phi^{-1}(\mathbb{A} \backslash X)$. Then $\mu \in \mathbb{A}_\tau$ such that $\Phi(\mu) \in \mathbb{A}$ and $\Phi(\mu) \notin X$. In particular, $\Phi(\mu) \in \mathbb{A}$ implies $\mu \neq \tau$ (since $\Phi(\tau) = \tau \notin \mathbb{A}$) and $\Phi(\mu) \notin \{\tau\}$. Thus, $\mu \in \mathbb{A}$ such that $\Phi(\mu) \notin X \cup \{\tau\}$ and, hence, $\mu \notin \Phi^{-1}(X \cup \{\tau\}) \supseteq X'$. We can conclude that $\mu \in \mathbb{A} \backslash X'$.

Now, let $\mu \in \mathbb{A} \backslash X'$. Then $\mu \neq \tau$ and $\mu \notin X'$ imply $\mu \notin \Phi^{-1}(X \cup \{\tau\})$ and, hence, $\Phi(\mu) \notin X \cup \{\tau\}$. Finally, $\Phi(\mu) \notin X$ and $\Phi(\mu) \neq \tau$ imply $\Phi(\mu) \in \mathbb{A} \backslash X$ and, hence, $\mu \in \Phi^{-1}(\mathbb{A} \backslash X)$.

1. Assume that $Q \xrightarrow{X}_r Q_1'[\Phi_u] = Q'$ and $Y \cap \mathcal{U}(Q, A) = Y \cap \Phi(\mathcal{U}(Q_1, \Phi^{-1}(A))) = \emptyset$. Then $Q_1 \xrightarrow{X'}_r Q_1'$ and $\Phi^{-1}(Y) \cap \mathcal{U}(Q_1, \Phi^{-1}(A)) = \emptyset$. By induction hypothesis $Q_1 \xrightarrow{X' \cup (\Phi^{-1}(Y) \backslash \Phi^{-1}(A))}_r Q_1'$. Moreover, since $\Phi(\tau) = \tau$ and $Y \subseteq \mathbb{A}$ imply $\tau \notin \Phi^{-1}(Y)$, $\tau \notin \Phi^{-1}(Y) \backslash \Phi^{-1}(A) = \Phi^{-1}(Y \backslash A)$ and, hence, $\Phi^{-1}(Y \backslash A) = \Phi^{-1}(Y \backslash A) \backslash \{\tau\}$, we have
$X' \cup (\Phi^{-1}(Y) \backslash \Phi^{-1}(A)) = (\Phi^{-1}(X \cup \{\tau\}) \backslash \{\tau\}) \cup (\Phi^{-1}(Y \backslash A) \backslash \{\tau\}) =$
$(\Phi^{-1}(X \cup \{\tau\}) \cup \Phi^{-1}(Y \backslash A)) \backslash \{\tau\} = \Phi^{-1}(X \cup \{\tau\} \cup (Y \backslash A)) \backslash \{\tau\} =$
$\Phi^{-1}((X \cup (Y \backslash A)) \cup \{\tau\}) \backslash \{\tau\}$.
Then, by operational rules, $Q \xrightarrow{X \cup (Y \backslash A)}_r Q_1'[\Phi_u] = Q'$

2. Assume $Q \xrightarrow{X}_r$ and, hence, $Q_1 \xrightarrow{X'}_r$. By induction hypothesis $\mathcal{U}(Q_1, \mathbb{A} \backslash X') = \mathcal{U}(Q_1, \Phi^{-1}(\mathbb{A} \backslash X)) = \emptyset$. Thus $\mathcal{U}(Q, \mathbb{A} \backslash X) = \Phi(\mathcal{U}(Q_1, \Phi^{-1}(\mathbb{A} \backslash X))) = \emptyset$.

3. Assume Q guarded and $\mathcal{U}(Q, \mathbb{A} \backslash X) = \Phi(\mathcal{U}(Q_1, \Phi^{-1}(\mathbb{A} \backslash X))) = \emptyset$. Then we have also Q_1 guarded and $\mathcal{U}(Q_1, \Phi^{-1}(\mathbb{A} \backslash X)) = \mathcal{U}(Q_1, \mathbb{A} \backslash X') = \emptyset$. By induction hypothesis $Q_1 \xrightarrow{X'}_r$ ad, hence, $Q \xrightarrow{X}_r$.

Rec: $Q = \mathsf{rec}\ x_u.Q_1$.

1. If $Q \xrightarrow{X}_r Q'$ then $Q_1 \xrightarrow{X}_r Q_1'$ and $Q' = \mathsf{rec}\ x_u.Q_1'$. Now assume $Y \cap \mathcal{U}(Q, A) = Y \cap \mathcal{U}(Q_1) = \emptyset$. By induction hypothesis we have that $Q_1 \xrightarrow{X \cup (Y \backslash A)}_r Q_1'$ and, by operational semantics, $Q \xrightarrow{X \cup (Y \backslash A)}_r \mathsf{rec}\ x_u.Q_1' = Q'$.

2. Assume $Q \xrightarrow{X}_r$. Then, by operational semantics, we have that $Q_1 \xrightarrow{X}_r$ and hence, by induction hypothesis, $\mathcal{U}(Q, \mathbb{A}\backslash X) = \mathcal{U}(Q_1, \mathbb{A}\backslash X) = \emptyset$.

3. Q guarded implies Q_1 guarded. Moreover, $\mathcal{U}(Q, \mathbb{A}\backslash X) = \mathcal{U}(Q_1, \mathbb{A}\backslash X) = \emptyset$ implies, by induction hypothesis, $Q_1 \xrightarrow{X}_r$ and, hence, $Q \xrightarrow{X}_r$.

$\square$

Proposition 3.12 Let $Q \in \mathsf{L}(\tilde{\mathbb{P}})$ and $A \subseteq \mathbb{A}_\tau$. Then:

1. $Q \xrightarrow{1}$ implies $\mathcal{U}(Q) = \emptyset$;

2. Q guarded and $\mathcal{U}(Q) = \emptyset$ implies $Q \xrightarrow{1}$;

3. $\mathcal{U}(Q, A) = \emptyset$ if and only if $\mathsf{UE}(Q, A) = \emptyset$.

Proof: Items 1. and 2. are corollary of Lemma 9.1. Item 3. is proven by induction on Q.

Nil, Var: $Q = \mathsf{nil}_u$, $Q = x_u$. In these cases $\mathcal{U}(Q, A) = \emptyset$ and $\mathsf{UE}(Q, A) = \emptyset$.

Pref: $Q = \alpha_u.P_1$ or $Q = \underline{\alpha}_u.P_1$. Consider only the latter case (the former is similar to the previous ones). $\mathcal{U}(Q, A) = \emptyset$ if and only if $\alpha \in A$ and, hence, if and only if $\mathsf{UE}(Q, A) = \emptyset$.

Sum: $Q = Q_1 +_u Q_2$. $\mathcal{U}(Q, A) = \emptyset$ iff $\mathcal{U}(Q_1, A) = \emptyset$ and $\mathcal{U}(Q_2, A) = \emptyset$, iff, by induction hypothesis, $\mathsf{UE}(Q_1, A) = \emptyset$, $\mathsf{UE}(Q_2, A) = \emptyset$ and $\mathsf{UE}(Q, A) = \emptyset$.

Par: $Q = Q_1\|_B^u Q_2$. $\mathcal{U}(Q, A) = \emptyset$ iff $\mathcal{U}(Q_1, A \cup B) = \emptyset$, $\mathcal{U}(Q_2, A \cup B) = \emptyset$, and $A = \mathcal{U}(Q_1, \mathbb{A}_\tau\backslash(B\backslash A)) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau\backslash(B\backslash A)) = \emptyset$. By induction hypothesis $\mathcal{U}(Q_i, A \cup B) = \emptyset$ iff $\mathsf{UE}(Q_i, A \cup B) = \emptyset$.

Now, we prove that $A = \emptyset$ iff $\bigcup_{\alpha \in B\backslash A}(\mathsf{UE}(Q_1, \mathbb{A}_\tau\backslash\{\alpha\}) \times \mathsf{UE}(Q_2, \mathbb{A}_\tau\backslash\{\alpha\})) = \emptyset$. We need the following statements:

(i) $A = \emptyset$ if and only if $A_\alpha = \mathcal{U}(Q_1, \mathbb{A}_\tau\backslash\{\alpha\}) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau\backslash\{\alpha\}) = \emptyset$ for any $\alpha \in B\backslash A$. Assume $A = \emptyset$ and let $\alpha \in B\backslash A$. $\{\alpha\} \subseteq B\backslash A$ implies $\mathbb{A}_\tau\backslash(B\backslash A) \subseteq \mathbb{A}_\tau\backslash\{\alpha\}$ and, by Proposition 7.1-3, $\mathcal{U}(Q_i, \mathbb{A}_\tau\backslash\{\alpha\}) \subseteq \mathcal{U}(Q_i, \mathbb{A}_\tau\backslash(B\backslash A))$ for $i = 1, 2$. Thus $A_\alpha = \mathcal{U}(Q_1, \mathbb{A}_\tau\backslash\{\alpha\}) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau\backslash\{\alpha\}) \subseteq A = \emptyset$. On the other hand, assume $A_\alpha = \emptyset$ for any $\alpha \in B\backslash A$ and, by contradiction, $\beta \in A \neq \emptyset$. We have that $\beta \in \mathcal{U}(Q_i, \mathbb{A}_\tau\backslash(B\backslash A))$ for $i = 1, 2$. By Proposition 7.1-1, $\beta \notin \mathbb{A}_\tau\backslash(B\backslash A)$ and, hence, $\beta \in B\backslash A$. Moreover, clearly, $\beta \notin \mathbb{A}_\tau\backslash\{\beta\}$. Then $\beta \in \mathcal{U}(Q_i, \mathbb{A}_\tau\backslash(B\backslash A))$ implies, by Proposition 7.1-2, $\beta \in \mathcal{U}(Q_i, \mathbb{A}_\tau\backslash\{\beta\})$ for $i = 1, 2$. Thus $\beta \in B\backslash A$ and $\beta \in A_\beta = \emptyset$.

(ii) $A_\alpha = \emptyset$ if and only if either $\mathcal{U}(Q_1, \mathbb{A}_\tau\backslash\{\alpha\}) = \emptyset$ or $\mathcal{U}(Q_2, \mathbb{A}_\tau\backslash\{\alpha\}) = \emptyset$. Since $\mu \in \mathcal{U}(Q_i, \mathbb{A}_\tau\backslash\{\alpha\})$ implies, again by Proposition 7.1-1, $\mu \notin \mathbb{A}_\tau\backslash\{\alpha\}$ and, hence, $\mu = \alpha$, we have that $\mathcal{U}(Q_i, \mathbb{A}_\tau\backslash\{\alpha\}) \subseteq \{\alpha\}$, for $i = 1, 2$. Then $A_\alpha = \emptyset$ iff $\mathcal{U}(Q_1, \mathbb{A}_\tau\backslash\{\alpha\}) = \emptyset$ or $\mathcal{U}(Q_2, \mathbb{A}_\tau\backslash\{\alpha\}) = \emptyset$.

(i) and (ii) together prove that $A = \emptyset$ iff, for any $\alpha \in B \backslash A$, either $\mathcal{U}(Q_1, \mathbb{A}_\tau \backslash \{\alpha\}) = \emptyset$ or $\mathcal{U}(Q_2, \mathbb{A}_\tau \backslash \{\alpha\}) = \emptyset$, iff, by induction hypothesis, either $\mathsf{UE}(Q_1, \mathbb{A}_\tau \backslash \{\alpha\}) = \emptyset$ or $\mathsf{UE}(Q_2, \mathbb{A}_\tau \backslash \{\alpha\}) = \emptyset$ and, hence, iff $\mathsf{UE}(Q_1, \mathbb{A}_\tau \backslash \{\alpha\}) \times \mathsf{UE}(Q_2, \mathbb{A}_\tau \backslash \{\alpha\}) = \emptyset$. We can conclude that $A = \emptyset$ iff $\bigcup_{\alpha \in B \backslash A}(\mathsf{UE}(Q_1, \mathbb{A}_\tau \backslash \{\alpha\}) \times \mathsf{UE}(Q_2, \mathbb{A}_\tau \backslash \{\alpha\})) = \emptyset$. Finally we have $\mathcal{U}(Q, A) = \emptyset$ iff $\mathsf{UE}(Q, A) = \emptyset$.

Rel: $Q = Q_1[\Phi_u]$. $\mathcal{U}(Q, A) = \Phi(\mathcal{U}(Q_1, \Phi^{-1}(A))) = \emptyset$ iff $\mathcal{U}(Q_1, \Phi^{-1}(A)) = \emptyset$ iff, by induction hypothesis, $\mathsf{UE}(Q_1, \Phi^{-1}(A)) = \mathsf{UE}(Q, A) = \emptyset$.

Rec: $Q = \mathsf{rec}\ x_u.Q_1$. In this case $\mathcal{U}(Q, A) = \mathcal{U}(Q_1, A) = \emptyset$ if and only if, by induction hypothesis, $\mathsf{UE}(Q_1, A) = \mathsf{UE}(Q, A) = \emptyset$.

$\square$

10 Appendix D: A Proof of Proposition 3.13

This section is devoted to proving Proposition 3.13. A preliminary lemma is needed.

Proposition 10.1 Let $Q, \in \mathsf{L}(\tilde{\mathbb{P}})$, $A, X \subseteq \mathbb{A}$. Then $\mathsf{clean}(Q, A) \xrightarrow{X}_r Q'$ implies $Q \xrightarrow{X \backslash A}_r Q'$.

Proof: By induction on $Q \in \mathsf{L}(\tilde{\mathbb{P}})$

Nil, Var: $Q = \mathsf{nil}_u$, $Q = x_u$. The latter case is not possible since $\mathsf{clean}(Q, A) = x_u \xcancel{\xrightarrow{X}}_r$.
Assume $Q = \mathsf{nil}_u$. Then $\mathsf{clean}(Q, A) = \mathsf{nil}_u \xrightarrow{X}_r \mathsf{nil}_u$ and $Q \xrightarrow{X \backslash A}_r \mathsf{nil}_u$.

Pref: $Q = \alpha_u.P_1$ or $Q = \underline{\alpha}_u.P_1$. We prove only the latter case (the former one is similar to the Nil-case). We have two possible subcases:

- $\alpha \in A$. In this case $\mathsf{clean}(Q, A) = \alpha_u.P_1 \xrightarrow{X}_r \underline{\alpha}_u.P_1$. Moreover $\alpha \in A \subseteq \mathbb{A}$ implies $\alpha \notin (X \backslash A) \cup \{\tau\}$ and, by operational rules, $Q \xrightarrow{X \backslash A}_r \underline{\alpha}_u.P_1$.

- $\alpha \notin A$ In this case $\mathsf{clean}(Q, A) = \underline{\alpha}_u.P_1 \xrightarrow{X}_r \underline{\alpha}_u.P_1$ implies $\alpha \notin X \cup \{\tau\} \supseteq (X \backslash A) \cup \{\tau\}$. By operational rules, $Q \xrightarrow{X \backslash A}_r \underline{\alpha}_u.P_1$.

Sum: $Q = Q_1 +_u Q_2$. By operational rules $\mathsf{clean}(Q, A) = \mathsf{clean}(Q_1, A) +_u \mathsf{clean}(Q_2, A) \xrightarrow{X}_r Q'$ implies $\mathsf{clean}(Q_1, A) \xrightarrow{X}_r Q_1'$, $\mathsf{clean}(Q_2, A) \xrightarrow{X}_r Q_2'$ and $Q' = Q_1' +_u Q_2'$. By induction hypothesis $Q_1 \xrightarrow{X \backslash A}_r Q_1'$, $Q_2 \xrightarrow{X \backslash A}_r Q_2'$ and, by operational rules, $Q \xrightarrow{X \backslash A}_r Q'$.

Par: $Q = Q_1 \|_B^u Q_2$. Let $A_1 = (\mathcal{U}(Q_1) \backslash \mathcal{U}(Q_2)) \cap B$ and $A_2 = (\mathcal{U}(Q_2) \backslash \mathcal{U}(Q_1)) \cap B$. Assume that $\mathsf{clean}(Q, A) = \mathsf{clean}(Q_1, A \cup A_1) \|_B^u \mathsf{clean}(Q_2, A \cup A_2) \xrightarrow{X}_r Q'$. Then there exist $X_1, X_2 \subseteq \mathbb{A}$ such that $\mathsf{clean}(Q_1, A \cup A_1) \xrightarrow{X_1}_r Q_1'$, $\mathsf{clean}(Q_2, A \cup A_2) \xrightarrow{X_2}_r Q_2'$, $X \subseteq (B \cap (X_1 \cup X_2)) \cup ((X_1 \cap X_2) \backslash B)$ and $Q' = \mathsf{clean}(Q_1' \|_B^u Q_2')$. By induction hypothesis we have that $Q_1 \xrightarrow{X_1 \backslash (A \cup A_1)}_r Q_1'$ and $Q_2 \xrightarrow{X_2 \backslash (A \cup A_2)}_r Q_2'$. Moreover, since $A_1 \cap \mathcal{U}(Q_2) = A_2 \cap \mathcal{U}(Q_1) = \emptyset$, by Proposition 9.1-1, we also have $Q_1 \xrightarrow{X_1'}_r Q_1'$ and $Q_2 \xrightarrow{X_2'}_r Q_2'$, where $X_1' = (X_1 \backslash (A \cup A_1)) \cup A_2$ and $X_2' = (X_2 \backslash (A \cup A_2)) \cup A_1$. By operational semantics, it remains to prove that $X \backslash A \subseteq (B \cap (X_1' \cup X_2')) \cup ((X_1' \cap X_2') \backslash B)$. Let $\mu \in X \backslash A$.

- $\mu \in B$. Then $\mu \in X$ implies either $\mu \in X_1$ or $\mu \in X_2$. Assume $\mu \in X_1$ (if $\mu \in X_2$ the statement can be proved similarly). If $\mu \notin A_1$ then, trivially, $\mu \in X_1 \backslash (A \cup A_1) \subseteq X_1'$. Otherwise, if $\mu \in A_1$, then $\mu \in (X_2 \backslash (A \cup A_2)) \cup A_1 = X_2'$. In both cases $\mu \in X_1' \cup X_2'$.

- $\mu \notin B$. Then $\mu \in X$ implies both $\mu \in X_1$ or $\mu \in X_2$. Moreover $\mu \notin A_1, A_2 \subseteq B$ implies $\mu \in X_1 \backslash (A \cup A_1) \subseteq X_1'$ and $\mu \in X_2 \backslash (A \cup A_2) \subseteq X_2'$. Thus $\mu \in X_1' \cap X_2'$.

Rel: $Q = Q_1[\Phi_u]$. Assume that $\mathsf{clean}(Q, A) = \mathsf{clean}(Q_1, \Phi^{-1}(A))[\Phi_u] \xrightarrow{X}_r Q'$. Then, by operational semantics, $\mathsf{clean}(Q_1, \Phi^{-1}(A)) \xrightarrow{X'}_r Q_1'$, with $X' = \Phi^{-1}(X \cup \{\tau\}) \backslash \{\tau\}$, and $Q' = Q_1'[\Phi_u]$. By induction hypothesis $Q_1 \xrightarrow{X' \backslash \Phi^{-1}(A)}_r Q_1'$. Moreover
$X' \backslash \Phi^{-1}(A) = (\Phi^{-1}(X \cup \{\tau\}) \backslash \{\tau\}) \backslash \Phi^{-1}(A) = (\Phi^{-1}(X \cup \{\tau\}) \backslash \Phi^{-1}(A)) \backslash \{\tau\} = (\Phi^{-1}((X \cup \{\tau\}) \backslash A)) \backslash \{\tau\} = (\Phi^{-1}((X \backslash A) \cup \{\tau\}) \backslash \{\tau\}$.

By operational rules $Q \xrightarrow{X \backslash A}_r Q_1'[\Phi_u] = Q'$.

Rec: $Q = \text{rec } x_u.Q_1$. $\text{clean}(Q, A) = \text{rec } x_u.\text{clean}(Q_1, A) \xrightarrow{X}_r Q'$ implies $\text{clean}(Q_1, A) \xrightarrow{X}_r Q_1'$ and $Q' = \text{rec } x_u.Q_1'$. By induction hypothesis $Q_1 \xrightarrow{X \backslash A}_r Q_1'$ and, by operational rules $Q \xrightarrow{X \backslash A}_r \text{rec } x_u.Q_1' = Q'$.

$\square$

Lemma 10.2 Let $Q, Q', Q'' \in \mathsf{L}(\tilde{\mathbb{P}})$ and $X, X' \subseteq \mathbb{A}$. Then:

1. $Q \xrightarrow{X}_r Q' \xrightarrow{X'}_r Q''$ implies $Q \not\xrightarrow{\mu}$ and $Q' \not\xrightarrow{\mu}$ for any $\mu \in X' \cup \{\tau\}$. Moreover $Q' = Q''$;

2. Q guarded and $Q \not\xrightarrow{\mu}$ for any $\mu \in X \cup \{\tau\}$ implies $Q \xrightarrow{X}_r Q'$ and $Q' \not\xrightarrow{\mu}$ for any $\mu \in X \cup \{\tau\}$.

Proof: We prove these items together by induction on Q.

Var: $Q = x_u$. This case is not possible since $Q \not\xrightarrow{X}_r$.

Nil: $Q = \text{nil}_u$.

    1. $Q \xrightarrow{X}_r \text{nil}_u = Q' \xrightarrow{X'}_r \text{nil}_u = Q''$, $Q = Q' = \text{nil}_u \not\xrightarrow{\mu}$ for any $\mu \in X' \cup \{\tau\}$ and, trivially, $\text{nil}_u = \text{nil}_u$.

    2. Q is guarded, $Q \not\xrightarrow{\mu}$ for any $\mu \in X \cup \{\tau\}$, $Q \xrightarrow{X}_r \text{nil}_u = Q'$ and $Q' \not\xrightarrow{\mu}$ for any $\mu \in X \cup \{\tau\}$.

Pref: $Q = \alpha_u.P_1$ or $Q = \underline{\alpha}_u.P_1$. Consider only the latter case (the former case is simpler).

    1. $\underline{\alpha}_u.P_1 \xrightarrow{X}_r \underline{\alpha}_u.P_1 = Q' \xrightarrow{X'}_r \underline{\alpha}_u.P_1 = Q''$ implies $\alpha \notin X \cup \{\tau\}$ and $\alpha \notin X' \cup \{\tau\}$. Thus, by operational semantics, both Q and $Q' \not\xrightarrow{\mu}$ for any $\mu \in X' \cup \{\tau\}$. Clearly $Q' = Q''$.

    2. In this case Q is guarded. Moreover $Q \not\xrightarrow{\mu}$ for any $\mu \in X \cup \{\tau\}$ implies $\alpha \notin X \cup \{\tau\}$ and, by operational rules, $Q \xrightarrow{X}_r \underline{\alpha}_u.P_1 = Q'$. Again by operational rules, $\alpha \notin X \cup \{\tau\}$ implies $Q' \not\xrightarrow{\mu}$ for any $\mu \in X \cup \{\tau\}$.

Sum: $Q = Q_1 +_u Q_2$.

    1. If $Q \xrightarrow{X}_r Q' \xrightarrow{X'}_r Q''$ then $Q_1 \xrightarrow{X}_r Q_1' \xrightarrow{X'}_r Q_1''$, $Q_2 \xrightarrow{X}_r Q_2' \xrightarrow{X'}_r Q_2''$, $Q' = Q_1' +_u Q_2'$ and $Q'' = Q_1'' +_u Q_2''$. By induction hypothesis $Q_1 \not\xrightarrow{\mu}$, $Q_1' \not\xrightarrow{\mu}$ and $Q_2 \not\xrightarrow{\mu}$, $Q_2' \not\xrightarrow{\mu}$ for any $\mu \in X \cup \{\tau\}$ and, hence, by operational rules, both $Q \not\xrightarrow{\mu}$ and $Q' \not\xrightarrow{\mu}$ for any $\mu \in X' \cup \{\tau\}$. Again by induction hypothesis, $Q_1' = Q_1''$ and $Q_2' = Q_2''$. Thus $Q' = Q''$.

    2. Q guarded implies both Q_1 and Q_2 guarded. Assume $Q \not\xrightarrow{\mu}$ for any $\mu \in X \cup \{\tau\}$. Then, by operational rules, $Q_1 \not\xrightarrow{\mu}$ and $Q_2 \not\xrightarrow{\mu}$ for any $\mu \in X \cup \{\tau\}$. By induction hypothesis, $Q_1 \xrightarrow{X}_r Q_1'$, $Q_2 \xrightarrow{X}_r Q_2'$ and, hence, $Q \xrightarrow{X}_r Q_1' +_u Q_2' = Q'$. Moreover, again by induction hypothesis, we have that both Q_1' and $Q_2' \not\xrightarrow{\mu}$ for any $\mu \in X \cup \{\tau\}$. Then, by operational rules, also $Q' \not\xrightarrow{\mu}$ for any $\mu \in X \cup \{\tau\}$.

Par: $Q = Q_1 \|_B^u Q_2$.

1. $Q \xrightarrow{X}_r Q'$ implies that there exist X_1, X_2 such that $Q_1 \xrightarrow{X_1}_r Q'_1$, $Q_2 \xrightarrow{X_2}_r Q'_2$ with $X \subseteq (B \cap (X_1 \cup X_2)) \cup ((X_1 \cap X_2) \setminus B)$ and $Q' = \mathsf{clean}(Q'_1 \|_B^u Q'_2)$. Let $A_1 = B \setminus \mathcal{U}(Q'_2)$ and $A_2 = B \setminus \mathcal{U}(Q'_1)$ and assume $Q' = \mathsf{clean}(Q'_1, A_1) \|_B^u \mathsf{clean}(Q'_2, A_2) \xrightarrow{X'}_r Q''$. Again, there exist X'_1, X'_2 such that $\mathsf{clean}(Q'_1, A_1) \xrightarrow{X'_1}_r Q''_1$, $\mathsf{clean}(Q'_2, A_2) \xrightarrow{X'_2}_r Q''_2$ with $X' \subseteq (B \cap (X'_1 \cup X'_2)) \cup ((X'_1 \cap X'_2) \setminus B)$ and $Q'' = \mathsf{clean}(Q''_1 \|_B^u Q''_2)$. If $\mathsf{clean}(Q'_1, A_1) \xrightarrow{X'_1}_r Q''_1$ then $Q'_1 \xrightarrow{X'_1 \setminus A_1}_r Q''_1$ (by Proposition 10.1). Moreover $A_2 \cap \mathcal{U}(Q'_1) = \emptyset$ and Proposition 9.1-1 imply $Q'_1 \xrightarrow{(X'_1 \setminus A_1) \cup A_2}_r Q''_1$. Similarly $Q'_2 \xrightarrow{(X'_2 \setminus A_2) \cup A_1}_r Q''_2$. By induction hypothesis $Q_1 \xrightarrow{X_1}_r Q'_1 \xrightarrow{(X'_1 \setminus A_1) \cup A_2}_r Q''_1$ and $Q_2 \xrightarrow{X_2}_r Q'_2 \xrightarrow{(X'_2 \setminus A_2) \cup A_1}_r Q''_2$ imply (i) $Q_1, Q'_1 \not\xrightarrow{\mu}$ for any $\mu \in (X'_1 \setminus A_1) \cup A_2 \cup \{\tau\}$ and (ii) $Q_2, Q'_2 \not\xrightarrow{\mu}$ for any $\mu \in (X'_2 \setminus A_2) \cup A_1 \cup \{\tau\}$.

   Now we prove that $Q' \not\xrightarrow{\mu}$ (and similarly Q) for any $\mu \in X' \cup \{\tau\}$. First $Q'_1 \not\xrightarrow{\tau}$ and $Q'_2 \not\xrightarrow{\tau}$ imply $Q' \not\xrightarrow{\tau}$. Let $\mu \in X'$ and consider the following subcases:

   - $\mu \in B$. Then $\mu \in X$ implies $\mu \in X'_1 \cup X'_2$ and, hence, either $\mu \in X'_1$ or $\mu \in X'_2$. Assume $\mu \in X'_1$ (the case in which $\mu \in X_2$ can be proved similarly). If $\mu \in A_1$ then, by (ii), $Q'_2 \not\xrightarrow{\mu}$. If $\mu \notin A_1$ and, hence, $\mu \in X'_1 \setminus A_1$, then, by (i), $Q'_1 \not\xrightarrow{\mu}$. In both cases $Q' \not\xrightarrow{\mu}$.
   - $\mu \notin B$. In this case $\mu \in X'_1$ and $\mu \in X'_2$. Moreover $\mu \notin A_1, A_2 \subseteq B$ Thus $\mu \in X'_1 \setminus A_1$ and $\mu \in X'_2 \setminus A_2$ imply, by (i) and (ii), $Q'_1 \not\xrightarrow{\mu}$, $Q'_2 \not\xrightarrow{\mu}$ and, hence, $Q' \not\xrightarrow{\mu}$.

   Again by induction hypothesis we have $Q'_1 = Q''_1$ and $Q'_2 = Q''_2$. Then also $Q' = Q''$.

2. Assume Q guarded and, hence, both Q_1 and Q_2 guarded. Now, assume $Q \not\xrightarrow{\mu}$ for any $\mu \in X \cup \{\tau\}$. By operational semantics we have that: (i) $Q_1 \not\xrightarrow{\mu}$ and $Q_2 \not\xrightarrow{\mu}$, for any $\mu \in (X \setminus B) \cup \{\tau\}$ and (ii) for any $\mu \in X \cap B$ either $Q_1 \not\xrightarrow{\mu}$ or $Q_2 \not\xrightarrow{\mu}$. Let $X'_i = \{\mu \in X \cap B \mid Q_i \not\xrightarrow{\mu}\} \subseteq X \cap B \subseteq B$ and $X_i = (X \setminus B) \cup X'_i$. Then, Q_i guarded and $Q_i \not\xrightarrow{\mu}$ for any $\mu \in X_i \cup \{\tau\}$ implies, by induction hypothesis, $Q_i \xrightarrow{X_i} Q'_i$ for $i = 1, 2$. Moreover, $B \cap (X_1 \cup X_2) = B \cap ((X \setminus B) \cup X'_1 \cup X'_2) = X'_1 \cup X'_2$, $(X_1 \cap X_2) \setminus B = (X_1 \setminus B) \cup (X_2 \setminus B) = (X \setminus B) \cup (X \setminus B) = X \setminus B$ and, by (ii), $X \cap B = X'_1 \cup X'_2$.

   Finally $(B \cap (X_1 \cup X_2)) \cup ((X_1 \cap X_2) \setminus B) = (X \cap B) \cup (X \setminus B) = X$ and, by operational rules, $Q \xrightarrow{X} \mathsf{clean}(Q'_1 \|_B^u Q'_2) = Q'$. Again by induction hypothesis we have that $Q'_1 \not\xrightarrow{\mu}$ for any $\mu \in X_1 \cup \{\tau\}$ and $Q'_2 \not\xrightarrow{\mu}$ for any $\mu \in X_2 \cup \{\tau\}$. Also in this case, $Q'_1 \not\xrightarrow{\tau}$ and $Q'_2 \not\xrightarrow{\tau}$ imply $Q' \not\xrightarrow{\tau}$. Now, let $\mu \in X$. If $\mu \in X \setminus B$ then $\mu \in X_1$ and $\mu \in X_2$ implies both $Q'_1 \not\xrightarrow{\mu}$ and $Q'_2 \not\xrightarrow{\mu}$ and, hence, $Q' \not\xrightarrow{\mu}$. If $\mu \in X \cap B = X'_1 \cup X'_2$ we have either $\mu \in X'_1 \subseteq X_1$ or $\mu \in X'_2 \subseteq X_2$. Thus, either $Q'_1 \not\xrightarrow{\mu}$ or $Q'_2 \not\xrightarrow{\mu}$. Also in this case $Q' \not\xrightarrow{\tau}$.

Rel: $Q = Q_1[\Phi_u]$. Let $X, X' \subseteq \mathbb{A}$, $Y = \Phi^{-1}(X \cup \{\tau\}) \setminus \{\tau\}$ and $Y' = \Phi^{-1}(X' \cup \{\tau\}) \setminus \{\tau\}$. Then $\Phi(\tau) = \tau$ implies $\tau \in \Phi^{-1}(X \cup \{\tau\})$ and $\Phi^{-1}(X \cup \{\tau\}) = (\Phi^{-1}(X \cup \{\tau\}) \setminus \{\tau\}) \cup \{\tau\} = Y \cup \{\tau\}$. Similarly, we have $\Phi^{-1}(X' \cup \{\tau\}) = Y' \cup \{\tau\}$.

1. By operational rules $Q \xrightarrow{X}_r Q' \xrightarrow{X'}_r Q''$ implies $Q_1 \xrightarrow{Y}_r Q'_1 \xrightarrow{Y'}_r Q''_1$, $Q' = Q'_1[\Phi_u]$ and $Q'' = Q''_1[\Phi_u]$. By induction hypothesis $Q_1, Q'_1 \xrightarrow{\mu'}\!\!\!\!/\;$ for any $\mu' \in Y' \cup \{\tau\} = \Phi^{-1}(X' \cup \{\tau\})$ and, hence, $Q, Q' \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in X' \cup \{\tau\}$. Again by induction hypothesis $Q'_1 = Q''_1$ and, hence, also $Q' = Q''$.

2. Q guarded implies Q_1 guarded. Now, assume $Q \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in X \cup \{\tau\}$. Then $Q_1 \xrightarrow{\mu'}\!\!\!\!/\;$ for any $\mu' \in \Phi^{-1}(X \cup \{\tau\}) = Y \cup \{\tau\}$. By induction hypothesis $Q_1 \xrightarrow{Y}_r Q'_1$ and, hence, $Q \xrightarrow{X}_r Q'_1[\Phi] = Q'$. Again by induction hypothesis we have that $Q'_1 \xrightarrow{\mu'}\!\!\!\!/\;$ for any $\mu' \in \Phi^{-1}(X \cup \{\tau\})$ and, by operational semantics, $Q' \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in X \cup \{\tau\}$.

Rec: $Q = \mathsf{rec}\, x_u.Q_1$

1. $Q \xrightarrow{X}_r \mathsf{rec}\, x_u.Q'_1 = Q' \xrightarrow{X'}_r \mathsf{rec}\, x_u.Q''_1 = Q''$ implies $Q_1 \xrightarrow{X}_r Q'_1 \xrightarrow{X'}_r Q''_1$. By induction hypothesis, $Q_1, Q'_1 \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in X' \cup \{\tau\}$. Thus, x guarded in Q_1 and, hence, in Q'_1 imply, by Proposition 7.3-3, $Q_1\{\!|\mathsf{rec}\, x_u.\mathsf{unmark}(Q_1)/x|\!\} \xrightarrow{\mu}\!\!\!\!/\;$ $Q'_1\{\!|\mathsf{rec}\, x_u.\mathsf{unmark}(Q'_1)/x|\!\} \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in X' \cup \{\tau\}$. Finally, by operational semantics, $Q, Q' \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in X \cup \{\tau\}$. Again by induction hypothesis, we also have $Q'_1 = Q''_1$ and, clearly, $Q' = Q''$.

2. Assume Q and, hence, Q_1 guarded. In this case $Q \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in X \cup \{\tau\}$ implies $Q_1\{\!|\mathsf{rec}\, x_u.\mathsf{unmark}(Q_1)/x|\!\} \xrightarrow{\mu}\!\!\!\!/\;$ and, since x is guarded in Q_1, also $Q_1 \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in X \cup \{\tau\}$ (see Proposition 7.3-3). Then, by induction hypothesis, $Q_1 \xrightarrow{X}_r Q'_1$ and, by operational semantics, $Q \xrightarrow{X}_r \mathsf{rec}\, x_u.Q'_1 = Q'$. Moreover, again by induction hypothesis, $Q'_1 \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in X \cup \{\tau\}$. Then, since x guarded in Q_1 implies x in Q'_1, by Proposition 7.3-3 we have that $Q'_1\{\!|\mathsf{rec}\, x_u.\mathsf{unmark}(Q'_1)/x|\!\} \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in X \cup \{\tau\}$. Finally, by operational rules, $Q' \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in X \cup \{\tau\}$.

$\square$

Proposition 3.13 Let $Q, Q', Q'' \in \mathsf{L}(\tilde{\mathbb{P}})$.

1. $Q \xrightarrow{1} Q' \xrightarrow{1} Q''$ implies $Q \xrightarrow{\mu}\!\!\!\!/\;$ and $Q' \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in \mathbb{A}_\tau$. Moreover $Q' = Q''$;

2. Q guarded and $Q \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in \mathbb{A}_\tau$ implies $Q \xrightarrow{1} Q' \xrightarrow{1} Q'$

Proof:

1. If $Q \xrightarrow{1} Q' \xrightarrow{1} Q''$ then, by Lemma 10.2-1 and $Q, Q' \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in \mathbb{A}_\tau$ and $Q' = Q''$.

2. By Lemma 10.2-2, Q guarded and $Q \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in \mathbb{A}_\tau$ implies that $Q \xrightarrow{1} Q'$ and $Q' \xrightarrow{\mu}\!\!\!\!/\;$ for any $\mu \in \mathbb{A}_\tau$. Moreover since Q guarded implies also Q' guarded, again by Lemma 10.2-2, there exists Q'' such that $Q' \xrightarrow{1} Q''$. By Item 1, $Q' = Q''$ and the statement follows.

$\square$

11   Appendix E: A Proof of Proposition 4.4

Proposition 4.4 Let $Q \in \mathsf{L}(\tilde{\mathbb{P}})$ and $P \in \mathsf{L}(\tilde{\mathbb{P}}_1)$ such that $P = \mathsf{unmark}(Q)$. Then:

1. $\mathsf{LE}(P, A) = \mathsf{LE}(Q, A)$, for every A;

2. $Q \xrightarrow{\mu} Q'$ implies $P \xrightarrow{\mu} P'$ and $P' = \mathsf{unmark}(Q')$. Moreover $\mathsf{UE}(Q', A) \subseteq \mathsf{UE}(Q, A)$ and $\mathsf{UE}(Q') = \emptyset$ implies $Q' = P'$;

3. $P \xrightarrow{\mu} P'$ implies $Q \xrightarrow{\mu} Q'$ and $P' = \mathsf{unmark}(Q')$;

Proof: We prove Item 1 by induction on Q and Item 2 by induction of length of derivation $Q \xrightarrow{\mu} Q'$. The proof for Item 3 is similar to the one for 2 and hence omitted. We proceed by case analysis on the structure of Q.

Nil, Var: $Q = \mathsf{nil}_u$, $Q = x_u$. In both case $P = \mathsf{unmark}(Q)$ implies $P = Q$ and items 1. and 2. hold trivially.

Pref: $Q = \mu_u.P_1$ or $Q = \underline{\mu}_u.P_1$. In the first case (the other is similar) $P = \mathsf{unmark}(Q)$ implies $P = \mu_u.P_1$. Then:

1. $\mathsf{LE}(P, A) = \mathsf{LE}(Q, A) = \{\langle u \rangle\}$ if $\mu \notin A$, $\mathsf{LE}(P, A) = \mathsf{LE}(Q, A) = \emptyset$ otherwise.

2. $Q \xrightarrow{\mu} P_1$, $P \xrightarrow{\mu} P_1$ and, since $P_1 \in \mathsf{L}(\tilde{\mathbb{P}}_1)$, $P_1 = \mathsf{unmark}(P_1)$. Moreover we have that $\mathsf{UE}(P_1, A) = \emptyset \subseteq \mathsf{UE}(Q, A)$, $\mathsf{UE}(P_1) = \emptyset$ and, sure, $P_1 = P_1$.

Sum: $Q = Q_1 +_u Q_2$. In this case $P = \mathsf{unmark}(Q)$ implies $P_i = \mathsf{unmark}(Q_i)$, for $i = 1, 2$, and $P = P_1 +_u P_2$.

1. By induction hypothesis, for every A,
   $\mathsf{LE}(P, A) = \mathsf{LE}(P_1, A) \cup \mathsf{LE}(P_2, A) = \mathsf{LE}(Q_1, A) \cup \mathsf{LE}(Q_2, A) = \mathsf{LE}(Q, A)$.

2. By the operational rules, $Q \xrightarrow{\mu} Q'$ if either (i) $Q_1 \xrightarrow{\mu} Q'$ or (ii) $Q_2 \xrightarrow{\mu} P'$. Consider case (i). By induction hypothesis, $Q_1 \xrightarrow{\mu} Q'$ implies $P_1 \xrightarrow{\mu} P'$, and hence $P \xrightarrow{\mu} P'$, and $P' = \mathsf{unmark}(Q')$. Moreover, again by induction hypothesis, we have that $\mathsf{UE}(Q', A) \subseteq \mathsf{UE}(Q_1, A) \subseteq \mathsf{UE}(Q, A)$ and $\mathsf{UE}(Q') = \emptyset$ implies $Q' = P'$.

Par: $Q = Q_1 \|_B^u Q_2$. By definition $\mathsf{unmark}(Q) = \mathsf{unmark}(Q_1) \|_B^u \mathsf{unmark}(Q_2)$. Thus $P = P_1 \|_B^u P_2$, where $P_i = \mathsf{unmark}(Q_i)$ for every $i = 1, 2$. Then:

1. By induction hypothesis, $\mathsf{LE}(P, A) =$
   $\mathsf{LE}(P_1, A \cup B) \cup \mathsf{LE}(P_2, A \cup B) \cup \bigcup_{\alpha \in B \setminus A}(\mathsf{LE}(P_1, \mathbb{A}_\tau \setminus \{\alpha\}) \times \mathsf{LE}(P_2, \mathbb{A}_\tau \setminus \{\alpha\})) =$
   $\mathsf{LE}(Q_1, A \cup B) \cup \mathsf{LE}(Q_2, A \cup B) \cup \bigcup_{\alpha \in B \setminus A}(\mathsf{LE}(Q_1, \mathbb{A}_\tau \setminus \{\alpha\}) \times \mathsf{LE}(Q_2, \mathbb{A}_\tau \setminus \{\alpha\})) =$
   $\mathsf{LE}(Q, A)$.

2. Assume $Q \xrightarrow{\mu} Q'$ and consider the following three possible cases:

   i. $\mu \in B$ and $Q_i \xrightarrow{\mu} Q_i'$ for $i = 1, 2$ and $Q' = \mathsf{clean}(Q_1' \|_B^u Q_2')$. By induction hypothesis, $Q_i \xrightarrow{\mu} Q_i'$ implies $P_i \xrightarrow{\mu} P_i'$, $P_i' = \mathsf{unmark}(Q_i')$ and $\mathsf{UE}(Q_i', A) \subseteq \mathsf{UE}(Q_i, A)$, for every i and A. Then: $Q \xrightarrow{\mu} Q'$ implies $P \xrightarrow{\mu} \mathsf{clean}(P_1' \|_B^u P_2') = P_1' \|_B^u P_2' = P'$ and $P' = \mathsf{unmark}(Q')$. Moreover, by Lemma 7.10, $\mathsf{UE}(Q', A) = \mathsf{UE}(Q_1' \|_B^u Q_2', A) =$

$\mathsf{UE}(Q_1', A\cup B)\cup\mathsf{UE}(Q_2', A\cup B)\cup\bigcup_{\alpha\in B\setminus A}(\mathsf{UE}(Q_1', \mathbb{A}_\tau\setminus\{\alpha\})\times\mathsf{UE}(Q_2', \mathbb{A}_\tau\setminus\{\alpha\}))\subseteq$
$\mathsf{UE}(Q_1, A\cup B)\cup\mathsf{UE}(Q_2, A\cup B)\cup\bigcup_{\alpha\in B\setminus A}(\mathsf{UE}(Q_1, \mathbb{A}_\tau\setminus\{\alpha\})\times\mathsf{UE}(Q_2, \mathbb{A}_\tau\setminus\{\alpha\}))=$
$\mathsf{UE}(Q, A)$.

Now assume $\mathsf{UE}(Q')=\emptyset$. By Lemma 7.10 and Proposition 3.12-3, we also have $\mathsf{UE}(Q_1'\parallel_B^u Q_2')=\emptyset$ and $\mathcal{U}(Q_1'\parallel_B^u Q_2')=\emptyset$. Finally, $\mathcal{U}(Q_1'\parallel_B^u Q_2')=\emptyset$ and Proposition 7.5-3 imply $Q'=\mathsf{clean}(Q_1'\parallel_B^u Q_2')=\mathsf{unmark}(Q_1'\parallel_B^u Q_2')=P_1'\parallel_B^u P_2'=P'$.

  ii. $\mu\notin B$ and $Q_1\xrightarrow{\mu}Q_1'$ and $Q'=\mathsf{clean}(Q_1'\parallel_B^u Q_2)$. By induction hypothesis, $Q_1\xrightarrow{\mu}Q_1'$ implies $P_1\xrightarrow{\mu}P_1'$, $P_1'=\mathsf{unmark}(Q_1')$ and $\mathsf{UE}(Q_1', A)\subseteq\mathsf{UE}(Q_1, A)$, for every A. Then: $Q\xrightarrow{\alpha}Q'$ implies $P\xrightarrow{\alpha}\mathsf{clean}(P_1'\parallel_B^u P_2)=P_1'\parallel_B^u P_2=P'$ and $P'=\mathsf{unmark}(Q')$. Moreover, $\mathsf{UE}(Q', A)=$
$\mathsf{UE}(Q_1', A\cup B)\cup\mathsf{UE}(Q_2, A\cup B)\cup\bigcup_{\alpha\in B\setminus A}(\mathsf{UE}(Q_1', \mathbb{A}_\tau\setminus\{\alpha\})\times\mathsf{UE}(Q_2, \mathbb{A}_\tau\setminus\{\alpha\}))\subseteq$
$\mathsf{UE}(Q_1, A\cup B)\cup\mathsf{UE}(Q_2, A\cup B)\cup\bigcup_{\alpha\in B\setminus A}(\mathsf{UE}(Q_1, \mathbb{A}_\tau\setminus\{\alpha\})\times\mathsf{UE}(Q_2, \mathbb{A}_\tau\setminus\{\alpha\}))=$
$\mathsf{UE}(Q, A)$. Similar to the previous case we can prove that $\mathsf{UE}(Q')=\emptyset$ implies $Q'=P'$.

  iii. $\mu\notin B$ and $Q_2\xrightarrow{\mu}Q_2'$ and $Q'=\mathsf{clean}(Q_1\parallel_B^u Q_2')$. This case is similar to the previous one.

Rel: $Q=Q_1[\Phi_u]$. By definition $\mathsf{unmark}(Q)=\mathsf{unmark}(Q_1)[\Phi_u]$. Thus $P=P_1[\Phi_u]$ with $P_1=\mathsf{unmark}(Q_1)$. Then:

  1. $\mathsf{LE}(P, A)=\mathsf{LE}(P_1, \Phi^{-1}(A))=\mathsf{LE}(Q_1, \Phi^{-1}(A))=\mathsf{LE}(Q, A)$, by induction hypothesis.

  2. By the operational rules, $Q\xrightarrow{\mu}Q_1'[\Phi_u]=Q'$ if and only if there exists $\mu'\in\Phi^{-1}(\mu)$ such that $Q_1\xrightarrow{\mu'}Q_1'$. By induction hypothesis, $Q_1\xrightarrow{\mu'}Q_1'$ implies $P_1\xrightarrow{\mu'}P_1'$ and $P_1'=\mathsf{unmark}(Q_1')$ which implies $P\xrightarrow{\mu}P_1'[\Phi_u]=P'$ and $P'=\mathsf{unmark}(Q_1')[\Phi_u]=\mathsf{unmark}(Q')$. Moreover, we conclude from the induction hypothesis that $\mathsf{UE}(Q', A)=\mathsf{UE}(Q_1', \Phi^{-1}(A))\subseteq\mathsf{UE}(Q_1, \Phi^{-1}(A))=\mathsf{UE}(Q, A)$. Finally, again by induction hypothesis $\mathsf{UE}(Q')=\mathsf{UE}(Q_1')=\emptyset$ implies $Q_1'=P_1'$ and, hence, $Q'=P'$.

Rec: $Q=\mathsf{rec}\ x_u.Q_1$. In this case $P=\mathsf{unmark}(Q)$ implies $P=\mathsf{rec}\ x_u.P_1$ with $P_1=\mathsf{unmark}(Q_1)$.

  1. By induction hypothesis, $\mathsf{LE}(P, A)=\mathsf{LE}(P_1, A)=\mathsf{LE}(Q_1, A)=\mathsf{LE}(Q, A)$.

  2. Let $R=Q_1\{\!|\mathsf{rec}\ x_u.\mathsf{unmark}(Q_1)/x|\!\}=Q_1\{\!|\mathsf{rec}\ x_u.P_1/x|\!\}$ and $S=\mathsf{unmark}(R)$. Then x guarded in Q_1 and Propositions 7.3-2 and 7.7-2 imply (i) $\mathsf{UE}(R, A)=\mathsf{UE}(Q_1, A)=\mathsf{UE}(Q, A)$ and (ii) $S=\mathsf{unmark}(Q_1)\{\!|\mathsf{rec}x_u.P_1/x|\!\}=P_1\{\!|\mathsf{rec}x_u.P_1/x|\!\}$. Now assume $Q\xrightarrow{\mu}Q'$ and hence, by operational rules, $R\xrightarrow{\mu}Q'$. By induction hypothesis we have that $S\xrightarrow{\mu}P'$, $P'=\mathsf{unmark}(Q')$ and $\mathsf{UE}(Q', A)\subseteq\mathcal{U}(R, A)=\mathsf{UE}(Q, A)$. Moreover, again by induction hypothesis $\mathsf{UE}(Q')=\emptyset$ implies $Q'=P'$.

$\hfill\square$

12 Appendix F: Proof of Proposition 5.3

First we report the formal definition of $\mathcal{U}\mathcal{U}(Q, A)$.

Definition 12.1 (*urgency and unfolding*)

Let $Q \in \tilde{\mathbb{P}}$ and $A \subseteq \mathbb{A}$. Define $\mathcal{U}\mathcal{U}(Q, A)$ by induction on Q as follows (where we extend the PAFAS-operators to sets of process terms in the natural way):

Nil: $\quad \mathcal{U}\mathcal{U}(\mathsf{nil}, A) = \{\mathsf{nil}\}$

Var: $\quad \mathcal{U}\mathcal{U}(x, A) = \{x\}$

Pref: $\quad \mathcal{U}\mathcal{U}(\alpha.P, A) = \begin{cases} \{\alpha.P, \underline{\alpha}.P\} & \text{if } \alpha \in A \\ \{\alpha.P\} & \text{otherwise} \end{cases}$

$\qquad \mathcal{U}\mathcal{U}(\underline{\alpha}.P, A) = \{\underline{\alpha}.P\}$

Sum: $\quad \mathcal{U}\mathcal{U}(Q_1 + Q_2, A) = \mathcal{U}\mathcal{U}(Q_1, A) + \mathcal{U}\mathcal{U}(Q_2, A)$

Par: $\quad \mathcal{U}\mathcal{U}(Q_1 \parallel_B Q_2, A) = \left\{ R \in \mathcal{U}\mathcal{U}(Q_1, A \cup A_1) \parallel_B \mathcal{U}\mathcal{U}(Q_2, A \cup A_2) \,\middle|\, \mathcal{U}(R) \subseteq \mathcal{U}(Q) \cup A \right\}$
$\qquad$ where $A_1 = B \backslash \mathcal{U}(Q_2)$ and $A_2 = B \backslash \mathcal{U}(Q_1)$

Rel: $\quad \mathcal{U}\mathcal{U}(Q[\Phi], A) = (\mathcal{U}\mathcal{U}(Q, \Phi^{-1}(A)))[\Phi]$

Rec: $\quad \mathcal{U}\mathcal{U}(\mathsf{rec}\, x.Q, A) = \begin{cases} \{\mathsf{rec}\, x.Q\} \cup \mathcal{U}\mathcal{U}(Q, A)\{\mathsf{rec}\, x.Q/x\} & \text{if } Q \in \tilde{\mathbb{P}}_1 \\ \mathcal{U}\mathcal{U}(Q, A)\{\mathsf{rec}\, x.\mathsf{unmark}(Q)/x\} & \text{otherwise} \end{cases}$

As usual, we denote with $\mathcal{U}\mathcal{U}(Q)$ the set $\mathcal{U}\mathcal{U}(Q, \emptyset)$

The following lemma can be easily proved by induction on P.

Lemma 12.2 If $P \in \tilde{\mathbb{P}}_1$ and $A \subseteq \mathbb{A}$, then $P \in \mathcal{U}\mathcal{U}(P, A)$.

The following proposition states some useful properties of the set $\mathcal{U}\mathcal{U}(Q, A)$.

Proposition 12.3 Let $Q, R \in \tilde{\mathbb{P}}$ and $A, A' \subseteq \mathbb{A}$ such that $R \in \mathcal{U}\mathcal{U}(Q, A)$. Then:

1. $\mathcal{U}(Q) \subseteq \mathcal{U}(R)$ and $\mathcal{U}(R) \subseteq \mathcal{U}(Q) \cup A$.

2. $A \subseteq A'$ implies $R \in \mathcal{U}\mathcal{U}(Q, A')$;

3. $A' \subseteq A$ implies $R \in \mathcal{U}\mathcal{U}(\mathsf{clean}(Q, A'), A)$.

Proof: We prove all items by induction on $Q \in \tilde{\mathbb{P}}$.

Nil, Var: $Q = \mathsf{nil}$, $Q = x$. In these cases $R \in \mathcal{U}\mathcal{U}(Q, A) = \{Q\}$ implies $R = Q$. Moreover:

1. $\mathcal{U}(Q) = \mathcal{U}(R) = \emptyset$ and $\mathcal{U}(R) = \emptyset \subseteq \mathcal{U}(Q) \cup A$.

2. $R \in \mathcal{UU}(Q, A') = \{Q\}$ for all A'.

3. Trivial since $\mathsf{clean}(Q, A') = Q$ for all A'.

Pref: $Q = \alpha.P_1$ or $Q = \underline{\alpha}.P_1$. Assume $Q = \alpha.P_1$, $R \in \mathcal{UU}(Q, A)$ and consider the following cases:

- $\alpha \in A$. Then $\mathcal{UU}(Q, A) = \{\alpha.P_1, \underline{\alpha}.P_1\}$ implies either $R = \alpha.P_1$ or $R = \underline{\alpha}.P_1$. In both cases:

  1. $\mathcal{U}(Q) = \emptyset \subseteq \mathcal{U}(R)$ and $\mathcal{U}(R) \subseteq \{\alpha\} \subseteq \mathcal{U}(Q) \cup A$.
  2. $\alpha \in A \subseteq A'$ implies $\mathcal{UU}(Q, A') = \{\alpha.P_1, \underline{\alpha}.P_1\}$ and $R \in \mathcal{UU}(Q, A')$.
  3. Trivial since $\mathsf{clean}(Q, A') = Q$ for any A'.

- $\alpha \notin A$ in this case $R \in \mathcal{UU}(Q, A) = \{\alpha.P_1\}$ implies $R = \alpha.P_1$.

  1. $\mathcal{U}(Q) = \mathcal{U}(R) = \emptyset$ and $\mathcal{U}(R) \subseteq \mathcal{U}(Q) \cup A$.
  2. $\alpha \notin A'$ implies $\mathcal{UU}(Q, A') = \{\alpha.P_1\}$, while if $\alpha \in A'$ then $\mathcal{UU}(Q, A') = \{\alpha.P_1, \underline{\alpha}.P_1\}$. In both cases $R \in \mathcal{UU}(Q, A')$.
  3. Again, for each A', $\mathsf{clean}(Q, A') = Q$.

Now we can assume $Q = \underline{\alpha}.P_1$. In this case $\mathcal{UU}(Q, A) = \{\underline{\alpha}.P_1\}$ and $R \in \mathcal{UU}(Q, A)$ imply $R = \underline{\alpha}.P_1$.

1. $\mathcal{U}(Q) = \mathcal{U}(R) = \{\alpha\}$ and $\mathcal{U}(R) = \mathcal{U}(Q) \subseteq \mathcal{U}(Q) \cup A$.
2. In this case, by Definition 12.1, $\mathcal{UU}(Q, A') = \{\underline{\alpha}.P_1\}$ and, hence, $R \in \mathcal{UU}(Q, A')$.
3. If $\alpha \in A' \subseteq A$ then $\mathcal{UU}(\mathsf{clean}(Q, A'), A) = \mathcal{UU}(\alpha.P_1, A) = \{\alpha.P_1, \underline{\alpha}.P_1\}$. Otherwise, if $\alpha \notin A'$, $\mathcal{UU}(\mathsf{clean}(Q, A'), A) = \mathcal{UU}(\underline{\alpha}.P_1, A) = \{\underline{\alpha}.P_1\}$. In both cases $R \in \mathcal{UU}(\mathsf{clean}(Q, A'), A)$.

Sum: $Q = Q_1 + Q_2$. Assume $R \in \mathcal{UU}(Q, A)$. Then, by Definition 12.1, $R = R_1 + R_2$ with $R_i \in \mathcal{UU}(Q_i, A)$ for $i = 1, 2$. By induction hypothesis:

1. $\mathcal{U}(Q) = \mathcal{U}(Q_1) \cup \mathcal{U}(Q_2) \subseteq \mathcal{U}(R_1) \cup \mathcal{U}(R_2) = \mathcal{U}(R)$ and $\mathcal{U}(R) = \mathcal{U}(R_1) \cup \mathcal{U}(R_2) \subseteq (\mathcal{U}(Q_1) \cup A) \cup (\mathcal{U}(Q_2) \cup A) = (\mathcal{U}(Q_1) \cup \mathcal{U}(Q_2)) \cup A = \mathcal{U}(Q) \cup A$.
2. $A \subseteq A'$ implies $R = R_1 + R_2 \in \mathcal{UU}(Q_1, A') + \mathcal{UU}(Q_2, A') = \mathcal{UU}(Q, A')$.
3. $A' \subseteq A$ implies $R = R_1 + R_2 \in \mathcal{UU}(\mathsf{clean}(Q_1, A'), A) + \mathcal{UU}(\mathsf{clean}(Q_2, A'), A) = \mathcal{UU}(\mathsf{clean}(Q_1, A') + \mathsf{clean}(Q_2, A'), A) = \mathcal{UU}(\mathsf{clean}(Q, A'), A)$.

Par: $Q = Q_1 \parallel_B Q_2$. Let $A_1 = B \backslash \mathcal{U}(Q_2)$ and $A_2 = B \backslash \mathcal{U}(Q_1)$. In this case $R \in \mathcal{UU}(Q, A)$ implies $R = R_1 \parallel_B R_2$ with $R_i \in \mathcal{UU}(Q_i, A \cup A_i)$, for $i = 1, 2$, and $\mathcal{U}(R) \subseteq \mathcal{U}(Q) \cup A$. Then:

1. By induction hypothesis $\mathcal{U}(Q_i) \subseteq \mathcal{U}(R_i)$. Thus, for a generic $C \subseteq \mathbb{A}_\tau$, we also have that $\mathcal{U}(Q_i, C) = \mathcal{U}(Q_i) \backslash C \subseteq \mathcal{U}(R_i) \backslash C = \mathcal{U}(R_i, C)$. Then, by Definition 2.3, $\mathcal{U}(Q) \subseteq \mathcal{U}(R)$. Trivially $\mathcal{U}(R) \subseteq \mathcal{U}(Q) \cup A$.
2. $A \subseteq A'$ implies $A \cup A_i \subseteq A' \cup A_i$ and hence, by induction hypothesis, $R_i \in \mathcal{UU}(Q_i, A' \cup A_i)$ (for $i = 1, 2$). Moreover $\mathcal{U}(R) \subseteq \mathcal{U}(Q) \cup A \subseteq \mathcal{U}(Q) \cup A'$. Thus we can conclude that $R \in \mathcal{UU}(Q, A')$.

3. Let $A_1' = (\mathcal{U}(Q_1)\backslash\mathcal{U}(Q_2)) \cap B$ and $A_2' = (\mathcal{U}(Q_2)\backslash\mathcal{U}(Q_1)) \cap B$. By Definition 2.4, $\mathsf{clean}(Q, A') = \mathsf{clean}(Q_1, A' \cup A_1') \parallel_B \mathsf{clean}(Q_2, A' \cup A_2') = Q_1' \parallel_B Q_2' = Q'$. Now let $A_1'' = B\backslash\mathcal{U}(Q_2')$ and $A_2'' = B\backslash\mathcal{U}(Q_1')$. Then Definition 12.1 implies that $R \in \mathcal{U}\mathcal{U}(Q', A)$ iff (i) $R_i \in \mathcal{U}\mathcal{U}(Q_i', A \cup A_i'')$, for $i = 1, 2$ and (ii) $\mathcal{U}(R) \subseteq \mathcal{U}(Q') \cup A\}$. Assume $A' \subseteq A$.

   (i) $A_1' = (\mathcal{U}(Q_1)\backslash\mathcal{U}(Q_2)) \cap B = \mathcal{U}(Q_1) \cap (B\backslash\mathcal{U}(Q_2)) = \mathcal{U}(Q_1) \cap A_1 \subseteq A_1$ and $A' \subseteq A$ imply $A' \cup A_1' \subseteq A \cup A_1$. By induction hypothesis, $R_1 \in \mathcal{U}\mathcal{U}(Q_1, A \cup A_1)$ and $A' \cup A_1' \subseteq A \cup A_1$ imply $R_1 \in \mathcal{U}\mathcal{U}(\mathsf{clean}(Q_1, A' \cup A_1'), A \cup A_1) = \mathcal{U}\mathcal{U}(Q_1', A \cup A_1)$. Moreover $\mathcal{U}(Q_2') = \mathcal{U}(\mathsf{clean}(Q_2, A' \cup A_2')) = \mathcal{U}(Q_2, A' \cup A_2') \subseteq \mathcal{U}(Q_2)$ (see Propositions 7.6-2 and 7.1-3) implies $A_1 = B\backslash\mathcal{U}(Q_2) \subseteq B\backslash\mathcal{U}(Q_2') = A_1''$ and, hence, $A \cup A_1 \subseteq A \cup A_1''$. Then $R_1 \in \mathcal{U}\mathcal{U}(Q_1', A \cup A_1)$ and $A \cup A_1 \subseteq A \cup A_1''$ imply, by Item 2, $R_1 \in \mathcal{U}\mathcal{U}(Q_1', A \cup A_1'')$. Similarly, we have that $R_2 \in \mathcal{U}\mathcal{U}(Q_2', A \cup A_2'')$.

   (ii) Propositions 7.6-2 and 7.1-4 imply that $\mathcal{U}(Q') = \mathcal{U}(\mathsf{clean}(Q, A')) = \mathcal{U}(Q, A') = \mathcal{U}(Q)\backslash A'$. Thus, since $A' \subseteq A$, we have that $\mathcal{U}(R) \subseteq \mathcal{U}(Q) \cup A = (\mathcal{U}(Q)\backslash A') \cup (\mathcal{U}(Q) \cap A') \cup A \subseteq \mathcal{U}(Q') \cup (A' \cup A) = \mathcal{U}(Q') \cup A$.

Rel: $Q = Q_1[\Phi]$. $R \in \mathcal{U}\mathcal{U}(Q, A) = \mathcal{U}\mathcal{U}(Q_1, \Phi^{-1}(A))[\Phi]$ implies $R = R_1[\Phi]$ with $R_1 \in \mathcal{U}\mathcal{U}(Q_1, \Phi^{-1}(A))$

   1. By induction hypothesis $\mathcal{U}(Q) = \Phi(\mathcal{U}(Q_1)) \subseteq \Phi(\mathcal{U}(R_1)) = \mathcal{U}(R)$. Again by induction hypothesis $\mathcal{U}(R) = \Phi(\mathcal{U}(R_1)) \subseteq \Phi(\mathcal{U}(Q_1) \cup \Phi^{-1}(A)) = \Phi(\mathcal{U}(Q_1)) \cup \Phi(\Phi^{-1}(A)) \subseteq \mathcal{U}(Q) \cup A$.

   2. $A \subseteq A'$ implies $\Phi^{-1}(A) \subseteq \Phi^{-1}(A')$. By induction hypothesis we have that $R_1 \in \mathcal{U}\mathcal{U}(Q_1, \Phi^{-1}(A'))$ and, hence, $R \in \mathcal{U}\mathcal{U}(Q, A')$

   3. $A' \subseteq A$ implies $\Phi^{-1}(A') \subseteq \Phi^{-1}(A)$. Then, by induction hypothesis, we have that $R_1 \in \mathcal{U}\mathcal{U}(\mathsf{clean}(Q_1, \Phi^{-1}(A')), \Phi^{-1}(A))$. Moreover $\mathcal{U}\mathcal{U}(\mathsf{clean}(Q, A'), A) = \mathcal{U}\mathcal{U}(\mathsf{clean}(Q_1, \Phi^{-1}(A'))[\Phi], A) = \mathcal{U}\mathcal{U}(\mathsf{clean}(Q_1, \Phi^{-1}(A')), \Phi^{-1}(A))[\Phi]$. We can conclude that $R \in \mathcal{U}\mathcal{U}(\mathsf{clean}(Q, A'), A)$.

Rec: $Q = \mathsf{rec}\, x.Q_1$. We have to consider the following subcases:

   - $Q_1 \notin \tilde{\mathbb{P}}_1$. In this case $R \in \mathcal{U}\mathcal{U}(Q, A)$ implies $R = R_1\{\mathsf{rec}\, x.S_1/x\}$ where $R_1 \in \mathcal{U}\mathcal{U}(Q_1, A)$ and $S_1 = \mathsf{unmark}(Q_1)$.

     1. By induction hypothesis $\mathcal{U}(Q) = \mathcal{U}(Q_1) \subseteq \mathcal{U}(R_1)$. Moreover x guarded in Q_1 implies x guarded in $R_1 \in \mathcal{U}\mathcal{U}(Q_1, A)$ and $\mathcal{U}(R) = \mathcal{U}(R_1\{\mathsf{rec}\, x.S_1/x\}) = \mathcal{U}(R_1)$ (see Proposition 7.2). We can conclude that $\mathcal{U}(Q) \subseteq \mathcal{U}(R)$. Again by induction hypothesis $\mathcal{U}(R) = \mathcal{U}(R_1) \subseteq \mathcal{U}(Q_1) \cup A = \mathcal{U}(Q) \cup A$.

     2. By induction hypothesis $A \subseteq A'$ implies $R_1 \in \mathcal{U}\mathcal{U}(Q_1, A')$ and, hence, $R \in \mathcal{U}\mathcal{U}(Q, A') = \mathcal{U}\mathcal{U}(Q_1, A')\{\mathsf{rec}\, x.S_1/x\}$.

     3. Let $Q' = \mathsf{clean}(Q, A') = \mathsf{rec}\, x.\mathsf{clean}(Q_1, A') = \mathsf{rec}\, x.Q_1'$. By induction hypothesis $A' \subseteq A$ implies $R_1 \in \mathcal{U}\mathcal{U}(\mathsf{clean}(Q_1, A'), A) = \mathcal{U}\mathcal{U}(Q_1', A)$. Now consider the following subcases:

        - $Q_1' \in \tilde{\mathbb{P}}_1$. By Definition 12.1 $\mathcal{U}\mathcal{U}(Q', A) \supseteq \mathcal{U}\mathcal{U}(Q_1', A)\{\mathsf{rec}\, x.Q_1'/x\}$ with $Q_1' = \mathsf{unmark}(Q_1') = \mathsf{unmark}(\mathsf{clean}(Q_1, A)) = \mathsf{unmark}(Q_1) = S_1$ (see Proposition 7.5-1).

- $Q_1' \notin \tilde{\mathbb{P}}_1$. Similarly $\mathcal{UU}(Q', A) \supseteq \mathcal{UU}(Q_1', A)\{\text{rec } x.\text{unmark}(Q_1')/x\}$ with $\text{unmark}(Q_1') = \text{unmark}(\text{clean}(Q_1, A)) = \text{unmark}(Q_1) = S_1$. (again by Proposition 7.5-1)

  In both cases we have that $R \in \mathcal{UU}(Q', A)$.

- $Q_1 \in \tilde{\mathbb{P}}_1$. In this case $R \in \mathcal{UU}(Q, A) = \{Q\} \cup \mathcal{UU}(Q_1, A)\{\text{rec } x.Q_1/x\}$ implies either $R = Q$ or $R = R_1\{\text{rec } x.Q_1/x\}$, where $R_1 \in \mathcal{UU}(Q_1, A)$. We prove only the latter case (the former can be proved similarly to the case in which $Q_1 \notin \tilde{\mathbb{P}}_1$).

  1. Similar to the previous case.
  2. By induction hypothesis $A \subseteq A'$ implies $R_1 \in \mathcal{UU}(Q_1, A')$ and, hence, $R \in \mathcal{UU}(Q_1, A')\{\text{rec } x.Q_1/x\} \subseteq \mathcal{UU}(Q, A')$.
  3. $Q_1 \in \tilde{\mathbb{P}}_1$ implies $\text{clean}(Q, A') = Q$, for all A', and, hence, the item follows follows easily.

$\square$

Proposition 12.4 Let $Q \in \tilde{\mathbb{P}}$ and $A \subseteq \mathbb{A}_\tau$. Then $Q \xrightarrow{\alpha} Q'$ implies $\mathcal{U}(Q', A) \subseteq \mathcal{U}(Q, A)$.

Proof: We prove the statement by induction on length of derivation $Q \xrightarrow{\alpha} Q'$. We proceed by case analysis on the structure of Q.

Nil, Var: $Q = \text{nil}$, $Q = x$. These case are not possible since $Q \not\xrightarrow{}$.

Pref: $Q = \alpha.P_1$ or $Q = \underline{\alpha}.P_1$. In both cases $Q \xrightarrow{\alpha} P_1 \in \tilde{\mathbb{P}}_1$ and $\mathcal{U}(P_1, A) = \emptyset \subseteq \mathcal{U}(Q, A)$.

Sum: $Q = Q_1 + Q_2$. By operational rules we have either (i) $Q_1 \xrightarrow{\alpha} Q'$ or (ii) $Q_2 \xrightarrow{\alpha} Q'$. Consider the (i) case (the (ii)-case is symmetric). Then by induction hypothesis, $\mathcal{U}(Q', A) \subseteq \mathcal{U}(Q_1, A) \subseteq \mathcal{U}(Q, A)$.

Par: $Q = Q_1 \parallel_B Q_2$. Assume $Q \xrightarrow{\alpha} Q'$ and consider the following cases:

- $\alpha \notin B$, $Q_1 \xrightarrow{\alpha} Q_1'$ and $Q' = \text{clean}(Q_1' \parallel_B Q_2)$. By induction hypothesis we have that $\mathcal{U}(Q_1', A) \subseteq \mathcal{U}(Q_1, A)$. Thus, by Proposition 7.6-2, $\mathcal{U}(Q', A) = \mathcal{U}(Q_1' \parallel_B Q_2, A) = \mathcal{U}(Q_1', A \cup B) \cup \mathcal{U}(Q_2, A \cup B) \cup (\mathcal{U}(Q_1', \mathbb{A}_\tau \backslash (B \backslash A)) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau \backslash (B \backslash A))) \subseteq \mathcal{U}(Q_1, A \cup B) \cup \mathcal{U}(Q_2, A \cup B) \cup (\mathcal{U}(Q_1, \mathbb{A}_\tau \backslash (B \backslash A)) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau \backslash (B \backslash A))) = \mathcal{U}(Q, A)$.

- $\alpha \notin B$, $Q_2 \xrightarrow{\alpha} Q_2'$ and $Q' = \text{clean}(Q_1 \parallel_B Q_2')$. Similar to the previous case.

- $\alpha \in B$, $Q_i \xrightarrow{\alpha} Q_i'$, for $i = 1, 2$ and $Q' = \text{clean}(Q_1' \parallel_B Q_2')$. By induction hypothesis, $\mathcal{U}(Q_i', A) \subseteq \mathcal{U}(Q_i, A)$. Again by Proposition 7.6-2, $\mathcal{U}(Q', A) = \mathcal{U}(Q_1' \parallel_B Q_2', A) = \mathcal{U}(Q_1', A \cup B) \cup \mathcal{U}(Q_2', A \cup B) \cup (\mathcal{U}(Q_1', \mathbb{A}_\tau \backslash (B \backslash A)) \cap \mathcal{U}(Q_2', \mathbb{A}_\tau \backslash (B \backslash A))) \subseteq \mathcal{U}(Q_1, A \cup B) \cup \mathcal{U}(Q_2, A \cup B) \cup (\mathcal{U}(Q_1, \mathbb{A}_\tau \backslash (B \backslash A)) \cap \mathcal{U}(Q_2, \mathbb{A}_\tau \backslash (B \backslash A))) = \mathcal{U}(Q, A)$.

Rel: $Q = Q_1[\Phi]$. Assume $Q \xrightarrow{\alpha} Q'$. By operational rules there exists $\beta \in \Phi^{-1}(\alpha)$ such that $Q_1 \xrightarrow{\beta} Q_1'$ and $Q' = Q_1'[\Phi]$. By induction hypothesis we have that $\mathcal{U}(Q', A) = \Phi(\mathcal{U}(Q_1', \Phi^{-1}(A))) \subseteq \Phi(\mathcal{U}(Q_1, \Phi^{-1}(A))) = \mathcal{U}(Q, A)$.

Rec: $Q = \text{rec } x.Q_1$. Let $S_1 = \text{unmark}(Q_1)$ and $S = Q_1\{\text{rec } x.S_1/x\}$; x guarded in Q_1 and Proposition 7.2 imply $\mathcal{U}(S, A) = \mathcal{U}(Q_1\{\text{rec } x.S_1/x\}, A) = \mathcal{U}(Q_1, A) = \mathcal{U}(Q, A)$. Now assume $Q \xrightarrow{\alpha} Q'$. Then, by operational rules, $S \xrightarrow{\alpha} Q'$ and, by induction hypothesis, $\mathcal{U}(Q', A) \subseteq \mathcal{U}(S, A) = \mathcal{U}(Q, A)$.

$\square$

Proposition 12.5 Let $Q \in \tilde{\mathbb{P}}$, $P \in \tilde{\mathbb{P}}_1$ and $x \in \mathcal{X}$ guarded in Q. Then $\mathcal{UU}(Q\{P/x\}, A) = \mathcal{UU}(Q, A)\{P/x\}$.

Proof: We proceed by induction on $Q \in \tilde{\mathbb{P}}$.

Nil: $Q = \text{nil}$. In this case trivially x is guarded in Q and $Q\{P/x\} = \text{nil}$ $\mathcal{UU}(Q\{P/x\}, A) = \mathcal{UU}(\text{nil}, A) = \{\text{nil}\} = \{\text{nil}\}\{P/x\} = \mathcal{UU}(Q, A)\{P/x\}$.

Var: $Q = y$. x guarded in Q implies $x \neq y$ and $Q\{P/x\} = y$. Similar to the Nil-case.

Pref: $Q = \alpha.P_1$ or $Q = \underline{\alpha}.P_1$. We prove only the former case. The latter one is simpler. In this case x is guarded in Q and $Q\{P/x\} = \alpha.(P_1\{P/x\})$. Consider the following possible subcases:

- $\alpha \in A$. In this case $\mathcal{UU}(Q\{P/x\}, A) = \{\alpha.(P_1\{P/x\}), \underline{\alpha}.(P_1\{P/x\})\} = \{((\alpha.P_1)\{P/x\}), ((\underline{\alpha}.P_1)\{P/x\})\} = \{\alpha.P_1, \underline{\alpha}.P_1\}\{P/x\} = \mathcal{UU}(Q, A)\{P/x\}$.
- $\alpha \notin A$. In this case $\mathcal{UU}(Q\{P/x\}, A) = \{\alpha.(P_1\{P/x\})\} = \{(\alpha.P_1)\{P/x\}\} = (\{\alpha.P_1\})\{P/x\} = \mathcal{UU}(Q, A)\{P/x\}$.

Sum: $Q = Q_1 + Q_2$. In this case x is guarded in Q implies x is guarded in Q_1 and in Q_2. By induction hypothesis we have that

$\mathcal{UU}(Q\{P/x\}, A) = \mathcal{UU}(Q_1\{P/x\}, A) \cup \mathcal{UU}(Q_2\{P/x\}, A) =$

$\big(\mathcal{UU}(Q_1, A)\{P/x\}\big) \cup \big(\mathcal{UU}(Q_2, A)\{P/x\}\big) = \big(\mathcal{UU}(Q_1, A) \cup \mathcal{UU}(Q_2, A)\big)\{P/x\} =$

$\mathcal{UU}(Q, A)\{P/x\}$.

Par: $Q = Q_1 \|_B Q_2$. Assume x guarded in Q and, hence, in Q_1 and in Q_2. Let us denote with $A_1 = B \backslash \mathcal{U}(Q_2\{P/x\})$ and $A_2 = B \backslash \mathcal{U}(Q_1\{P/x\})$. Then x guarded both in Q_1 and in Q_2 implies $\mathcal{U}(Q_1\{P/x\} = \mathcal{U}(Q_1)$ and $\mathcal{U}(Q_2\{P/x\} = \mathcal{U}(Q_2)$ (Proposition 7.2) and, hence, $A_1 = B \backslash \mathcal{U}(Q_2)$ and $A_2 = B \backslash \mathcal{U}(Q_1)$. By induction hypothesis we have that:

$\mathcal{S} = \mathcal{UU}(Q_1\{P/x\}, A \cup A_1) \|_B \mathcal{UU}(Q_2\{P/x\}, A \cup A_2) =$

$\big(\mathcal{UU}(Q_1, A \cup A_1)\{P/x\}\big) \|_B \big(\mathcal{UU}(Q_2, A \cup A_2)\{P/x\}\big) =$

$\big(\mathcal{UU}(Q_1, A \cup A_1) \|_B \mathcal{UU}(Q_2, A \cup A_2)\big)\{P/x\} = \mathcal{S}'\{P/x\}$

Now, let $R \in \mathcal{UU}(Q\{P/x\}, A)$. Then, by Definition 12.1, $R \in \mathcal{S}$ such that $\mathcal{U}(R) \subseteq \mathcal{U}(Q\{P/x\}) \cup A = \mathcal{U}(Q) \cup A$ (x guarded in Q and Proposition 7.2 imply $\mathcal{U}(Q\{P/x\}) = \mathcal{U}(Q)$). $R \in \mathcal{S}$ implies $R = R'\{P/x\}$ with $R' \in \mathcal{S}'$. On the other hand $R' \in \mathcal{S}'$ implies $R' = R'_1 \|_B R'_2$ with $R'_i \in \mathcal{UU}(Q_i, A \cup A_i)$, for $i = 1, 2$. Moreover x guarded in Q_i implies x guarded in $R'_i \in \mathcal{UU}(Q_i, A \cup A_i)$ and, hence, x guarded in R'. Again by Proposition 7.2, we have that $\mathcal{U}(R) = \mathcal{U}(R'\{P/x\}) = \mathcal{U}(R')$. We can conclude that $R' \in \mathcal{S}'$ such that $\mathcal{U}(R') = \mathcal{U}(R) \subseteq \mathcal{U}(Q) \cup A$. Then, by Definition 12.1, $R' \in \mathcal{UU}(Q, A)$ and, hence, $R \in \mathcal{UU}(Q, A)\{P/x\}$. Similarly we can prove that $R \in \mathcal{UU}(Q, A)\{P/x\}$ implies $R \in \mathcal{UU}(Q\{P/x\}, A)$.

Rel: $Q = Q_1[\Phi]$. Assume x guarded in Q and, hence, x in Q_1. By induction hypothesis $\mathcal{UU}(Q\{P/x\}, A) = \mathcal{UU}(Q_1\{P/x\}, \Phi^{-1}(A))[\Phi] = \big(\mathcal{UU}(Q_1, \Phi^{-1}(A))\{P/x\}\big)[\Phi] = \big(\mathcal{UU}(Q_1, \Phi^{-1}(A))[\Phi]\big)\{P/x\} = \mathcal{UU}(Q, A)\{P/x\}$.

Rec: $Q = \mathsf{rec}\, y.Q_1$. If $x = y$ then $Q\{P/x\} = Q$ and the statement follows easily. We can assume $x \neq y$. In this case x guarded in Q implies x guarded in Q_1 and $Q\{P/x\} = \mathsf{rec}\, y.(Q_1\{P/x\})$. We have to consider two possible subcases:

- $Q_1 \in \tilde{\mathbb{P}}_1$ (and, hence, $Q_1\{P/x\} \in \tilde{\mathbb{P}}_1$). In this case
  $\mathcal{UU}(Q\{P/x\}, A) = \{\mathsf{rec}\, y.(Q_1\{P/x\})\} \cup \mathcal{UU}(Q_1\{P/x\}, A)\{\mathsf{rec}\, y.(Q_1\{P/x\})/y\}$.
  By induction hypothesis $\mathcal{UU}(Q_1\{P/x\}, A)\{\mathsf{rec}\, y.(Q_1\{P/x\})/y\} =$
  $\big(\mathcal{UU}(Q_1, A)\{P/x\}\big)\{\mathsf{rec}\, y.(Q_1\{P/x\})/y\} =$
  $\big(\mathcal{UU}(Q_1, A)\{\mathsf{rec}\, y.Q_1/y\}\big)\{P/x\}$.
  Thus $\mathcal{UU}(Q\{P/x\}, A) =$
  $\{\mathsf{rec}\, y.(Q_1\{P/x\})\} \cup \big((\mathcal{UU}(Q_1, A)\{\mathsf{rec}\, y.Q_1/y\})\{P/x\}\big) =$
  $\big((\{\mathsf{rec}\, y.Q_1\})\{P/x\}\big) \cup \big((\mathcal{UU}(Q_1, A)\{\mathsf{rec}\, y.Q_1/y\})\{P/x\}\big) =$
  $\big(\{\mathsf{rec}\, y.Q_1\} \cup \mathcal{UU}(Q_1, A)\{\mathsf{rec}\, y.Q_1/y\}\big)\{P/x\} = \mathcal{UU}(Q, A)\{P/x\}$

- $Q_1 \notin \tilde{\mathbb{P}}_1$ (and, hence, $Q_1\{P/x\} \notin \tilde{\mathbb{P}}_1$). Let $S = \mathsf{unmark}(Q_1\{P/x\})$; x guarded in Q_1 and Proposition 7.7-2 imply $S = \mathsf{unmark}(Q_1)\{P/x\} = S_1\{P/x\}$. In this case
  $\mathcal{UU}(Q\{P/x\}, A) = \mathcal{UU}(Q_1\{P/x\}, A)\{\mathsf{rec}\, y.(S_1\{P/x\})/y\}$.
  By induction hypothesis $\mathcal{UU}(Q_1\{P/x\}, A)\{\mathsf{rec}\, y.(S_1\{P/x\})/y\} =$
  $\big(\mathcal{UU}(Q_1, A)\{P/x\}\big)\{\mathsf{rec}\, y.(S_1\{P/x\})/y\} =$
  $\big(\mathcal{UU}(Q_1, A)\{\mathsf{rec}\, y.S_1/y\}\big)\{P/x\} = \mathcal{UU}(Q, A)\{P/x\}$.

$\square$

Proposition 12.6 Let $Q \in \tilde{\mathbb{P}}$, X and $Y \subseteq \mathbb{A}$. Then $Q \xmapsto{X}_r Q'$ and $Y \cap \mathcal{U}(Q, A) = \emptyset$ implies $Q \xmapsto{X \cup (Y \backslash A)}_r Q'$.

Proof: The proof is by induction on length of derivation $Q \xmapsto{X}_r Q'$ and by case analysis on structure of Q. We only consider the the Rec-case. The other cases follow similar lines as Lemma 9.1-1.

Rec: $Q = \mathsf{rec}\, x.Q_1$. Assume that $Q \xmapsto{X}_r Q'$. Then, by operational rules, we also have that $S = Q_1\{\mathsf{rec}\, x.Q_1/x\} \xmapsto{X}_r Q'$. Now assume $Y \cap \mathcal{U}(Q, A) = Y \cap \mathcal{U}(Q_1, A) = \emptyset$. Since x guarded in Q_1 and Proposition 7.2 imply $\mathcal{U}(S, A) = \mathcal{U}(Q_1, A)$ and $Y \cap \mathcal{U}(S, A) = \emptyset$, by induction hypothesis $S \xmapsto{X \cup (Y \backslash A)}_r Q'$. Again by operational semantics, we can conclude that $Q \xmapsto{X \cup (Y \backslash A)}_r Q'$.

$\square$

Proposition 5.3 Let $Q, R \in \tilde{\mathbb{P}}$ and $A \subseteq \mathbb{A}$ such that $R \in \mathcal{UU}(Q, A)$. Then:

1. $Q \xrightarrow{\alpha} Q'$ implies $R \xmapsto{\alpha} R'$ for some $R' \in \mathcal{UU}(Q', A)$. Moreover $\mathcal{U}(R') \subseteq \mathcal{U}(R)$;

2. $Q \xrightarrow{X}_r Q'$ implies $R \xmapsto{X\backslash A}_r R'$ for some $R' \in \mathcal{U}\mathcal{U}(Q')$;

3. $R \xmapsto{\alpha} R'$ implies $Q \xrightarrow{\alpha} Q'$ for some Q' such that $R' \in \mathcal{U}\mathcal{U}(Q', A)$;

4. $R \xmapsto{X}_r R'$ implies $Q \xrightarrow{X}_r Q'$ for some Q' such that $R' \in \mathcal{U}\mathcal{U}(Q')$.

Proof: We prove Item 1 by induction on length of derivation $Q \xrightarrow{\alpha} Q'$ and Item 2 by induction on Q. The proof of Items 3 and 4 is omitted since they can be proved as Items 1 and 3 respectively. We proceed by case analysis on Q.

Nil, Var: $Q = \text{nil}$, $Q = x$. The latter case is not possible since $Q \not\xrightarrow{}$ and $Q \not\xrightarrow{}_r$. We can assume $Q = \text{nil}$. In this case $R \in \mathcal{U}\mathcal{U}(Q, A)$ implies $R = \text{nil}$.

  1. This case is not possible since $Q \not\xrightarrow{}$.

  2. $Q \xrightarrow{X}_r \text{nil} = Q'$, $R \xmapsto{X\backslash A}_r \text{nil} = R'$ and $R' \in \mathcal{U}\mathcal{U}(Q') = \{\text{nil}\}$.

Pref: $Q = \alpha.P_1$ or $Q = \underline{\alpha}.P_1$. Assume $Q = \alpha.P_1$ and consider the following cases:

  - $\alpha \in A$. $R \in \mathcal{U}\mathcal{U}(Q, A) = \{\alpha.P_1, \underline{\alpha}.P_1\}$ implies either $R = \alpha.P_1$ or $R = \underline{\alpha}.P_1$. Assume $R = \underline{\alpha}.P_1$ (the former case is simpler).

    1. $Q \xrightarrow{\alpha} P_1$, $R \xmapsto{\alpha} P_1$ and, by Lemma 12.2, $P_1 \in \tilde{\mathbb{P}}_1$ implies $P_1 \in \mathcal{U}\mathcal{U}(P_1, A)$. Moreover $P_1 \in \tilde{\mathbb{P}}_1$ implies $\mathcal{U}(P_1) = \emptyset \subseteq \mathcal{U}(R)$.

    2. $Q \xrightarrow{X}_r \underline{\alpha}.P_1 = Q'$, $\alpha \in A \subseteq \mathbb{A}$ implies $\alpha \notin (X\backslash A) \cup \{\tau\}$ and, by operational rules, $R \xmapsto{X\backslash A}_r \underline{\alpha}.P_1 = R'$. Finally, by Definition 12.1 $R' \in \mathcal{U}\mathcal{U}(Q') = \{\underline{\alpha}.P_1\}$.

  - $\alpha \notin A$. $R \in \mathcal{U}\mathcal{U}(Q, A) = \{\alpha.P_1\}$ implies $R = \alpha.P_1$. Similar to the previous case.

  Now assume $Q = \underline{\alpha}.P_1$. Then $R \in \mathcal{U}\mathcal{U}(Q, A) = \{\underline{\alpha}.P_1\}$ implies $R = \underline{\alpha}.P_1$.

  1. Also in this case $Q \xrightarrow{\alpha} P_1$, $R \xmapsto{\alpha} P_1$, $P_1 \in \mathcal{U}\mathcal{U}(P_1, A)$ and $\mathcal{U}(P_1) \subseteq \mathcal{U}(R)$.

  2. $Q \xrightarrow{X}_r \underline{\alpha}.P_1 = Q'$, implies $\alpha \notin X \cup \{\tau\} \supseteq (X\backslash A) \cup \{\tau\}$. Then, by operational semantics, $R \xmapsto{X\backslash A}_r \underline{\alpha}.P_1 = R'$ and $R' \in \mathcal{U}\mathcal{U}(Q') = \{\underline{\alpha}.P_1\}$.

Sum: $Q = Q_1 + Q_2$. By Definition 12.1 $R \in \mathcal{U}\mathcal{U}(Q, A)$ implies $R = R_1 + R_2$ with $R_i \in \mathcal{U}\mathcal{U}(Q_i, A)$ for $i = 1, 2$.

  1. $Q \xrightarrow{\alpha} Q'$ implies either (i) $Q_1 \xrightarrow{\alpha} Q'$ or (ii) $Q_2 \xrightarrow{\alpha} Q'$. Consider the (i) case (the (ii) case is symmetric). By induction hypothesis $R_1 \xmapsto{\alpha} R'$ and, hence, $R \xmapsto{\alpha} R'$ for some $R' \in \mathcal{U}\mathcal{U}(Q', A)$. Moreover, again by induction hypothesis, $\mathcal{U}(R') \subseteq \mathcal{U}(R_1) \subseteq \mathcal{U}(R_1) \cup \mathcal{U}(R_2) = \mathcal{U}(R)$.

  2. $Q \xrightarrow{X}_r Q'$ implies $Q_1 \xrightarrow{X} Q'_1$, $Q_2 \xrightarrow{X} Q'_2$ and $Q' = Q'_1 + Q'_2$. By induction hypothesis $R_1 \xmapsto{X\backslash A} R'_1$, $R_2 \xmapsto{X\backslash A} R'_2$ for some $R'_1 \in \mathcal{U}\mathcal{U}(Q'_1)$ and $R'_2 \in \mathcal{U}\mathcal{U}(Q'_2)$. Thus, $R \xmapsto{X\backslash A} R'_1 + R'_2 = R'$ and $R' \in \mathcal{U}\mathcal{U}(Q'_1) + \mathcal{U}\mathcal{U}(Q'_2) = \mathcal{U}\mathcal{U}(Q')$.

Par: $Q = Q_1 \|_B Q_2$. Let $A_1 = B\backslash\mathcal{U}(Q_2)$ and $A_2 = B\backslash\mathcal{U}(Q_1)$. In this case $R \in \mathcal{U}\mathcal{U}(Q, A)$ implies $R = R_1 \|_B R_2$ with $R_i \in \mathcal{U}\mathcal{U}(Q_i, A \cup A_i)$, for $i = 1, 2$, and $\mathcal{U}(R) \subseteq \mathcal{U}(Q) \cup A$.

1. Assume $Q \xrightarrow{\alpha} Q'$ and consider the following subcases:

   - $\alpha \notin B$, $Q_1 \xrightarrow{\alpha} Q_1'$ and $Q' = \mathsf{clean}(Q_1' \|_B Q_2)$. By induction $R_1 \xmapsto{\alpha} R_1'$ for some $R_1' \in \mathcal{UU}(Q_1', A \cup A_1)$ and, by operational semantics, $R \xmapsto{\alpha} R_1' \|_B R_2 = R'$. Now we want prove that $R' \in \mathcal{UU}(Q_1' \|_B Q_2)$. By Definition 12.1 we have to prove that (1) $R_1' \in \mathcal{UU}(Q_1', A \cup A_1)$, (2) $R_2 \in \mathcal{UU}(Q_2, A \cup A_2')$, where $A_2' = B \backslash \mathcal{U}(Q_1')$ and (3) $\mathcal{U}(R') \subseteq \mathcal{U}(Q_1' \|_B Q_2)$. (1) follows by induction hypothesis; let us prove (2) and (3).

   (2) $Q_1 \xrightarrow{\alpha} Q_1'$ implies $\mathcal{U}(Q_1') \subseteq \mathcal{U}(Q_1)$ and $A_2 = B \backslash \mathcal{U}(Q_1) \subseteq B \backslash \mathcal{U}(Q_1') = A_2'$ (see Proposition 12.4). Then, $R_2 \in \mathcal{UU}(Q_2, A \cup A_2)$, $A \cup A_2 \subseteq A \cup A_2'$ and Proposition 12.3-2, imply $R_2 \in \mathcal{UU}(Q_2, A \cup A_2')$;

   (3) $R_1' \in \mathcal{UU}(Q_1', A \cup A_1)$ and $R_2 \in \mathcal{UU}(Q_2, A \cup A_2)$ implies, by Proposition 12.3-1, $\mathcal{U}(R_1') \subseteq \mathcal{U}(Q_1') \cup A \cup A_1$ and $\mathcal{U}(R_2) \subseteq \mathcal{U}(Q_2) \cup A \cup A_2$. Let $\mu \in \mathcal{U}(R') \backslash A$ (if $\mu \in A$ trivially $\mu \in (Q_1' \|_B Q_2) \cup A$) and consider the following possible cases:

      - $\mu \notin B$. In this case, Definition 2.3 and $\mu \in \mathcal{U}(R')$ imply either $\mu \in \mathcal{U}(R_1')$ or $\mu \in \mathcal{U}(R_2)$. Moreover $\mathcal{U}(R_1') \backslash B \subseteq (\mathcal{U}(Q_1') \cup A \cup A_1) \backslash B = (\mathcal{U}(Q_1') \backslash B) \cup (A \backslash B) \cup (A_1 \backslash B) = \mathcal{U}(Q_1', B) \cup (A \backslash B) \subseteq \mathcal{U}(Q_1' \|_B Q_2) \cup A$. Similarly we have $\mathcal{U}(R_2) \backslash B \subseteq \mathcal{U}(Q_2, B) \cup (A \backslash B) \subseteq \mathcal{U}(Q_1' \|_B Q_2) \cup A$ and, in both cases, $\mu \in \mathcal{U}(Q_1' \|_B Q_2) \cup A$.

      - $\mu \in B$. In this case $\mu \in \mathcal{U}(R')$ implies $\mu \in \mathcal{U}(R_1')$ and $\mu \in \mathcal{U}(R_2)$. Moreover $\mathcal{U}(R_1') \subseteq \mathcal{U}(Q_1') \cup A \cup A_1$, $\mathcal{U}(R_2) \subseteq \mathcal{U}(Q_2) \cup A \cup A_2$ and $\mu \notin A$ imply $\mu \in (\mathcal{U}(Q_1') \cup A_1) \cap (\mathcal{U}(Q_2) \cup A_2)$. Thus, since $\mathcal{U}(Q_1') \cap A_2 \subseteq \mathcal{U}(Q_1') \cap A_2' = \mathcal{U}(Q_1') \cap (B \backslash \mathcal{U}(Q_1')) = \emptyset$ and $\mathcal{U}(Q_2) \cap A_1 = \mathcal{U}(Q_2) \cap (B \backslash \mathcal{U}(Q_2)) = \emptyset$, we have $\mu \in (\mathcal{U}(Q_1') \cap \mathcal{U}(Q_2)) \cup (A_1 \cap A_2)$.
      Now, assume, by contradiction, $\mu \in A_1 \cap A_2$. Then $\mu \in B$ such that $\mu \notin \mathcal{U}(Q_1)$ and $\mu \notin \mathcal{U}(Q_2)$. By Definition 2.3 we have that $\mu \notin \mathcal{U}(Q)$ and $\mu \notin \mathcal{U}(R) \subseteq \mathcal{U}(Q) \cup A$. Again by Definition 2.3, $\mu \in B$, $\mu \notin \mathcal{U}(R)$ and $\mu \in \mathcal{U}(R_2)$ imply $\mu \notin \mathcal{U}(R_1)$. This case is not possible since $\mu \in \mathcal{U}(R_1') \subseteq \mathcal{U}(R_1)$. Finally, $\mu \in \mathcal{U}(Q_1') \cap \mathcal{U}(Q_2)$ and $\mu \in B$ imply $\mu \in \mathcal{U}(Q_1' \|_B Q_2) \subseteq \mathcal{U}(Q_1' \|_B Q_2 \cup A)$.

   Finally, $R' \in \mathcal{UU}(Q_1' \|_B Q_2, A)$ implies $R' \in \mathcal{UU}(\mathsf{clean}(Q_1' \|_B Q_2), A) = \mathcal{UU}(Q', A)$ (see Proposition 12.3-3).
   It remains to prove that $\mathcal{U}(R') \subseteq \mathcal{U}(R)$. Again by induction hypothesis, we have $\mathcal{U}(R_1') \subseteq \mathcal{U}(R_1)$. Then, by Proposition 7.1-4, $\mathcal{U}(R_1', C) \subseteq \mathcal{U}(R_1, C)$ for a generic $C \subseteq \mathbb{A}_\tau$ and, by Definition 2.3, $\mathcal{U}(R') \subseteq \mathcal{U}(R)$.

   - $\alpha \notin B$, $Q_2 \xrightarrow{\alpha} Q_2'$ and $Q' = \mathsf{clean}(Q_1 \|_B Q_2')$. Similar to the previous case.

   - $\alpha \in B$, $Q_i \xrightarrow{\alpha} Q_i'$ for $i = 1, 2$ and $Q' = \mathsf{clean}(Q_1' \|_B Q_2')$. By induction hypothesis, we have $R_1 \xmapsto{\alpha} R_1'$, $R_2 \xmapsto{\alpha} R_2'$ for some $R_1' \in \mathcal{UU}(Q_1', A \cup A_1)$ and $R_2' \in \mathcal{UU}(Q_2', A \cup A_2)$. By operational rules $R \xmapsto{\alpha} R_1' \|_B R_2' = R'$. Now, we prove that $R' \in \mathcal{UU}(Q_1' \|_B Q_2')$. Then, by Definition 12.1, we have to prove that (1) $R_1' \in \mathcal{UU}(Q_1', A \cup A_1')$, $R_2' \in \mathcal{UU}(Q_2', A \cup A_2')$, where $A_1' = B \backslash \mathcal{U}(Q_2')$, $A_2' = B \backslash \mathcal{U}(Q_1')$, and (2) $\mathcal{U}(R') \subseteq \mathcal{U}(Q_1' \|_B Q_2')$.

   (1) $Q_2 \xrightarrow{\alpha} Q_2'$ implies $\mathcal{U}(Q_2') \subseteq \mathcal{U}(Q_2)$ and $A_1 = B \backslash \mathcal{U}(Q_2) \subseteq B \backslash \mathcal{U}(Q_2') = A_1'$ (see Proposition 12.4). Then, $R_1' \in \mathcal{UU}(Q_1', A \cup A_1)$, $A \cup A_1 \subseteq A \cup A_1'$ and Proposition 12.3-2 imply $R_1' \in \mathcal{UU}(Q_1', A \cup A_1')$. Similarly we have $R_2' \in \mathcal{UU}(Q_2', A \cup A_2')$;

(2) $R'_1 \in \mathcal{UU}(Q'_1, A \cup A_1)$ and $R'_2 \in \mathcal{UU}(Q'_2, A \cup A_2)$ implies, by Proposition 12.3-1, $\mathcal{U}(R'_1) \subseteq \mathcal{U}(Q'_1) \cup A \cup A_1$ and $\mathcal{U}(R'_2) \subseteq \mathcal{U}(Q'_2) \cup A \cup A_2$. Assume, again, $\mu \in \mathcal{U}(R')\backslash A$ and consider the following possible cases:

- $\mu \notin B$. Definition 2.3 and $\mu \in \mathcal{U}(R')$ imply either $\mu \in \mathcal{U}(R'_1)$ or $\mu \in \mathcal{U}(R'_2)$. Moreover $\mathcal{U}(R'_1)\backslash B \subseteq (\mathcal{U}(Q'_1) \cup A \cup A_1)\backslash B = (\mathcal{U}(Q'_1)\backslash B) \cup (A\backslash B) \cup (A_1\backslash B) = \mathcal{U}(Q'_1, B) \cup (A\backslash B) \subseteq \mathcal{U}(Q'_1 \parallel_B Q'_2) \cup A$. Similarly $\mathcal{U}(R'_2)\backslash B \subseteq \mathcal{U}(Q'_2, B) \cup (A\backslash B) \subseteq \mathcal{U}(Q'_1 \parallel_B Q'_2) \cup A$. In both cases $\mu \in \mathcal{U}(Q'_1 \parallel_B Q'_2) \cup A$.

- $\mu \in B$ and $\mu \notin A$. In this case $\mu \in \mathcal{U}(R')$ implies $\mu \in \mathcal{U}(R'_1)$ and $\mu \in \mathcal{U}(R'_2)$. Moreover $\mathcal{U}(R'_1) \subseteq \mathcal{U}(Q'_1) \cup A \cup A_1$, $\mathcal{U}(R'_2) \subseteq \mathcal{U}(Q'_2) \cup A \cup A_2$ and $\mu \notin A$ imply $\mu \in (\mathcal{U}(Q'_1) \cup A_1) \cap (\mathcal{U}(Q'_2) \cup A_2)$. Now, since $\mathcal{U}(Q'_1) \cap A_2 \subseteq \mathcal{U}(Q'_1) \cap A'_2 = \mathcal{U}(Q'_1) \cap (B\backslash\mathcal{U}(Q'_1)) = \emptyset$ and, similarly, $\mathcal{U}(Q'_2) \cap A_1 = \emptyset$, $\mu \in (\mathcal{U}(Q'_1) \cap \mathcal{U}(Q'_2)) \cup (A_1 \cap A_2)$.
Assume, by contradiction, $\mu \in A_1 \cap A_2$. Then $\mu \in B$ such that $\mu \notin \mathcal{U}(Q_1)$ and $\mu \notin \mathcal{U}(Q_2)$. By Definition 2.3 we have that $\mu \notin \mathcal{U}(Q)$. Thus, $\mathcal{U}(R) \subseteq \mathcal{U}(Q) \cup A$, $\mu \notin \mathcal{U}(Q)$ and $\mu \notin A$ imply $\mu \notin \mathcal{U}(R)$ and either $\mu \notin \mathcal{U}(R_1)$ or $\mu \notin \mathcal{U}(R_2)$ (again by Definition 2.3). We have either $\mu \in \mathcal{U}(R'_1)$ and $\mu \notin \mathcal{U}(R_1)$ or $\mu \in \mathcal{U}(R'_2)$ and $\mu \notin \mathcal{U}(R_2)$. Both cases are not possible since $\mathcal{U}(R'_1) \subseteq \mathcal{U}(R_1)$ and $\mathcal{U}(R'_2) \subseteq \mathcal{U}(R_2)$.
We can conclude that $\mu \in \mathcal{U}(Q'_1) \cap \mathcal{U}(Q'_2)$ and since $\mu \in B$ implies $\mu \notin \mathbb{A}_\tau\backslash B$, $\mu \in \mathcal{U}(Q'_1, \mathbb{A}_\tau\backslash B) \cap \mathcal{U}(Q'_2, \mathbb{A}_\tau\backslash B) \subseteq \mathcal{U}(Q'_1 \parallel_B Q'_2)$.

Again $R' \in \mathcal{UU}(Q'_1\parallel_B Q'_2, A)$ implies $R' \in \mathcal{UU}(\mathsf{clean}(Q'_1\parallel_B Q'_2), A) = \mathcal{UU}(Q', A)$. Moreover, again by induction hypothesis, $\mathcal{U}(R'_1) \subseteq \mathcal{U}(R_1)$ and $\mathcal{U}(R'_2) \subseteq \mathcal{U}(R_2)$. Then, similarly to the previous cases, we can prove that $\mathcal{U}(R') \subseteq \mathcal{U}(R)$.

2. By operational rules, $Q \xrightarrow{X}_r Q'$ if there exist $X_1, X_2 \subseteq \mathbb{A}$ such that $Q_1 \xrightarrow{X_1}_r Q'_1$, $Q_2 \xrightarrow{X_2}_r Q'_2$ with $X \subseteq ((X_1 \cup X_2) \cap B) \cup ((X_1 \cap X_2)\backslash B)$ and $Q' = \mathsf{clean}(Q'_1 \parallel_B Q'_2)$. Then, by induction hypothesis, $R_1 \xrightarrow{X_1\backslash(A\cup A_1)}_r R'_1$, $R_2 \xrightarrow{X_2\backslash(A\cup A_2)}_r R'_2$ for some $R'_1 \in \mathcal{UU}(Q'_1)$ and $R'_2 \in \mathcal{UU}(Q'_2)$.

Now, let $X'_i = (X_i\backslash(A \cup A_i)) \cup (B\backslash\mathcal{U}(R_i))$, for $i = 1, 2$. $R_i \xrightarrow{X_i\backslash(A\cup A_i)}_r R'_i$ and $(B\backslash\mathcal{U}(R_i)) \cap \mathcal{U}(R_i) = \emptyset$ imply, by Proposition 12.6, $R_i \xrightarrow{X'_i}_r R'_i$. Now, we prove that $X\backslash A \subseteq ((X'_1 \cup X'_2) \cap B) \cup ((X'_1 \cap X'_2)\backslash B)$. Let $\mu \in X\backslash A$, i.e. $\mu \in X$ such that $\mu \notin A$, and consider the following possible cases:

- $\mu \in B$. Then $\mu \in X$ implies either $\mu \in X_1$ or $\mu \in X_2$. Assume $\mu \in X_1$ (the case in which $\mu \in X_2$ can be proved similarly). If $\mu \notin A_1$ then $\mu \in X_1\backslash(A \cup A_1) \subseteq X'_1 \subseteq X'_1 \cup X'_2$. We can assume $\mu \in A_1$ and, hence, $\mu \notin \mathcal{U}(Q_2)$. By Definition 2.3 $\mu \in B$ and $\mu \notin \mathcal{U}(Q_2)$ implies $\mu \notin \mathcal{U}(Q)$ and, hence, $\mu \notin \mathcal{U}(R) \subseteq \mathcal{U}(Q) \cup A$. Again by Definition 2.3, $\mu \in B$ and $\mu \notin \mathcal{U}(R)$ implies either $\mu \notin \mathcal{U}(R_1)$ or $\mu \notin \mathcal{U}(R_2)$. Thus we have either $\mu \in B\backslash\mathcal{U}(R_1) \subseteq X'_1$ or $\mu \in B\backslash\mathcal{U}(R_2) \subseteq X'_2$ and, hence, $\mu \in X'_1 \cup X'_2$.

- $\mu \notin B$. In this case $\mu \in X$ implies $\mu \in X_1$ and $\mu \in X_2$. Moreover, since $\mu \notin A_1, A_2 \subseteq B$, we have $\mu \in X_1\backslash(A \cup A_1) \subseteq X'_1$ and $\mu \in X_2\backslash(A \cup A_2) \subseteq X'_2$. Thus $\mu \in X'_1 \cap X'_2$.

By operational semantics, we can conclude that $R \xrightarrow{X\backslash A} R'_1 \parallel_B R'_2 = R'$. It remains

to prove that $R' \in \mathcal{UU}(Q')$. Let $A'_1 = B \backslash \mathcal{U}(Q'_2)$ and $A'_2 = B \backslash \mathcal{U}(Q'_1)$. $R'_i \in \mathcal{UU}(Q'_i)$ and Proposition 12.3-2 imply $R'_i \in \mathcal{UU}(Q'_i, A'_i)$ for $i = 1, 2$. Moreover $R'_i \in \mathcal{UU}(Q'_i)$ and Proposition 12.3-1 imply $\mathcal{U}(R'_i) = \mathcal{U}(Q'_i)$ for $i = 1, 2$. We can conclude that $\mathcal{U}(R') = \mathcal{U}(Q'_1 \|_B Q'_2)$. By Definition 12.1, $R'_i \in \mathcal{UU}(Q'_i, A'_i)$, for $i = 1, 2$, and $\mathcal{U}(R') = \mathcal{U}(Q'_1 \|_B Q'_2)$ imply $R' \in \mathcal{UU}(Q'_1 \|_B Q'_2)$. Then, by Proposition 12.3-3, we can conclude that $R' \in \mathcal{UU}(\mathsf{clean}(Q'_1 \|_B Q'_2)) = \mathcal{UU}(Q')$.

Rel: $Q = Q_1[\Phi]$. In this case $R \in \mathcal{UU}(Q, A)$ implies $R = R_1[\Phi]$ with $R_1 \in \mathcal{UU}(Q_1, \Phi^{-1}(A))$.

1. Assume $Q \xrightarrow{\alpha} Q'$. By operational semantics there exists $\beta \in \Phi^{-1}(\alpha)$ such that $Q_1 \xrightarrow{\beta} Q'_1$ and $Q' = Q'_1[\Phi]$. By induction hypothesis $R_1 \xmapsto{\beta} R'_1$ for some $R'_1 \in \mathcal{UU}(Q'_1, \Phi^{-1}(A))$. Thus, $R \xmapsto{\alpha} R'_1[\Phi] = R'$ and $R' \in \mathcal{UU}(Q'_1, \Phi^{-1}(A))[\Phi] = \mathcal{UU}(Q', A)$. Moreover, again by induction hypothesis, $\mathcal{U}(R'_1) \subseteq \mathcal{U}(R_1)$ and $\mathcal{U}(R') = \Phi(\mathcal{U}(R'_1)) \subseteq \Phi(\mathcal{U}(R_1)) = \mathcal{U}(R)$.

2. Assume that $Q \xrightarrow{X}_r Q'_1[\Phi] = Q'$ and let $Y = \Phi^{-1}(X \cup \{\tau\}) \backslash \{\tau\}$. By operational rules we have that $Q_1 \xrightarrow{Y}_r Q'_1$ and, by induction hypothesis, $R_1 \xmapsto{Y \backslash \Phi^{-1}(A)}_r R'_1$ for some $R'_1 \in \mathcal{UU}(Q'_1)$. Moreover, $\tau \notin \Phi^{-1}(A)$ implies $Y \backslash \Phi^{-1}(A) = (\Phi^{-1}(X \cup \{\tau\}) \backslash \{\tau\}) \backslash \Phi^{-1}(A) = (\Phi^{-1}(X \cup \{\tau\}) \backslash \Phi^{-1}(A)) \backslash \{\tau\} = \Phi^{-1}((X \cup \{\tau\}) \backslash A) \backslash \{\tau\} = \Phi^{-1}((X \backslash A) \cup \{\tau\}) \backslash \{\tau\}$. By operational semantics, $R \xmapsto{X \backslash A}_r R'_1[\Phi] = R'$ and $R' = R'_1[\Phi] \in \mathcal{UU}(Q'_1)[\Phi] = \mathcal{UU}(Q')$.

Rec: $Q = \mathsf{rec}\, x.Q_1$. Consider the following subcases:

- $Q_1 \notin \tilde{\mathbb{P}}_1$. By Definition 12.1, $R \in \mathcal{UU}(Q, A)$ implies $R = R_1\{\mathsf{rec}\, x.S_1/x\}$ with $R_1 \in \mathcal{UU}(Q_1, A)$ and $S_1 = \mathsf{unmark}(Q_1)$.

  1. Let $S = Q_1\{\mathsf{rec}\, x.S_1/x\} \xrightarrow{\alpha} Q'$. x guarded in Q_1 and Proposition 12.5 imply $\mathcal{UU}(S, A) = \mathcal{UU}(Q_1\{\mathsf{rec}\, x.S_1/x\}, A) = \mathcal{UU}(Q_1, A)\{\mathsf{rec}\, x.S_1/x\}$ and, hence, $R \in \mathcal{UU}(S, A)$. Now, assume that $Q \xrightarrow{\alpha} Q'$. By operational semantics, we also have $S \xrightarrow{\alpha} Q'$. Thus, by induction hypothesis, $R \xmapsto{\alpha} R'$ for some $R' \in \mathcal{UU}(Q', A)$ and $\mathcal{U}(R') \subseteq \mathcal{U}(R)$.

  2. $Q \xrightarrow{X}_r Q'$ implies $Q_1 \xrightarrow{X}_r Q'_1$ and $Q' = \mathsf{rec}\, x.Q'_1$. By induction hypothesis we have $R_1 \xmapsto{X \backslash A} R'_1$ for some $R'_1 \in \mathcal{UU}(Q'_1)$. $R_1 \xmapsto{X \backslash A} R'_1$ implies, by Proposition in [6], $R = R_1\{\mathsf{rec}\, x.S_1/x\} \xmapsto{X \backslash A} R'_1\{\mathsf{rec}\, x.S_1/x\} = R'$. It remains to prove that $R' \in \mathcal{UU}(Q')$. $Q'_1 \in \tilde{\mathbb{P}}_1$ implies $\mathcal{UU}(Q') = \{\mathsf{rec}\, x.Q'_1\} \cup \mathcal{UU}(Q'_1)\{\mathsf{rec}\, x.Q'_1/x\}$ with $Q'_1 = \mathsf{unmark}(Q'_1) = \mathsf{unmark}(Q_1) = S_1$ (by Proposition 7.8-2). Otherwise, we have $\mathcal{UU}(Q') = \mathcal{UU}(Q'_1)\{\mathsf{rec}\, x.\mathsf{unmark}(Q'_1)/x\}$ with $\mathsf{unmark}(Q'_1) = \mathsf{unmark}(Q_1) = S_1$. In both cases, $R'_1 \in \mathcal{UU}(Q'_1)$ implies $R' = R'_1\{\mathsf{rec}\, x.S_1/x\} \in \mathcal{UU}(Q'_1)\{\mathsf{rec}\, x.S_1/x\} \subseteq \mathcal{UU}(Q')$.

- $Q_1 \in \tilde{\mathbb{P}}_1$. In this case $R \in \mathcal{UU}(Q, A)$ implies either $R = \mathsf{rec}\, x.Q_1$ or $R = R_1\{\mathsf{rec}\, x.Q_1/x\}$ with $R_1 \in \mathcal{UU}(Q_1, A)$. Assume $R = \mathsf{rec}\, x.Q_1$ (the case in which $R = R_1\{\mathsf{rec}\, x.Q_1/x\}$ can be proved as the previous one).
  Let $S = Q_1\{\mathsf{rec}\, x.Q_1/x\} = Q_1\{\mathsf{rec}\, x.\mathsf{unmark}(Q_1)/x\}$. $Q_1 \in \tilde{\mathbb{P}}_1$ implies $S \in \tilde{\mathbb{P}}_1$ and, by Lemma 12.2, $Q_1 \in \mathcal{UU}(Q_1, A)$ and $S \in \mathcal{UU}(S, A)$.

1. Assume that $Q \xrightarrow{\alpha} Q'$. By operational semantics, $S \xrightarrow{\alpha} Q'$ and, by induction hypothesis, $S \xmapsto{\alpha} R'$ for some $R' \in \mathcal{UU}(Q', A)$. By operational rules $S \xmapsto{\alpha} R'$ implies $R \xmapsto{\alpha} R'$. Moreover, again by induction hypothesis, $\mathcal{U}(R') \subseteq \mathcal{U}(S)$. Finally, x guarded in Q_1 and Proposition 7.2 implies $\mathcal{U}(R') \subseteq \mathcal{U}(S) = \mathcal{U}(Q_1) = \mathcal{U}(R)$.

2. $Q \xrightarrow{X}_r \operatorname{rec} x.Q_1' = Q'$ implies $Q_1 \xrightarrow{X}_r Q_1'$. Since $Q_1 \in \mathcal{UU}(Q_1, A)$, by induction hypothesis we have $Q_1 \xmapsto{X \backslash A}_r R_1'$ for some $R_1' \in \mathcal{UU}(Q_1')$. Now, $Q_1 \xmapsto{X \backslash A}_r R_1'$ implies $Q_1\{\operatorname{rec} x.Q_1/x\} \xmapsto{X \backslash A}_r R_1'\{\operatorname{rec} x.Q_1/x\} = R'$ and, by operational rules, $R \xmapsto{X \backslash A}_r R'$. Moreover, since, as in the previous case, $\mathcal{UU}(Q_1')\{\operatorname{rec} x.Q_1/x\} \subseteq \mathcal{UU}(Q')$, $R_1' \in \mathcal{UU}(Q_1')$ implies $R' = R_1'\{\operatorname{rec} x.Q_1/x\} \in \mathcal{UU}(Q')$.

$\square$