Dynamic Probabilistic Input Output Automata (Extended Version)

Pierre Civit
Sorbonne Université, CNRS, LIP6, France

Maria Potop-Butucaru
Sorbonne Université, CNRS, LIP6, France

Abstract

We present *probabilistic dynamic I/O automata*, a framework to model dynamic probabilistic systems. Our work extends *dynamic I/O Automata* formalism of Attie & Lynch [2] to probabilistic setting. The original dynamic I/O Automata formalism included operators for parallel composition, action hiding, action renaming, automaton creation, and behavioral sub-typing by means of trace inclusion. They can model mobility by using signature modification. They are also hierarchical: a dynamically changing system of interacting automata is itself modeled as a single automaton. Our work extends to probabilistic settings all these features. Furthermore, we prove necessary and sufficient conditions to obtain the implementation monotonicity with respect to automata creation and destruction. Our construction uses a novel proof technique based on homomorphism that can be of independent interest. Our work lays down the foundations for extending *composable secure-emulation* of Canetti et al. [5] to dynamic settings, an important tool towards the formal verification of protocols combining probabilistic distributed systems and cryptography in dynamic settings (e.g. blockchains, secure distributed computation, cybersecure distributed protocols etc).

2012 ACM Subject Classification Theory of computation Distributed algorithms

Keywords and phrases Automata, Distributed Computing, Formal Verification, Dynamic systems

1 Introduction

Distributed computing area faces today important challenges coming from modern applications such as peer-to-peer networks, cooperative robotics, dynamic sensor networks, adhoc networks and more recently, cryptocurrencies and blockchains which have a tremendous impact in our society. These newly emerging fields of distributed systems are characterized by an extreme dynamism in terms of structure, content and load. Moreover, they have to offer strong guarantees over large scale networks which is usually impossible in deterministic settings. Therefore, most of these systems use probabilistic algorithms and randomized techniques in order to offer scalability features. However, the vulnerabilities of these systems may be exploited with the aim to provoke an unforeseen execution that diverges from the understanding or intuition of the developers. Therefore, formal validation and verification of these systems has to be realized before their industrial deployment.

It is difficult to attribute the first formalization of concurrent systems to some particular authors [18, 9, 1, 17, 10, 14, 8]. Lynch and Tuttle [11] proposed the formalism of *Input/Output Automata* to model deterministic asynchronous distributed systems. Relationship between process algebra and I/O automata are discussed in [21, 16]. Later, this formalism is extended by Segala in [20] with Markov decision processes [19]. In order to model randomized distributed systems Segala proposes *Probabilistic Input/Output Automata*. In this model each process in the system is an automaton with probabilistic transitions. The probabilistic protocol is the parallel composition of the automata modeling each participant.

The modelisation of dynamic behavior in distributed systems has been addressed by Attie & Lynch in [2] where they propose *Dynamic Input Output Automata* formalism. This formalism extends the *Input/Output Automata* with the ability to change their signature dynamically (i.e. the set of actions in which the automaton can participate) and to create other I/O automata or destroy existing I/O automata. The formalism introduced in [2] does not cover the case of probabilistic distributed systems and therefore cannot be used in the verification of recent blockchains such as Algorand [6].

In order to respond to the need of formalisation in secure distributed systems, Canetti & al. proposed in [3] *task-structured probabilistic Input/Output automata* (TPIOA) specifically designed for the analysis of cryptographic protocols. Task-structured probabilistic Input/Output automata are Probabilistic Input/Output automata extended with tasks that are equivalence classes on the set of actions. The task-structure allows a generalisation of "off-line scheduling" where the non-determinism of the system is resolved in advance by a *task-scheduler*, i.e. a sequence of tasks chosen in advance that trigger the actions among the enabled ones. They define the parallel composition for this type of automata. Inspired by the literature in security area they also define the notion of implementation for TPIOA. Informally, the implementation of a Task-structured probabilistic Input/Output automata should look "similar" to the specification whatever will be the external environment of execution. Furthermore, they provide compositional results for the implementation relation. Even thought the formalism proposed in [5] (built on top of the one of [3]) has been already used in the formal proof of various cryptographic protocols [4, 22], this formalism does not capture the dynamicity of probabilistic dynamic systems such as peer-to-peer networks or blockchains systems where the set of participants dynamically changes.

Our contribution. In order to cope with dynamicity and probabilistic nature of modern distributed systems we propose an extension of the two formalisms introduced in [2] and [3]. Our extension uses a refined definition of probabilistic configuration automata in order to cope with dynamic actions. The main result of our formalism is as follows: the implementation of probabilistic configuration automata is monotonic to automata creation and destruction. That is, if systems $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ differ only in that $X_{\mathcal{A}}$ dynamically creates and destroys automaton $\mathcal{A}$ instead of creating and destroying automaton $\mathcal{B}$ as $X_{\mathcal{B}}$ does, and if $\mathcal{A}$ implements $\mathcal{B}$ (in the sense they cannot be distinguished by any external observer), then $X_{\mathcal{A}}$ implements $X_{\mathcal{B}}$. This result enables a design and refinement methodology based solely on the notion of externally visible behavior and permits the refinement of components and subsystems in isolation from the rest of the system. In our construction, we exhibit the need of considering only *creation-oblivious* schedulers in the implementation relation, i.e. a scheduler that, upon the (dynamic) creation of a sub-automaton $\mathcal{A}$, does not take into account the previous internal actions of $\mathcal{A}$ to output (randomly) a transition. Surprisingly, the task-schedulers introduced by Canetti & al. [3] are not creation-oblivious. Interestingly, an important contribution of the paper of independent interest is the proof technique we used in order to obtain our results. Differently from [2] and [3] which build their constructions mainly on induction techniques, we developed an elegant homomorphism based technique which aim to render the proofs modular. This proof technique can be easily adapted in order to further extend our framework with cryptography and time.

It should be noted that our work is an intermediate step before extending composable secure-emulation [5] to dynamic settings. This extension is necessary for formal verification of secure dynamic distributed systems (e.g. blockchain systems).

Paper organization. The paper is organized as follow. Section 3 is dedicated to a brief introduction of the notion of probabilistic measure and recalls notations used in defining Signature I/O automata of [2]. Section 4 builds on the frameworks proposed in [2] and [3] in order to lay down the preliminaries of our formalism. More specifically, we introduce the definitions of probabilistic signed I/O automata and define their composition and implementation. In Section 5 we extend the definition of configuration automata proposed in [2] to probabilistic configuration automata then we define the composition of probabilistic configuration automata and prove its closeness in Section 7. Section 6 contains definitions related to the behavioural semantic of automata, e.g. executions, traces, etc. Section 8 introduces implementation relationship, which allows to formalise the idea that a concrete system is meeting the specification of an abstract object. The key result of our formalisation, the monotonicity of PSIOA implementations with respect to creation and destruction, is presented in the end of Section 9 and demonstrated in the remaining sections, up to Section 14). Section 15 explains why the off-line scheduler introduced by Canetti & al. [5] is not creation-oblivious and therefore cannot be used to obtain our key result.

2 Warm up

In this section we describe the paper in a very informal way, giving some intuitions on the role of each section. The section 3 gives some preliminaries on probability and measure, while a glossary can be found at the end of the document, section 17.

2.1 Probabilistic Signature Input/Output Automata (PSIOA)

The section 4 defines the notion of probabilistic signature Input/Output automata (PSIOA). A PSIOA $\mathcal{A}$ is an automaton that can move from one *state* to another through *actions*. The set of states of $\mathcal{A}$ is then denoted $Q_{\mathcal{A}}$, while we note $\bar{q}_{\mathcal{A}} \in Q_{\mathcal{A}}$ the unique start state of $\mathcal{A}$. At each state $q \in Q_{\mathcal{A}}$ some actions can be triggered in its signature $sig(\mathcal{A})(q)$. Such an action leads to a new state with a certain probability. The measure of probability triggered by an action a in a state q is denoted $\eta_{(\mathcal{A},q,a)}$. The model aims to allow the composition of several automata (noted $\mathcal{A}_1||...||\mathcal{A}_n$) to capture the idea of an interaction between them. That is why a signature is composed by three categories of actions: the input actions, the output actions and the internal actions. In practice the input actions of an automaton potentially aim to be the ouput action of another automaton and vice-versa. Hence an automaton can influence another one through a shared action. The comportment of the entire system is formalised by the automaton issued from the compostion of the automata of the system.

After this, we can speak about an execution of an automaton, which is an alternating sequence of states and actions. We can also speak about a trace of an automaton, which is the projection of an execution on the external actions uniquely. This allows us to speak about external behaviour of a system, that is, what can we observe from an outside point of view.

2.2 Scheduler

We remarked in the example of figure 2 that an inherent non-determinism has to be solved to be able to define a measure of probability on the executions. This is the role of the scheduler which is a function $\sigma : Frags^*(\mathcal{A}) \rightarrow SubDisc(D_{\mathcal{A}})$ that (consistently) maps an execution fragment to a discrete sub-probability distributions on set of discrete transitions of the concerned PSIOA $\mathcal{A}$. Loosely speaking, the scheduler σ decides (probabilistically) which transition to take after each finite execution fragment α. Since this decision is a discrete sub-probability measure, it may be the case that σ chooses to halt after α with non-zero probability: $1 - \sigma(\alpha)(D_{\mathcal{A}}) > 0$.

A scheduler σ generate a measure ϵ_σ on the sigma-field $\mathcal{F}_{Execs(\mathcal{A})}$ generated by cones of executions (of the form $C_{\alpha^x} = \{\alpha^x \frown \alpha^y | \alpha^y \in Frags(\mathcal{A})\}$), and so a measure on the measurable

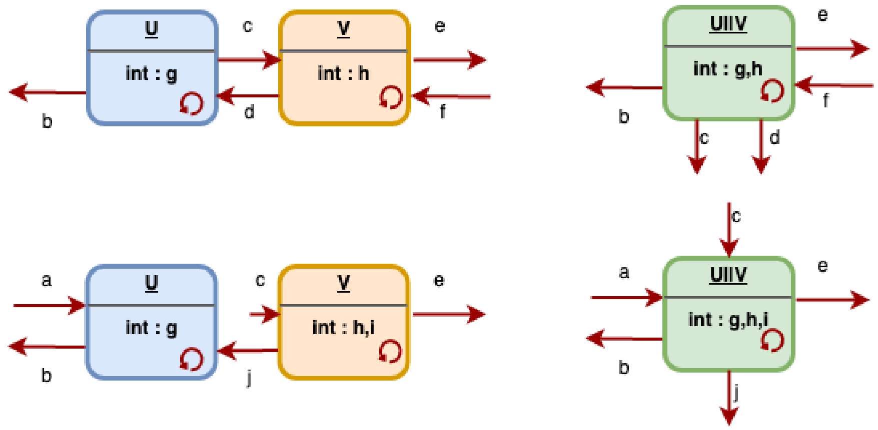


Figure 1 A representation of two automata U and V. In the top line, we see the PSIOA U in a state q_U^1, s.t. $sig(U)(q_U^1) = (out(U)(q_U^1), in(U)(q_U^1), int(U)(q_U^1)) = (\{b, c\}, \{d\}, \{g\})$, the PSIOA V in a state q_V^1, s.t. $sig(V)(q_V^1) = (out(V)(q_V^1), in(V)(q_V^1), int(V)(q_V^1)) = (\{d, e\}, \{c, f\}, \{h\})$ and the result of their composition, the PSIOA $U||V$ in a state (q_U^1, q_V^1), s.t. $sig(U||V)((q_U^1, q_V^1)) = (out(U||V)((q_U^1, q_V^1)), in(U||V)((q_U^1, q_V^1)), int(U||V)((q_U^1, q_V^1)) = (\{b, c, d, e\}, \{f\}, \{g, h\})$. In the second line we see the same PSIOA but in different states, with different signatures.

space $(G, \mathcal{F}_G)$ for any measurable function f from $(Execs(\mathcal{A}), \mathcal{F}_{Execs(\mathcal{A})})$ to $(G, \mathcal{F}_G)$. Hence, when a scheduler is made explicit, we can state the probability that a cone of execution is reached and that a property holds. We denote by $\epsilon_\sigma : Execs(\mathcal{A}) \rightarrow [0, 1]$ the execution distribution generated by the scheduler σ.

2.3 Environment, external behavior, implementation

Now it is possible to define the crucial concept of implementation that captures the idea that an automaton $\mathcal{A}$ "mimics" another automaton $\mathcal{B}$. To do so, we define an environment $\mathcal{E}$ which takes on the role of a "distinguisher" for $\mathcal{A}$ and $\mathcal{B}$. In general, an environment of an automaton $\mathcal{A}$ is just an automaton compatible with $\mathcal{A}$ but some additional minor technical properties can be assumed. The set of environments of the automaton $\mathcal{A}$ is denoted $env(\mathcal{A})$. The information used by an environment to attempt a distinction between two automata $\mathcal{A}$ and $\mathcal{B}$ s.t. $\mathcal{E} \in env(\mathcal{A}) \cap env(\mathcal{B})$ is captured by a function $f_{(.,.)}$ that we call *insight function*. In the literature, we very often deal with (i) $f_{(\mathcal{E},\mathcal{A})} = trace_{(\mathcal{E},\mathcal{A})}$ or (ii) $proj_{(\mathcal{E},\mathcal{A})} : \alpha \in Execs(\mathcal{E}||\mathcal{A}) \mapsto \alpha \upharpoonright \mathcal{E}$, the function that maps every execution to its projection on the environment. The philosophy of the two approaches are the same ones, but we proved monotonicity of external behaviour inclusion only for $proj_{(.,.)}$.

For any insight function $f_{(.,.)}$, we denote by $f\text{-}dist_{\mathcal{E},\mathcal{A}}(\sigma)$ the image measure of ϵ_σ under $f_{(\mathcal{E},\mathcal{A})}$. From here, this is classic to define the f-external behaviour of $\mathcal{A}$, denoted $ExtBeh^f_{\mathcal{A}} : \mathcal{E} \in env(\mathcal{A}) \mapsto \{f\text{-}dist_{\mathcal{A},\mathcal{E}}(\sigma) | \sigma \in schedulers(\mathcal{E}||\mathcal{A})\}$. Such an object capture all the possible measures of probability on the external interaction of the concerned automaton $\mathcal{A}$ and an arbitrary environment $\mathcal{E}$. Finally we can say that $\mathcal{A}$ f-implements $\mathcal{B}$ if $\forall \mathcal{E} \in env(\mathcal{A}) \cap env(\mathcal{B}), ExtBeh^f_{\mathcal{A}}(\mathcal{E}) \subseteq ExtBeh^f_{\mathcal{B}}(\mathcal{E})$, i.e. for any "distinguisher" $\mathcal{E}$ for $\mathcal{A}$ and $\mathcal{B}$, for any possible distribution $f\text{-}dist_{(\mathcal{E},\mathcal{A})}(\sigma)$ of the interaction between $\mathcal{E}$ and $\mathcal{A}$ generated by a scheduler $\sigma \in schedulers(\mathcal{E}||\mathcal{A})$, there exists a scheduler $\sigma' \in schedulers(\mathcal{E}||\mathcal{B})$ s.t. the distribution $f\text{-}dist_{(\mathcal{E},\mathcal{B})}(\sigma')$ of the interaction between $\mathcal{E}$ and $\mathcal{B}$ generated by σ' is the same, i.e. for every external perception $\zeta \in range(f_{(\mathcal{E},\mathcal{A})}) \cup range(f_{(\mathcal{E},\mathcal{B})})$, $f\text{-}dist_{(\mathcal{E},\mathcal{A})}(\sigma)(\zeta) = f$-

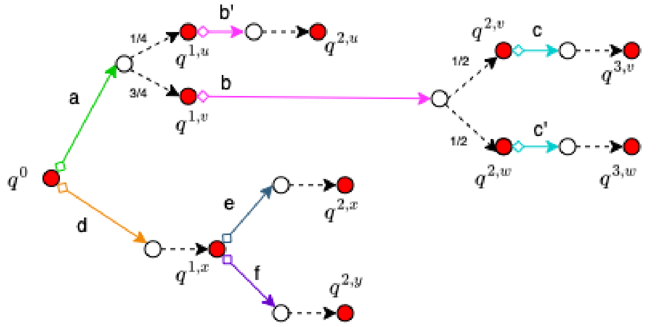


Figure 2 The figure represents a tree of possible executions for a PSIOA $\mathcal{A}$. The red dots $(q^0, q^{1,\cdot}, q^{2,\cdot}, q^{3,\cdot})$ represents some states of the PSIOA. The PSIOA can move from on state to another through actions $(a, b, c, d, e, f, ...)$ represented with colored solid arrows. Such an action act, triggered from a specif state q does not lead directly to another state q' but to a probabilistic distribution on states $\eta_{(\mathcal{A},q,act)}$ represented by a white dot and as many dashed black arrows as states in the support of $\eta_{(\mathcal{A},q,act)}$.For example, the PSIOA $\mathcal{A}$ can be in state q^0, trigger the action a that leads him to $\eta_{(\mathcal{A},q,a)}$ and hence to $q^{1,u}$ with probability $1/4$ and to $q^{1,v}$ with probability $3/4$. The sequence $q^0, a, q^{1,v}, b, q^{2,w}$ is an example of execution. If b is an internal actions, then a, c is an example of trace. A non-determinism is appearing since the choice of an action at a particular state is not determined a priori (e.g. between a and d at state q^0). This non-determinism will be solved by the *scheduler*, introduced later.

$dist_{(\mathcal{E},\mathcal{B})}(\sigma')(\zeta)$, noted $f\text{-}dist_{(\mathcal{E},\mathcal{A})}(\sigma) \equiv f\text{-}dist_{(\mathcal{E},\mathcal{B})}(\sigma')$. This a way to formalise that there is no way to distinguish $\mathcal{A}$ from $\mathcal{B}$. (see figure 3).

However, as already mentioned in [20], the correctness of an algorithm may be based on some specific assumptions on the scheduling policy that is used. Thus, in general, we are interested only in a subset of $schedulers(\mathcal{E}||\mathcal{A})$. A function that maps any automaton W to a subset of $schedulers(W)$ is called a *scheduler schema*. Among the most noteworthy examples are the fair schedulers, the off-line, a.k.a. oblivious schedulers, defined in opposition with the online-schedulers. So, we note $ExtBeh^{f,S}_{\mathcal{A}} : \mathcal{E} \in env(\mathcal{A}) \mapsto \{f\text{-}dist_{\mathcal{A},\mathcal{E}}(\sigma) | \sigma \in S(\mathcal{E}||\mathcal{A})\}$ where S is a scheduler schema and we say that $\mathcal{A}$ f-implements $\mathcal{B}$ according to a scheduler schema S if $\forall \mathcal{E} \in env(\mathcal{A}) \cap env(\mathcal{B}), ExtBeh^{f,S}_{\mathcal{A}}(\mathcal{E}) \subseteq ExtBeh^{f,S}_{\mathcal{B}}(\mathcal{E})$. In the remaining, we will have a great interest for two certain classes of oblivious schedulers, i.e. i) the creation-oblivious scheduler (introduced later) and ii) the task-scheduler: an off-line scheduler already introduced in [3], which is relevant for cryptographic analysis. The previous notions can be adapted with a particular class of scheduler schema.

2.4 Probabilistic Configuration Automata (PCA)

The section 5 introduces the notion of probabilistic configuration automata (PCA). (see figure 4). A PCA is very closed to a PSIOA, but each state is mapped to a *configuration* $C = (\mathbf{A}, \mathbf{S})$ which is a pair constituted by a set $\mathbf{A}$ of PSIOA and the current states of each member of the set (with a mapping function $\mathbf{S} : \mathcal{A} \in \mathbf{A} \mapsto q_{\mathcal{A}} \in Q_{\mathcal{A}}$. The idea is that the composition of the attached set can change during the execution of a PCA, which allows us to formalise the notion of dynamicity, that is the potential creation and potential destruction of a PSIOA in a dynamic system. Some particular precautions have to be taken to make it consistent.

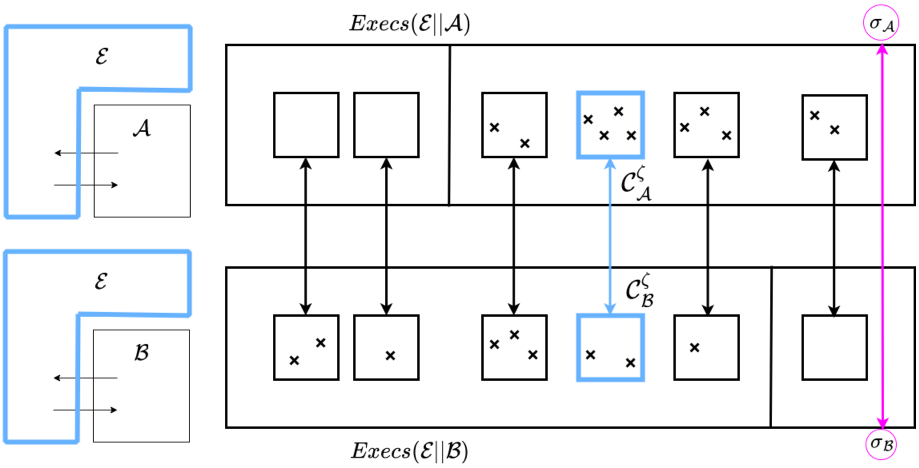


Figure 3 An environment $\mathcal{E}$, which is nothing more than a PSIOA compatible with both $\mathcal{A}$ and $\mathcal{B}$, tries to distinguish $\mathcal{A}$ from $\mathcal{B}$. We say that $\mathcal{A}$ implements $\mathcal{B}$ if no environment $\mathcal{E}$ is able to distinguish $\mathcal{A}$ from $\mathcal{B}$, that is $\forall \sigma \in schedulers(\mathcal{E}||\mathcal{A})\ \exists \sigma' \in schedulers(\mathcal{E}||\mathcal{B})$ (linked by pink arrow) s.t. every pair of corresponding classes of equivalence of executions, related to the same perception by the environment (e.g. $(C^{\zeta}_{\mathcal{A}}, C^{\zeta}_{\mathcal{B}})$ in blue for perception ζ) are equiprobable, i.e. $f\text{-}dist_{(\mathcal{E},\mathcal{A})}(\sigma)(\zeta) = f\text{-}dist_{(\mathcal{E},\mathcal{B})}(\sigma')(\zeta)$.

2.5 Road to monotonicity

The rest of the paper is dedicated to the proof of implementation monotonicity. We show that, under certain technical conditions, automaton creation is monotonic with respect to external behavior inclusion, i.e. if a system X creates automaton $\mathcal{A}$ instead of (previously) creating automaton $\mathcal{B}$ and the external behaviors of $\mathcal{A}$ are a subset of the external behaviors of $\mathcal{B}$, then the set of external behaviors of the overall system is possibly reduced, but not increased. Such an external behavior inclusion result enables a design and refinement methodology based solely on the notion of externally visible behavior, and which is therefore independent of specific methods of establishing external behavior inclusion. It permits the refinement of components and subsystems in isolation from the entire system. To do so, we develop different mathematical tools.

2.5.1 Execution-matching

First, we define in section 10, the notion of executions-matching (see figure 5) to capture the idea that two automata have the same "comportment" along some corresponding executions. Basically an execution-matching from a PSIOA $\mathcal{A}$ to a PSIOA $\mathcal{B}$ is a morphism f^{ex} : $Execs'_{\mathcal{A}} \rightarrow Execs(\mathcal{B})$ where $Execs'_{\mathcal{A}} \subseteq Execs(\mathcal{A})$. This morphism preserves some properties along the pair of matched executions: signature, transition, ... in such a way that for every pair $(\alpha, \alpha') \in Execs(\mathcal{A}) \times Execs(\mathcal{B})$ s.t. $\alpha' = f^{ex}(\alpha)$, $\epsilon_{\sigma}(\alpha) = \epsilon_{\sigma'}(\alpha')$ for every pair of scheduler (σ, σ') (so-called *alter ego*) that are "very similar" in the sense they take into account only the "structure" of the argument to return a sub-probability distribution, i.e. $\alpha' = f^{ex}(\alpha)$ implies $\sigma(\alpha) = \sigma'(\alpha')$. When the executions-matching is a bijection function from $Execs(\mathcal{A})$ to $Execs(\mathcal{B})$, we say $\mathcal{A}$ and $\mathcal{B}$ are semantically-equivalent (they differ only syntactically).

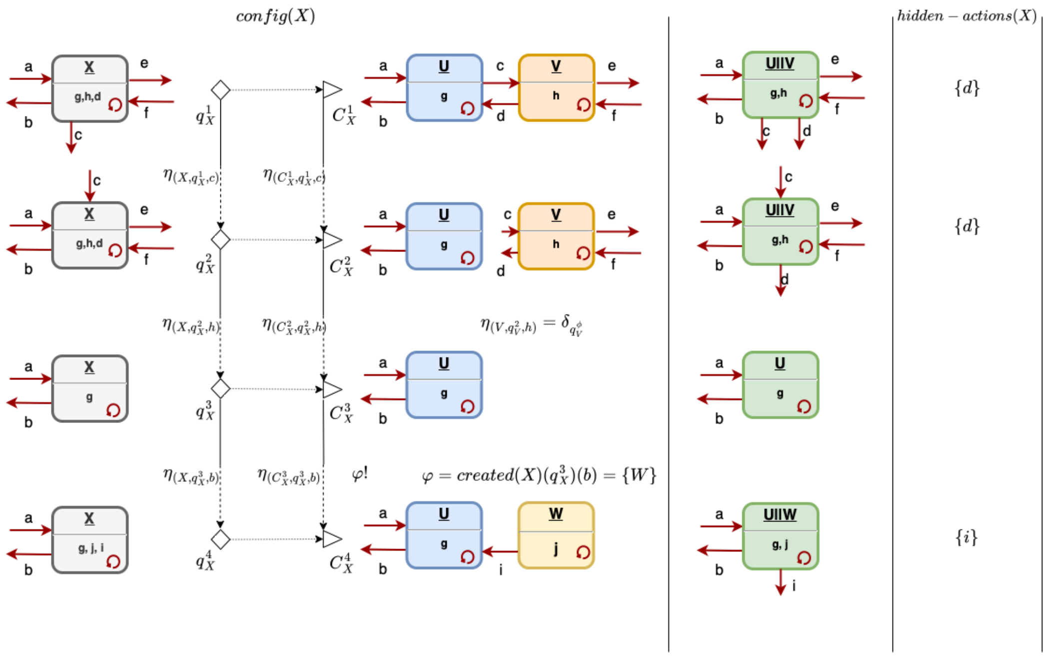


Figure 4 The figure represents an execution fragment $(q_X^1, c, q_X^2, h, q_X^3, b, q_X^4)$ of a PCA X. In the left column, we see different states q_X^1, q_X^2, q_X^3 and q_X^4 of the PCA X, represented with white diamonds ($\diamond$). Each of these states q_X^i is mapped through the mapping $config(X)$ (represented with right dotted arrows) to a *configuration* C_X^i, represented with a white triangle ($\triangleright$). For example the state q_X^1 is mapped with the configuration $C_X^1 = (\mathbf{A}^1, \mathbf{S}^1)$ with $\mathbf{A}^1 = \{U, V\}$, $\mathbf{S}^1(U) = q_U^1$ and $\mathbf{S}^1(V) = q_V^1$. The signature of the PCA X at state q_X^i is the one of the composition of automata, in their current states in the attached configuration C_X^i, modulo some external actions $hidden\text{-}actions(X)(q_X^i)$ for C_X^i that are hidden and become internal for X. For example, the configuration C_X^1 has a signature $sig(C_X^1) = (out(C_X^1), in(C_X^1), int(C_X^1)) = (\{b, e, c, d\}, \{a, f\}, \{g, h\})$, while the signature of X at corresponding state is $sig(X)(q_X^1) = (out(X)(q_X^1), in(X)(q_X^1), int(X)(C_X^1)) = (\{b, e, c\}, \{a, f\}, \{g, h, d\})$ since the unique action $d \in hidden\text{-}actions(X)(q_X^1)$ is hidden and hence becomes an internal action. We can define discrete transitions for configurations in a similar way as what we do for PSIOA, but adding some tools (formally defined in section 5) to allow the creation and the destruction of automata. For example, the automaton V is destroyed during the step (q_X^2, h, q_X^3), while W is created during the step (q_X^3, b, q_X^4) which is made explicit by the fact that $created(X)(q_X^3)(b) = \{X\}$ where $created(X)$ is a mapping function defined for any PCA X. Some intuitive consistency rules have to be respected by pair of "corresponding transitions" $((q_X^i, act, \eta_{(X,q_X^i,act)}); (C_X^i, act, \eta_{(C_X^i,q_X^i,act)}))$ represented by pair of parallel downward arrows (one between two diamonds $\diamond$ and one between two triangles $\triangleright$) . For example, the probability $\eta_{(X,q_X^1,c)}(q_X^2)$ of reaching q_X^2 by triggering c from q_X^1 is equal to the probability $\eta_{(C_X^1,q_X^1,act)}(C_X^2)$ of reaching C_X^2 by triggering c from C_X^1. Moreover, a configuration transition has to respect some of other consistency rules with respect to the sub-automata that compose the configuration. Typically, the destruction of V in step (C_X^2, h, C_X^3) comes from the fact that the triggering the action h from state q_V^2 of sub-automaton V leads to a probabilistic states distribution $\eta_{(V,q_V^2,h)}$ equal to $\delta_{q_V^\phi}$ which is a Dirac distribution for a special state q_V^ϕ with $sig(V)(q_V^\phi) = (\emptyset, \emptyset, \emptyset)$ that means V "has been destroyed".

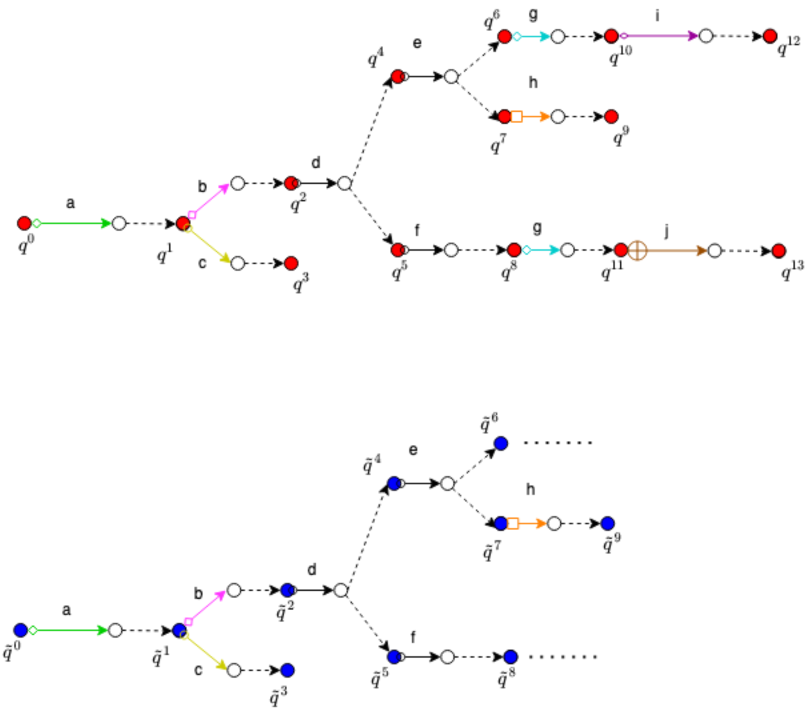


Figure 5 The figure represents the respective executions tree of two automata $\mathcal{A}$ and $\mathcal{B}$ with some strong similarities. The states of $\mathcal{A}$ (resp. $\mathcal{B}$) are represented with red (resp. blue) dots. The actions are represented with solid arrows. An action leads to a discrete probability distribution on states η, represented with a white dot and dashed arrows reaching the different states of the support of η. In section 10, we define these strong similarities with what we call an executions-matching (f, f^{tr}, f^{ex}) where $f : Q'_{\mathcal{A}} \rightarrow Q_{\mathcal{B}}$, $f^{tr} : D'_{\mathcal{A}} \rightarrow D_{\mathcal{B}}$, $f^{ex} : Execs'_{\mathcal{A}} \rightarrow Execs(\mathcal{B})$ with $Q'_{\mathcal{A}} \subseteq Q_{\mathcal{A}}$, $D'_{\mathcal{A}} \subseteq D_{\mathcal{A}}$, $Execs'_{\mathcal{A}} \subseteq Execs(\mathcal{A})$. The mappings f, f^{tr} and f^{ex} preserves the important properties: signature for corresponding states, name of the action and measure of probability of corresponding states for corresponding transitions, etc. In the example the similarities exist until the states q^6, q^8 and q^9, hence we have $Q'_{\mathcal{A}} = \{q^0, q^1, ..., q^9\} \subsetneq Q_{\mathcal{A}}$. The *states-matching* f is then defined s.t. $\forall k \in [1, 9], f(q^k) = \tilde{q}^k$. Thereafter, we define define $Act = \{a, b, c, d, e, f, h\}$ and f^{trans}, s.t. $\forall k \in [1, 9]$, $\forall act \in Act$, for every transition $(q^k, act, \eta_{(\mathcal{A}, q^k, act)})$, $f^{trans}((q^k, act, \eta_{(\mathcal{A}, q^k, act)}))$ $= (\tilde{q}^k, act, \eta_{(\mathcal{B}, q^k, act)})$. Each pair of mapped transition gives the same probability to pair of mapped states, e.g. $\eta_{(\mathcal{A}, q^2, d)}(q^4) = \eta_{(\mathcal{B}, \tilde{q}^2, d)}(\tilde{q}^4)$. Then we can define $Execs'_{\mathcal{A}} \subset Execs(\mathcal{A})$ the set of executions composed only with states in $Q'_{\mathcal{A}}$ and actions in Act. Finally $f^{ex} : \alpha = q^0 a^1 ... a^n q^n \in Execs'_{\mathcal{A}} \mapsto f(q^0) a^1 ... a^n f(q^n)$ is an execution-matching. The point is that if two schedulers σ and σ' only look at the preserved properties to output a measure of probability on the actions to take, the attached measures of probability will be equal, i.e. $\epsilon_{\sigma}(\alpha) = \epsilon_{\sigma'}(\alpha')$

2.5.2 A PCA $X_{\mathcal{A}}$ deprived from a PSIOA $\mathcal{A}$

Second, we define in section 11 the notion of a PCA $X_{\mathcal{A}}$ deprived from a PSIOA $\mathcal{A}$ noted $(X_{\mathcal{A}} \setminus \{\mathcal{A}\})$. Such an automaton corresponds to the intuition of a similar automaton where $\mathcal{A}$ is systematically removed from the configuration of the original PCA (see figure 6a and 6b).

2.5.3 Reconstruction: $(X_{\mathcal{A}} \setminus \{\mathcal{A}\})||\tilde{\mathcal{A}}^{sw}$

Thereafter we show in section 12 that under technical minor assumptions $X_{\mathcal{A}} \setminus \{\mathcal{A}\}$ and $\tilde{\mathcal{A}}^{sw}$ are composable where $\tilde{\mathcal{A}}^{sw}$ and $\mathcal{A}$ are semantically equivalent in the sense loosely introduced in the section 2.5.1 . In fact $\tilde{\mathcal{A}}^{sw}$ is the simpleton wrapper of $\mathcal{A}$, that is a PCA that only owns $\mathcal{A}$ in its attached configuration (see figure 7). Let us note that if $\mathcal{A}$ implements $\mathcal{B}$, then $\tilde{\mathcal{A}}^{sw}$ implements $\tilde{\mathcal{B}}^{sw}$.

Then we show that there is an (incomplete) execution-matching from $X_{\mathcal{A}}$ to $(X_{\mathcal{A}} \setminus \{\mathcal{A}\})||\tilde{\mathcal{A}}^{sw}$ (see figure 8). The domain of this executions-matching is the set of executions where $\mathcal{A}$ is not (re-)created.

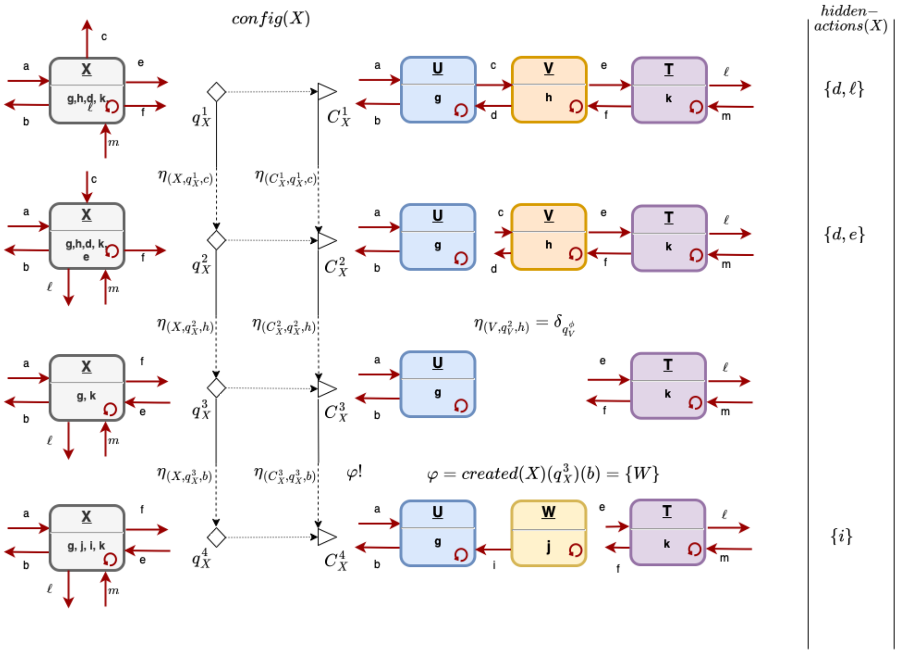
After this, we always try to reduce any reasoning on $X_{\mathcal{A}}$ (resp. $X_{\mathcal{B}}$) on a reasoning on $(X_{\mathcal{A}} \setminus \{\mathcal{A}\})||\tilde{\mathcal{A}}^{sw}$ (resp. $(X_{\mathcal{B}} \setminus \{\mathcal{B}\})||\tilde{\mathcal{B}}^{sw}$).

2.5.4 Corresponding PCA

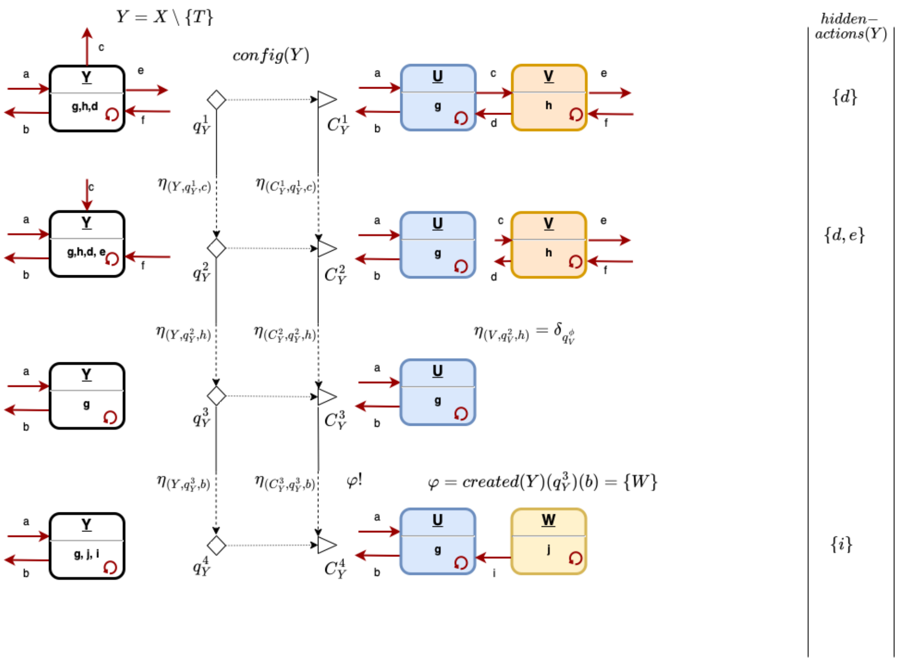
We show in section 13 that, under certain reasonable technical assumptions (captured in the definition of corresponding PCA w.r.t. $\mathcal{A}$, $\mathcal{B}$), $(X_{\mathcal{A}} \setminus \{\mathcal{A}\})$ and $(X_{\mathcal{B}} \setminus \{\mathcal{B}\})$ are semantically-equivalent. We can note Y an arbitrary PCA semantically-equivalent to $(X_{\mathcal{A}} \setminus \{\mathcal{A}\})$ and $(X_{\mathcal{B}} \setminus \{\mathcal{B}\})$. Finally, a reasoning on $\mathcal{E}||X_{\mathcal{A}}$ (resp. $\mathcal{E}||X_{\mathcal{B}}$) can be reduced to a reasoning on $\mathcal{E}'||\tilde{\mathcal{A}}^{sw}$ (resp. $\mathcal{E}'||\tilde{\mathcal{B}}^{sw}$) with $\mathcal{E}' = \mathcal{E}||Y$. Since $\tilde{\mathcal{A}}^{sw}$ implements $\tilde{\mathcal{B}}^{sw}$, we have already some results on $\mathcal{E}'||\tilde{\mathcal{A}}^{sw}$ and $\mathcal{E}'||\tilde{\mathcal{B}}^{sw}$ and so on $\mathcal{E}||X_{\mathcal{A}}$ and $\mathcal{E}||X_{\mathcal{B}}$. However, these results are a priori valid only for the subset of executions without creation of neither $\mathcal{A}$ nor $\mathcal{B}$ before very last action). This reduction is represented in figures 9a and 9b.

2.5.5 Cut-paste execution fragments creation at the endpoints

The reduction roughly described in figures 9a and 9b holds only for executions fragments that do not create the automata $\mathcal{A}$ and $\mathcal{B}$ after their destruction (or at very last action). Some technical precautions have to be taken to be allowed to paste these fragments together to finally say that $\mathcal{A}$ implements $\mathcal{B}$ implies $X_{\mathcal{A}}$ implements $X_{\mathcal{B}}$. In fact, such a pasting is generally not possible for a fully information online scheduler. This observation motivated us to introduce the *creation-oblivious scheduler* that outputs (randomly) a transition without taking into account the internal actions and internal states of a sub-automaton $\mathcal{A}$ preceding its last destruction. We prove monotonicity of external behaviour inclusion for schema of creation oblivious scheduler in section 14. Surprisingly, the fully-offline task-scheduler introduced in [3] (slightly modified to be adapted to dynamic setting) is not creation-oblivious (see section 15) and so does not allow monotonicity of external behaviour inclusion. The figure 10 represents the issue with non-creation-oblivious scheduler.



(a) Projection on PCA, part 1/2: The figure represents a PCA X like in figure 4. A sub-automaton T (in purple) appears in the configurations attached to the states visited by X. The PCA $Y = X \setminus \{T\}$ where the sub-automaton T is systematically removed is represented in figure 6b.



(b) Projection on PCA, part 2/2: the figure represents the PCA $Y = X \setminus \{T\}$ while the original PCA X is represented in figure 6a. We can see that the sub-automaton T (in purple in figure 6a) has been systematically removed from the configurations attached to the states visited by Y.

Figure 6 PCA deprived of a sub-PSIOA

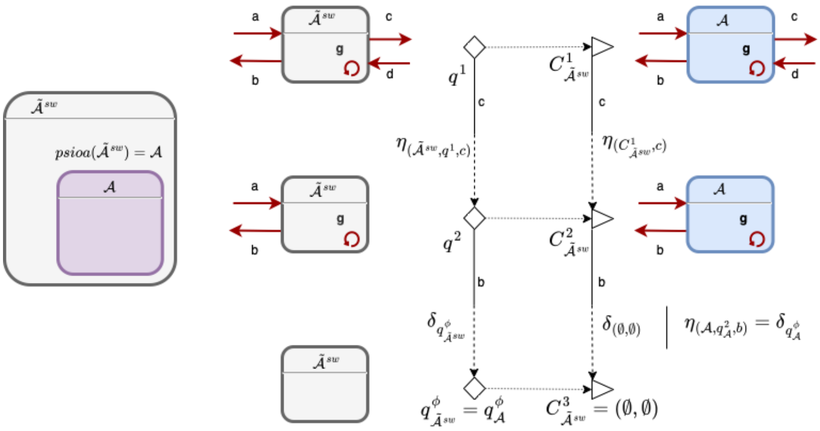


Figure 7 The figure represents the simpleton wrapper $\tilde{A}^{sw}$ of an automaton $\mathcal{A}$. The automaton $\tilde{A}^{sw}$ is a PCA that only encapsulates one unique sub-automaton which is $\mathcal{A}$. We can confuse $\mathcal{A}$ and $\tilde{A}^{sw}$ without impact. Intuitively, we can see $\tilde{A}^{sw}$ as a wrapper of $\mathcal{A}$ that does not provide anything.

3 Preliminaries on probability and measure

We assume our reader is comfortable with basic notions of probability theory, such as σ-algebra and (discrete) probability measures. A measurable space is denoted by $(S, \mathcal{F}_S)$, where S is a set and $\mathcal{F}_S$ is a σ-algebra over S that is $\mathcal{F}_S \subseteq \mathcal{P}(S)$, is closed under countable union and complementation and its members are called measurable sets ($\mathcal{P}(S)$ denotes the power set of S). The union of a collection $\{S_i\}_{i \in I}$ of pairwise disjoint sets indexed by a set I is written as $\biguplus_{i \in I} S_i$. A measure over $(S, \mathcal{F}_S)$ is a function $\eta : \mathcal{F}_S \to \mathbb{R}^{\geq 0}$, such that $\eta(\emptyset) = 0$ and for every countable collection of disjoint sets $\{S_i\}_{i \in I}$ in $\mathcal{F}_S$, $\eta(\biguplus_{i \in I} S_i) = \Sigma_{i \in I} \eta(S_i)$. A probability measure (resp. sub-probability measure) over $(S, \mathcal{F}_S)$ is a measure η such that $\eta(S) = 1$ (resp. $\eta(S) \leq 1$). A measure space is denoted by $(S, \mathcal{F}_S, \eta)$ where η is a measure on $(S, \mathcal{F}_S)$.

The product measure space $(S_1, \mathcal{F}_{s_1}, \eta_1) \otimes (S_2, \mathcal{F}_{s_2}, \eta_2)$ is the measure space $(S_1 \times S_2, \mathcal{F}_{s_1} \otimes \mathcal{F}_{s_2}, \eta_1 \otimes \eta_2)$, where $\mathcal{F}_{s_1} \otimes \mathcal{F}_{s_2}$ is the smallest σ-algebra generated by sets of the form $\{A \times B | A \in \mathcal{F}_{s_1}, B \in \mathcal{F}_{s_2}\}$ and $\eta_1 \otimes \eta_2$ is the unique measure s.t. for every $C_1 \in \mathcal{F}_{s_1}, C_2 \in \mathcal{F}_{s_2}$, $\eta_1 \otimes \eta_2(C_1 \times C_2) = \eta_1(C_1) \cdot \eta_2(C_2)$. If S is countable, we note $\mathcal{P}(S) = 2^S$. If S_1 and S_2 are countable, we have $2^{S_1} \otimes 2^{S_2} = 2^{S_1 \times S_2}$.

A discrete probability measure on a set S is a probability measure η on $(S, 2^S)$, such that, for each $C \subset S, \eta(C) = \sum_{c \in C} \eta(\{c\})$. We define $Disc(S)$ and $SubDisc(S)$ to be respectively, the set of discrete probability and sub-probability measures on S. In the sequel, we often omit the set notation when we denote the measure of a singleton set. For a discrete probability measure η on a set S, $supp(\eta)$ denotes the support of η, that is, the set of elements $s \in S$ such that $\eta(s) \neq 0$. Given set S and a subset $C \subset S$, the Dirac measure δ_C is the discrete probability measure on S that assigns probability 1 to C. For each element $s \in S$, we note δ_s for $\delta_{\{s\}}$.

If $\{m_i\}_{i \in I}$ is a countable family of measures on $(S, \mathcal{F}_S)$, and $\{p_i\}_{i \in I}$ is a family of non-negative values, then the expression $\sum_{i \in I} p_i m_i$ denotes a measure m on $(S, \mathcal{F}_S)$ such that, for each $C \in \mathcal{F}_S, m(C) = \sum_{i \in I} m_i f_i(C)$. A function $f : X \to Y$ is said to be measurable from $(X, \mathcal{F}_X) \to (Y, \mathcal{F}_Y)$ if the inverse image of each element of $\mathcal{F}_Y$ is an element of $\mathcal{F}_X$, that is, for each $C \in \mathcal{F}_Y$, $f^{-1}(C) \in \mathcal{F}_X$. In such a case, given a measure η on $(X, \mathcal{F}_X)$,

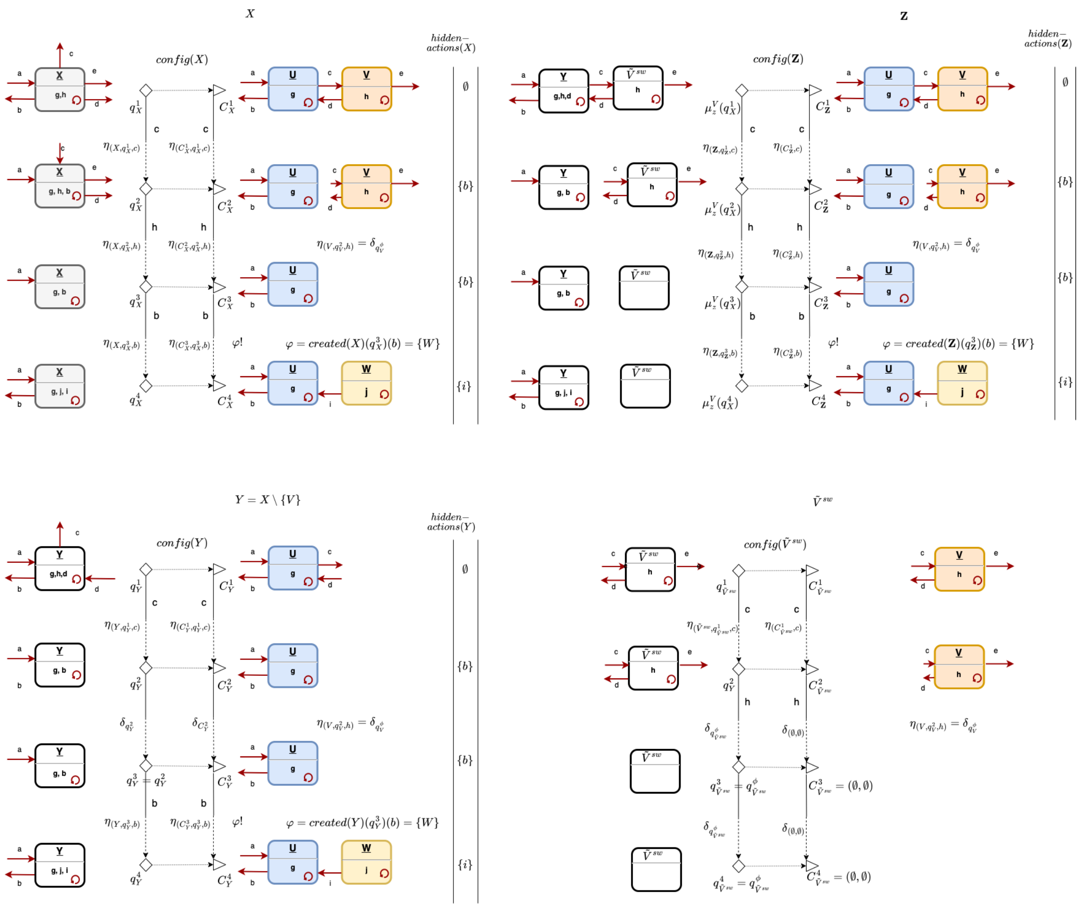
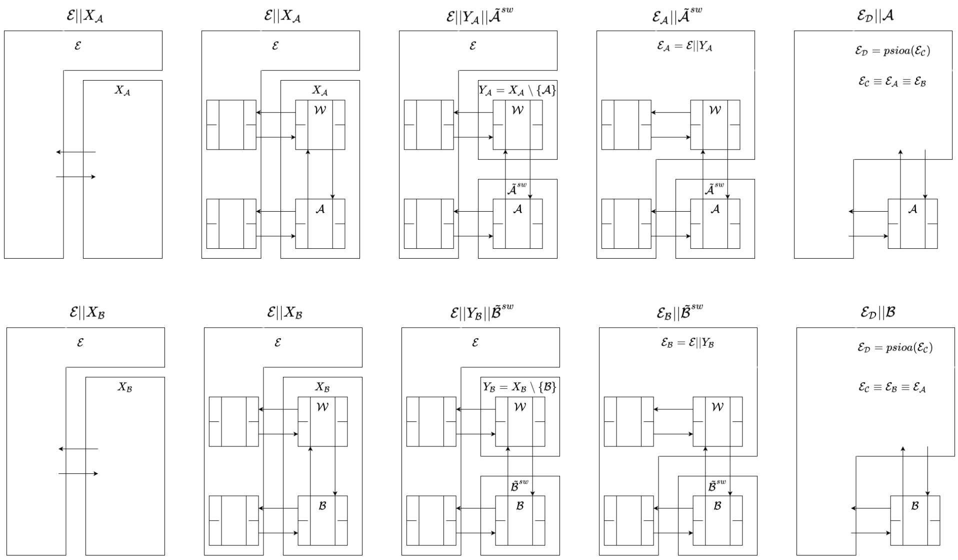
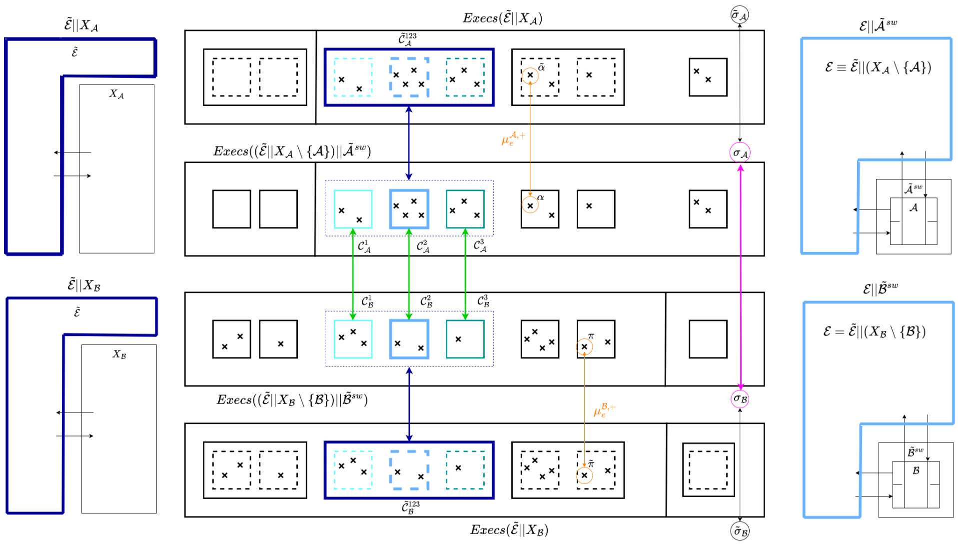


Figure 8 The figure shows the similarities between two PCA X and $Z = (X \setminus \{V\})||\tilde{V}^{sw}$ represented in the top line. The two components of Z, i.e. $(X \setminus \{V\})$ and $\tilde{V}^{sw}$ are represented in the bottom line like in figure 6b and 7. These similarities are captured by the notions of executions-matching and hold as long as the the sub-automaton V is not created by X after a destruction. The idea is to reduce any reasoning on X to a reasoning on $(X \setminus \{V\})||\tilde{V}^{sw}$.



(a) The figure represents successive steps to reduce the problem of an environment $\mathcal{E}$ that tries to distinguish two PCA $X_\mathcal{A}$ and $X_\mathcal{B}$ (represented at first column) to a problem of an environment $\mathcal{E}_\mathcal{D}$ that tries to distinguish the automata $\mathcal{A}$ and $\mathcal{B}$ (represented at last column).



(b) The figure represents the homomorphism enabling the reduction reasoning, for set of executions that do not create neither $\mathcal{A}$ nor $\mathcal{B}$ before last action. For every environment $\mathcal{E}$, For every scheduler $\sigma_\mathcal{A}$, there exists a corresponding scheduler $\sigma_\mathcal{B}$ (mapped with pink arrow) s.t. for every possible perception ζ (represented in light blue), the probability to observe ζ is the same for $\mathcal{E}$ in each world. There is an homomorphism $\mu_e^{\mathcal{A},+}$ (orange arrow) between $\tilde{\mathcal{E}}||X_\mathcal{A}$ and $\mathcal{E}||\tilde{\mathcal{A}}^{sw}$ (and similarly for $X_\mathcal{B}$ and $\tilde{\mathcal{B}}^{sw}$) s.t. for every scheduler $\tilde{\sigma}_\mathcal{A}$, alter-ego of $\sigma_\mathcal{A}$, the measure of each corresponding perception is preserved. Hence, for every environment $\tilde{\mathcal{E}}$, for every scheduler $\tilde{\sigma}_\mathcal{A}$, there exists a corresponding scheduler $\tilde{\sigma}_\mathcal{B}$ s.t. for every possible perception $\tilde{\zeta}$ (represented in dark blue), the probability to observe $\tilde{\zeta}$ is the same for $\tilde{\mathcal{E}}$ in each world.

Figure 9 homomorphism-based-proof

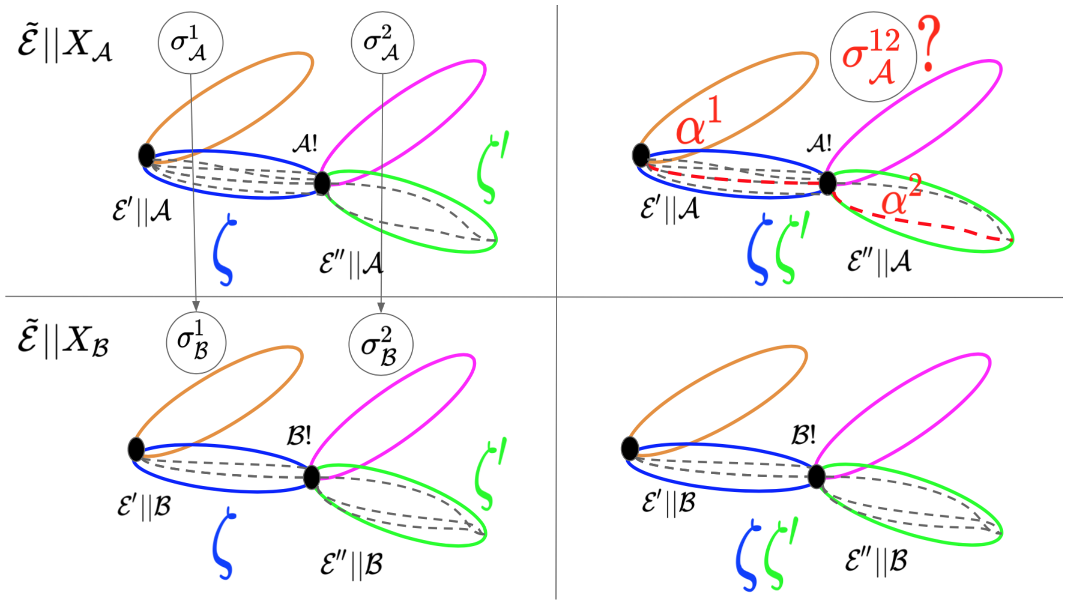


Figure 10 Necessity of creation oblivious scheduler. The reduction described before holds only for set of executions that do not create neither $\mathcal{A}$ nor $\mathcal{B}$ before last action (represented on the left). What if the scheduler $\sigma^{12}_{\mathcal{A}}$ break independence of probabilities between executing α^1 and executing α^2 after α^1 ? In that case, we cannot cut-paste the different reductions and the monotonicity of implementation does not hold, i.e. there is no reason there exists a scheduler counterpart $\sigma^{12}_{\mathcal{B}}$ s.t. that observing $\zeta^\frown\zeta'$ (represented in blue and green) has the same probability to occur in $\mathcal{A}$-world and in $\mathcal{B}$-world.

the function $f(\eta)$ defined on $\mathcal{F}_Y$ by $f(\eta)(C) = \eta(f^{-1}(C))$ for each $C \in Y$ is a measure on $(Y, \mathcal{F}_Y)$ and is called the image measure of η under f.

Let $(Q_1, 2^{Q_1})$ and $(Q_2, 2^{Q_2})$ be two measurable sets. Let $(\eta_2, \eta_2) \in Disc(Q_1) \times Disc(Q_2)$. Let $f : Q_1 \rightarrow Q_2$. We note $\eta_1 \overset{f}{\leftrightarrow} \eta_2$ if the following is verified: (1) the restriction $\tilde{f}$ of f to $supp(\eta_1)$ is a bijection from $supp(\eta_1)$ to $supp(\eta_2)$ and (2) $\forall q \in supp(\eta)$, $\eta(q_1) = \eta_2(f(q_1))$.

4 Probabilistic Signature Input/Output Automata (PSIOA)

This section aims to introduce the first brick of our formalism: the probabilistic signature input/output automata (PSIOA).

4.1 Background

Here, we quickly survey the literature on I/O automata that led to PSIOA. We first present the very well known Labeled Transition Systems (LTS). Then we briefly discuss the new features brought by I/O Automata, probabilistic I/O Automata and signature I/O Automata.

4.1.1 Labeled Transition System (LTS)

Roberto Segala describes LTS as follows ([20], section 3.2, p. 37): "A Labeled Transition System is a state machine with labeled transitions. The labels, also called *actions*, are used to model communication between a system and its external environment." A possible definition of an LTS, using notation of [13], is $\mathcal{A} = (Q_{\mathcal{A}}, \bar{q}_{\mathcal{A}}, \check{sig}(\mathcal{A}), steps(\mathcal{A}))$ where $Q_{\mathcal{A}}$ represents the states of $\mathcal{A}$, $\bar{q}_{\mathcal{A}}$ represents the start state of $\mathcal{A}$, $\check{sig}(\mathcal{A}) = (\check{ext}(\mathcal{A}), \check{int}(\mathcal{A}))$ represents the *signature* of $\mathcal{A}$, i.e. the set of actions that can be triggered, that are partitioned into external and internal actions, and $steps(\mathcal{A}) \subseteq Q_{\mathcal{A}} \times acts(\mathcal{A}) \times Q_{\mathcal{A}}$ represent the possible transition of the transition with $acts(\mathcal{A}) = \check{ext}(\mathcal{A}) \cup \check{int}(\mathcal{A})$. We can note $enabled(\mathcal{A}) : q \in Q_{\mathcal{A}} \mapsto \{a \in acts(\mathcal{A}) | \exists (q, a, q') \in steps(\mathcal{A})\}$ to model the actions enabled at a certain state. "The external actions model communication with the external environment; the internal actions model internal communication, not visible from the external environment."

It is possible to make several LTS communicate with each others through shared external actions in CSP [8] style. Typically, if $\mathcal{A}$ and $\mathcal{B}$ are two LTS s.t. the compatibility condition $acts(\mathcal{A}) \cap \check{int}(\mathcal{B}) = acts(\mathcal{B}) \cap \check{int}(\mathcal{A}) = \emptyset$ is verified, we can define their composition, $\mathcal{A}||\mathcal{B}$ with

- $Q_{\mathcal{A}||\mathcal{B}} = Q_{\mathcal{A}} \times Q_{\mathcal{B}}$,
- $\bar{q}_{\mathcal{A}||\mathcal{B}} = (\bar{q}_{\mathcal{A}}, \bar{q}_{\mathcal{B}})$,
- $\check{sig}(\mathcal{A}||\mathcal{B}) = (\check{ext}(\mathcal{A}) \cup \check{ext}(\mathcal{B}), \check{int}(\mathcal{A}) \cup \check{int}(\mathcal{B}))$,
- $steps(\mathcal{A}||\mathcal{B}) = \{((q_{\mathcal{A}}, q_{\mathcal{B}}), a, (q'_{\mathcal{A}}, q_{\mathcal{B}})') \in Q_{\mathcal{A}||\mathcal{B}} \times \times acts(\mathcal{A}||\mathcal{B}) Q_{\mathcal{A}||\mathcal{B}} | a \in enabled(\mathcal{A}) \cup enabled(\mathcal{B}) \wedge \forall \mathcal{K} \in \{\mathcal{A}, \mathcal{B}\}, (q_{\mathcal{K}}, a, q'_{\mathcal{K}}) \notin steps(\mathcal{K}) \implies (a \notin enabled(\mathcal{K}) \wedge q'_{\mathcal{K}} = q_{\mathcal{K}})\})$.

An *execution* of an LTS $\mathcal{A}$ is an alternating sequence of states and actions $q^0 a^1 q^1 a^2 ...$ such that each $(q^{i-1}, a^i, q^i) \in steps(\mathcal{A})$. A *trace* is the restriction to external actions of an execution. A LTS $\mathcal{A}$ implements another LTS $\mathcal{B}$ if $Traces(\mathcal{A}) \subseteq Traces(\mathcal{B})$, where $Traces(\mathcal{K})$ represents the set of traces of $\mathcal{K}$.

4.1.2 I/O Automata

The input output Automata (IOA) [12] are LTS with the following additional points:

- (I/O partitioning) There is a partition $(\check{in}(\mathcal{A}), \check{out}(\mathcal{A}))$ of $\check{ext}(\mathcal{A})$ where $\check{in}(\mathcal{A})$ denotes the *input* actions and $\check{out}(\mathcal{A})$ denotes the *output* actions. Moreover, $\check{loc}(\mathcal{A})$ denotes the *local* actions.
- (Output compatibility) The compatibility condition requires $out(\mathcal{A}) \cap out(\mathcal{B}) = \emptyset$ in addition.
- (I/O composition) After composition, we have in addition $out(\mathcal{A}||\mathcal{B}) = out(\mathcal{A}) \cup out(\mathcal{B})$ and $in(\mathcal{A}||\mathcal{B}) = in(\mathcal{A}) \cup in(\mathcal{B}) \setminus out(\mathcal{A}||\mathcal{B})$
- (Input enabling) $\forall q \in Q_{\mathcal{A}}, in(\mathcal{A}) \subseteq enabled(\mathcal{A})(q)$

The interests of this additional restrictions for formal verification are subtle (e.g. input enabling can avoid trivial liveness property implementation, locality allows simple definitions of fairness and oblivious scheduler, I/O partitioning allows intuitive definition of forwarding, ...). However, they do not add complexity in the analysis of this paper. Typically, they are never required in the key results of this paper. Adapting this paper to LTS is straightforward. We have kept I/O automata to be as close as possible from [2] and [3].

4.1.3 PIOA

The probabilistic input output automata (PIOA) [20] are kind of I/O automata where transitions are randomized, i.e. triggering an action leads to a probability measure on states instead to a particular state. The transitions are then elements of $D_{\mathcal{A}} \subseteq Q_{\mathcal{A}} \times acts(\mathcal{A}) \times Disc(Q_{\mathcal{A}})$. Now, the set of steps is $steps(\mathcal{A}) = \{(q, a, q') | \exists (q, a, \eta) \in D_{\mathcal{A}} \wedge q' \in supp(\eta)\}$. To define a measure of probability on the set of executions, it is convenient to call on a scheduler σ that will resolve the non-determinism and enable the construction of a measure of probability ϵ_σ on executions. The notion of implementation has to be adapted to probabilistic setting to be relevant.

4.1.4 SIOA

The signature I/O automata (SIOA) [2] are kind of I/O automata where the signature is evolving during the time. This feature is particularly convenient to model dynamicity. The signature of the automaton $\mathcal{A}$ becomes a function mapping each state q to a signature $sig(\mathcal{A})(q)$.

4.1.5 PSIOA

A PSIOA is the result of the generalization of probabilistic input/output automata (PIOA) [20] and signature input/output automata (SIOA) [2]. A PSIOA is thus an automaton that can randomly move from one *state* to another in response to some *actions*. The set of possible actions is the *signature* of the automaton and is partitioned into *input*, *output* and *internal* actions. An action can often be both the input of one automaton and the output of another one to captures the idea that the behavior of an automaton can influence the behavior of another one. As for the SIOA [2], the signature of a PSIOA can change according to the current state of the automaton, which allows us to formalise dynamicity later. The figure 11 gives a first intuition of what is a PSIOA.

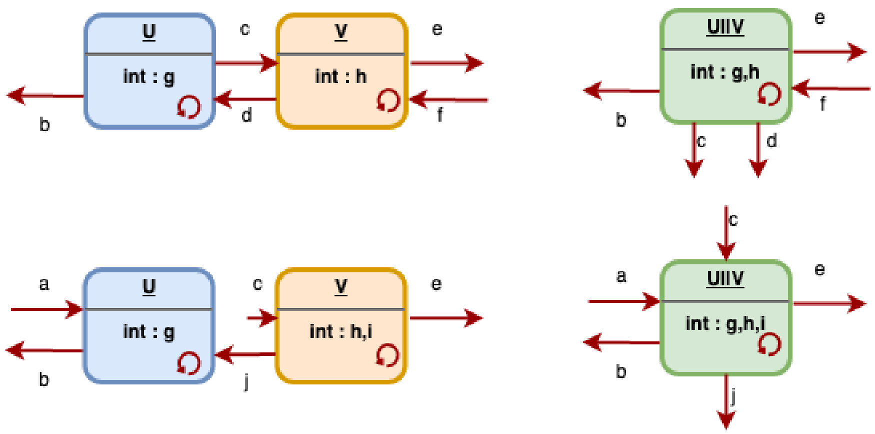


Figure 11 A representation of two automata U and V. In the top line, we see the PSIOA U in a state q^1_U, s.t. $sig(U)(q^1_U) = (out(U)(q^1_U), in(U)(q^1_U), int(U)(q^1_U)) = (\{b, c\}, \{d\}, \{g\})$, the PSIOA V in a state q^1_V, s.t. $sig(V)(q^1_V) = (out(V)(q^1_V), in(V)(q^1_V), int(V)(q^1_V)) = (\{d, e\}, \{c, f\}, \{h\})$ and the result of their composition, the PSIOA $U||V$ in a state (q^1_U, q^1_V), s.t. $sig(U||V)((q^1_U, q^1_V)) = (out(U||V)((q^1_U, q^1_V)), in(U||V)((q^1_U, q^1_V)), int(U||V)((q^1_U, q^1_V) = (\{b, c, d, e\}, \{f\}, \{g, h\})$. In the second line we see the same PSIOA but in different states. We see the PSIOA U in a state q^2_U, s.t. $sig(U)(q^2_U) = (out(U)(q^2_U), in(U)(q^2_U), int(U)(q^2_U)) = (\{b\}, \{a, j\}, \{g\})$, the PSIOA V in a state q^2_V, s.t. $sig(V)(q^2_V) = (out(V)(q^2_V), in(V)(q^2_V), int(V)(q^2_V)) = (\{e, j\}, \{c\}, \{h, i\})$ and the result of their composition, the PSIOA $U||V$ in a state (q^2_U, q^2_V), s.t. $sig(U||V)((q^2_U, q^2_V)) = (out(U||V)((q^2_U, q^2_V)), in(U||V)((q^2_U, q^2_V)), int(U||V)((q^2_U, q^2_V) = (\{b, e, j\}, \{a, c\}, \{g, h, i\})$.

4.2 Action Signature

We use the signature approach from [2]. We assume the existence of a countable set *Autids* of unique probabilistic signature input/output automata (PSIOA) identifiers, an underlying universal set *Auts* of PSIOA, and a mapping $aut : Autids \rightarrow Auts$. $aut(\mathcal{A})$ is the PSIOA with identifier $\mathcal{A}$. We use "the automaton $\mathcal{A}$" to mean "the PSIOA with identifier $\mathcal{A}$". We use the letters $\mathcal{A}, \mathcal{B}$, possibly subscripted or primed, for PSIOA identifiers. The executable actions of a PSIOA $\mathcal{A}$ are drawn from a signature $sig(\mathcal{A})(q) = (in(\mathcal{A})(q), out(\mathcal{A})(q), int(\mathcal{A})(q))$, called the state signature, which is a function of the current state q of $\mathcal{A}$.

$in(\mathcal{A})(q), out(\mathcal{A})(q), int(\mathcal{A})(q)$ are pairwise disjoint sets of input, output, and internal actions, respectively. We define $ext(\mathcal{A})(q)$, the external signature of $\mathcal{A}$ in state q, to be $ext(\mathcal{A})(q) = (in(\mathcal{A})(q), out(\mathcal{A})(q))$.

We define $loc(\mathcal{A})(q)$, the local signature of $\mathcal{A}$ in state q, to be $loc(\mathcal{A})(q) = (out(\mathcal{A})(q), int(\mathcal{A})(q))$. For any signature component, generally, the $\widehat{\cdot}$ operator yields the union of sets of actions within the signature, e.g., $\widehat{sig}(\mathcal{A}) : q \in Q \mapsto \widehat{sig}(\mathcal{A})(q) = in(\mathcal{A})(q) \cup out(\mathcal{A})(q) \cup int(\mathcal{A})(q)$. Also we define $acts(\mathcal{A}) = \bigcup_{q \in Q} \widehat{sig}(\mathcal{A})(q)$, that is $acts(\mathcal{A})$ is the "universal" set of all actions that A could possibly trigger, in any state. In the same way $UI(\mathcal{A}) = \bigcup_{q \in Q} in(\mathcal{A})(q)$, $UO(\mathcal{A}) = \bigcup_{q \in Q} out(\mathcal{A})(q)$, $UH(\mathcal{A}) = \bigcup_{q \in Q} int(\mathcal{A})(q)$, $UL(\mathcal{A}) = \bigcup_{q \in Q} \widehat{loc}(\mathcal{A})(q)$, $UE(\mathcal{A}) = \bigcup_{q \in Q} \widehat{ext}(\mathcal{A})(q)$.

4.3 PSIOA

We combine the SIOA of [2] with the PIOA of [20]:

▶ **Definition 1** (PSIOA). *A PSIOA* $\mathcal{A} = (Q_{\mathcal{A}}, \bar{q}_{\mathcal{A}}, sig(\mathcal{A}), D_{\mathcal{A}})$*, where:*

- $Q_{\mathcal{A}}$ *is a countable set of* states*,* $(Q_{\mathcal{A}}, 2^{Q_{\mathcal{A}}})$ *is the* state space*,*
- $\bar{q}_{\mathcal{A}}$ *is the unique* start state*.*
- $sig(\mathcal{A}) : q \in Q_{\mathcal{A}} \mapsto sig(\mathcal{A})(q) = (in(\mathcal{A})(q), out(\mathcal{A})(q), int(\mathcal{A})(q))$ *is the signature function that maps each state to a triplet of mutually disjoint countable set of* actions*, respectively called* input*,* output *and* internal *actions.*
- $D_{\mathcal{A}} \subset Q_{\mathcal{A}} \times acts(\mathcal{A}) \times Disc(Q_{\mathcal{A}})$ *is the set of* probabilistic discrete transitions *where* $\forall (q, a, \eta) \in D_{\mathcal{A}} : a \in \widehat{sig}(\mathcal{A})(q)$*. If* (q, a, η) *is an element of* $D_{\mathcal{A}}$*, we write* $q \xrightarrow{a} \eta$ *and action* a *is said to be* enabled *at* q*. We note* $enabled(\mathcal{A}) : q \in Q_{\mathcal{A}} \mapsto enabled(\mathcal{A})(q)$ *where* $enabled(\mathcal{A})(q)$ *denotes the set of enabled actions at state* q*. We also note* $steps(\mathcal{A}) \triangleq \{(q, a, q') \in Q_{\mathcal{A}} \times acts(\mathcal{A}) \times Q_{\mathcal{A}} | \exists (q, a, \eta) \in D_{\mathcal{A}}, q' \in supp(\eta)\}$*.*

In addition $\mathcal{A}$ must satisfy the following conditions

- $\mathbf{E_1}$ (input enabling) $\forall q \in Q_{\mathcal{A}},\ in(\mathcal{A})(q) \subseteq enabled(\mathcal{A})(q)$.[1]
- $\mathbf{T_1}$ (Transition determinism): For every $q \in Q_{\mathcal{A}}$ and $a \in \widehat{sig}(\mathcal{A})(q)$ there is at most one $\eta_{(\mathcal{A},q,a)} \in Disc(Q_{\mathcal{A}})$, such that $(q, a, \eta_{(\mathcal{A},q,a)}) \in D_{\mathcal{A}}$.

Later, we will define *execution fragments* as alternating sequences of states and actions with classic and natural consistency rules. But a subtlety will appear with the composability of set of automata at reachable states. Hence, we will define *execution fragments* after "local composability" and "probabilistic configuration automata".

4.4 Local composition

The main aim of a formalism of concurrent systems is to compose several automata $\mathbf{A} = \{\mathcal{A}_1, ..., \mathcal{A}_n\}$ and provide guarantees by composing the guarantees of the different elements of the system. Some syntactical rules have to be satisfied before defining the composition operation.

---

[1] Since the signature is dynamic, we could require $\widehat{sig}(\mathcal{A}) = enabled(\mathcal{A})$

▶ **Definition 2** (Compatible signatures). *Let $S = \{sig_i\}_{i \in \mathcal{I}}$ be a set of signatures. Then S is compatible iff, $\forall i, j \in \mathcal{I}$, $i \neq j$, where $sig_i = (in_i, out_i, int_i)$, $sig_j = (in_j, out_j, int_j)$, we have: 1. $(in_i \cup out_i \cup int_i) \cap int_j = \emptyset$, and 2. $out_i \cap out_j = \emptyset$.*

▶ **Definition 3** (Composition of Signatures). *Let $\Sigma = (in, out, int)$ and $\Sigma' = (in', out', int')$ be compatible signatures. Then we define their composition $\Sigma \times \Sigma = (in \cup in' - (out \cup out'), out \cup out', int \cup int')$[2].*

Signature composition is clearly commutative and associative. Now we can define the compatibility of several automata at a state with the compatibility of their attached signatures. First we define compatibility at a state, and discrete transition for a set of automata for a particular compatible state.

▶ **Definition 4** (compatibility at a state). *Let $\mathbf{A} = \{\mathcal{A}_1, ..., \mathcal{A}_n\}$ be a set of PSIOA. A* state *of $\mathbf{A}$ is an element $q = (q_1, ..., q_n) \in Q_{\mathbf{A}} \triangleq Q_{\mathcal{A}_1} \times ... \times Q_{\mathcal{A}_n}$. We note $q \upharpoonright \mathcal{A}_i \triangleq q_i$. We say $\mathcal{A}_1, ..., \mathcal{A}_n$ are (or $\mathbf{A}$ is)* compatible at state q *if $\{sig(\mathcal{A}_1)(q_1), ..., sig(\mathcal{A}_n)(q_n)\}$ is a set of compatible signatures. In this case we note $sig(\mathbf{A})(q) \triangleq sig(\mathcal{A}_1)(q_1) \times ... \times sig(\mathcal{A}_n)(q_n)$ as per definition 3 and we note $\eta_{(\mathbf{A},q,a)} \in Disc(Q_{\mathbf{A}})$, s.t. $\forall a \in \widehat{sig}(\mathbf{A})(q)$, $\eta_{(\mathbf{A},q,a)} = \eta_1 \otimes ... \otimes \eta_n$ where $\forall j \in [1, n]$, $\eta_j = \eta_{(\mathcal{A}_j, q_j, a)}$ if $a \in sig(\mathcal{A}_j)(q_j)$ and $\eta_j = \delta_{q_j}$ otherwise. Moreover, we note $steps(\mathbf{A}) = \{(q, a, q') | q, q' \in Q_{\mathbf{A}}, a \in sig(\mathbf{A})(q), q' \in supp(\eta_{(\mathbf{A},q,a)})\}$. Finally, we note $\bar{q}_{\mathbf{A}} = (\bar{q}_{\mathcal{A}_1}, ..., \bar{q}_{\mathcal{A}_n})$.*

Let us note that an action a shared by two automata becomes an output action and not an internal action after composition. First, it permits the possibility of further communication using a. Second, it allows associativity. If this property is counter-intuitive, it is always possible to use the classic hiding operator that "hides" the output actions transforming them into internal actions.

▶ **Definition 5** (hiding operator). *Let $sig = (in, out, int)$ be a signature and H a set of actions. We note $hide(sig, H) \triangleq (in, out \setminus H, int \cup (out \cap H))$.*

Let $\mathcal{A} = (Q_{\mathcal{A}}, \bar{q}_{\mathcal{A}}, sig(\mathcal{A}), D_{\mathcal{A}})$ be a PSIOA. Let $h : q \in Q_{\mathcal{A}} \mapsto h(q) \subseteq out(\mathcal{A})(q)$. We note $hide(\mathcal{A}, h) \triangleq (Q_{\mathcal{A}}, \bar{q}_{\mathcal{A}}, sig'(\mathcal{A}), D_{\mathcal{A}})$, where $sig'(\mathcal{A}) : q \in Q_{\mathcal{A}} \mapsto hide(sig(\mathcal{A})(q), h(q))$. Clearly, $hide(\mathcal{A}, h)$ is a PSIOA.

▶ **Lemma 6** (hiding and composition are commutative). *Let $sig_a = (in_a, out_a, int_a)$, $sig_b = (in_b, out_b, int_b)$ be compatible signature and H_a, H_b some set of actions, s.t.*
- *$(H_a \cap out_a) \cap \widehat{sig}_b = \emptyset$ and*
- *$(H_b \cap out_b) \cap \widehat{sig}_b = \emptyset$,*

then $sig'_a \triangleq hide(sig, H_a) \triangleq (in'_a, out'_a, int'_a)$ and $sig'_b \triangleq hide(sig_b, H_b) \triangleq (in'_b, out'_b, int'_b)$ are compatible. Furthermore, if
- *$out_b \cap H_a = \emptyset$,and*
- *$out_a \cap H_b = \emptyset$*

then $sig'_a \times sig'_b = hide(sig_a \times sig_b, H_a \cup H_b)$.

Proof.
- compatibility: After hiding operation, we have:
  - $in'_a = in_a$, $in'_b = in_b$
  - $out'_a = out_a \setminus H_a$, $out'_b = out_b \setminus H_b$

[2] not to be confused with Cartesian product. We keep this notation to stay as close as possible to the literature.

- $int'_a = int_a \cup (out_a \cap H_a)$, $int'_b = int_b \cup (out_b \cap H_b)$

Since $out_a \cap out_b = \emptyset$, a fortiori $out'_a \cap out'_b = \emptyset$. $int_a \cap \widehat{sig}_b = \emptyset$, thus if $(out_a \cap H_a) \cap \widehat{sig}_b = \emptyset$, then $int'_a \cap \widehat{sig}_b = \emptyset$ and with the symetric argument, $int'_b \cap \widehat{sig}_a = \emptyset$. Hence, sig'_a and sig'_b are compatible.

- commutativity:

After composition of $sig'_c = sig'_a \times sig'_b$ operation, we have:
- $out'_c = out'_a \cup out'_b = (out_a \setminus H_a) \cup (out_b \setminus H_b)$. If $out_b \cap H_a = \emptyset$ and $out_a \cap H_b = \emptyset$, then $out'_c = (out_a \cup out_b) \setminus (H_a \cup H_b)$.
- $in'_c = in'_a \cup in'_b \setminus out'_c = in_a \cup in_b \setminus out'_c$
- $int'_c = int'_a \cup int'_b = int_a \cup (out_a \cap H_a) int_b \cup (out_b \cap H_b) = int_a \cup int_b \cup (out_a \cap H_a) \cup (out_b \cap H_b)$. If $out_b \cap H_a = \emptyset$ and $out_a \cap H_b = \emptyset$, then $int'_c = int_a \cup int_b \cup ((out_a \cup out_b) \cap (H_a \cup H_b)$.

and after composition of $sig_d = sig_a \times sig_b$
- $out_d = out_a \cup out_b$
- $in_d = in_a \cup in_b \setminus out_d$
- $int_d = int_a \cup int_b$

Finally, after hiding operation $sig'_d = hide(sig_d, H_a \cup H_b)$ we have :
- $in'_d = in_d$
- $out'_d = out_d \setminus H_a \cup H_b = (out_a \cup out_b) \setminus (H_a \cup H_b)$
- $int'_d = int_d \cup (out_d \cap (H_a \cup H_b)) = (int_a \cup int_b) \cup (out_d \cap (H_a \cup H_b))$

Thus, if $out_b \cap H_a = \emptyset$ and $out_a \cap H_b = \emptyset$
- $in'_d = in'_c$
- $out'_d = out'_c$
- $int'_d = int'_c$

◀

▶ Remark 7. We can restrict hiding operation to set of actions included in the set of output actions of the signature ($H \subseteq out$). In this case, since we alreay have $out_a \cap out_b = \emptyset$ by compatibility, we immediatly have $out_a \cap H_b = \emptyset$ and $out_b \cap H_a = \emptyset$. Thus to obtain compatibility, we only need $in_b \cap H_a = \emptyset$ and $in_a \cap H_b = \emptyset$. Later, the compatibility of PCA will implicitly assume this predicate (otherwise the PCA could not be compatible).

4.5 Renaming operators

We introduce some classic, and sometimes useful operators.

4.5.1 State renaming

We anticipate the definition of isomorphism between PSIOA that differs only syntactically.

▶ **Definition 8.** *(State renaming for PSIOA) Let $\mathcal{A}$ be a PSIOA with $Q_{\mathcal{A}}$ as set of states, let $Q_{\mathcal{A}'}$ be another set of states and let $r : Q_{\mathcal{A}} \rightarrow Q_{\mathcal{A}'}$ be a bijective mapping. Then $r(\mathcal{A})$ (we abuse the notation) is the automaton given by:*
- $\bar{q}_{r(\mathcal{A})} = r(\bar{q}_{\mathcal{A}})$
- $Q_{r(\mathcal{A})} = r(Q_{\mathcal{A}})$
- $\forall q_{\mathcal{A}'} \in Q_{r(\mathcal{A})}, sig(r(\mathcal{A}))(q_{\mathcal{A}'}) = sig(\mathcal{A})(r^{-1}(q_{\mathcal{A}'}))$
- $\forall q_{\mathcal{A}'} \in Q_{r(\mathcal{A})}, \forall a \in sig(r(\mathcal{A}))(q_{\mathcal{A}'})$, *if* $(r^{-1}(q_{\mathcal{A}'}), a, \eta) \in D_{\mathcal{A}}$, *then* $(q_{\mathcal{A}'}, a, \eta') \in D_{r(\mathcal{A})}$ *where* $\eta' \in Disc(Q_{\mathcal{A}'}, \mathcal{F}_{Q_{\mathcal{A}'}})$ *and for every* $q_{\mathcal{A}''} \in Q_{r(\mathcal{A})}$, $\eta'(q_{\mathcal{A}''}) = \eta(r^{-1}(q_{\mathcal{A}''}))$.

▶ **Definition 9.** *(State renaming for PSIOA execution) Let $\mathcal{A}$ and $\mathcal{A}'$ be two PSIOA s.t. $\mathcal{A}' = r(\mathcal{A}')$. Let $\alpha = q^0 a^1 q^1 ...$ be an execution fragment of $\mathcal{A}$. We note $r(\alpha)$ the sequence $r(q^0)a^1 r(q^1)....$*

▶ **Lemma 10.** *Let $\mathcal{A}$ and $\mathcal{A}'$ be two PSIOA s.t. $\mathcal{A}' = r(\mathcal{A})$ with $r : Q_{\mathcal{A}} \rightarrow Q_{\mathcal{A}'}$ being a bijective map. Let α be an execution fragment of $\mathcal{A}$. The sequence $r(\alpha)$ is an execution fragment of $\mathcal{A}$.*

Proof. Let $q^j a^{j+1} q^{j+1}$ be a subsequence of α. $r(q^j) \in Q_{\mathcal{A}'}$ by definition, $a^j \in sig(\mathcal{A}')(r(q^j))$ since $sig(\mathcal{A}')(r(q^j)) = sig(\mathcal{A})(q^j)$, and $\eta_{(\mathcal{A}', r(q^j), a^{j+1})}(r(q^{j+1})) = \eta_{(\mathcal{A}, q^j, a^{j+1})}(q^{j+1}) > 0$. ◀

4.5.2 Action renaming

Action renaming is useful to make automata compatible. This operator is used in the proof of theorem 48 of transitivity of implementation relationship .

▶ **Definition 11** (Action renaming for PSIOA)**.** *Let $\mathcal{A}$ be a PSIOA and let r be a partial function on $Q_{\mathcal{A}} \times acts(\mathcal{A})$, s.t. $\forall q \in Q_{\mathcal{A}}$, $r(q)$ is an injective mapping with $\widehat{sig}(\mathcal{A})(q)$ as domain. Then $r(\mathcal{A})$ is the automata given by:*

1. $\bar{q}_{r(\mathcal{A})} = \bar{q}_{\mathcal{A}}$.
2. $Q_{r(\mathcal{A})} = Q_{\mathcal{A}}$.
3. $\forall q \in Q_{\mathcal{A}}$, $sig(r(\mathcal{A}))(q) = (in(r(\mathcal{A}))(q)\ , out(r(\mathcal{A}))\ (q)\ , int(r(\mathcal{A}))(q))$ *with*
   - $out(r(\mathcal{A}))(q) = r(out(\mathcal{A})(q))$,
   - $in(r(\mathcal{A}))(q) = r(in(\mathcal{A})(q))$,
   - $int(r(\mathcal{A}))(q) = r(int\ (\mathcal{A})(q))$.
4. $D_{r(\mathcal{A})} = \{(q, r(a), \eta) | (q, a, \eta) \in D_{\mathcal{A}}\}$ *(we note $\eta_{(r(\mathcal{A}), q, r(a))}$ the element of $Disc(Q_{r(\mathcal{A})})$ which is equal to $\eta_{(\mathcal{A}, q, a)}$.*

▶ **Lemma 12** (PSIOA closeness under action-renaming)**.** *Let $\mathcal{A}$ be a PSIOA and let r be a partial function on $Q_{\mathcal{A}} \times acts(\mathcal{A})$, s.t. $\forall q \in Q_{\mathcal{A}}$, $r(q)$ is an injective mapping with $\widehat{sig}(\mathcal{A})(q)$ as domain. Then $r(\mathcal{A})$ is a PSIOA.*

Proof. We need to show (1) $\forall (q, a, \eta), (q, a, \eta') \in D_{\mathcal{A}}$, $\eta = \eta'$ and $a \in \widehat{sig}(\mathcal{A})(q)$, (2) $\forall q \in Q_{\mathcal{A}}, \forall a \in \widehat{sig}(\mathcal{A})(q)$, $\exists \eta \in Disc(Q_{\mathcal{A}})$, $(q, a, \eta) \in D_{\mathcal{A}}$ and (3) $\forall q \in Q_{\mathcal{A}} : in(\mathcal{A})(q) \cap out(\mathcal{A})(s) = in(\mathcal{A})(q) \cap int(\mathcal{A})(q) = out(\mathcal{A})(q) \cap int(\mathcal{A})(q) = \emptyset$.

- Constraint 1: From definition 11, we have, for any $q \in Q_{r(\mathcal{A})}$: $\widehat{sig}(r(\mathcal{A}))(q) = out(r(\mathcal{A}))(q) \cup in(r(\mathcal{A}))\ (q) \cup int(r(\mathcal{A}))(q) = r(out(\mathcal{A})(q)) \cup r(in(\mathcal{A})(q)) \cup r(int(\mathcal{A})(q)) = r(\widehat{sig}(\mathcal{A})(q))$. Since $\mathcal{A}$ is a PSIOA, we have $\forall (q, a, \eta), (q, a, \eta') \in D_{\mathcal{A}} : a \in \widehat{sig}(\mathcal{A})(q)$ and $\eta = \eta'$. From definition 11, $D_{r(\mathcal{A})} = \{(q, r(a), \eta) |\ (q, a, \eta) \in D_{\mathcal{A}}\}$ Hence, if $(q, r(a), \eta)$, $(q, r(a), \eta')$ are arbitrary element of $D_{r(\mathcal{A})}$, then $(q, a, \eta), (q, a, \eta') \in D_{\mathcal{A}}$, and so $\eta = \eta'$ and $a \in \widehat{sig}(\mathcal{A})(q)$. Hence $r(a) \in r(\widehat{sig}(\mathcal{A})(q))$. Since $r(\widehat{sig}(\mathcal{A})(q)) = \widehat{sig}(r(\mathcal{A}))(q)$, we conclude $r(a) \in \widehat{sig}(r(\mathcal{A}))(q)$. Hence, $\forall (q, r(a), \eta)$, $(q, r(a), \eta') \in D_{r(\mathcal{A})} : r(a) \in \widehat{sig}(r(\mathcal{A}))(q)$ and $\eta = \eta'$. Thus, Constraint 1 holds for $r(\mathcal{A})$.
- Constraint 2: From definition 11, $D_{r(\mathcal{A})} = \{(q, r(a), \eta) | (q, a, \eta) \in D_{\mathcal{A}}\}$, $Q_{r(\mathcal{A})} = Q_{\mathcal{A}}$, and for all $q \in Q_{r(\mathcal{A})}, in(r(\mathcal{A}))(q) = r(in(\mathcal{A})(q))$. Let q be any state of $r(\mathcal{A})$, and let $q \in \widehat{sig}(r(\mathcal{A}))(q)$. Then $b = r(a)$ for some $a \in \widehat{sig}(\mathcal{A})(q)$. We have $(q, a, \eta) \in D_{\mathcal{A}}$ for some η, by Constraint 2 of action enabling for $\mathcal{A}$. Hence $(q, a, \eta) \in D_{r(\mathcal{A})}$. Hence $(q, b, \eta) \in D_{r(\mathcal{A})}$. Hence Constraint 2 holds for $r(\mathcal{A})$.
- Constraint 3: $\mathcal{A}$ is a PSIOA and so satisfies Constraint 3. From this and definition 11 and the requirement that r be injective, it is easy to see that $r(\mathcal{A})$ also satisfies Constraint 3.

◀

5 Probabilistic Configuration Automata

We combine the notion of configuration of [2] with the probabilistic setting of [20]. A configuration is a set of automata attached with their current states. This will be a very useful tool to define dynamicity by mapping the state of an automaton of a certain "layer" to a configuration of automata of lower layer, where the set of automata in the configuration can dynamically change from on state of the automaton of the upper level to another one.

5.1 configuration

▶ **Definition 13** (Configuration). *A configuration is a pair* $(\mathbf{A}, \mathbf{S})$ *where*

- $\mathbf{A} = \{\mathcal{A}_1, ..., \mathcal{A}_n\}$ *is a finite set of PSIOA identifiers and*
- $\mathbf{S}$ *maps each* $\mathcal{A}_k \in \mathbf{A}$ *to a state of* $\mathcal{A}_k$.

In distributed computing, configuration usually refers to the union of states of ***all*** *the automata of the "system". Here, there is a subtlety, since it captures a set of some automata* $(\mathbf{A})$ *in their current state* $(\mathbf{S})$*, but the set of automata of the systems will not be fixed in the time.*

We note Q^{conf} *the (countable) set of configurations.*

▶ **Proposition 14.** *The set* Q_{conf} *of configurations is countable.*

Proof. (1) $\{\mathbf{A} \in \mathcal{P}(Autids) | \mathbf{A} \text{ is finite}\}$ is countable, (2) $\forall \mathcal{A} \in Autids, Q_{\mathcal{A}}$ is countable by definition 1 of PSIOA and (3) the cartesian product of countable sets is a countable set. ◀

▶ **Definition 15** (Compatible configuration). *A configuration* $(\mathbf{A}, \mathbf{S})$*, with* $\mathbf{A} = \{\mathcal{A}_1, ..., \mathcal{A}_n\}$*, is compatible iff the set* $\mathbf{A}$ *is compatible at state* $(\mathbf{S}(\mathcal{A}_1), ..., \mathbf{S}(\mathcal{A}_n))$ *as per definition 4*

▶ **Definition 16** (Intrinsic attributes of a configuration). *Let* $C = (\mathbf{A}, \mathbf{S})$ *be a compatible configuration. Then we define*

- $auts(C) = \mathbf{A}$ *represents the automata of the configuration,*
- $map(C) = \mathbf{S}$ *maps each automaton of the configuration with its current state,*
- $TS(C) = (\mathbf{S}(\mathcal{A}_1), ..., \mathbf{S}(\mathcal{A}_n))$ *yields the tuple of states of the automata of the configuration.*
- $sig(C) = (in(C), out(C), int(C)) = sig(auts(C), TS(C))$ *in the sense of definition 4, is called the intrinsic signature of the configuration*

Here we define a reduced configuration as a configuration deprived of the automata that are in the very particular state where their current signatures are the empty set. This mechanism will be used later to capture the idea of destruction of an automaton.

▶ **Definition 17** (Reduced configuration). $reduce(C) = (\mathbf{A}', \mathbf{S}')$*, where* $\mathbf{A}' = \{\mathcal{A} | \mathcal{A} \in \mathbf{A} \text{ and } sig(\mathcal{A})(\mathbf{S}(\mathcal{A})) \neq \emptyset\}$ *and* $\mathbf{S}'$ *is the restriction of* $\mathbf{S}$ *to* $\mathbf{A}'$*, noted* $\mathbf{S} \upharpoonright \mathbf{A}'$ *in the remaining.*

A configuration C *is a reduced configuration iff* $C = reduce(C)$.

We will define some probabilistic transition from configurations to others where some automata can be destroyed or created. To define it properly, we start by defining "preserving transition" where no automaton is neither created nor destroyed and then we define above this definition the notion of configuration transition.

▶ **Definition 18** (From preserving distribution to intrinsic transition).

- *(preserving distribution) Let $\eta_p \in Disc(Q_{conf})$. We say η_p is a* preserving distribution *if it exists a finite set of automata $\mathbf{A}$, called* family support *of η_p, s.t. $\forall (\mathbf{A}', \mathbf{S}') \in supp(\eta_p), \mathbf{A} = \mathbf{A}'$.*
- *(preserving configuration transition $C \stackrel{a}{\rightharpoonup} \eta_p$) Let $C = (\mathbf{A}, \mathbf{S})$ be a compatible configuration, $a \in \widehat{sig}(C)$. Let η_p be the unique preserving distribution of $Disc(Q_{conf})$ such that (1) the family support of η_p is $\mathbf{A}$ and (2) $\eta_p \stackrel{TS}{\leftrightarrow} \eta_{(\mathbf{A},TS(C),a)}$. We say that (C, a, η_p) is a* preserving configuration transition, *noted $C \stackrel{a}{\rightharpoonup} \eta_p$.*
- *($\eta_p \uparrow \varphi$) Let $\eta_p \in Disc(Q_{conf})$ be a preserving distribution with $\mathbf{A}$ as family support. Let φ be a finite set of of PSIOA identifiers with $\mathbf{A} \cap \varphi = \emptyset$. Let $C_\varphi = (\varphi, S_\varphi) \in Q_{conf}$ with $\forall \mathcal{A}_j \in \varphi, S_\varphi(\mathcal{A}_j) = \bar{q}_{\mathcal{A}_j}$. We note $\eta_p \uparrow \varphi$ the unique element of $Disc(Q_{conf})$ verifying $\eta_p \stackrel{u}{\leftrightarrow} (\eta_p \uparrow \varphi)$ with $u : C \in supp(\eta_p) \mapsto (C \cup C_\varphi)$.*
- *(distribution reduction) Let $\eta \in Disc(Q_{conf})$. We note $reduce(\eta)$ the element of $Disc(Q_{conf})$ verifying $\forall c \in Q_{conf}$, $(reduce(\eta))(c) = \Sigma_{(c' \in supp(\eta), c = reduce(c'))} \eta(c')$*
- *(intrinsic transition $C \stackrel{a}{\Longrightarrow}_\varphi \eta$) Let $C = (\mathbf{A}, \mathbf{S})$ be a compatible configuration, let $a \in \widehat{sig}(C)$, let φ be a finite set of of PSIOA identifiers with $\mathbf{A} \cap \varphi = \emptyset$. We note $C \stackrel{a}{\Longrightarrow}_\varphi \eta$, if $\eta = reduce(\eta_p \uparrow \varphi)$ with $C \stackrel{a}{\rightharpoonup} \eta_p$. In this case, we say that η is* generated by *η_p* and *φ.*

Preserving configuration transition (C, a, η_p) is the intuitive transition for configurations, corresponding to the transition $(TS(C), a, \eta_{(auts(C),TS(C),a)})$ (see figure 12). The operator $\uparrow \varphi$ describes the deterministic creation of automata in φ, who will be appear at their respective start states. The *reduce* operator enables to remove "destroyed" automata from the possibly returned configurations (see figure 13).

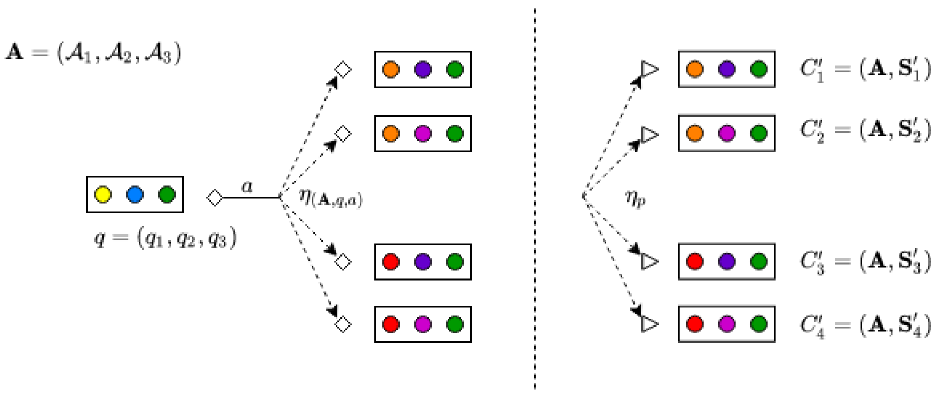


Figure 12 There is a trivial homomorphism between the preserving distribution η_p with $C = (\mathbf{A}, \mathbf{S}) \stackrel{a}{\rightharpoonup} \eta_p$ and the distribution $\eta_{(\mathbf{A},TS(C),a)}$.

5.2 probabilistic configuration automata (PCA)

Now we are ready to define our probabilistic configuration automata (see figure 14). Such an automaton define a strong link with a dynamic configuration.

▶ **Definition 19** (Probabilistic Configuration Automaton). *A probabilistic configuration automaton (PCA) X consists of the following components:*

- *1. A probabilistic signature I/O automaton $psioa(X)$. For brevity, we define $Q_X = Q_{psioa(X)}, \bar{q}_X = \bar{q}_{psioa(X)}$, $sig(X) = sig(psioa(X)), steps(X) = steps(psioa(X))$, and likewise for all other (sub)components and attributes of $psioa(X)$.*

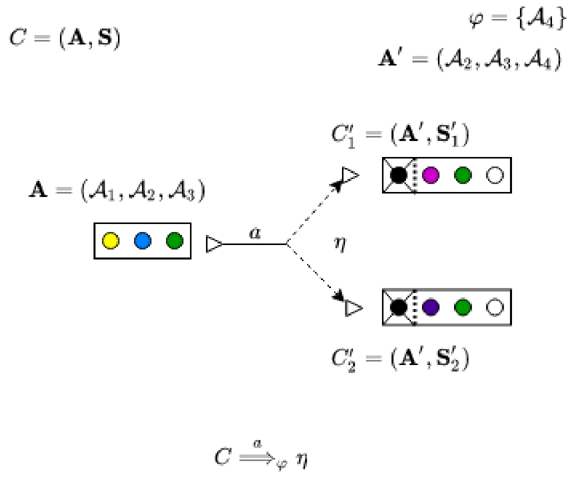


Figure 13 An intrinsic transition where $\mathcal{A}_1$ is destroyed deterministically and $\mathcal{A}_4$ is created deterministically. First, we have the preserving disribution η_p s.t. $C \stackrel{a}{\rightharpoonup} \eta_p$ with $\eta_p \stackrel{TS}{\leftrightarrow} \eta_{(\mathbf{A},TS(C),a)}$. Second, we take into account the created automata $\varphi = \{\mathcal{A}\}$, captured by the distribution $\eta_p \uparrow \varphi$. Third, we remove the automata in a particular state with associated empty signature. This is captured by distribution $reduce(\eta_p \uparrow \varphi)$.

- *2. A configuration mapping $config(X)$ with domain Q_X and such that, for all $q \in Q_X$, $config(X)(q)$ is a reduced compatible configuration.*
- *3. For each $q \in Q_X$, a mapping $created(X)(q)$ with domain $sig(X)(q)$ and such that $\forall a \in sig(X)(q)$, $created(X)(q)(a) \subseteq Autids$ with $created(X)(q)(a)$ finite.*
- *4. A hidden-actions mapping hidden-actions(X) with domain Q_X and such that hidden-actions$(X)(q) \subseteq out(config(X)(q))$.*

and satisfies the following constraints, for every $q \in Q_X$, $C = config(X)(q)$, $H =$ hidden-actions(q).

- *1. (start states preservation) If $config(X)(\bar{q}_X) = (\mathbf{A}, \mathbf{S})$, then $\forall \mathcal{A}_i \in \mathbf{A}, \mathbf{S}(\mathcal{A}_i) = \bar{q}_{\mathcal{A}_i}$.*
- *2. (top/down transition preservation) If $(q, a, \eta_{(X,q,a)}) \in D_X$, then $\exists \eta' \in Disc(Q_{conf})$ s.t. $\eta_{(X,q,a)} \stackrel{c}{\leftrightarrow} \eta'$ with $C \stackrel{a}{\Longrightarrow}_\varphi \eta'$, where $\varphi = created(X)(q)(a)$ and $c = config(X)$.*
- *3. (bottom/up transition preservation) If $q \in Q_X$ and $C \stackrel{a}{\Longrightarrow}_\varphi \eta'$ for some action a, $\varphi = created(X)(q)(a)$, and reduced compatible probabilistic measure $\eta' \in Disc(Q_{conf})$, then $(q, a, \eta_{(X,q,a)}) \in D_X$, and $\eta_{(X,q,a)} \stackrel{c}{\leftrightarrow} \eta'$ where $c = config(X)$.*
- *4. (signature preservation modulo hiding) $\forall q \in Q_X$, $sig(X)(q) = hide(sig(C), H)$.*

This definition, proposed in a deterministic fashion in [2], captures dynamicity of the system. Each state is linked with a configuration. The set of automata of the configuration can change during an execution. A sub-automaton $\mathcal{A}$ is created from state q by the action a if $\mathcal{A} \in created(X)(q)(a)$. A sub-automaton $\mathcal{A}$ is destroyed if the non-reduced attached configuration distribution leads to a configuration where $\mathcal{A}$ is in a state $q^\phi_{\mathcal{A}}$ s. t. $\widehat{sig}(\mathcal{A})(q^\phi_{\mathcal{A}}) = \emptyset$. Then the corresponding reduced configuration will not hold $\mathcal{A}$. The last constraint states that the signature of a state q of X must be the same as the signature of its corresponding configuration $config(X)(q)$, except for the possible effects of hiding operators, so that some outputs of $config(X)(q)$ may be internal actions of X in state q.

As for PSIOA, we can define hiding operator applied to PCA.

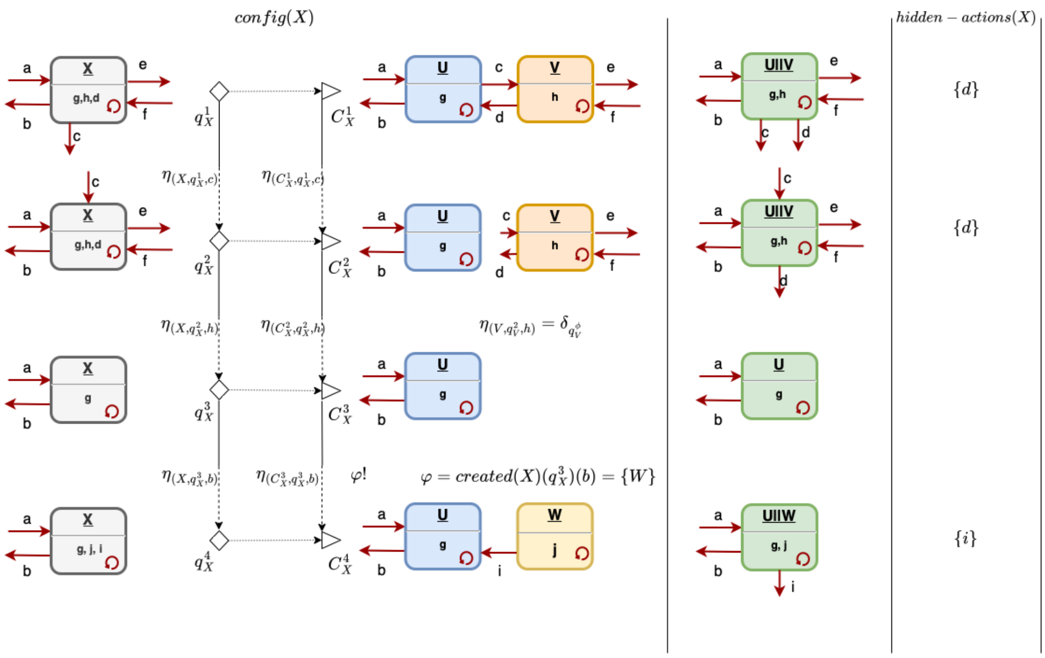


Figure 14 A PCA life cycle.

▶ **Definition 20** (hiding on PCA). *Let X be a PCA. Let $h : q \in Q_X \mapsto h(q) \subseteq out(X)(q)$. We note $hide(X, h)$ the PCA X' that differs from X only on*

- $psioa(X') = hide(psioa(X), h)$
- $sig(X') = hide(sig(X), h)$ *and*
- $\forall q \in Q_X = Q_{X'}$*,* $hidden\text{-}actions(X')(q) = hidden\text{-}actions(X)(q) \cup h(q)$*.*

The notion of local compatibility can be naturally extended to set of PCA.

▶ **Definition 21** (PCA compatible at a state). *Let $\mathbf{X} = \{X_1, ..., X_n\}$ be a set of PCA. Let $q = (q_1, ..., q_n) \in Q_{X_1} \times ... \times Q_{X_n}$. Let us note $C_i = (\mathbf{A}_i, \mathbf{S}_i) = config(X_i)(q_i)$, $\forall i \in [1, n]$. The PCA in $\mathbf{X}$ are compatible at state q iff*[3]*:*

1. *PSIOA compatibility: $psioa(X_1), ..., psioa(X_n)$ are compatible at $q_{\mathbf{X}}$.*
2. *Sub-automaton exclusivity: $\forall i, j \in [1 : n], i \neq j : \mathbf{A}_i \cap \mathbf{A}_j = \emptyset$.*
3. *Creation exclusivity: $\forall i, j \in [1 : n], i \neq j, \forall a \in \widehat{sig}(X_i)(q_i) \cap \widehat{sig}(X_j)(q_j)$: $created(X_i)(q_i)(a) \cap created(X_j)(q_j)(a) = \emptyset$.*

If $\mathbf{X}$ is compatible at state q, for every action $a \in \widehat{sig}(psioa(\mathbf{X}))(q)$, we note $\eta_{(\mathbf{X},q,a)} = \eta_{(psioa(\mathbf{X}),q,a)}$ and we extend this notation with $\eta_{(\mathbf{X},q,a)} = \delta_q$ if $a \notin \widehat{sig}(psioa(\mathbf{X}))(q)$.

6 Executions, reachable states, partially-compatible automata

6.1 Executions, reachable states, traces

In previous sections, we have described how to model probabilistic transitions that might lead to the creation and destruction of some components of the system. In this section, we

[3] We can remark that the conjunction of PSIOA compatibility and sub-automata exclusivity implies the compatibility of respective configurations as defined later in definition 27

will define pseudo execution fragments of a set of automata to model the run of a set **A** of several dynamic systems interacting with each others. With such a definition, we will kill two birds with one stone, since it will allow to define *reachable states* of **A** and then compatibility of **A** as compatibility of **A** at each reachable state.

▶ **Definition 22** (pseudo execution, reachable states, partial-compatibility). *Let* $\mathbf{A} = \{\mathcal{A}_1, ..., \mathcal{A}_n\}$ *be a finite set of PSIOA (resp. PCA). A* pseudo execution fragment *of* **A** *is a finite or infinite sequence* $\alpha = q^0 a^1 q^1 a^2 ...$ *of alternating states and actions, such that:*

1. *If* α *is finite, it ends with a state. In that case, we note* $lstate(\alpha)$ *the last state of* α.
2. **A** *is compatible at each state of* α*, with the potential exception of* $lstate(\alpha)$ *if* α *is finite.*
3. *for ever action* a^i, $(q^{i-1}, a^i, q^i) \in steps(\mathbf{A})$.

The first state of a pseudo execution fragment α *is noted* $fstate(\alpha)$*. A pseudo execution fragment* α *of* **A** *is a pseudo execution of* **A** *if* $fstate(\alpha) = \bar{q}_{\mathbf{A}}$*. The length* $|\alpha|$ *of a finite pseudo execution fragment* α *is the number of actions in* α*. A state* q *of* **A** *is said* reachable *if there is a pseudo execution* α *s.t.* $lstate(\alpha) = q$*. We note* $Reachable(\mathbf{A})$ *the set of* reachable *states of* **A***. If* **A** *is compatible at every reachable state* q*,* **A** *is said* partially-compatible.[4]

▶ **Definition 23** (executions, concatenations). *Let* $\mathcal{A}$ *be an automaton. An* execution fragment *(resp.* execution*) of* $\mathcal{A}$ *is a pseudo execution fragment (resp. pseudo execution) of* $\{\mathcal{A}\}$*. We use* $Frags(\mathcal{A})$ *(resp.,* $Frags^*(\mathcal{A})$*) to denote the set of all (resp., all finite) execution fragments of* $\mathcal{A}$*.* $Execs(\mathcal{A})$ *(resp.* $Execs^*(\mathcal{A})$*) denotes the set of all (resp., all finite) executions of* $\mathcal{A}$*.*

We define a concatenation operator $\frown$ *for execution fragments as follows:*
If $\alpha = q^0 a^1 q^1 ... a^n q^n \in Frags^*(\mathcal{A})$ *and* $\alpha' = q^{0\prime} a^{1\prime} q^{1\prime} ... \in Frags^*(\mathcal{A})$*, we define* $\alpha^\frown\alpha' \triangleq q^0 a^1 q^1 ... a^n q^n\ a^{1\prime} q^{1\prime} ...$ *only if* $s^0 = q^n$*, otherwise* $\alpha^\frown\alpha'$ *is undefined. Hence the notation* $\alpha^\frown\alpha'$ *implicitly means* $fstate(\alpha') = lstate(\alpha)$*.*

Let $\alpha, \alpha' \in Frags(\mathcal{A})$*, then* α *is a proper prefix of* α' *iff* $\exists \alpha'' \in Frags(\mathcal{A})$ *such that* $\alpha' = \alpha^\frown\alpha''$ *with* $\alpha \neq \alpha'$*. In that case, we note* $\alpha < \alpha'$*. We note* $\alpha \leq \alpha'$ *if* $\alpha < \alpha'$ *or* $\alpha = \alpha'$ *and say that* α *is a prefix of* α'*. Finally,* α, α' *are said comparable if either* $\alpha \leq \alpha'$ *or* $\alpha' \leq \alpha$*.*

▶ **Definition 24** (traces). *The* trace *of an execution* α *represents its externally visible part, i.e. the external actions. Let* $\mathcal{A}$ *be a PSIOA (resp. PCA). Let* $q^0 \in Q_{\mathcal{A}}$*,* $(q, a, q') \in steps(\mathcal{A})$*,* $\alpha, \alpha' \in Execs^*(\mathcal{A}) \times Execs(\mathcal{A})$ *with* $fstate(\alpha') = lstate(\alpha)$*.*

$trace_{\mathcal{A}}(q^0)$ *is the empty sequence, noted* λ*,*

$$trace_{\mathcal{A}}(qaq') \begin{cases} a \text{ if } a \in \widehat{ext}(\mathcal{A})(q) \\ \lambda \text{ otherwise.} \end{cases},$$

$trace_{\mathcal{A}}(\alpha^\frown\alpha') = trace_{\mathcal{A}}(\alpha)^\frown trace_{\mathcal{A}}(\alpha')$

We say that β *is a trace of* $\mathcal{A}$ *if* $\exists \alpha \in Execs(\mathcal{A})$ *with* $\beta = trace_{\mathcal{A}}(\alpha)$*. We note* $Traces(\mathcal{A})$ *(resp.* $Traces^*(\mathcal{A})$*, resp.* $Traces^\omega(\mathcal{A})$*) the set of traces (resp. finite traces, resp. infinite traces) of* $\mathcal{A}$*. When the automaton* $\mathcal{A}$ *is understood from context, we write simply* $trace(\alpha)$*.*

The projection of a pseudo-execution α on an automaton $\mathcal{A}_i$, noted $\alpha \upharpoonright \mathcal{A}_i$, represents the contribution of $\mathcal{A}_i$ to this execution.

▶ **Definition 25** (projection). *Let* **A** *be a set of PSIOA (resp. PCA), let* $\mathcal{A}_i \in \mathbf{A}$*. We define projection operator* $\upharpoonright$ *recursively as follows: For every* $(q, a, q') \in steps(\mathbf{A})$*, for every* α, α' *being two pseudo executions of* **A** *with* $fstate(\alpha') = lstate(\alpha)$*.*

---

[4] In [2], compatible set of PCA are compatible at every (potentially non-reachable) state of the associated Cartesian product.

$$(q, a, q') \upharpoonright \mathcal{A}_i = \begin{cases} (q \upharpoonright \mathcal{A}_i), a, (q' \upharpoonright \mathcal{A}_i) \ \textit{if } a \in \widehat{sig}(\mathcal{A}_i)(q \upharpoonright \mathcal{A}_i) \\ (q \upharpoonright \mathcal{A}_i) = (q' \upharpoonright \mathcal{A}_i) \ \textit{otherwise.} \end{cases},$$

$$(\alpha^\frown\alpha') \upharpoonright \mathcal{A}_i = (\alpha \upharpoonright \mathcal{A}_i)^\frown(\alpha' \upharpoonright \mathcal{A}_i)$$

6.2 PSIOA and PCA composition

We are ready to define composition operator, the most important operator for concurrent systems.

▶ **Definition 26** (PSIOA partial-composition). *If $\mathbf{A} = \{\mathcal{A}_1, ..., \mathcal{A}_n\}$ is a partially-compatible set of PSIOA, with $\mathcal{A}_i = (Q_{\mathcal{A}_i}, \bar{q}_{\mathcal{A}_i}, sig(\mathcal{A}_i), D_{\mathcal{A}_i})$, then their partial-composition $\mathcal{A}_1||...||\mathcal{A}_n$, is defined to be $\mathcal{A} = (Q_\mathcal{A}, \bar{q}_\mathcal{A}, sig(\mathcal{A}), D_\mathcal{A})$, where:*
- $Q_\mathcal{A} = Reachable(\mathbf{A})$
- $\bar{q}_\mathcal{A} = (\bar{q}_{\mathcal{A}_1}, ..., \bar{q}_{\mathcal{A}_n})$
- $sig(\mathcal{A}) : q \in Q_\mathcal{A} \mapsto sig(\mathcal{A})(q) = sig(\mathbf{A})(q)$
- $D_\mathcal{A} = \{(q, a, \eta_{(\mathbf{A},q,a)}) | q \in Q_\mathcal{A}, a \in \widehat{sig}(\mathbf{A})(q)\}$

▶ **Definition 27** (Union of configurations). *Let $C_1 = (\mathbf{A}_1, \mathbf{S}_1)$ and $C_2 = (\mathbf{A}_2, \mathbf{S}_2)$ be configurations such that $\mathbf{A}_1 \cap \mathbf{A}_2 = \emptyset$. Then, the union of C_1 and C_2, denoted $C_1 \cup C_2$, is the configuration $C = (\mathbf{A}, \mathbf{S})$ where $\mathbf{A} = \mathbf{A}_1 \cup \mathbf{A}_2$ and $\mathbf{S}$ agrees with $\mathbf{S}_1$ on $\mathbf{A}_1$, and with $\mathbf{S}_2$ on $\mathbf{A}_2$. Moreover, if $C_1 \cup C_2$ is a compatible configuration, we say that C_1 and C_2 are* compatible configurations*. It is clear that configuration union is commutative and associative. Hence, we will freely use the n-ary notation $C_1 \cup ... \cup C_n$, whenever $\forall i, j \in [1 : n], i \neq j, auts(C_i) \cap auts(C_j) = \emptyset$.*

▶ **Lemma 28.** *Let $C_1 = (\mathbf{A}_1, \mathbf{S}_1)$ and $C_2 = (\mathbf{A}_2, \mathbf{S}_2)$ be configurations such that $\mathbf{A}_1 \cap \mathbf{A}_2 = \emptyset$. Let $C = (\mathbf{A}, \mathbf{S}) = C_1 \cup C_2$ be a compatible configuration. Then $sig(C) = sig(C_1) \times sig(C_2)$ (in the sense of definition 3).*

Proof.

$$\begin{aligned} out(C) &= \bigcup_{\mathcal{A}_k \in \mathbf{A}} out(\mathcal{A}_k)(\mathbf{S}(\mathcal{A}_k)) \\ &= (\bigcup_{\mathcal{A}_i \in \mathbf{A}_1} out(\mathcal{A}_i)(\mathbf{S}(\mathcal{A}_i))) \cup (\bigcup_{\mathcal{A}_j \in \mathbf{A}_2} out(\mathcal{A}_j)(\mathbf{S}(\mathcal{A}_j))) \\ &= (\bigcup_{\mathcal{A}_i \in \mathbf{A}_1} out(\mathcal{A}_i)(\mathbf{S}_1(\mathcal{A}_i))) \cup (\bigcup_{\mathcal{A}_j \in \mathbf{A}_2} out(\mathcal{A}_j)(\mathbf{S}_2(\mathcal{A}_j))) \\ &= out(C_1) \cup out(C_2) \end{aligned}$$

$$\begin{aligned} in(C) &= \bigcup_{\mathcal{A}_k \in \mathbf{A}} in(\mathcal{A}_k)(\mathbf{S}(\mathcal{A}_k)) \setminus out(C) \\ &= (\bigcup_{\mathcal{A}_i \in \mathbf{A}_1} in(\mathcal{A}_i)(\mathbf{S}(\mathcal{A}_i))) \cup (\bigcup_{\mathcal{A}_j \in \mathbf{A}_2} in(\mathcal{A}_j)(\mathbf{S}(\mathcal{A}_j))) \setminus out(C) \\ &= (\bigcup_{\mathcal{A}_i \in \mathbf{A}_1} in(\mathcal{A}_i)(\mathbf{S}_1(\mathcal{A}_i))) \cup (\bigcup_{\mathcal{A}_j \in \mathbf{A}_2} in(\mathcal{A}_j)(\mathbf{S}_2(\mathcal{A}_j))) \setminus out(C) \\ &= in(C_1) \cup in(C_2) \setminus (out(C_1) \cup out(C_2)) \end{aligned}$$

$$int(C) = \bigcup_{\mathcal{A}_k \in \mathbf{A}} int(\mathcal{A}_k)(\mathbf{S}(\mathcal{A}_k))$$
$$= (\bigcup_{\mathcal{A}_i \in \mathbf{A}_1} int(\mathcal{A}_i)(\mathbf{S}(\mathcal{A}_i))) \cup (\bigcup_{\mathcal{A}_j \in \mathbf{A}_2} int(\mathcal{A}_j)(\mathbf{S}(\mathcal{A}_j)))$$
$$= (\bigcup_{\mathcal{A}_i \in \mathbf{A}_1} int(\mathcal{A}_i)(\mathbf{S}_1(\mathcal{A}_i))) \cup (\bigcup_{\mathcal{A}_j \in \mathbf{A}_2} int(\mathcal{A}_j)(\mathbf{S}_2(\mathcal{A}_j)))$$
$$= int(C_1) \cup int(C_2)$$

◀

▶ **Definition 29** (PCA partial-composition). *If $\mathbf{X} = \{X_1, ..., X_n\}$ is a partially-compatible set of PCA, then their partial-composition $X_1||...||X_n$, is defined to be the PCA X (proved in theorem 38 in section 7) s.t. $psioa(X) = psioa(X_1)||...||psioa(X_n)$ and $\forall q \in Q_X$:*

- $config(X)(q) = \bigcup_{i \in [1,n]} config(X_i)(q \upharpoonright X_i)$
- $\forall a \in \widehat{sig}(X)(q)$, $created(X)(q)(a) = \bigcup_{i \in [1,n]} created(X_i)(q \upharpoonright X_i)(a)$, *with the convention* $created(X_i)(q_i)(a) = \emptyset$ *if* $a \notin \widehat{sig}(X_i)(q_i)$
- $hidden\text{-}actions(q) = \bigcup_{i \in [1,n]} hidden\text{-}actions(X_i)(q \upharpoonright X_i)$

7 Toolkit for configurations & PCA closeness under composition

In this section, we define some tools to manipulate measure preserving bijections between probability distributions (relations of the form $\eta \overset{f}{\leftrightarrow} \eta'$). This tools will be used to prove (1) the closeness of PCA under parallel composition (theorem 38) and some intermediate results in the proof of monotonicity of implementation relationship w.r.t. creation/destruction of PSIOA.

Merge, join, split

▶ **Definition 30** (join). *Let $\tilde{\eta} = (\eta_1, ..., \eta_n) \in Disc(Q_1) \times ... \times Disc(Q_n)$ with each Q_i being a set. We define, join($\tilde{\eta}$):* $\begin{cases} Q_1 \times ... \times Q_n & \rightarrow & [0,1] \\ \tilde{q} & \mapsto & (\eta_1 \otimes ... \otimes \eta_n)(\tilde{q}) \end{cases}$

▶ **Lemma 31** (Joint preserving probability distribution for union of configuration). *Let $n \in \mathbb{N}$, let $\{C_k\}_{k\in[1:n]}$ be a set of compatible configurations and $C_0 = \bigcup_{k\in[1:n]} C_k$. Let $(\eta_p^0, ..., \eta_p^n) \in Disc(Q_{conf})^{n+1}$ s.t. $\forall k \in [0:n]$, $C_k \overset{a}{\rightharpoonup} \eta_p^k$ if $a \in \widehat{sig}(C_k)$ and $\eta_p^k = \delta_{C_k}$ otherwise.*
Then, $\forall (C'_1, ..., C'_n) \in Q^n_{conf}$, s.t. $\forall k \in [1:n], aut(C'_k) = aut(C_k)$,
$\eta_p^0(\bigcup_{k\in[1:n]} C'_k) = (\eta_p^1 \otimes ... \otimes \eta_p^n)(C'_1, ..., C'_n)$.

Proof. We note $\{C_k = (\mathbf{A}_k, \mathbf{S}_k)\}_{k\in[1:n]}$, $C_0 = (\mathbf{A}_0, \mathbf{S}_0)$, $q_k = TS(C_k)$ for every $k \in [0:n]$. We note $(\mathcal{I}, \mathcal{J})$ the partition of $[1:n]$ s.t. $\forall i \in \mathcal{I}, a \in \widehat{sig}(C_i)$ and $\forall j \in \mathcal{J}, a \notin \widehat{sig}(C_j)$. Since $\mathbf{A}_0 = \bigcup_{k\in[1:n]} \mathbf{A}_k$ and $\mathbf{S}_0$ agrees with $\mathbf{S}_k$ on $\mathcal{A} \in \mathbf{A}_k$ for every $k \in [1:n]$, we have $\eta_{\mathbf{A}_0,q_0,a} = \eta_{(\mathbf{A}_1,q_1,a)} \otimes ... \otimes \eta_{(\mathbf{A}_n,q_n,a)}$ with the convention $\eta_{(\mathbf{A}_j,q_j,a)} = \delta_{q_j}$, $\forall j \in \mathcal{J}$. Furthermore, for every $k \in [1,n], \eta_p^k \overset{TS}{\leftrightarrow} \eta_{(\mathbf{A}_k,q_k,a)}$, that is for every $(C'_k, q'_k) \in Q_{conf} \times Q_{\mathbf{A}_k}$ with $q'_k = TS(C'_k)$, $\eta_p^k(C'_k) = \eta_{(\mathbf{A}_k,q_k,a)}(q'_k)$. Hence for every $((C'_1, ..., C'_n), (q'_1, ..., q'_n)) \in Q^n_{conf} \times Q_{\mathbf{A}_0}$ with $q'_1 = TS(C'_1), ..., q'_n = TS(C'_n)$, $\eta_{(\mathbf{A}_0,q_0,a)}((q'_1, ..., q'_n)) = (\eta_{(\mathbf{A}_1,q_1,a)} \otimes ... \otimes \eta_{(\mathbf{A}_n,q_n,a)}))((q'_1, ..., q'_n)) = (\eta_p^1 \otimes ... \otimes \eta_p^n((C'_1, ..., C'_n))$ (*).
By definition of η_p^0, $\forall (C'_0, q'_0) \in Q_{conf} \times Q_{\mathbf{A}_0}$, with $q'_0 = TS(C'_0)$, $\eta_{(\mathbf{A}_0,q_0,a)}(q'_0) = \eta_p^0(C'_0)$.

Since we deal with preserving distribution and $\mathbf{A}_0 = \bigcup_{k\in[1:n]} \mathbf{A}_k$, q'_0 is of the form $(q'_1, ..., q'_n)$ with $q'_k \in Q_{\mathbf{A}_k}$ and verifies $C'_0 = C'_1 \cup ... \cup C'_n$ with $auts(C'_k) = \mathbf{A}_k$ and $TS(C'_k) = q'_k$ (**). Hence we compose (*) and (**) to obtain for every configuration $C'_0 = (\mathbf{A}_0, \mathbf{S}'_0)$, for every finite set of configurations $\{C'_k = (\mathbf{A}_k, \mathbf{S}'_k)\}_{k\in[1:n]}$, s.t. $C'_0 = \bigcup_{k\in[1:n]} C'_k$, then $\eta^0_p(C'_0) = (\eta^1_p \otimes ... \otimes \eta^n_p)((C'_1, ..., C'_n))$.

◀

▶ **Definition 32** (merge). *Let $\tilde{\eta} = (\eta_1, ...\eta_n) \in Disc(Q_{conf})^n$. We define*

$$merge(\tilde{\eta}) : \begin{cases} Q_{conf} & \rightarrow & [0,1] \\ C & \mapsto & \sum_{(C'_1,...,C'_n)\in Q^n_{conf}} join(\tilde{\eta})((C'_1, ..., C'_n)) \cdot \mathbb{1}_{(C'_1\cup...\cup C'_n)=C} \end{cases}$$

▶ **Lemma 33** (Preserving-merging). *Let $n \in \mathbb{N}$, let $\{C_k\}_{k\in[1:n]}$ be a set of compatible configurations. Let $\tilde{\eta}_p = (\eta^1_p, ..., \eta^n_p) \in Disc(Q_{conf})^n$. Assume $\forall k \in [1 : n]$, if $a \in \widehat{sig}(C_k)$, then $C_k \xrightarrow{a} \eta^k_p$ and otherwise, $\eta^k_p = \delta_{C_k}$.*

Then, $\forall C'_0 \in supp(merge(\tilde{\eta}_p))$, it exists a unique $(C'_1, ..., C'_n)$, noted $split_{\tilde{\eta}}(C'_0)$, s.t. (a) $C'_0 = \bigcup_{k\in[1:n]} C'_k$ and (b) $\forall k \in [1, n], C'_k \in supp(\eta^k_p)$.

We note $split_{\tilde{\eta}} : \begin{cases} supp(merge(\tilde{\eta}_p)) & \rightarrow & supp(\eta^1_p) \times ... \times supp(\eta^n_p) \\ C'_0 & \mapsto & split_{\tilde{\eta}_p}(C'_0) \end{cases}$

Moreover, $merge(\tilde{\eta}_p) \overset{s}{\leftrightarrow} join(\tilde{\eta}_p)$ with $s = split_{\tilde{\eta}_p}$

Proof. (Uniqueness) Let us imagine two candidates $(C'_1, ..., C'_n)$ and $(C''_1, ..., C''_n)$ verifying both (a) and (b). Let $k, \ell \in [1 : n], k \neq \ell$. First, by compatibility of C_0, $\varphi_k \cap \varphi_\ell = \emptyset$. Hence $auts(C'_k) \cap auts(C''_\ell) = auts(C_k) \cap auts(C_\ell) = \emptyset$. Since $auts(\bigcup_{k\in[1:n]} C'_k) = auts(\bigcup_{k\in[1:n]} C'_k)$, $\forall k \in [1 : n]$, $auts(C'_k) = auts(C''_k)$. By equality, $\forall k \in [1 : n]$, $map(C'_k) = map(C''_k)$ and so $\forall k \in [1 : n]$, $C'_k = C''_k$. (Existence) By construction of $merge$. By uniqueness and existence properties, $s = split_{\tilde{\eta}_p}$ is then a bijection from $supp(merge(\tilde{\eta}_p))$ and $supp(\eta^1_p) \times ... \times supp(\eta^n_p)$. Let $C'_0 \in supp(merge(\tilde{\eta}_p))$. By definition $merge(\tilde{\eta}_p)(C'_0) = \sum_{(C'_1,...,C'_n)\in Q^n_{conf}} join(\tilde{\eta}_p)((C'_1, ..., C'_n)) \cdot \mathbb{1}_{(C'_1\cup...\cup C'_n)=C'_0}$. By bijectivity, $merge(\tilde{\eta}_p)(C'_0) = join(\tilde{\eta}_p)(split_{\tilde{\eta}_p}(C'_0))$. ◀

▶ **Definition 34** (deter-dest, base). *Let $C = (\mathbf{A}, \mathbf{S})$ be a configuration. For every $\mathcal{A} \in \mathbf{A}$, we note $q = \mathbf{S}(\mathcal{A})$. Let $\varphi \in \mathcal{P}(Autids)$. We define*

- *$deter\text{-}dest(C, a) = \{\mathcal{A} \in \mathbf{A} | \eta_{\mathcal{A},q_{\mathcal{A}},a} = \delta_{q^{\phi}_{\mathcal{A}}}\}$ if $a \in \widehat{sig}(\mathcal{A})(q)$ and $\emptyset$ otherwise. It represents the set of automata that will be deterministically destroyed.*
- *$base(C, a, \varphi) = \mathbf{A} \cup \varphi \setminus deter\text{-}dest(C, a)$. It represents the automata present in $supp(\eta)$ with $C \overset{a}{\Longrightarrow}_\varphi \eta$.*

▶ **Lemma 35** (Merging). *Let $n \in \mathbb{N}$, Let $(\varphi_1, ..., \varphi_n) \in \mathcal{P}(Autids)^n$ with $\forall k, \ell \in [1 : n]$, $\varphi_k \cap \varphi_\ell = \emptyset$. Let $\{C_k\}_{k\in[1:n]}$ be a set of compatible configurations. Let $\tilde{\eta} = (\eta_1, ..., \eta_n) \in Disc(Q_{conf})^n$. Assume $\forall k \in [1 : n]$, if $a \in \widehat{sig}(C_k)$, then $C_k \overset{a}{\Longrightarrow}_{\varphi_k} \eta^k$ and otherwise, $\eta^k = \delta_{C_k}$ and $\varphi_k = \emptyset$. We note $\varphi_0 = \bigcup_{k\in[1:n]} \varphi_k$ and $C_0 = \bigcup_{k\in[1:n]} C_k$.*

1. *Assume, $\forall k, \ell \in [1 : n], k \neq \ell, \varphi_k \cap auts(C_\ell) \subseteq deter\text{-}dest(C_\ell, a)$.*
   a. *$\forall C'_0 \in supp(merge(\tilde{\eta}))$, it exists a unique $(C'_1, ..., C'_n)$, noted $split_{\tilde{\eta}}(C'_0)$, s.t. (a) $C'_0 = \bigcup_{k\in[1:n]} C'_k$ and (b) $\forall k \in [1, n], C'_k \in supp(\eta_k)$.*
   We note $split_{\tilde{\eta}} : \begin{cases} supp(merge(\tilde{\eta})) & \rightarrow & supp(\eta_1) \times ... \times supp(\eta_n) \\ C'_0 & \mapsto & split_{\tilde{\eta}}(C'_0) \end{cases}$
   b. *$merge(\tilde{\eta}) \overset{s}{\leftrightarrow} join(\tilde{\eta})$ with $s = split_{\tilde{\eta}}$*
   c. *$merge(\tilde{\eta}) = reduce(merge(\tilde{\eta}_p) \uparrow \varphi_0)$.*
   d. *$C_0 \overset{a}{\Longrightarrow}_{\varphi_0} merge(\tilde{\eta})$ if $a \in \widehat{sig}(C_0)$ and $merge(\tilde{\eta}) = \delta_{C_0}$ otherwise.*

2. *Assume* $\forall C'_0 \in supp(merge(\tilde{\eta}))$, C'_0 *is compatible. Then,* $\forall k, \ell \in [1:n], k \neq \ell, \varphi_k \cap auts(C_\ell) \subseteq deter\text{-}dest(C_\ell, a)$.

Proof. **1.**

- **a.** Indeed, let us imagine two candidates $(C'_1, ..., C'_n)$ and $(C''_1, ..., C''_n)$ verifying both (a) and (b). Let $k, \ell \in [1:n], k \neq \ell$. By contradiction, let $\mathcal{A} \in auts(C'_k) \cap auts(C''_\ell)$. By compatibility, $\mathcal{A} \notin auts(C_k) \cap auts(C_\ell)$. W.l.o.g., $\mathcal{A} \in \varphi_k \cap auts(C_\ell)$. By assumption $\mathcal{A} \in deter\text{-}dest(C_\ell, a)$ and so $mathcal A \notin auts(C''_\ell)$ which leads to a contradiction. Hence, $\forall k \in [1:n]$, $auts(C'_k) = auts(C''_k)$. Since $auts(\bigcup_{k\in[1:n]} C'_k) = auts(\bigcup_{k\in[1:n]} C'_k)$, $\forall k \in [1:n]$, $auts(C'_k) = auts(C''_k)$. By equality, $\forall k \in [1:n]$, $map(C'_k) = map(C''_k)$ and so $\forall k \in [1:n]$, $C'_k = C''_k$. The existence is by construction of *join*.
- **b.** The fact that $s = split_{\tilde{\eta}}$ is a bijection from $supp(merge(\tilde{\eta}))$ and $supp(\eta_1) \times ... \times supp(\eta_1)$ comes from the existence and the uniqueness of pre-image proved in item 1a. Let $C'_0 \in supp(merge(\tilde{\eta}))$. By definition $merge(\tilde{\eta})(C'_0) = \sum_{(C'_1,...,C'_n)\in Q^n_{conf}} join(\tilde{\eta})((C'_1, ..., C'_n)) \cdot \mathbb{1}_{(C'_1 \cup ... \cup C'_n) = C'_0}$. By bijectivity, $merge(\tilde{\eta})(C'_0) = join(\tilde{\eta})(split_{\tilde{\eta}}(C'_0))$.
- **c.** We want to show that $merge(\tilde{\eta}) \triangleq merge((reduce(\eta^1_p \uparrow \varphi_1), ..., (reduce(\eta^n_p \uparrow \varphi_n)) = reduce(merge(\tilde{\eta}_p) \uparrow \bigcup_{k\in[1:n]} \varphi_k) \triangleq reduce(merge(\tilde{\eta}_p) \uparrow \varphi_0)$. Intuitively, it comes from 1b that gives $merge(\tilde{\eta}) \overset{s}{\leftrightarrow} join(\tilde{\eta})$ with $s = split_{\tilde{\eta}}$ and $\forall k \in [1:n]$, $\eta^k = reduce(\eta^k_p \uparrow \varphi_k)$, with $\forall k, \ell \in [1:n], k \neq \ell$, $\varphi_k \cap \varphi_\ell = \emptyset$. Let us elaborate.

  Let $C'_0 \in supp(merge(\tilde{\eta}))$. $merge(\tilde{\eta})(C'_0) = join(\tilde{\eta})(split_{\tilde{\eta}}(C'_0))$ by 1b.

  Hence, $merge(\tilde{\eta})(C'_0) = \Pi_{k\in[1:n]}(reduce(\eta^k_p \uparrow \varphi_k)(C'_k)$ with $split_{\tilde{\eta}}(C'_0) = (C'_1, ..., C'_n)$.

  Thus, for every $k \in [1, n]$, $C'_k = (\mathbf{A}'_k, \mathbf{S}'_k)$ with (i) $\mathbf{A}'_k = \mathbf{A}''_k \cup \varphi_k$, (ii) $\forall \mathcal{A} \in \varphi_k, \mathbf{S}'_k(\mathcal{A}) = \bar{q}_\mathcal{A}$ (iii) $\forall \mathcal{A} \in \mathbf{A}'_k, \mathbf{S}'_k(\mathcal{A}) \neq q^\phi_\mathcal{A}$ (*). This leads to $merge(\tilde{\eta})(C'_0) = \Pi_{k\in[1:n]}(reduce(\eta^k_p))(C''_k)$ with $C''_k = (\mathbf{A}''_k, \mathbf{S}''_k)$ where $\mathbf{S}''_k = \mathbf{S}'_k \upharpoonright \mathbf{A}''_k$.

  Hence, $merge(\tilde{\eta})(C'_0) = \Pi_{k\in[1:n]}(\sum_{C''_{k,\ell}, reduce(C''_{k,\ell}) = C''_k} \eta^k_p(C''_{k,\ell}))$ where every $C''_{k,\ell} = (\mathbf{A}''_{k,\ell}, \mathbf{S}''_{k,\ell}) \in supp(\eta^k_p)$ with $reduce(C''_{k,\ell}) = C''_k$ verifies $\mathbf{A}''_{k,\ell} = \mathbf{A}_k$ and $\mathbf{S}''_{k,\ell} \upharpoonright \mathbf{A}''_k = \mathbf{S}''_k$ (**).

  Second, for every $k \in [1:n]$, we note $\mathbf{A}^d_k = deter\text{-}dest(C_k, a)$, $\eta^k_{p,d}$ the unique preserving distribution such that $\eta^k_p \overset{dest^k}{\leftrightarrow} \eta^k_{p,d}$ with $dest^k : (\mathbf{A}'_k, \mathbf{S}'_k) \mapsto (\mathbf{A}'_k \setminus \mathbf{A}^d_k, \mathbf{S}'_k \upharpoonright (\mathbf{A}'_k \setminus \mathbf{A}^d_k))$ and we note $\eta^k_{p,d,\uparrow} = \eta^k_{p,d} \uparrow \varphi_k$. We note $\tilde{\eta}_{p,d,\uparrow} = (\eta^1_{p,d,\uparrow}, ..., \eta^n_{p,d,\uparrow})$. Clearly, $(reduce(merge(\tilde{\eta}_p) \uparrow \varphi_0)) = (reduce(merge(\tilde{\eta}_{p,d,\uparrow}))$.

  $(reduce(merge(\tilde{\eta}_{p,d,\uparrow}))(C'_0) = \sum_{C'_{0,d,\ell}, reduce(C'_{0,d,\ell}) = C'_0} (merge(\tilde{\eta}_{p,d,\uparrow}))(C'_{0,d,\ell})$, where every $C'_{0,d,\ell} = (\mathbf{A}'_{0,d,\ell}, \mathbf{S}'_{0,d,\ell}) \in supp((merge(\tilde{\eta}_{p,d,\uparrow}))$ with $reduce(C'_{0,d,\ell}) = C'_0$ verifies $\mathbf{A}'_{0,\ell} = \mathbf{A}_0 \setminus \bigcup_{k[1:n]} \mathbf{A}^d_k$ and $\mathbf{S}'_{0,d,\ell} \upharpoonright \mathbf{A}'_0 = \mathbf{S}'_0$.

  By lemma 33, for each ℓ, $(merge(\tilde{\eta}_{p,d,\uparrow}))(C'_{0,d,\ell}) = split_{\tilde{\eta}_{p,d,\uparrow}}(C'_{0,d,\ell}) = \Pi_{k\in[1:n]}\eta^k_{p,d,\uparrow}(C''_{k,d,\ell})$, with $split_{\tilde{\eta}_{p,d,\uparrow}}(C'_{0,d,\ell}) \triangleq (C'_{1,d,\ell}, ..., C'_{n,d,\ell})$.

  Moreover, every $C'_{k,d,\ell} \triangleq (\mathbf{A}'_{k,d,\ell}, \mathbf{S}'_{k,d,\ell}) \in supp(\eta^k_{p,d} \uparrow \varphi_k))$ with $reduce(C'_{k,d,\ell}) = C'_{k,d}$, $\mathbf{A}'_{k,d,\ell} = (\mathbf{A}_k \setminus \mathbf{A}^d_k) \cup \varphi_k$, $\mathbf{S}'_{k,d,\ell} \upharpoonright \mathbf{A}'_k = \mathbf{S}'_k$. We obtain $(reduce(merge(\tilde{\eta}_{p,d,\uparrow}))(C'_0) = \sum_{C'_{0,d,\ell}, reduce(C'_{0,d,\ell}) = C'_0} (join(\tilde{\eta}_{p,d,\uparrow})(split_{\tilde{\eta}_{p,d,\uparrow}}(C'_{0,d,\ell})))$ and so

  $(reduce(merge(\tilde{\eta}_{p,d,\uparrow}))(C'_0) = \sum_{C'_{0,d,\ell}, reduce(C'_{0,d,\ell}) = C'_0} (\Pi_{k\in[1:n]}(\eta^k_{p,d,\uparrow})(C'_{k,d,\ell}))$ (***).

  Clearly, for every $k \in [1:n]$, $(\eta^k_p \uparrow \varphi_k) \overset{dest^k}{\leftrightarrow} \eta^k_{p,d,\uparrow}$.

  Combined with (**) and (***), we find $merge(\tilde{\eta})(C'_0) = (reduce(merge(\tilde{\eta}_p) \uparrow \varphi))(C'_0)$ for every $C'_0 \in supp(merge(\tilde{\eta}))$, which ends the proof.

d. If $a \notin \widehat{sig}(C_0)$, the result is trivial. Assume $a \in \widehat{sig}(C_0)$ Let $\tilde{\eta}_p = (\eta_p^1, ..., \eta_p^n) \in Disc(Q_{conf})^n$ s.t. $\forall k \in [1:n]$, $C_k \xrightarrow{a} \eta_p^k$ if $a \in \widehat{sig}(C_k)$ and $\eta_p^k = \delta_{C_k}$ otherwise. For every $k \in [1:n]$, $\eta^k = reduce(\eta_p^1 \uparrow \varphi_k)$. By compatibility of C_0, for every $k, \ell \in [1,n], k \neq \ell$, $\mathbf{A}_k^p \cap \mathbf{A}_\ell^p = \emptyset$. Hence, we can apply lemma 31 and we have $C_0 \xrightarrow{a} merge(\tilde{\eta}_p)$. Thus, $C_0 \overset{a}{\Longrightarrow}_{\varphi_0} reduce(merge(\tilde{\eta}_p) \uparrow \varphi_0)$. Finally, $merge(\tilde{\eta}) = reduce(merge(\tilde{\eta}_p) \uparrow \varphi_0)$ by 1c.

2. By contradiction. W.l.o.g., let us assume $\mathcal{A} \in \varphi_k \cap auts(C_\ell) \setminus deter\text{-}dest(C_\ell, a)$. Since C is compatible, $\mathcal{A} \notin \mathbf{A}_k \cap \mathbf{A}_\ell$. By definition of *deter-dest* it exists $(C'_k, C'_\ell) \in supp(\eta_k) \times supp(\eta_\ell)$, $\mathcal{A} \in auts(C'_k) \cap auts(C'_\ell)$ and $C'_k \cup C'_\ell$ is not compatible. So it exists $(C'_1, ..., C'_n) \in supp(\eta_1 \otimes ... \otimes \eta_n)$ s.t. $(C'_1 \cup ... \cup C'_n)$ is not compatible.

◀

trivial results about homomorphisms between probability measures

▶ **Lemma 36.** *Let $(\eta_1, \eta_2, \eta_3) \in Disc(Q_1) \times Disc(Q_2) \times Disc(Q_3)$, with Q_i being a set for each $i \in \{1,2,3\}$. Let $f : Q_1 \rightharpoonup Q_2$ and $g : Q_1 \rightharpoonup Q_2$ defined on $supp(\eta_1)$ and $supp(\eta_2)$ respectively. Let $\tilde{f}$ (resp. $\tilde{g}$) denotes the restriction of f (resp. g) on $supp(\eta_1)$ (resp. $supp(\eta_2)$).*

If $\eta_1 \overset{f}{\leftrightarrow} \eta_2$ and $\eta_2 \overset{g}{\leftrightarrow} \eta_3$, then

1. *$\eta_1 \overset{h}{\leftrightarrow} \eta_3$ where the restriction $\tilde{h}$ of h on $supp(\eta_1)$ verifies $\tilde{h} = \tilde{g} \circ \tilde{f}$ and*

2. *$\eta_2 \overset{k}{\leftrightarrow} \eta_1$ where the restriction $\tilde{k}$ of k to $supp(\eta_2)$ verifies $\tilde{k} = \tilde{f}^{-1}$.*

Proof.

- (bijectivity) The composition of two bijection is a bijection and the reverse function of a bijection is a bijection.
- (measure preservation) In the first case, $\forall q \in supp(\eta_1), \eta_1(q) = \eta_2(f(q))$ with $f(q) \in supp(\eta_2)$ which means $\eta_2(f(q)) = \eta_3(g(f(q)))$. In the second case $\forall q' \in supp(\eta_2), \exists! q \in supp(\eta_1), \eta_1(q) = \eta_2(q' = \tilde{f}(q))$ and hence $\forall q' \in supp(\eta_2), \eta_2(q') = \eta_1(q = \tilde{f}^{-1}(q'))$.

◀

▶ **Lemma 37** (correspondence preservation for joint probability)**.** *Let $\tilde{\eta} = (\eta_1, ..., \eta_n) \in Disc(Q_1) \times ... \times Disc(Q_n)$, $\tilde{\eta}' = (\eta'_1, ..., \eta'_n) \in Disc(Q'_1) \times ... \times Disc(Q'_n)$ with each Q_i (resp. Q'_i) being a set. For each $i \in [1:n]$, let $f_i : Q_i \rightharpoonup Q'_i$, where $dom(f_i) \subseteq supp(\eta_i)$, with $\eta_i \overset{f_i}{\leftrightarrow} \eta'_i$.*

Then $join(\tilde{\eta}) \overset{f}{\leftrightarrow} join(\tilde{\eta}')$ with $f : \begin{cases} Q_1 \times ... \times Q_n & \rightharpoonup & range(f_1) \times ... \times range(f_n) \\ (x_1, ..., x_n) & \mapsto & (f_1(x_1), ..., f_n(x_n)) \end{cases}$.

Proof. The restriction $\tilde{f}$ of f on $supp(join(\tilde{\eta})) = supp(\eta_1) \times ... \times supp(\eta_n)$ is still a bijection and $\forall x = (x_1, ..., x_n) \in dom(f_1) \times ... \times dom(f_n)$, $join(\tilde{\eta})(x) = \eta_1(x_1) \cdot ... \cdot \eta_n(x_n) = \eta'_1(f_1(x_1)) \cdot ... \cdot \eta'_n(f_n(x_n)) = join(\tilde{\eta}')(f(x_1, ..., x_n))$. ◀

PCA closeness under composition

Now we are ready for the theorem that claims that a composition of PCA is a PCA.

▶ **Theorem 38** (PCA closeness under composition)**.** *Let $X_1, ..., X_n$, be partially-compatible PCA. Then $X = X_1 || ... || X_n$ is a PCA.*

Proof. We need to show that X verifies all the constraints of definition 19.

- (Constraint) 1: The demonstration is the same as the one in [2], section 5.1, proposition 21, p 32-33. Let $\bar{q}_X$ and $(\mathbf{A}, \mathbf{S}) = config(X)(\bar{q}_X)$. By the composition of psioa, then $\bar{q}_X = (\bar{q}_{X_1}, ..., \bar{q}_{X_n})$. By definition, $config(X)(\bar{q}_X) = config(X_1)(\bar{q}_{X_1}) \cup ... \cup config(X_n)(\bar{q}_{X_n})$. Since for every $j \in [1:n]$, X_j is a configuration automaton, we apply constraint 1 to X_j to conclude $\mathbf{S}(\mathcal{A}_\ell) = \bar{q}_{\mathcal{A}_\ell}$ for every $\mathcal{A}_\ell \in auts(config(X_j)(\bar{q}_{X_j})$. Since $(auts(config(X_1)(\bar{q}_{X_1}), ..., auts(config(X_n)(\bar{q}_{X_n}))$ is a partition of $\mathbf{A}$ by definition of composition, $\mathbf{S}(\mathcal{A}_\ell) = \bar{q}_{\mathcal{A}_\ell}$ for every $\mathcal{A}_\ell \in \mathbf{A}$ which ensures X verifies constraint 1.
- (Constraint 2)
  Let $(q, a, \eta_{(X,q,a)}) \in D_X$. We will establish $\exists \eta' \in Disc(Q_{conf})$ s.t. $\eta_{(X,q,a)} \overset{c}{\leftrightarrow} \eta'$ where $c = config(X)$ and $config(X)(q) \overset{a}{\Longrightarrow}_\varphi \eta'$ with $\varphi = created(X)(q)(a)$.
  For brevity, let $P_i = psioa(X_i)$ for every $i \in [1:n]$. By definition 29 of PCA compositon, $psioa(X) = psioa(X_1)||...||psioa(X_n) = P_1||...||P_n$. By definition 26 of PSIOA composition, $q = (q_1, ..., q_n) \in Q_{P_1} \times ... \times Q_{P_n}$, while $a \in \bigcup_{i\in[1:n]} \widehat{sig}(P_i)(q_i)$ and $\eta_{X,q,a} = \eta_{P_1,q_1,a} \otimes ... \otimes \eta_{P_n,q_n,a}$ with the convention $\eta_{P_i,q_i,a} = \delta_{q_i}$ if $a \notin \widehat{sig}(P_i)(q_i)$.
  Let $(\mathcal{I}, \mathcal{J})$ be a partition of $[1:n]$ s.t. $\forall i \in \mathcal{I}$, $a \in \widehat{sig}(P_i)(q_i)$ and $\forall j \in \mathcal{J}$, $a \notin \widehat{sig}(P_j)(q_j)$. Then by PCA top/down transition preservation, it exists $\eta'_i \in Disc(Q_{conf})$ s. t. $\eta_{X_i,q_i,a} = \eta_{P_i,q_i,a} \overset{c_i}{\leftrightarrow} \eta'_i$ with $c_i = config(X_i)$ and $config(X_i)(q_i) \overset{a}{\Longrightarrow}_{\varphi_i} \eta'_i$ with $\varphi_i = created(X_i)(q_i)(a)$. For every $j \in \mathcal{J}$, we note $\varphi_j = \emptyset$ and $\eta'_j = \delta_{config(X_j)(q_j)}$ that verifies $\delta_{q_j} \overset{c_j}{\leftrightarrow} \eta'_j$ with $c_j = config(X_j)$.
  We note $\tilde{\eta}' = (\eta'_1, ..., \eta'_n)$ and $\varphi = \bigcup_{i\in[1:n]} \varphi_i$. By definition 29 of PCA composition, $\varphi = created(X)(q)(a)$.
  We have $\eta_{X,q,a} \overset{c'}{\leftrightarrow} \eta'$ with $c' : q = (q_1, ..., q_n) \mapsto (c_1(q_1), ..., c_n(q_n))$ by lemma 37.
  Moreover $merge(\tilde{\eta}') \overset{s}{\leftrightarrow} join(\tilde{\eta}')$ with $s = split_{\tilde{\eta}}$ by lemma 35, item 1b.
  So $\eta_{X,q,a} \overset{c}{\leftrightarrow} merge(\tilde{\eta}')$ with $c = s^{-1} \circ c' = config(X)$.
  Moreover we have $config(X)(q) \overset{a}{\Longrightarrow}_\varphi merge(\tilde{\eta}')$ by lemma 35, item 1d.
- (Constraint 3)
  Let $q \in Q_X$, $C = config(X)(q)$, $a \in \widehat{sig}(X)(q)$, $\varphi = created(X)(q)(a)$ that verify $C \overset{a}{\Longrightarrow}_\varphi \eta'$. We need to show that it exists $(q, a, \eta_{(X,q,a)}) \in D_X$ s.t. $\eta_{(X,q,a)} \overset{c}{\leftrightarrow} \eta'$ with $c = config(X)$.
  For brevity, let $P_i = psioa(X_i)$ for every $i \in [1:n]$. By definition 29 of PCA composition $psioa(X) = psioa(X_1)||...||psioa(X_n) = P_1||...||P_n$. By definition 26 of PSIOA composition, $q = (q_1, ..., q_n) \in Q_{P_1} \times ... \times Q_{P_n}$, while $a \in \bigcup_{i\in[1:n]} \widehat{sig}(P_i)(q_i)$.
  Let $(\mathcal{I}, \mathcal{J})$ be a partition $[1:n]$ s.t. $\forall i \in \mathcal{I}$, $a \in \widehat{sig}(P_i)(q_i)$ and $\forall j \in \mathcal{J}$, $a \notin \widehat{sig}(P_j)(q_j)$.
  For every $i \in \mathcal{I}$, we note $\varphi_i = created(X_i)(q_i)(a)$,while for every $j \in \mathcal{J}$, we note $\varphi_j = \emptyset$ and $\eta'_j = \delta_{config(X_j)(q_j)}$ that verifies $\delta_{q_j} \overset{c_j}{\leftrightarrow} \eta'_j$ with $c_j = config(X_j)$.
  We note $\varphi = created(X)(q)(a)$. By pca-composition definition, $\varphi = \bigcup_{k\in[1:n]} \varphi_k$. For every $k \in [1:n]$, we note $C_k = config(X_k)(q_k)$ and for every $i \in \mathcal{I}$, $\eta'_i \in Disc(Q_{conf})$ s.t. $C_i \overset{a}{\Longrightarrow}_{\varphi_i} \eta'_i$. We note $\tilde{\eta}' = (\eta'_1, ..., \eta'_n)$
  By constraint 3 (bottom/up transition preservation), $\forall i \in \mathcal{I}$, $\exists (q_i, a, \eta_{X_i,q_i,a}) \in D_{X_i}$ s.t. $\eta_{X_i,q_i,a} \overset{c_i}{\leftrightarrow} \eta'_i$ with $c_i = config(X_i)$. by lemma 37, $\eta_{X,q,a} = \eta_{X_1,q_1,a} \otimes ... \otimes \eta_{X_n,q_n,a} \overset{c'}{\leftrightarrow} \eta'_1 \otimes ... \otimes \eta'_n = join(\tilde{\eta}')$ with the convention $\eta_{X_j,q_j,a} = \delta_{q_j}$ for $j \in \mathcal{J}$ and $c' : q = (q_1, ..., q_n) \in states(X) \mapsto (c_1(q_1), ..., c_n(q_n))$.
  By partial-compatibility, for every $C' \in supp(merge(\tilde{\eta}'))$, C' is compatible. Hence we can apply lemma 35, item 1b, which gives $merge(\tilde{\eta}') \overset{s}{\leftrightarrow} join(\tilde{\eta}')$ with $s = split_{\tilde{\eta}'}$. Hence $\eta_{X,q,a} \overset{c''}{\leftrightarrow} merge(\tilde{\eta}')$ with $c'' = s^{-1} \circ c'$, that is $\eta_{X,q,a} \overset{c}{\leftrightarrow} \eta'$ with $c = config(X)$ and the restriction of c'' on $supp(\eta_{X,q,a})$ is c. We can apply lemma 35 again, but for item 1d, which gives $C \overset{a}{\Longrightarrow}_\varphi merge(\tilde{\eta}')$.

- (Constraint 4).
  Let $q = (q_1, ..., q_n) \in Q_X$. For every $i \in [1, n]$, we note $h_i = \mathit{hidden\text{-}actions}(X_i)(q_i)$, $C_i = \mathit{config}(X_i)(q_i)$, $h = \bigcup_{i\in[1,n]} h_i$ and $C = \mathit{config}(X)(q)$. Since $X_1, ..., X_n$ are compatible at state q, we have both $\{C_i | i \in [1, n]\}$ compatible and $\forall i, j \in [1, n], in(C_i) \cap h_j = \emptyset$. By compatibility, $\forall i, j \in [1, n], i \neq j, out(C_i) \cap out(C_j) = int(C_i) \cap \widehat{sig}(C_j) = \emptyset$, which finally gives $\forall i, j \in [1, n], i \neq j$, $\widehat{sig}(C_i) \cap h_j = \emptyset$.
  Hence, we can apply lemma 6 of commutativity between hiding and composition to obtain $hide(sig(C_1) \times .... \times sig(C_n), h_1 \cup ... \cup h_n) = hide(sig(C_1), h_1) \times ... \times hide(sig(C_n), h_n)$ where $\times$ has to be understood in the sense of definition 3 of signature composition.
  That is $sig(psioa(X))(q) = sig(psioa(X_1))(q_1)) \times .... \times sig(psioa(X_n))(q_n))$, as per definition 3, with $sig(psioa(X))(q) = hide(sig(config(X)(x)), h)$. Furthermore $h \subseteq out(config(X)(q))$, since $\forall i \in [1, n]$, $h_i \subseteq out(C_i)$. This terminates the proof.

◀

8 Scheduler, measure on executions, implementation

An inherent non-determinism appears for concurrent systems. Indeed, after composition (or even before), it is natural to obtain a state with several enabled actions. The most common case is the reception of two concurrent messages in flight from two different processes. This non-determinism must be solved if we want to define a probability measure on the automata executions and be able to say that a situation is likely to occur or not. To solve the non-determinism, we use a scheduler that chooses an enabled action from a signature.

8.1 General definition and probabilistic space $(\mathit{Frags}(\mathcal{A}), \mathcal{F}_{\mathit{Frags}(\mathcal{A})}, \epsilon_{\sigma,\mu})$

A scheduler is hence a function that takes an execution fragment as input and outputs the probability distribution on the set of transitions that will be triggered. We reuse the formalism from [20] with the syntax from [3].

▶ **Definition 39** (scheduler). *A* scheduler *of a PSIOA (resp. PCA) $\mathcal{A}$ is a function $\sigma : \mathit{Frags}^*(\mathcal{A}) \to \mathit{SubDisc}(D_{\mathcal{A}})$ such that $(q, a, \eta) \in supp(\sigma(\alpha))$ implies $q = lstate(\alpha)$. Here $\mathit{SubDisc}(D_{\mathcal{A}})$ is the set of discrete sub-probability distributions on $D_{\mathcal{A}}$. Loosely speaking, σ decides (probabilistically) which transition to take after each finite execution fragment α. Since this decision is a discrete sub-probability measure, it may be the case that σ chooses to halt after α with non-zero probability: $1 - \sigma(\alpha)(D_{\mathcal{A}}) > 0$. We note $schedulers(\mathcal{A})$ the set of schedulers of $\mathcal{A}$.*

▶ **Definition 40** (measure $\epsilon_{\sigma,\alpha}$ generated by a scheduler and a fragment). *A scheduler σ and a finite execution fragment α generate a measure $\epsilon_{\sigma,\alpha}$ on the sigma-algebra $\mathcal{F}_{\mathit{Frags}(\mathcal{A})}$ generated by cones of execution fragments, where each cone $C_{\alpha'}$ is the set of execution fragments that have α' as a prefix, i.e. $C_{\alpha'} = \{\alpha \in \mathit{Frags}(\mathcal{A}) | \alpha' \leq \alpha\}$. The measure of a cone $C_{\alpha'}$ is defined recursively as follows:*

$$\epsilon_{\sigma,\alpha}(C_{\alpha'}) =: \begin{cases} 0 & \text{if both } \alpha' \nleq \alpha \text{ and } \alpha \nleq \alpha' \\ 1 & \text{if } \alpha' \leq \alpha \\ \epsilon_{\sigma,\alpha}(C_{\alpha''}) \cdot \sigma(\alpha'')(\eta_{(\mathcal{A},q',a)}) \cdot \eta_{(\mathcal{A},q',a)}(q) & \text{if } \alpha \leq \alpha'' \text{ and } \alpha' = \alpha'' \frown q'aq \end{cases}$$

Standard measure theoretic arguments [20] ensure that $\epsilon_{\sigma,\alpha}$ is well-defined. The proof of [20] (terminating with theorem 4.2.10, section 4.2) is very general and might appear discouraging for a brief reading. For sake of completeness, we adapt the proof of [20] to the formalism of [3][5].

First, for every set $\mathcal{C}$ of subset of a set Ω, we define $F_1(\mathcal{C})$, $F_2(\mathcal{C})$, $F_3(\mathcal{C})$, $\mathcal{F}_\Omega$ as follows:

- Let $F_1(\mathcal{C})$ be the be the family containing $\emptyset$, Ω, and all $C \subseteq \Omega$ such that either $C \in \mathcal{C}$ or $\Omega \setminus C \in \mathcal{C}$.
- $F_2(\mathcal{C})$ is the family containing all finite intersections of elements of $F_1(\mathcal{C})$.
- $F_3(\mathcal{C})$ is the family containing all finite unions of disjoint elements of $F_2(\mathcal{C})$.
- Clearly, $F_3(\mathcal{C})$ is a ring ("field" in [20]; a ring is also a semi-ring, which is enough to apply extension theorem [15]) on Ω, i.e. it is a family of subsets of Ω that contains Ω, and that is closed under complementation and finite union. When Ω is clear in the context, we say $F_3(\mathcal{C})$ is the ring generated by $\mathcal{C}$.

  $\mathcal{F}_\Omega$ is defined as the smallest sigma-algebra containing $F_3(\mathcal{C})$. (This is also the smallest sigma-algebra on Ω containg $\mathcal{C}$). We say $\mathcal{F}_\Omega$ is the sigma-algebra generated by $\mathcal{C}$. If μ is a measure on $F_3(\mathcal{C})$, by famous Carathéodory's extension theorem [7], there exists a unique extension μ' of μ to the sigma-algebra $\mathcal{F}_\Omega$, defining $\mu'(\biguplus_{k\in\mathbb{N}} E_k) \triangleq \sum_{k\in\mathbb{N}} \mu(E_k)$.

Let $\mathcal{C} = \{C_{\alpha'} | \alpha' \in \mathit{Frags}(\mathcal{A})\}$ be the set of cones. Clearly, $\mathcal{C}$ is a set of subsets of $\mathit{Frags}(\mathcal{A})$. As mentioned earlier, we define $\mathcal{F}_{\mathit{Frags}(\mathcal{A})}$ as the sigma-algebra on $\mathit{Frags}(\mathcal{A})$ generated by $\mathcal{C}$.

Also, for every pair of execution fragments α_1 and α_2, if α_1 and α_2 are non-comparable, then $C_{\alpha_1} \cup C_{\alpha_2}$ is not a cone, while if α_1 and α_2 are comparable, C_{α_1} and C_{α_2} are not disjoint. Hence, sigma-additivity is trivially ensured by $\epsilon_{\sigma,\alpha}$ on $\mathcal{C}$. Now, let us generate the appropriate sigma-algebra $\mathcal{F}_{\mathit{Frags}(\mathcal{A})}$ on $\mathit{Frags}(\mathcal{A})$ and let us extend $\epsilon_{\sigma,\alpha}$ to $\mathcal{F}_{\mathit{Frags}(\mathcal{A})}$.

- Let $F_1(\mathcal{C})$ be the be the family containing $\emptyset$, $\mathit{Frags}(\mathcal{A})$, and all $C \subseteq \mathit{Frags}(\mathcal{A})$ such that either $C \in \mathcal{C}$ or $\mathit{Frags}(\mathcal{A}) \setminus C \in \mathcal{C}$.

  There exists a unique extension $\epsilon^i_{\sigma,\alpha}$ of $\epsilon_{\sigma,\alpha}$ to $F_1(\mathcal{C})$. Indeed, there is a unique way to extend the measure of the cones to their complements since for each α', $\epsilon^i_{\sigma,\alpha}(C_{\alpha'}) + \epsilon^i_{\sigma,\alpha}(\mathit{Frags}(\mathcal{A}) \setminus C_{\alpha'}) = 1$. Therefore $\epsilon^i_{\sigma,\alpha}$ coincides with $\epsilon_{\sigma,\alpha}$ on the cones and $\epsilon^i_{\sigma,\alpha}$ is defined to be $1 - \epsilon^i_{\sigma,\alpha}(C_\alpha)$ for the complement of any cone C_α . By countably branching structure of $\mathit{Frags}(\mathcal{A})$ ($Q_\mathcal{A}$ and $\mathit{acts}(\mathcal{A})$ are both countable), the complement of a cone is a countable union of cones. Indeed, let $\alpha' \in \mathit{Frags}^*(\mathcal{A}), C_{\alpha'} \in \mathcal{C}$, then $\mathit{Frags}(\mathcal{A}) \setminus C_{\alpha'} = \bigcup_{\alpha'' \in \mathit{Frags}^*(\mathcal{A}), \alpha'' \nleq \alpha' \wedge \alpha' \nleq \alpha''} C_{\alpha''}$. Hence, σ-additivity is preserved.
- Let $F_2(\mathcal{C})$ be the family containing all finite intersections of elements of $F_1(\mathcal{C})$. There exists a unique extension $\epsilon^{ii}_{\sigma,\alpha}$ of $\epsilon^i_{\sigma,\alpha}$ to $F_2(\mathcal{C})$. Indeed, let us fix a pair of execution fragments α_1 and α_2, if α_1 and α_2 are non-comparable, then $C_{\alpha_1} \cap C_{\alpha_2} = \emptyset$ is not a cone, while if α_1 and α_2 are comparable, let say $\alpha_1 \leq \alpha_2$, then $C_{\alpha_1} \cap C_{\alpha_2} = C_{\alpha_2}$. Thus, intersection of finitely many sets of $F_1(\mathcal{C})$ is a countable union of cones. Therefore σ-additivity enforces a unique measure on the new sets of $F_1(\mathcal{C})$.
- Let $F_3(\mathcal{C})$ be the family containing all finite unions of disjoint elements of $F_2(\mathcal{C})$.

  There exists a unique extension $\epsilon^{iii}_{\sigma,\alpha}$ of $\epsilon^{ii}_{\sigma,\alpha}$ to $F_2(\mathcal{C})$. Indeed, there is a unique way of assigning a measure to the finite union of disjoint sets whose measure is known, i.e., adding up their measures. Since all the sets of $F_3(\mathcal{C})$ are countable unions of cones, σ-additivity is preserved.
- Clearly, $F_3(\mathcal{C})$ is a ring ("field" in [20]) on $\mathit{Frags}(\mathcal{A})$, i.e. it is a family of subsets of $\mathit{Frags}(\mathcal{A})$ that contains $\mathit{Frags}(\mathcal{A})$, and that is closed under complementation and finite union. $\mathcal{F}_{\mathit{Frags}(\mathcal{A})}$ is defined as the smallest sigma-algebra containing $F_3(\mathcal{C})$. (This is

[5] We are not aware of such an adaptation in the literature. This concise presentation might have its own pedagogical interest

also the smallest σ-algebra containg $\mathcal{C}$). By famous Carathéodory's extension theorem [7], there exists a unique extension $\epsilon^{iv}_{\sigma,\alpha}$ of $\epsilon^{iii}_{\sigma,\alpha}$ to the sigma-algebra $\mathcal{F}_{Frags(\mathcal{A})}$, defining $\epsilon^{iv}_{\sigma,\alpha}(\biguplus_{k\in\mathbb{N}} E_k) = \sum_{k\in\mathbb{N}} \epsilon^{iii}_{\sigma,\alpha}(E_k)$.
We can remark that $\forall \alpha' \in Frags^*(\mathcal{A}), \{\alpha'\} = C_{\alpha'} \setminus (\bigcup_{\alpha''\in Frags^*(\mathcal{A}),\alpha'<\alpha''} C_{\alpha''})$. In the same way, $\forall \alpha' \in Frags^{\omega}(\mathcal{A}), \{\alpha'\} = Frags(\mathcal{A}) \setminus (\bigcup_{i\in\mathbb{N}} \bigcup_{\alpha''\in Frags^*(\mathcal{A}),\alpha'|_i<\alpha'',\alpha'|_{i+1}\neq\alpha''|_{i+1}} C_{\alpha''})$. Hence $\forall \alpha' \in Frags(\mathcal{A}), \{\alpha'\} \in \mathcal{F}_{Frags(\mathcal{A})}$. Necessarily, we have $\forall \alpha' \in Frags^{\omega}(\mathcal{A}), \epsilon^{iv}_{\sigma,\alpha}(\alpha') = lim_{i\to\infty}\epsilon^{iv}_{\sigma,\alpha}(\alpha'|_i)$. Let us note that the limit is well-defined, since $\forall i \in \mathbb{N}$, (1) $\epsilon^{iv}_{\sigma,\alpha}(\alpha'|_{i+1}) \leq \epsilon^{iv}_{\sigma,\alpha}(\alpha'|_i)$ and (2) $\epsilon^{iv}_{\sigma,\alpha}(\alpha'|_i) \geq 0$. In the remaining, we abuse the notation and use $\epsilon_{\sigma,\alpha}$ to denotes its extension $\epsilon^{iv}_{\sigma,\alpha}$ on $\mathcal{F}_{Frags(\mathcal{A})}$.

We call the state $fstate(\alpha)$ the first state of $\epsilon_{\sigma,\alpha}$ and denote it by $fstate(\epsilon_{\sigma,\alpha})$. If α consists of the start state $\bar{q}_{\mathcal{A}}$ only, we call $\epsilon_{\sigma,\alpha}$ a probabilistic execution of $\mathcal{A}$. Let μ be a discrete probability measure over $Frags^*(\mathcal{A})$. We denote by $\epsilon_{\sigma,\mu}$ the measure $\sum_{\alpha\in supp(\mu)} \mu(\alpha)\cdot\epsilon_{\sigma,\alpha}$ and we say that $\epsilon_{\sigma,\mu}$ is generated by σ and μ. We call the measure $\epsilon_{\sigma,\mu}$ a generalized probabilistic execution fragment of $\mathcal{A}$. If every execution fragment in $supp(\mu)$ consists of a single state, then we call $\epsilon_{\sigma,\mu}$ a probabilistic execution fragment of $\mathcal{A}$.

The collection $F(\mathcal{C}_{Execs(\mathcal{A})})$ of sets obtained by taking the intersection of each element in $F_3(\mathcal{C})$ with $Execs(\mathcal{A})$ is a ring in $Execs(\mathcal{A})$. We note $\mathcal{F}_{Execs(\mathcal{A})}$ the smallest sigma-algebra containing $F(\mathcal{C}_{Execs(\mathcal{A})})$. In the remaining part of the paper, we will mainly focus on probabilistic executions of $\mathcal{A}$ of the form $\epsilon_\sigma \triangleq \epsilon_{\sigma,\delta_{\bar{q}_{\mathcal{A}}}} = \epsilon_{\sigma,\bar{q}_{\mathcal{A}}}$. Hence, we will deal with probablistic space of the form $(Execs(\mathcal{A}), \mathcal{F}_{Execs(\mathcal{A})}, \epsilon_\sigma)$.

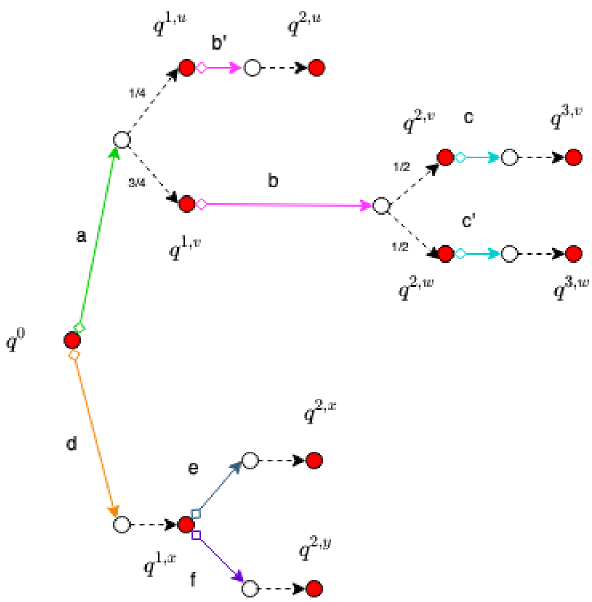


Figure 15 Non-deterministic execution: The scheduler allows us to solve the non-determinism, by triggering an action among the enabled one. Typically after execution $\alpha = q^0\ d\ q^{1,x}$, the actions e and f are enabled and the probability to take one transition is given by the scheduler σ that computes $\sigma(\alpha)$.

Scheduler Schema

Without restriction, a scheduler could become a too powerful adversary for practical applications. Hence, it is common to only consider a subset of schedulers, called a *scheduler* *schema*. Typically, a classic limitation is often described by a scheduler with "partial online information". Some formalism has already been proposed in [20] (section 5.6) to impose the scheduler that its choices are correlated for executions fragments in the same equivalence class where both the equivalence relation and the correlation must to be defined. This idea has been reused and simplified in [4] that defines equivalence classes on actions, called *tasks*. Then, a task-scheduler (a.k.a. "off-line" scheduler) selects a sequence of tasks $T_1, T_2, ...$ in advance that it cannot modify during the execution of the automaton. After each transition, the next task T_i triggers an enabled action if there is no ambiguity and is ignored otherwise. One of our main contribution, the theorem of implementation monotonicity w.r.t. PSIOA creation, is ensured only for a certain scheduler schema, so-called *creation-oblivious*. However, we will see that the practical set of task-schedulers are not creation-oblivious.

▶ **Definition 41** (scheduler schema). *A* scheduler schema *is a function that maps every PSIOA (resp. PCA) $\mathcal{A}$ to a subset of $schedulers(\mathcal{A})$.*

8.2 Implementation

In last subsection, we defined a measure of probability on executions with the help of a scheduler to solve non-determinism. Now we can define the notion of implementation. The intuition behind this notion is the fact that any environment $\mathcal{E}$ that would interact with both $\mathcal{A}$ and $\mathcal{B}$, would not be able to distinguish $\mathcal{A}$ from $\mathcal{B}$. The classic use-case is to formally show that a (potentially very sophisticated) algorithm implements a specification.

For us, an environment is simply a partially-compatible automaton, but in practice, he will play the role of a "distinguisher".

▶ **Definition 42** (Environment). *A probabilistic environment for PSIOA $\mathcal{A}$ is a PSIOA $\mathcal{E}$ such that $\mathcal{A}$ and $\mathcal{E}$ are partially-compatible. We note $env(\mathcal{A})$ the set of environments of $\mathcal{A}$.*

Now we define *insight function* which is a function that captures the insights that could be obtained by an external observer to attempt a distinction.

▶ **Definition 43** (insight function). *An* insight-function *is a function $f_{(.,.)}$ parametrized by a pair $(\mathcal{E}, \mathcal{A})$ of PSIOA where $\mathcal{E} \in env(\mathcal{A})$ s.t. $f_{(\mathcal{E},\mathcal{A})}$ is a measurable function from $(Execs(\mathcal{E}||\mathcal{A}), \mathcal{F}_{Execs(\mathcal{E}||\mathcal{A})})$ to some measurable space $(G_{(\mathcal{E},\mathcal{A})}, \mathcal{F}_{G_{(\mathcal{E},\mathcal{A})}})$.*

Some examples of insight-functions are the trace function and the environment projection function.

Since an insight-function $f_{(.,.)}$ is measurable, we can define the image measure of $\epsilon_{\sigma,\mu}$ under $f_{(\mathcal{E},\mathcal{A})}$, i.e. the probability to obtain a certain external perception under a certain scheduler σ and a certain probability distribution μ on the starting executions.

▶ **Definition 44** (f-*dist*). *Let $f_{(.,.)}$ be an insight-function. Let $(\mathcal{E}, \mathcal{A})$ be a pair of PSIOA where $\mathcal{E} \in env(\mathcal{A})$. Let μ be a probability measure on $(Execs(\mathcal{E}||\mathcal{A}), \mathcal{F}_{Execs(\mathcal{E}||\mathcal{A})})$, and $\sigma \in schedulers(\mathcal{E}||\mathcal{A})$. We define $f\text{-}dist_{(\mathcal{E},\mathcal{A})}(\sigma, \mu)$, to be the image measure of $\epsilon_{\sigma,\mu}$ under $f_{(\mathcal{E},\mathcal{A})}$ (i.e. the function that maps any $C \in \mathcal{F}_{G_{(\mathcal{E},\mathcal{A})}}$ to $\epsilon_{\sigma,\mu}(f^{-1}_{(\mathcal{E},\mathcal{A})}(C))$) . We note $f\text{-}dist_{(\mathcal{E},\mathcal{A})}(\sigma)$ for $f\text{-}dist_{(\mathcal{E},\mathcal{A})}(\sigma, \delta_{\bar{q}(\mathcal{E}||\mathcal{A})})$.*

We can see next definition of f-implementation as the incapacity of an environment to distinguish two automata if it uses only information filtered by the insight function f.

▶ **Definition 45** (f-implementation). *Let $f_{(.,.)}$ be an insight-function. Let S be a scheduler schema. We say that $\mathcal{A}$ f-*implements $\mathcal{B}$ according to *S, noted $\mathcal{A} \leq_0^{S,f} \mathcal{B}$, if $\forall \mathcal{E} \in env(\mathcal{A}) \cap env(\mathcal{B})$, $\forall \sigma \in S(\mathcal{E}||\mathcal{A})$, $\exists \sigma' \in S(\mathcal{E}||\mathcal{B})$, $f\text{-}dist_{(\mathcal{E},\mathcal{A})}(\sigma) \equiv f\text{-}dist_{(\mathcal{E},\mathcal{B})}(\sigma')$, i.e.*

- $supp(f\text{-}dist_{(\mathcal{E},\mathcal{A})}(\sigma)) = supp(f\text{-}dist_{(\mathcal{E},\mathcal{B})}(\sigma')) \triangleq \tilde{supp}$, and
- $\forall C \in \tilde{supp}$, $f\text{-}dist_{(\mathcal{E},\mathcal{A})}(\sigma)(C) = f\text{-}dist_{(\mathcal{E},\mathcal{B})}(\sigma')(C)$

We states a necessary and sufficient condition to obtain composability of f-implementation.

▶ **Definition 46** (Perception function). *Let $f_{(.,.)}$ be an insight-function. We say that $f_{(.,.)}$ is a* stable by composition *if for every quadruplet of PSIOA $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{B}, \mathcal{E})$, s.t. $\mathcal{B}$ is partially compatible with $\mathcal{A}_1$ and $\mathcal{A}_2$, $\mathcal{E} \in env(\mathcal{B}||\mathcal{A}_1) \cap env(\mathcal{B}||\mathcal{A}_2)$, for every $(C_1, C_2) \in \mathcal{F}_{Execs(\mathcal{E}||\mathcal{B}||\mathcal{A}_1)} \times \mathcal{F}_{Execs(\mathcal{E}||\mathcal{B}||\mathcal{A}_2)}$, $f_{(\mathcal{E}||\mathcal{B},\mathcal{A}_1)}(C_1) = f_{(\mathcal{E}||\mathcal{B},\mathcal{A}_2)}(C_2) \implies f_{(\mathcal{E},\mathcal{B}||\mathcal{A}_1)}(C_1) = f_{(\mathcal{E},\mathcal{B}||\mathcal{A}_2)}(C_2)$. An insight function stable by composition is said to be a* perception-function.

Substitutability

We can restate classic theorem of composability of implementation in a quite general form.

▶ **Theorem 47** (Implementation composability). *Let $f_{(.,.)}$ be a perception-function. Let S be a scheduler schema. Let $\mathcal{A}_1$, $\mathcal{A}_2$, $\mathcal{B}$ be PSIOA, s.t. $\mathcal{A}_1 \leq_0^{S,f} \mathcal{A}_2$. If $\mathcal{B}$ is partially compatible with $\mathcal{A}_1$ and $\mathcal{A}_2$ then $\mathcal{B}||\mathcal{A}_1 \leq_0^{S,f} \mathcal{B}||\mathcal{A}_2$.*

Proof. If $\mathcal{E}$ is an environment for both $\mathcal{B}||\mathcal{A}_1$ and $\mathcal{B}||\mathcal{A}_2$, then $\mathcal{E}' = \mathcal{E}||\mathcal{B}$ is an environment for both $\mathcal{A}_1$ and $\mathcal{A}_2$. By associativity of parallel composition, we have for every $i \in \{1, 2\}$, $(\mathcal{E}||\mathcal{B})||\mathcal{A}_i = \mathcal{E}||(\mathcal{B}||\mathcal{A}_i)$. Since $\mathcal{A}_1 \leq_0^{S,f} \mathcal{A}_2$, for any scheduler $\sigma \in S((\mathcal{E}||\mathcal{B})||\mathcal{A}_1)$, it exists a corresponding scheduler $\sigma' \in S((\mathcal{E}||\mathcal{B})||\mathcal{A}_2)$, s.t. $f\text{-}dist_{(\mathcal{E}||\mathcal{B}),\mathcal{A}_1}(\epsilon_\sigma) \equiv f\text{-}dist_{(\mathcal{E}||\mathcal{B}),\mathcal{A}_2}(\epsilon_{\sigma'})$. Thus, by stability by composition, for any scheduler $\sigma \in S(\mathcal{E}||(\mathcal{B}||\mathcal{A}_1))$, it exists a corresponding schedule $\sigma' \in S(\mathcal{E}||(\mathcal{B}||\mathcal{A}_2))$, s.t. $f\text{-}dist_{(\mathcal{E},(\mathcal{B}||\mathcal{A}_1))}(\epsilon_\sigma) \equiv f\text{-}dist_{(\mathcal{E},(\mathcal{B}||\mathcal{A}_2))}(\epsilon_{\sigma'})$, that is $\mathcal{A}_1||\mathcal{B} \leq_0^{S,f} \mathcal{A}_2||\mathcal{B}$. ◀

We also want restate classic theorem of f-implementation transitivity in the same form.

▶ **Theorem 48** (Implementation transitivity). *Let S be a scheduler schema. Let $f_{(.,.)}$ be an insight-function. Let $\mathcal{A}_1$, $\mathcal{A}_2$, $\mathcal{A}_3$ be PSIOA, s.t. $\mathcal{A}_1 \leq_0^{S,f} \mathcal{A}_2$ and $\mathcal{A}_2 \leq_0^{S,f} \mathcal{A}_3$, then $\mathcal{A}_1 \leq_0^{S,f} \mathcal{A}_3$.*

Proof. Let $\mathcal{E} \in env(\mathcal{A}_1) \cap env(\mathcal{A}_3)$.

Case 1: $\mathcal{E} \in env(\mathcal{A}_2)$. Let $\sigma_1 \in S(\mathcal{E}||\mathcal{A}_1)$ then, since $\mathcal{A}_1 \leq_0^{S,f} \mathcal{A}_2$ it exists $\sigma_2 \in S(\mathcal{E}||\mathcal{A}_2)$ $f\text{-}dist_{(\mathcal{E},\mathcal{A}_1)}(\sigma_1) \equiv f\text{-}dist_{(\mathcal{E},\mathcal{A}_2)}(\sigma_2)$ and since $\mathcal{A}_2 \leq_0^{S,f} \mathcal{A}_3$, it exists $\sigma_3 \in S(\mathcal{E}||\mathcal{A}_3)$ s.t. $f\text{-}dist_{(\mathcal{E},\mathcal{A}_2)}(\sigma_2) \equiv f\text{-}dist_{(\mathcal{E},\mathcal{A}_3)}(\sigma_3)$ and so for every $\sigma_1 \in S(\mathcal{E}||\mathcal{A}_1)$,it exists $\sigma_3 \in S(\mathcal{E}||\mathcal{A}_3)$ s.t. $f\text{-}dist_{(\mathcal{E},\mathcal{A}_1)}(\sigma_1) \equiv f\text{-}dist_{(\mathcal{E},\mathcal{A}_3)}(\sigma_3)$, i.e. $\mathcal{A}_1 \leq_0^{S,f} \mathcal{A}_3$.

Case 2: $\mathcal{E} \notin env(\mathcal{A}_2)$. A renaming procedure has to be performed before applying Case 1.

Let $\mathbf{A} = \{\mathcal{E}, \mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3\}$. We note $acts(\mathbf{A}) = \bigcup_{\mathcal{B}\in\mathbf{A}} acts(\mathcal{B})$. We use the special character ® for our renaming which is assumed to not be present in any syntactical representation of any action in $acts(\mathbf{A})$.

We note r_{int} the action renaming fonction s.t. $\forall q \in Q_\mathcal{E}$, $\forall a \in \widehat{sig}(\mathcal{E})(q)$, if $a \in int(\mathcal{E})(q)$, then $r_{int}(q)(a) = a_{®int}$ and $r_{int}(q)(a) = a$ otherwise. Then we note $\mathcal{E}' = r_{int}(\mathcal{E})$.

If $\mathcal{E}'$ and $\mathcal{A}_2$ are not partially-compatible, it is only because of some reachable state $(q_\mathcal{E}, q_{\mathcal{A}_2}) \in Q'_\mathcal{E} \times Q_{\mathcal{A}_2}$ s.t. $out(\mathcal{A}_2)(q_{\mathcal{A}_2}) \cap out(\mathcal{E}')(q_\mathcal{E}) \neq \emptyset$. Thus, we rename the actions for each state to avoid this conflict.

We note r_{out} the renaming function for $\mathcal{E}'$, s.t. $\forall q_\mathcal{E} \in Q_\mathcal{E}$, $\forall a \in \widehat{sig}(\mathcal{E})(q_\mathcal{E})$, $r_{out}(q_\mathcal{E})(a) = a_{®out}$ if $a \in out(\mathcal{E})(q_\mathcal{E})$ and a otherwise. In the same way, We note, for every $i \in \{1, 2, 3\}$ r^i_{in} the renaming function for $\mathcal{A}_i$, s.t. $\forall q_{\mathcal{A}_i} \in Q_{\mathcal{A}_i}$, $\forall a \in \widehat{sig}(\mathcal{A}_i)(q_{\mathcal{A}_i})$ $r_{in}(q_{\mathcal{A}_i})(a) = a_{®out}$ if $a \in in(\mathcal{A}_i)(q_{\mathcal{A}_i})$ and a otherwise. By lemma 12, $\mathcal{E}'' \triangleq r_{out}(\mathcal{E}')$ is a PSIOA. Finally, $\mathcal{E}''$ and $\mathcal{A}''_i = r^i_{in}(\mathcal{A}_i)$ are obviously partially-compatible (and even compatible) for each $i \in \{1, 2, 3\}$.

There is an obvious isomorphism between $\mathcal{E}''||\mathcal{A}''_1$ and $\mathcal{E}||\mathcal{A}_1$ and between $\mathcal{E}''||\mathcal{A}''_3$ and $\mathcal{E}||\mathcal{A}_3$ that allows us to apply case 1, which ends the proof.

◀

The two last theorems allows to state the classical theorem of substitutability.

▶ **Theorem 49** (Implementation substitutability). *Let $f_{(.,.)}$ be a perception-function. Let S be a scheduler schema. Let $\mathcal{A}_1$, $\mathcal{A}_2$, $\mathcal{B}_1$, $\mathcal{B}_2$ be PSIOA, s.t. $\mathcal{A}_1 \leq_0^{S,f} \mathcal{A}_2$ and $\mathcal{B}_1 \leq_0^{S,f} \mathcal{B}_2$. If both $\mathcal{B}_1$ and $\mathcal{B}_2$ are partially compatible with both $\mathcal{A}_1$ and $\mathcal{A}_2$ then $\mathcal{A}_1||\mathcal{B}_1 \leq_0^{S,f} \mathcal{A}_2||\mathcal{B}_2$.*

Proof. By theorem 47 of implementation composability, $\mathcal{A}_1||\mathcal{B}_1 \leq_0^{S,f} \mathcal{A}_2||\mathcal{B}_1$ and $\mathcal{A}_2||\mathcal{B}_1 \leq_0^{S,f} \mathcal{A}_2||\mathcal{B}_2$. By theorem 48 of implementation transitivity $\mathcal{A}_1||\mathcal{B}_1 \leq_0^{S,f} \mathcal{A}_2||\mathcal{B}_2$. ◀

Trace and projection on environment are perception-functions

▶ **Proposition 50** (trace is measurable). *Let $\mathcal{A}$ be a PSIOA (resp. PCA).*
$trace_{\mathcal{A}} : (Execs(\mathcal{A}), \mathcal{F}_{Execs(\mathcal{A})}) \rightarrow (Traces(\mathcal{A}), \mathcal{F}_{Traces(\mathcal{A})})$ is measurable.

Proof. This is enough to show that $\forall \beta \in Traces^*(\mathcal{A})$, $trace_{\mathcal{A}}^{-1}(C_\beta) \in \mathcal{F}_{Execs(\mathcal{A})}$. Yet, $trace_{\mathcal{A}}^{-1}(C_\beta) = \bigcup_{\alpha \in Execs^*(\mathcal{A}), trace_{\mathcal{A}}(\alpha)=\beta} C_\alpha$. Hence, this is a countable union of cones of executions of $\mathcal{A}$, i.e. an element of $\mathcal{F}_{Execs(\mathcal{A})}$. ◀

▶ **Proposition 51** (projection is measurable). *Let $\mathcal{A}$ be a PSIOA (resp. PCA) and $\mathcal{E} \in env(\mathcal{A})$.*

$$proj_{(\mathcal{E},\mathcal{A})} : \begin{cases} (Execs(\mathcal{E}||\mathcal{A}), \mathcal{F}_{Execs(\mathcal{E}||\mathcal{A})}) & \rightarrow \quad (Execs(\mathcal{E}), \mathcal{F}_{Execs(\mathcal{E})}) \\ \alpha & \mapsto \quad \alpha \restriction \mathcal{E} \end{cases} \text{ is measurable.}$$

Proof. This is enough to show that $\forall \alpha' \in Execs^*(\mathcal{E})$, $proj_{(\mathcal{E},\mathcal{A})}^{-1}(C_{\alpha'}) \in \mathcal{F}_{Execs(\mathcal{E}||\mathcal{A})}$. Yet, $proj_{(\mathcal{E},\mathcal{A})}^{-1}(C_{\alpha'}) = \bigcup_{\alpha \in Execs^*(\mathcal{A}), \alpha \restriction \mathcal{E} = \alpha'} C_\alpha$. Hence, this is a countable union of cones of executions of $\mathcal{E}||\mathcal{A}$, i.e. an element of $\mathcal{F}_{Execs(\mathcal{E}||\mathcal{A})}$. ◀

▶ **Lemma 52** (trace and projections are perception functions). *The function $trace_{(.,.)}$ and $proj_{(.,.)}$ parametrized with PSIOA $\mathcal{E}, \mathcal{A}$ where $\mathcal{E} \in env(\mathcal{A})$, (with $trace_{(\mathcal{E},\mathcal{A})} = trace_{(\mathcal{E}||\mathcal{A})}$) are both perception functions.*

Proof. **1.** (measurability) Immediate by propositions 50 and 51.

2. (stability by composition) Let $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{B}, \mathcal{E})$ be a quadruplet of PSIOA, s.t. $\mathcal{B}$ is compatible with $\mathcal{A}_1$ and $\mathcal{A}_2$, $\mathcal{E} \in env(\mathcal{B}||\mathcal{A}_1) \cap env(\mathcal{B}||\mathcal{A}_2)$. Let $(\alpha, \pi) \in Execs_{\mathcal{E}||\mathcal{B}||\mathcal{A}_1} \times Execs_{\mathcal{E}||\mathcal{B}||\mathcal{A}_2}$, clearly $\alpha \restriction (\mathcal{E}||\mathcal{B}) = \pi \restriction (\mathcal{E}||\mathcal{B}) \Longrightarrow \alpha \restriction (\mathcal{E}||\mathcal{B}) \restriction \mathcal{E} = \pi \restriction (\mathcal{E}||\mathcal{B}) \restriction \mathcal{E} \Longrightarrow \alpha \restriction \mathcal{E} = \pi \restriction \mathcal{E}$, while the traces stay the same.

◀

Thus, given an environment $\mathcal{E}$ of $\mathcal{A}$ probability measure μ on $\mathcal{F}_{Execs(\mathcal{E}||\mathcal{A})}$, and a scheduler σ of $(\mathcal{E}||\mathcal{A})$ we define $pdist_{(\mathcal{E},\mathcal{A})}(\sigma, \mu) \triangleq proj\text{-}dist_{(\mathcal{E},\mathcal{A})}(\sigma, \mu)$, to be the image measure of $\epsilon_{\sigma,\mu}$ under $proj_{(\mathcal{E},\mathcal{A})}$. We note $pdist_{(\mathcal{E},\mathcal{A})}(\sigma)$ for $pdist_{(\mathcal{E},\mathcal{A})}(\sigma, \delta_{\bar{q}_{\mathcal{E}||\mathcal{A}}})$.

This choice that slightly differs from $tdist_{(\mathcal{E},\mathcal{A})}(\sigma, \mu) = trace\text{-}dist_{(\mathcal{E},\mathcal{A})}(\sigma, \mu)$ used in [5], is motivated by the achievement of monotonicity of p-implementation w.r.t. PSIOA creation.

9 Introduction on PCA corresponding w.r.t. PSIOA $\mathcal{A}$, $\mathcal{B}$ to introduce monotonicity

In this section we take an interest in PCA $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ that differ only on the fact that $\mathcal{B}$ supplants $\mathcal{A}$ in $X_{\mathcal{B}}$. This definition is a key step to formally define monotonicity of a property. If a property is a binary relation on automata, a brave property P would verify monotonicity, i.e. if 1) $(\mathcal{A}, \mathcal{B}) \in P$, and 2) $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ are PCA that differ only on the fact that $\mathcal{B}$ supplants $\mathcal{A}$ in $X_{\mathcal{B}}$, then 3) $(X_{\mathcal{A}}, X_{\mathcal{B}}) \in P$. Monotonicity of implementation w.r.t. PSIOA creation is the main contribution of the paper.

9.1 Naive correspondence between two PCA

We formalize the idea that two configurations are identical except that the automaton $\mathcal{B}$ supplants $\mathcal{A}$ but with the same external signature. The following definition comes from [2].

▶ **Definition 53** ($\lhd_{\mathcal{AB}}$-corresponding configurations)**.** *(see figure 27) Let $\Phi \subseteq Autids$, and $\mathcal{A}, \mathcal{B}$ be PSIOA identifiers. Then we define $\Phi[\mathcal{B}/\mathcal{A}] = (\Phi \setminus \mathcal{A}) \cup \{\mathcal{B}\}$ if $\mathcal{A} \in \Phi$, and $\Phi[\mathcal{B}/\mathcal{A}] = \Phi$ if $\mathcal{A} \notin \Phi$. Let C, D be configurations. We define $C \lhd_{\mathcal{AB}} D$ iff (1) $auts(D) = auts(C)[\mathcal{B}/\mathcal{A}]$, (2) for every $\mathcal{A}' \notin auts(C) \setminus \{\mathcal{A}\}$: $map(D)(\mathcal{A}') = map(C)(\mathcal{A}')$, and (3) $ext(\mathcal{A})(s) = ext(\mathcal{B})(t)$ where $s = map(C)(\mathcal{A}), t = map(D)(\mathcal{B})$. That is, in $\lhd_{\mathcal{AB}}$-corresponding configurations, the SIOA other than $\mathcal{A}, \mathcal{B}$ must be the same, and must be in the same state. $\mathcal{A}$ and $\mathcal{B}$ must have the same external signature. In the sequel, when we write $\Psi = \Phi[\mathcal{B}/\mathcal{A}]$, we always assume that $\mathcal{B} \notin \Phi$ and $\mathcal{A} \notin \Psi$.*

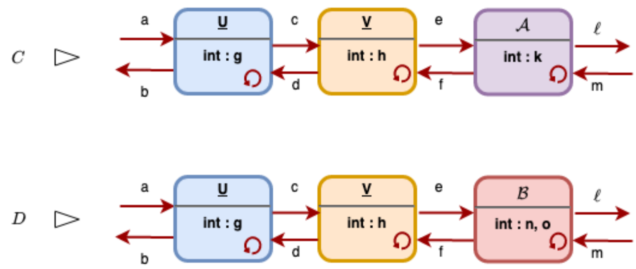


Figure 16 $\lhd_{\mathcal{AB}}$ corresponding-configuration

▶ Remark 54. It is possible to have two configurations C, D s.t. $C \lhd_{\mathcal{AA}} D$. That would mean that C and D only differ on the state of $\mathcal{A}$ (s or t) that has even the same external signature in both cases $ext(\mathcal{A})(s) = ext(\mathcal{A})(t)$, while we would have $int(\mathcal{A})(s) \neq int(\mathcal{A})(t)$.

Now, we formalise the fact that two PCA create some PSIOA in the same manner, excepting for $\mathcal{B}$ that supplants $\mathcal{A}$. Here again, this definition comes from [2].

▶ **Definition 55** (Creation corresponding configuration automata)**.** *Let X, Y be PCA and $\mathcal{A}, \mathcal{B}$ be PSIOA. We say that X, Y are creation-corresponding w.r.t. $\mathcal{A}, \mathcal{B}$ iff*

1. *X never creates $\mathcal{B}$ and Y never creates $\mathcal{A}$.*
2. *Let $(\alpha, \pi) \in Execs^*(X) \times Execs^*(Y)$ s.t. $trace_{\mathcal{A}}(\alpha) = trace_{\mathcal{B}}(\pi)$. Let $q = lstate(\alpha), q' = lstate(\pi)$. Then $\forall a \in \widehat{sig}(X)(q) \cap \widehat{sig}(Y)(q') : created(Y)(q')(a) = created(X)(q)(a)[\mathcal{B}/\mathcal{A}]$.*

In the same way than in definition 55, we formalise the fact that two PCA hide some output actions in the same manner. Here again, this definition is inspired by [2].

▶ **Definition 56** (Hiding corresponding configuration automata)**.** *Let X, Y be PCA and $\mathcal{A}, \mathcal{B}$ be PSIOA. We say that X, Y are hiding-corresponding w.r.t. $\mathcal{A}, \mathcal{B}$ iff*

1. *X never creates $\mathcal{B}$ and Y never creates $\mathcal{A}$.*
2. *Let $(\alpha, \pi) \in Execs^*(X) \times Execs^*(Y)$ s.t. $trace_{\mathcal{A}}(\alpha) = trace_{\mathcal{B}}(\pi)$. Let $q = lstate(\alpha), q' = lstate(\pi)$. Then $hidden\text{-}actions(Y)(q') = hidden\text{-}actions(X)(q)$.*

▶ **Definition 57** (creation&hiding-corresponding). *Let X, Y be PCA and $\mathcal{A}, \mathcal{B}$ be PSIOA. We say that X, Y are creation&hiding-corresponding w.r.t. $\mathcal{A}, \mathcal{B}$, if they are both creation-corresponding and hiding-corresponding w.r.t. $\mathcal{A}, \mathcal{B}$*

Now we define the notion of $\mathcal{A}$-exclusive action which corresponds to an action which is in the signature of $\mathcal{A}$ only. This definition is motivated by the fact that monotonicity induces that $\mathcal{A}$-exclusive (resp. $\mathcal{B}$-exclusive) actions do not create automata. Indeed, otherwise two internal action a and a' of $\mathcal{A}$ and $\mathcal{B}$ respectively could create different automata $\mathcal{C}$ and $\mathcal{D}$ and break the correspondence.

▶ **Definition 58** ($\mathcal{A}$-exclusive action). *Let $\mathcal{A} \in Autids$, X be a PCA. Let $q \in Q_X$, $(\mathbf{A}, \mathbf{S}) = config(X)(q)$, $act \in \widehat{sig}(X)(q)$. We say that act is $\mathcal{A}$-exclusive if for every $\mathcal{A}' \in \mathbf{A} \setminus \{\mathcal{A}\}$, $act \notin \widehat{sig}(\mathcal{A}')(\mathbf{S}(\mathcal{A}'))$ (and so $act \in \widehat{sig}(\mathcal{A})(\mathbf{S}(\mathcal{A}))$ only).*

The previous definitions 53, 55, 56 and 58 allow us to define a first (naive) definition of PCA corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$.

▶ **Definition 59** (naively corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$). *Let $\mathcal{A}, \mathcal{B} \in Autids$, $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ be PCA we say that $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ are naively corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$, if they verify:*

- $config(X_{\mathcal{A}})(\bar{q}_{X_{\mathcal{A}}}) \lhd_{AB} config(X_{\mathcal{B}})(\bar{q}_{X_{\mathcal{B}}})$.
- *$X_{\mathcal{A}}, X_{\mathcal{B}}$ are creation&hiding-corresponding w.r.t. $\mathcal{A}, \mathcal{B}$*
- *(No exclusive creation from $\mathcal{A}$ and $\mathcal{B}$) for each $\mathcal{K} \in \{\mathcal{A}, \mathcal{B}\}$, $\forall q \in Q_{X_{\mathcal{K}}}$, for every $\mathcal{K}$-exclusive action a, $created(X_{\mathcal{K}})(q)(a) = \emptyset$*

The last definition 59 of (naive) correspondence w.r.t. $\mathcal{A}$, $\mathcal{B}$ allows us to define a first (naive) definition 60 of monotonic relation.

▶ **Definition 60** (Naively monotonic relationship). *Let R be a binary relation on PSIOA. We say that R is naively monotonic if for every pair of PSIOA $(\mathcal{A}, \mathcal{B}) \in R$, for every pair of PCA $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ that are naively corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$, $(psioa(X_{\mathcal{A}}), psioa(X_{\mathcal{B}})) \in R$*

.

However, the relation of p-implementation introduced in subsection 8.2 is not proved monotonic without some additional technical assumptions presented in next subsection 9.2. Roughly speaking, it allows to 1) define a PCA $Y = X \setminus \{\mathcal{A}\}$ that corresponds to X "deprived" from $\mathcal{A}$ and 2) define the composition between Y and $\mathcal{A}$, 3) avoiding some ambiguities during the construction. In the first instance, the reader should skip the next subsection 9.2 on conservatism and keep in mind the intuition only. This sub-section 9.2 can be used to know the assumptions of the theorems of monotonicity and use them as black-boxes. The assumptions will be re-called during the proof.

9.2 Conservatism: the additional assumption for relevant definition of correspondence w.r.t. $\mathcal{A}, \mathcal{B}$

This subsection aims to define the notion of $\mathcal{A}$-conservative PCA.

Some definitions relative to configurations

In the remaining, it will often be useful to reason on the configurations. This is why we introduce some definitions that will be used again and again in the demonstrations.

The next definition captures the idea that two states of a certain layer represents the same situation for the bottom layer.

▶ **Definition 61** (configuration-equivalence between two states). *Let K, K' be PCA and $(q, q') \in Q_K \times Q_{K'}$. We say that q and q' are* config-equivalent*, noted $qR_{conf}q'$, if $config(K)(q) = config(K')(q')$. Furthermore, if*
- $config(K)(q) = config(K')(q')$,
- $hidden\text{-}actions(K)(q) = hidden\text{-}actions(K')(q')$ *and*
- $\forall a \in \widehat{sig}(K)(q) = \widehat{sig}(K')(q')$, $created(K)(q)(a) = created(K')(q')(a)$,

we say that q and q' are strictly-equivalent*, noted $qR_{strict}q'$.*

Now, we define a special subset of PCA that do not tolerate different configuration-equivalent states.

▶ **Definition 62** (Configuration-conflict-free PCA). *Let K be a PCA. We say K is configuration-conflict-free, if for every $q, q' \in Q_K$ s.t. $qR_{conf}q'$, then $q = q'$. The current state of a configuration-conflict-free PCA can be defined by its current attached configuration.*

For some elaborate definitions, we found useful to introduce the set of potential output actions of $\mathcal{A}$ in a configuration $config(X)(q)$ coming from a state q of a PCA X:

▶ **Definition 63** (potential ouput). *Let $\mathcal{A} \in autids$. Let X be a PCA. Let $q \in Q_X$. We note $pot\text{-}out(X)(q)(\mathcal{A})$ the set of potential output actions of $\mathcal{A}$ in $config(X)(q)$ that is*
- $pot\text{-}out(X)(q)(\mathcal{A}) = \emptyset$ *if* $\mathcal{A} \notin auts(config(X)(q))$
- $pot\text{-}out(X)(q)(\mathcal{A}) = out(\mathcal{A})(map(config(X)(q))(\mathcal{A}))$ *if* $\mathcal{A} \in auts(config(X)(q))$

Here, we define a configuration C deprived from an automaton $\mathcal{A}$ in the most natural way.

▶ **Definition 64** ($C \setminus \{\mathcal{A}\}$ Configuration deprived from an automaton). *$C = (\mathbf{A}, \mathbf{S})$. $C \setminus \{\mathcal{A}\} = (\mathbf{A}', \mathbf{S}')$ with $\mathbf{A}' = \mathbf{A} \setminus \{\mathcal{A}\}$ and $\mathbf{S}'$ the restriction of $\mathbf{S}$ on $\mathbf{A}'$*

The two last definitions 63 and 64 allows us to define in compact way a new relation between states that captures the idea that two states $q \in Q_X$ and $q' \in Q_Y$ are equivalent modulo a difference uniquely due to the presence of automaton $\mathcal{A}$ in $config(X)(q)$ and $config(Y)(q')$.

▶ **Definition 65** ($R^{\setminus\{\mathcal{A}\}}$ relationship (equivalent if we forget $\mathcal{A}$)). *Let $\mathcal{A} \in Autids$. Let $S = \{Q_X | X$ is a PCA $\}$ the set of states of any PCA. We defined the equivalence relation $R^{\setminus\{\mathcal{A}\}}_{conf}$ and $R^{\setminus\{\mathcal{A}\}}_{conf}$ on S defined by $\forall X, Y$ PCA, $\forall (q_X, q_Y) \in Q_X \times Q_Y$:*
- $q_X R^{\setminus\{\mathcal{A}\}}_{conf} q_Y \iff config(X)(q_X) \setminus \{A\} = config(Y)(q_Y) \setminus \{A\}$
- $q_X R^{\setminus\{\mathcal{A}\}}_{strict} q_Y \iff$ *the conjonction of the 3 following properties:*
  - $q_X R^{\setminus\{\mathcal{A}\}}_{conf} q_Y$
  - $\forall a \in \widehat{sig}(X)(q_X) \cap \widehat{sig}(Y)(q_Y)$, $created(Y)(q_Y)(a) \setminus \{\mathcal{A}\} = created(X)(q_X)(a) \setminus \{\mathcal{A}\}$
  - $hidden\text{-}actions(X)(q_X) \setminus pot\text{-}out(X)(q_X)(\mathcal{A}) = hidden\text{-}actions(Y)(q_Y) \setminus pot\text{-}out(Y)(q_Y)(\mathcal{A})$

$\mathcal{A}$-fair and $\mathcal{A}$-conservative: necessary assumptions to authorize the construction used in the proof

Now, we are ready to define $\mathcal{A}$-fairness and then $\mathcal{A}$-conservatism.

A $\mathcal{A}$-fair PCA is a PCA s.t. we can deduce its current properties from its current configuration deprived of $\mathcal{A}$. This assumption will allow us to define $Y = X \setminus \{\mathcal{A}\}$ in the proof of monotonicity.

▶ **Definition 66** ($\mathcal{A}$-fair PCA). *Let $\mathcal{A} \in Autids$. Let X be a PCA. We say that X is $\mathcal{A}$-fair if*

- *(configuration-conflict-free) X is configuration-conflict-free.*
- *(no conflict for projection) $\forall q_X, q'_X \in Q_X$, s.t. $q_X R^{\setminus\{\mathcal{A}\}}_{conf} q'_X$ then $q_X R^{\setminus\{\mathcal{A}\}}_{strict} q'_X$.*
- *(no exclusive creation by $\mathcal{A}$) $\forall q_X \in Q_X$, $\forall a \in \widehat{sig}(X)(q_X)$ $\mathcal{A}$-exclusive in q_X, $created(X)(q_X)(a) = \emptyset$*

This definition 66 allows the next definition 67 to be well-defined. A $\mathcal{A}$-conservative PCA is a $\mathcal{A}$-fair PCA that does not hide any output action that could be an external action of $\mathcal{A}$. This assumption will allow us to define the composition between $\mathcal{A}$ and $Y = X \setminus \{\mathcal{A}\}$ in the proof of monotonicity.

▶ **Definition 67** ($\mathcal{A}$-conservative PCA). *Let X be a PCA, $\mathcal{A} \in Autids$. We say that X is $\mathcal{A}$-conservative if it is $\mathcal{A}$-fair and for every state q_X, $C_X = config(X)(q_X)$ s.t. $\mathcal{A} \in aut(C_X)$ and $map(C_X)(\mathcal{A}) \triangleq q_{\mathcal{A}}$, $hidden\text{-}actions(X)(q_X) \cap \widehat{ext}(\mathcal{A})(q_{\mathcal{A}}) = \emptyset$.*

9.3 Corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$

We are closed to state all the technical assumptions to achieve monotonicity of p-implementation w.r.t. PSIOA creation. We introduce one last assumption so-called *creation-explicitness*, used in section 14 to reduce implementation of $X_{\mathcal{B}}$ by $X_{\mathcal{A}}$ to implementation of $\mathcal{B}$ by $\mathcal{A}$.

Intuitively, a PCA is $\mathcal{A}$-*creation-explicit* if the creation of a sub-automaton $\mathcal{A}$ is equivalent to the triggering of an action in a dedicated set. This property will allow to obtain the reduction of lemma 187.

▶ **Definition 68** (creation-explicit PCA). *Let $\mathcal{A}$ be a PSIOA and X be a PCA. We say that X is $\mathcal{A}$-*creation-explicit *iff: it exists a set of actions, noted $creation\text{-}actions(X)(\mathcal{A})$, s.t. $\forall q_X \in Q_X$, $\forall a \in \widehat{sig}(X)(q_X)$, if we note $\mathbf{A}_X = auts(config(X)(q_X))$ and $\varphi_X = created(X)(q_X)(a)$, then $\mathcal{A} \notin \mathbf{A}_X \wedge \mathcal{A} \in \varphi_X \iff a \in creation\text{-}actions(X)(\mathcal{A})$.*

Now we can define new (non naively) correspondence w.r.t. PSIOA $\mathcal{A}$, $\mathcal{B}$ to define (non naively) monotonic relationship.

▶ **Definition 69** (corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$). *Let $\mathcal{A}, \mathcal{B} \in Autids$, $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ be PCA we say that $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ are corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$, if 1) they are naively corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$, 2) they are $\mathcal{A}$-conservative and $\mathcal{B}$-conservative respectively and 3) they are $\mathcal{A}$-creation explicit and $\mathcal{B}$-creation explicit respectively with $creation\text{-}actions(X_{\mathcal{A}})(\mathcal{A}) = creation\text{-}actions(X_{\mathcal{B}})(\mathcal{B})$ i.e. they verify:*

- *$X_{\mathcal{A}}$ is $\mathcal{A}$-conservative and $X_{\mathcal{B}}$ is $\mathcal{B}$-conservative*
- *$X_{\mathcal{A}}$ is $\mathcal{A}$-creation explicit and $X_{\mathcal{B}}$ is $\mathcal{B}$-creation explicit with $creation\text{-}actions(X_{\mathcal{A}})(\mathcal{A}) = creation\text{-}actions(X_{\mathcal{B}})(\mathcal{B})$*
- *$config(X_{\mathcal{A}})(\bar{q}_{X_{\mathcal{A}}}) \lhd_{AB} config(X_{\mathcal{B}})(\bar{q}_{X_{\mathcal{B}}})$.*
- *$X_{\mathcal{A}}, X_{\mathcal{B}}$ are creation&hiding-corresponding w.r.t. $\mathcal{A}, \mathcal{B}$*
- *(No exclusive creation from $\mathcal{A}$ and $\mathcal{B}$) for each $\mathcal{K} \in \{\mathcal{A}, \mathcal{B}\}$, $\forall q \in Q_{X_{\mathcal{K}}}$, for every $\mathcal{K}$-exclusive action a, $created(X_{\mathcal{K}})(q)(a) = \emptyset$*

▶ **Definition 70** (Monotonic relationship). *Let R be a binary relation on PSIOA. We say that R is monotonic if for every pair of PSIOA $(\mathcal{A}, \mathcal{B}) \in R$, for every pair of PCA $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ that are corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$, $(psioa(X_{\mathcal{A}}), psioa(X_{\mathcal{B}})) \in R$.*

We would like to state the monotonicy of p-implementation, but it holds only for a certain class of schedulers, so-called *creation-oblivious* introduced in next subsection 9.4

9.4 Creation-oblivious scheduler

Here we present a particular scheduler schema, that do not take into account previous internal actions of a particular sub-automaton to output its probability over transitions to trigger.

We start by defining *strict oblivious-schedulers* that output the same transition with the same probability for pair of execution fragments that differ only by prefixes in the same class of equivalence. This definition is inspired by the one provided in the thesis of Segala, but is more restrictive since we require a strict equality instead of a correlation (section 5.6.2 in [20]).

▶ **Definition 71** (oblivious scheduler)**.** *Let $\tilde{W}$ be a PCA or a PSIOA, let $\tilde{\sigma} \in schedulers(\tilde{W})$ and let $\equiv$ be an equivalence relation on $Frags^*(\tilde{W})$ verifying $\forall \tilde{\alpha}_1, \tilde{\alpha}_2 \in Frags^*(\tilde{W})$ s.t. $\tilde{\alpha}_1 \equiv \tilde{\alpha}_2$, $lstate(\alpha_1) = lstate(\alpha_2)$. We say that $\tilde{\sigma}$ is $(\equiv)$-strictly oblivious if $\forall \tilde{\alpha}_1, \tilde{\alpha}_2, \tilde{\alpha}_3 \in Frags^*(\tilde{W})$ s.t. 1) $\alpha_1 \equiv \alpha_2$ and 2) $fstate(\tilde{\alpha}_3) = lstate(\tilde{\alpha}_2) = lstate(\tilde{\alpha}_1)$, then $\tilde{\sigma}(\tilde{\alpha}_1 ^\frown \tilde{\alpha}_3) = \tilde{\sigma}(\tilde{\alpha}_2 ^\frown \tilde{\alpha}_3)$.*

Now we define the relation of equivalence that defines our subset of creation-oblivious schedulers. Intuitively, two executions fragments ending on $\mathcal{A}$ creation are in the same equivalence class if they differ only in terms of internal actions of $\mathcal{A}$.

▶ **Definition 72** ($\tilde{\alpha} \equiv^{cr}_{\mathcal{A}} \tilde{\alpha}'$)**.** *Let $\mathcal{A}$ be a PSIOA, and $\tilde{W}$ be a PCA. For every $\tilde{\alpha}, \tilde{\alpha}' \in Frags^*(\tilde{W})$, we say $\tilde{\alpha} \equiv^{cr}_{\mathcal{A}} \tilde{\alpha}'$ iff:*

1. *$\tilde{\alpha}, \tilde{\alpha}'$ both ends on $\mathcal{A}$-creation.*
2. *$\tilde{\alpha}$ and $\tilde{\alpha}'$ differ only in the $\mathcal{A}$-exclusive actions and the states of $\mathcal{A}$, i.e. $\mu(\tilde{\alpha}) = \mu(\tilde{\alpha}')$ where $\mu(\tilde{\alpha} = \tilde{q}^0 a^1 \tilde{q}^1 ... a^n \tilde{q}^n) \in Frags^*(\tilde{W})$ is defined as follows:*
   - *remove the $\mathcal{A}$-exclusive actions*
   - *replace each state $\tilde{q}^i$ by its configuration $Config(\tilde{W})(\tilde{q}) = (\mathbf{A}^i, \mathbf{S}^i)$*
   - *replace each configuration $(\mathbf{A}^i, \mathbf{S}^i)$ by $(\mathbf{A}^i, \mathbf{S}^i) \setminus \{\mathcal{A}\}$*
   - *replace the (non-alternating) sequences of identical configurations (due to $\mathcal{A}$-exclusiveness of removed actions) by one unique configuration.*
3. *$lstate(\tilde{\alpha}) = lstate(\tilde{\alpha}')$*

We can remark that the items 3 can be deduced from 1 and 2 if X is configuration-conflict-free.

▶ **Definition 73** (creation-oblivious scheduler)**.** *Let $\tilde{A}$ be a PSIOA, $\tilde{W}$ be a PCA, $\tilde{\sigma} \in schedulers(\tilde{W})$. We say that $\tilde{\sigma}$ is $\mathcal{A}$-*creation oblivious *if it is $(\equiv^{cr}_{\mathcal{A}})$-strictly oblivious.*

We say that $\tilde{\sigma}$ is creation-oblivious *if it is $\mathcal{A}$-creation oblivious for every sub-automaton $\mathcal{A}$ of $\tilde{W}$ ($\mathcal{A} \in \bigcup_{q \in states(\tilde{W})} auts(config(\tilde{W})(q))$). We note CrOb the function that maps any PCA $\tilde{W}$ to the set of creation-oblivious schedulers of $\tilde{W}$.*

We have formally defined our notion of creation-oblivious scheduler. This will be a key property to ensure lemma 187 that allows to reduce the measure of a class of comportment as a function of measures of classes of shorter comportment where no creation of $\mathcal{A}$ or $\mathcal{B}$ occurs excepting potentially at very last action. This reduction is more or less necessary to obtain monotonicity of implementation relation:

▶ **Theorem 74** ($\leq_0^{CrOb,p}$ is monotonic)**.** *Let $\mathcal{A}, \mathcal{B} \in Autids$, $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ be PCA corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$. Let $S = CrOb$ and $p = proj_{(.,.)}$. If $\mathcal{A} \leq_0^{S,p} \mathcal{B}$, then $X_{\mathcal{A}} \leq_0^{S,p} X_{\mathcal{B}}$*

The remaining sections are dedicated to the proof of this theorem 74. We start by defining in section 10 a morphism between executions of automata, so called *executions-matching*, that preserves structure and measure of probability under *alter ego schedulers*. Next, we define in section 11 the notion of an automaton $X_{\mathcal{A}}$ deprived from a PSIOA $\mathcal{A}$, noted $X_{\mathcal{A}} \setminus \{\mathcal{A}\}$. Furthermore, we show in section 12 that there is an executions-matching from a PCA $X_{\mathcal{A}}$ to $(X_{\mathcal{A}} \setminus \{\mathcal{A}\})||\tilde{\mathcal{A}}^{sw}$ where $\tilde{\mathcal{A}}^{sw}$ is the *simpleton wrapper* of $\mathcal{A}$, i.e. a PCA that only handle $\mathcal{A}$. The section 14 uses the morphism of section 12 to reduce the implementation of $X_{\mathcal{B}}$ by $X_{\mathcal{A}}$ to the implementation of $\mathcal{B}$ by $\mathcal{A}$ and finally obtain the monotonicity of implementation w.r.t. PSIOA creation. Finally section 15 explains why the task-scheduler introduced in [5] is not creation-oblivious.

10 Executions-matching

In this section, we introduce some tools to formalise the fact that two automata have the same comportment for the same scheduler. This section is composed by two sub-sections on PSIOA executions-matching and PCA executions-matching. Basically, an executions-matching execution from an automaton $\mathcal{A}$ to another automaton $\mathcal{B}$ is a morphism f^{ex} from $Execs(\mathcal{A})$ to $Execs(\mathcal{B})$ that is structure-preserving. In the remaining, we will often use an executions-matching to show that a pair of executions $(\alpha, \pi = f^{ex}(\alpha)) \in Execs(\mathcal{A}) \times Execs(\mathcal{B})$ have the same probability $\epsilon_{\sigma}(\alpha) = \epsilon_{\sigma'}(\pi)$ under a pair of so-called *alter-ego* schedulers $(\sigma, \sigma') \in schedulers(\mathcal{A}) \times schedulers(\mathcal{B})$ that have corresponding comportment after corresponding executions fragment $(\alpha', \pi' = f^{ex}(\alpha')) \in Frags^*(\mathcal{A}) \times Frags^*(\mathcal{B})$.

10.1 PSIOA executions-matching and semantic equivalence

This first subsection is about PSIOA executions-matching.

matching execution

An executions-matching need a states-matching (see definition 75) and a transitions-matching (see definition 77) to be defined itself.

▶ **Definition 75** (states-matching). *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA, let $Q'_{\mathcal{A}} \subset Q_{\mathcal{A}}$ and let $f : Q'_{\mathcal{A}} \to Q_{\mathcal{B}}$ be a mapping that verifies:*

- *Starting state preservation: If $\bar{q}_{\mathcal{A}} \in Q'_{\mathcal{A}}$ then $f(\bar{q}_{\mathcal{A}}) = \bar{q}_{\mathcal{B}}$*
- *Signature preservation (modulo an hiding operation): $\forall (q, q') \in Q'_{\mathcal{A}} \times Q_{\mathcal{B}}$, s.t. $q' = f(q)$, $sig(\mathcal{A})(q) = hide(sig(\mathcal{B})(q'), h(q'))$ with $h(q') \subseteq out(\mathcal{B})(q')$ (resp. with $h(q') = \emptyset$, that is $sig(\mathcal{A})(q) = sig(\mathcal{B})(q')$).*

then we say that f is a weak *(resp.* strong*)* states-matching *from $\mathcal{A}$ to $\mathcal{B}$. If $Q'_{\mathcal{A}} = Q_{\mathcal{A}}$, then we say that f is a* complete *(weak or strong) states-matching from $\mathcal{A}$ to $\mathcal{B}$.*

Before being able to define transitions-matching, some requirements have to be ensured. A set of transition that would ensure these requirements would be called *eligible to transitions-matching*.

▶ **Definition 76** (transitions set eligible to transitions matching). *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA,let $Q'_{\mathcal{A}} \subset Q_{\mathcal{A}}$ and let $f : Q'_{\mathcal{A}} \to Q_{\mathcal{B}}$ be a states-matching from $\mathcal{A}$ to $\mathcal{B}$. Let $D'_{\mathcal{A}} \subseteq D_{\mathcal{A}}$ be a subset of transition. If $D'_{\mathcal{A}}$ verifies that $\forall (q, a, \eta_{(\mathcal{A},q,a)}) \in D'_{\mathcal{A}}$:*

- *Matched states preservation: $q \in Q'_{\mathcal{A}}$ and*
- *Equitable corresponding distribution: $\forall q'' \in supp(\eta_{(\mathcal{A},q,a)}), q'' \in Q'_{\mathcal{A}}$ and $\eta_{(\mathcal{A},q,a)} \overset{f}{\longleftrightarrow} \eta_{(\mathcal{B},f(q),a)}$*

then we say that $D'_{\mathcal{A}}$ is eligible to transitions-matching domain from f. *We omit to mention the states-matching f when this is clear in the context.*

Now, we are able to define a transitions-matching, which is a property-preserving mapping from a set of transitions $D'_{\mathcal{A}} \subseteq D_{\mathcal{A}}$ to another set of transitions $D'_{\mathcal{B}} \subseteq D_{\mathcal{B}}$.

▶ **Definition 77** (transitions-matching). *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA, let $Q'_{\mathcal{A}} \subset Q_{\mathcal{A}}$ and let $f : Q'_{\mathcal{A}} \to Q_{\mathcal{B}}$ be a states-matching from $\mathcal{A}$ to $\mathcal{B}$. Let $D'_{\mathcal{A}} \subseteq D_{\mathcal{A}}$ be a subset of transition eligible to transitions-matching domain from f.*

We define the transitions-matching (f, f^{tr}) from $\mathcal{A}$ to $\mathcal{B}$ induced by the states-matching f and the subset of transition $D'_{\mathcal{A}}$ *s.t.* $f^{tr} : D'_{\mathcal{A}} \to D_{\mathcal{B}}$ *is defined by* $f^{tr}((q, a, \eta_{(\mathcal{A},q,a)})) = (f(q), a, \eta_{(\mathcal{B},f(q),a)})$ *. If f is complete and $D'_{\mathcal{A}} = D_{\mathcal{A}}$, (f, f^{tr}) is said to be a complete transitions-matching. If f is weak (resp. strong) (f, f^{tr}) is said to be a weak (resp. strong) transitions-matching. If f is clear in the context, with a slight abuse of notation, we say that f^{tr} is a transitions-matching.*

The function f^{tr} needs to verify some constraints imposed by f, but if the set $D'_{\mathcal{A}}$ of concerned transitions is correctly-chosen to ensure the 2 properties of definition 76, then such a transitions-matching is unique.

Now, we can easily define an executions-matching with a transitions-matching, which is a property-preserving mapping from a set of execution fragments $F'_{\mathcal{A}} \subseteq Frags(\mathcal{A})$ to another set of execution fragments $F'_{\mathcal{B}} \subseteq Frags(\mathcal{B})$.

▶ **Definition 78** (executions-matching). *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA. Let (f, f^{tr}) be a transitions-matching from $\mathcal{A}$ to $\mathcal{B}$. Let $F'_{\mathcal{A}} = \{\alpha \triangleq q^0 a^1 q^1 ... a^n q^n ... \in Frags(\mathcal{A}) | \forall i \in [0 : |\alpha| - 1], (q^i, a^{i+1}, \eta_{(\mathcal{A},q^i,a^{i+1})}) \in dom(f^{tr})\}$. Let $f^{ex} : F'_{\mathcal{A}} \to Frags(\mathcal{B})$, built from (f, f^{tr}) s.t. $\forall \alpha = q^0_{\mathcal{A}} a^1 q^1_{\mathcal{A}} ... a^n q^n_{\mathcal{A}} ... \in F'_{\mathcal{A}}$, $f^{ex}(\alpha) = f(q^0_{\mathcal{A}}) a^1 f(q^1_{\mathcal{A}}) ... a^n f(q^n_{\mathcal{A}}) ...$*

We say that (f, f^{tr}, f^{ex}) is an executions-matching *from $\mathcal{A}$ to $\mathcal{B}$. Furthermore, if (f, f^{tr}) is complete and $F'_{\mathcal{A}} = Frags(\mathcal{A})$, (f, f^{tr}, f^{ex}) is said to be a complete executions-matching. If (f, f^{tr}) is weak (resp. strong) (f, f^{tr}, f^{ex}) is said to be a weak (resp. strong) executions-matching. When (f, f^{tr}) is clear in the context, with a slight abuse of notation, we say that f^{ex} is an executions-matching.*

The function f^{ex} is completely defined by (f, f^{tr}), hence we call (f, f^{tr}, f^{ex}) the executions-matching induced by the transition matching (f, f^{tr}) or the executions-matching induced by the states-matching f and the subset of transitions $dom(f^{tr})$.

The construction of f^{ex} allows us to see two executions mapped by an executions-mapping as a sequence of pairs of transitions mapped by the attached transitions-matching. This result is formalised in next lemma 79.

▶ **Lemma 79** (executions-matching seen as a sequence of transitions-matchings). *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA. Let (f, f^{tr}, f^{ex}) be an executions-matching from $\mathcal{A}$ to $\mathcal{B}$. Let $\alpha = q^0_{\mathcal{A}} a^1 q^1_{\mathcal{A}} ... a^n q^n_{\mathcal{A}} ... \in dom(f^{ex})$ and $\pi = f^{ex}(\alpha) = q^0_{\mathcal{B}} a^1 q^1_{\mathcal{B}} ... a^n q^n_{\mathcal{B}} ... = f(q^0_{\mathcal{A}}) a^1 f(q^1_{\mathcal{A}}) ... a^n f(q^n_{\mathcal{A}}) ....$ Then for every $i \in [0 : |\alpha| - 1]$, $(q^i_{\mathcal{B}}, a^{i+1}, \eta_{(\mathcal{B},q^i_{\mathcal{B}},a^{i+1})}) = f^{tr}((q^i_{\mathcal{A}}, a^{i+1}, \eta_{(\mathcal{A},q^i_{\mathcal{A}},a^i)}))$*

Proof. First, matched states preservation and action preservation are ensured by construction. By definition, for every $i \in [0 : |\alpha| - 1]$, $(q^i_{\mathcal{A}}, a^{i+1}, \eta_{(\mathcal{A},q^i_{\mathcal{A}},a^{i+1})}) \in dom(f^{tr})$. We note $tr^i_{\mathcal{B}} \triangleq f^{tr}((q^i_{\mathcal{A}}, a^{i+1}, \eta_{(\mathcal{A},q^i_{\mathcal{A}},a^{i+1})}))$. By definition, $tr^i_{\mathcal{B}}$ is of the form $(f(q^i_{\mathcal{A}}), a^{i+1}, \eta)$. But a transition of this form is unique, which means $tr^i_{\mathcal{B}} = (f(q^i_{\mathcal{A}}), a^{i+1}, \eta_{(\mathcal{B},f(q^i_{\mathcal{A}}),a^{i+1})})$ which ends the proof. ◀

Now we overload the definition of executions-matching to be able to state the main result of this paragraph i.e. theorem 83

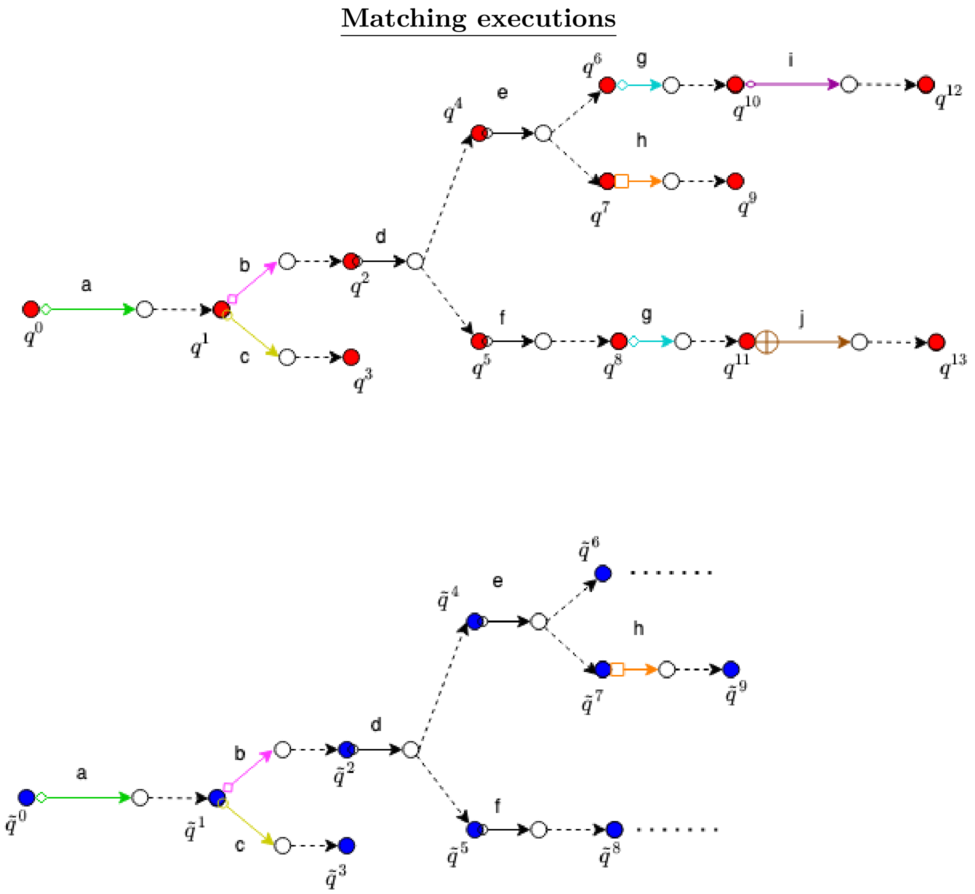


Figure 17 Here we have $Q'_{\mathcal{A}} = \{q^0, q^1, ..., q^9\} \subsetneq Q_{\mathcal{A}}$, we define the state-matching $f : Q'_{\mathcal{A}} \rightarrow Q_{\mathcal{B}}$ s.t. $\forall k \in [1, 9], f(q^k) = \tilde{q}^k$, and $D'_{\mathcal{A}} = \{(q^0, a, \eta_{(\mathcal{A},q^0,a)}), (q^1, b, \eta_{(\mathcal{A},q^1,b)}), (q^1, c, \eta_{(\mathcal{A},q^1,c)}), (q^2, d, \eta_{(\mathcal{A},q^2,d)}), (q^4, e, \eta_{(\mathcal{A},q^4,e)}), (q^5, f, \eta_{(\mathcal{A},q^5,f)}), (q^7, h, \eta_{(\mathcal{A},q^7,h)})\}$. We can define the execution matching (f, f^{tr}, f^{ex}) induced by f and $D'_{\mathcal{A}}$.

▶ **Definition 80** (executions-matching overload: pre-execution-distribution)**.** *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA. Let (f, f^{tr}, f^{ex}) be an executions-matching from $\mathcal{A}$ to $\mathcal{B}$.*

Let $(\mu, \mu') \in Disc(Frags(\mathcal{A})) \times Disc(Frags(\mathcal{B}))$ s.t. $\mu \overset{f^{ex}}{\leftrightarrow} \mu'$. Then we say that (f, f^{tr}, f^{ex}) is an executions-matching *from $(\mathcal{A}, \mu)$ to $(\mathcal{B}, \mu')$.*

In practice, we will often use executions-matching from $(\mathcal{A}, \delta_{\bar{q}_{\mathcal{A}}})$ to $(\mathcal{B}, \delta_{\bar{q}_{\mathcal{B}}})$.

Continued executions-matching

Motivated by PSIOA creation that would break the states-matching from a PCA $X_{\mathcal{A}}$ to the PCA $Z_{\mathcal{A}} \triangleq (X \setminus \{\mathcal{A}\})||\tilde{\mathcal{A}}^{sw}$ defined in section 12, we introduce the notion of continuation of executions-matching.

▶ **Definition 81** (Continued executions-matching)**.** *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA. Let (f, f^{tr}, f^{ex}) be an executions-matching from $\mathcal{A}$ to $\mathcal{B}$ with $dom(f) \triangleq Q'_{\mathcal{A}} \subset Q_{\mathcal{A}}$ and $dom(f^{tr}) \triangleq D'_{\mathcal{A}} \subset D_{\mathcal{A}}$.*

Let $f^+ : Q''_{\mathcal{A}} \rightarrow Q_{\mathcal{B}}$ with $Q''_{\mathcal{A}} \subset Q_{\mathcal{A}}$. Let $D''_{\mathcal{A}} \subset D_{\mathcal{A}}$ be a subset of transitions verifying for every $(q, a, \eta_{(\mathcal{A},q,a)}) \in D''_{\mathcal{A}} \setminus D'_{\mathcal{A}}$:

- *Matched states preservation: $q \in Q'_{\mathcal{A}}$*

- *Extension of equitable corresponding distribution:* $\forall q'' \in supp(\eta_{(\mathcal{A},q,a)}), q'' \in Q''_{\mathcal{A}}$ *and* $\eta_{(\mathcal{A},q,a)} \overset{f^+}{\longleftrightarrow} \eta_{(\mathcal{B},f(q),a)}$.

We define the $(f^+, D''_{\mathcal{A}})$-continuation of f^{tr} *as the function* $f^{tr,+} : D'_{\mathcal{A}} \cup D''_{\mathcal{A}} \to D_{\mathcal{B}}$ *s.t.* $\forall (q, a, \eta_{(\mathcal{A},q,a)}) \in D'_{\mathcal{A}} \cup D''_{\mathcal{A}}, f^{tr,+}((q, a, \eta_{(\mathcal{A},q,a)})) = (f(q), a, \eta_{(\mathcal{B},f(q),a)})$.

Let $F''_{\mathcal{A}} = dom(f^{ex}) \cup \{\alpha ^\frown qaq' \in Execs^*(\mathcal{A}) | \alpha \in dom(f^{ex}) \wedge (q, a, \eta_{(\mathcal{A},q,a))} \in D''_{\mathcal{A}}\}$. *We define the* $(f^{tr,+})$-continuation of f^{ex} *as the function* $f^{ex,+} : F''_{\mathcal{A}} \to Frags(\mathcal{B})$ *s.t.* $\forall \alpha \in dom(f^{ex}), f^{ex,+}(\alpha) = f^{ex}(\alpha)$ *and* $\forall \alpha' = \alpha ^\frown q, a, q' \in F''_{\mathcal{A}} \setminus dom(f^{ex})$, $f^{ex,+}(\alpha') = f^{ex}(\alpha) ^\frown f(q), a, f^+(q')$.

Then, we say that $((f, f^+), f^{tr,+}, f^{ex,+})$ *is the* $(f^+, D''_{\mathcal{A}})$-continuation *of* (f, f^{tr}, f^{ex}) *which is a* continuation *of* (f, f^{tr}, f^{ex}) *and a* continued executions-matching from $\mathcal{A}$ to. $\mathcal{B}$.

Moreover, if $(\mu, \mu') \in Disc(Frags(\mathcal{A})) \times Disc(Frags(\mathcal{B}))$ *s.t.* $\mu \overset{f^{ex,+}}{\longleftrightarrow} \mu'$, *then we say that* $((f, f^+), f^{tr,+}, f^{ex,+})$ *is a* continued executions-matching *from* $(\mathcal{A}, \mu)$ *to* $(\mathcal{B}, \mu')$.

From executions-matching to probabilistic distribution preservation

We want to states that a (potentially-continued) executions-matching preserves measure of probability of the corresponding executions.

To do so, we define alter egos schedulers to a certain executions-matching. Such pair of schedulers are very similar in the sense that their outputs depends only on the semantic structure of the input, preserved by the executions-matching.

▶ **Definition 82** ((f, f^{tr}, f^{ex})-alter egos schedulers)**.** *Let* $\mathcal{A}$ *and* $\mathcal{B}$ *be two PSIOA. Let* (f, f^{tr}, f^{ex}) *be an executions-matching from* $\mathcal{A}$ *to* $\mathcal{B}$. *Let* $(\tilde{\sigma}, \sigma) \in schedulers(\mathcal{A}) \times schedulers(\mathcal{B})$. *We say that* $(\tilde{\sigma}, \sigma)$ *are* (f, f^{tr}, f^{ex})-alter egos *(or* f^{ex}-alter egos*) if, and only if, for every* $(\tilde{\alpha}, \alpha) \in Frags^*(\mathcal{A}) \times Frags^*(\mathcal{B})$ *s.t.* $\alpha = f^{ex}(\tilde{\alpha})$ *(which means* $\widehat{sig}(\mathcal{A})(\tilde{q}) = \widehat{sig}(\mathcal{B})(q) \triangleq sig$ *with* $\tilde{q} = lstate(\tilde{\alpha})$ *and* $q = lstate(\alpha)$ *by signature preservation property of the associated states-matching),* $\forall a \in sig, \tilde{\sigma}(\tilde{\alpha})((\tilde{q}, a, \eta_{(\mathcal{A},\tilde{q},a)})) = \sigma(\alpha)((q, a, \eta_{(\mathcal{B},q,a)}))$.

Let us remark that the previous definition implies that the probability of halting after corresponding executions fragments $(\tilde{\alpha}, \alpha)$ is also the same.

Now we are ready to states an intuitive result that will be often used in the remaining.

▶ **Theorem 83** (Executions-matching preserves general probabilistic distribution)**.** *Let* $\mathcal{A}$ *and* $\mathcal{B}$ *be two PSIOA. Let* $(\tilde{\mu}, \mu) \in Disc(Frags(\mathcal{A})) \times Disc(Frags(\mathcal{B}))$. *Let* (f, f^{tr}, f^{ex}) *be an executions-matching from* $(\mathcal{A}, \tilde{\mu})$ *to* $(\mathcal{B}, \mu)$. *Let* $(\tilde{\sigma}, \sigma) \in schedulers(\mathcal{A}) \times schedulers(\mathcal{B})$, *s.t.* $(\tilde{\sigma}, \sigma)$ *are* (f, f^{tr}, f^{ex})-alter egos. *Let* $(\tilde{\alpha}, \alpha) \in Frags^*(\mathcal{A}) \times Frags^*(\mathcal{B})$ *s.t.* $\alpha = f^{ex}(\tilde{\alpha})$. *Then* $\epsilon_{\tilde{\sigma},\tilde{\mu}}(C_{\tilde{\alpha}}) = \epsilon_{\sigma,\mu}(C_{\alpha})$ *and* $\epsilon_{\tilde{\sigma},\tilde{\mu}}(\tilde{\alpha}) = \epsilon_{\sigma,\mu}(\alpha)$.

Proof. First, by definition 80 of executions-matching, f^{ex} is a bijection from $supp(\tilde{\mu})$ to $supp(\mu)$ where $\forall \tilde{\alpha}_o \in supp(\tilde{\mu}), \mu(f^{ex}(\tilde{\alpha}_o)) = \tilde{\mu}(\tilde{\alpha}_o)$ (*). Second, by definition 40 of measure generated by a scheduler, $\epsilon_{\sigma,\mu}(C_{\alpha'}) = \Sigma_{\alpha_o \in supp(\mu)} \mu(\alpha_o) \cdot \epsilon_{\sigma,\alpha_o}(C_{\alpha'})$ and $\epsilon_{\tilde{\sigma},\tilde{\mu}}(C_{\tilde{\alpha}'}) = \Sigma_{\tilde{\alpha}_o \in supp(\tilde{\mu})} \tilde{\mu}(\tilde{\alpha}_o) \cdot \epsilon_{\tilde{\sigma},\tilde{\alpha}_o}(C_{\tilde{\alpha}'})$ (**). Hence, by combining (*) and (**), we only need to show that for every $(\tilde{\alpha}_o, \alpha_o) \in supp(\tilde{\mu}) \times supp(\mu)$ with $f^{ex}(\tilde{\alpha}_o) = \alpha_o$, for every $(\tilde{\alpha}', \alpha') \in Frags^*(\mathcal{A}) \times Frags^*(\mathcal{B})$ with $f^{ex}(\tilde{\alpha}') = \alpha'$, we have $\epsilon_{\sigma,\alpha_o}(C_{\alpha'}) = \epsilon_{\tilde{\sigma},\tilde{\alpha}_o}(C_{\tilde{\alpha}'})$ that we show by induction on the size $s = |\tilde{\alpha}| = |\alpha|$. We fix $(\tilde{\alpha}_o, \alpha_o) \in supp(\tilde{\mu}) \times supp(\mu)$ with $f^{ex}(\tilde{\alpha}_o) = \alpha_o$.

Basis: $s = 0$

Let $\tilde{\alpha}' = \tilde{q}' \in Frags^*(\mathcal{A})$, $\alpha' = q' \in Frags^*(\mathcal{B})$ with $\alpha' = f^{ex}(\tilde{\alpha}')$. We have $|\tilde{\alpha}'| = |\alpha'| = 0$. By definition 40 of measure generated by a scheduler,

$$\epsilon_{\tilde{\sigma},\tilde{\alpha}_o}(C_{\tilde{\alpha}'}) =: \begin{cases} 0 & \text{if both } \tilde{\alpha}' \nleq \tilde{\alpha}_o \text{ and } \tilde{\alpha}_o \nleq \tilde{\alpha}' \\ 1 & \text{if } \tilde{\alpha}' \leq \tilde{\alpha}_o \\ \epsilon_{\tilde{\sigma},\tilde{\alpha}_o}(C_{\tilde{\alpha}}) \cdot \tilde{\sigma}(\tilde{\alpha})(\eta_{(\mathcal{A},\tilde{q},a)}) \cdot \eta_{(\mathcal{A},\tilde{q},a)}(\tilde{q}') & \text{if } \tilde{\alpha}_o \leq \tilde{\alpha} \text{ and } \tilde{\alpha}' = \tilde{\alpha}^\frown \tilde{q} a \tilde{q}' \end{cases}$$

and

$$\epsilon_{\sigma,\alpha_o}(C_{\alpha'}) =: \begin{cases} 0 & \text{if both } \alpha' \nleq \alpha_o \text{ and } \alpha_o \nleq \alpha' \\ 1 & \text{if } \alpha' \leq \alpha_o \\ \epsilon_{\sigma,\alpha_o}(C_{\alpha}) \cdot \sigma(\alpha)(\eta_{(\mathcal{B},q,a)}) \cdot \eta_{(\mathcal{B},q,a)}(q') & \text{if } \alpha_o \leq \alpha \text{ and } \alpha' = \alpha^\frown q a q' \end{cases}$$

Since $|\tilde{\alpha}'| = |\alpha'| = 0$ the third case is never met. The second case can be written: $\tilde{\alpha}' \leq \tilde{\alpha}_o$ (resp. $\alpha' \leq \alpha_o$) iff $fstate(\tilde{\alpha}_o) = \tilde{q}'$ (resp. $fstate(\alpha_o) = q'$). Hence, for every $(\tilde{\alpha}_o, \alpha_o)$ s.t. $f^{ex}(\tilde{\alpha}_o) = \alpha_o$, $\epsilon_{\tilde{\sigma},\tilde{\alpha}_o}(C_{\tilde{\alpha}'}) = \epsilon_{\sigma,\alpha_o}(C'_{\alpha})$ which ends the basis.

Induction: We assume the result to be true up to size s and we show it implies the result is true for size $s+1$. Let $(\tilde{\alpha}', \tilde{\alpha}, \alpha', \alpha) \in Frags^*(\mathcal{A})^2 \times Frags^*(\mathcal{B})^2$ with $\tilde{\alpha}' = \tilde{\alpha}^\frown \tilde{q} a \tilde{q}'$ and $\alpha' = \alpha^\frown q a q'$ s.t. $\alpha' = f^{ex}(\tilde{\alpha}')$ with $|\tilde{\alpha}'| = |\alpha'| = s+1$. We want to show that $\epsilon_{\tilde{\sigma},\tilde{\mu}}(C_{\tilde{\alpha}'}) = \epsilon_{\sigma,\mu}(C_{\alpha'})$. By definition 40 of measure generated by a scheduler,

$$\epsilon_{\tilde{\sigma},\tilde{\alpha}_o}(C_{\tilde{\alpha}'}) =: \begin{cases} 0 & \text{if both } \tilde{\alpha}' \nleq \tilde{\alpha}_o \text{ and } \tilde{\alpha}_o \nleq \tilde{\alpha}' \\ 1 & \text{if } \tilde{\alpha}' \leq \tilde{\alpha}_o \\ \epsilon_{\tilde{\sigma},\tilde{\alpha}_o}(C_{\tilde{\alpha}}) \cdot \tilde{\sigma}(\tilde{\alpha})(\eta_{(\mathcal{A},\tilde{q},a)}) \cdot \eta_{(\mathcal{A},\tilde{q},a)}(\tilde{q}') & \text{if } \tilde{\alpha}_o \leq \tilde{\alpha} \text{ and } \tilde{\alpha}' = \tilde{\alpha}^\frown \tilde{q} a \tilde{q}' \end{cases}$$

and

$$\epsilon_{\sigma,\alpha_o}(C_{\alpha'}) =: \begin{cases} 0 & \text{if both } \alpha' \nleq \alpha_o \text{ and } \alpha_o \nleq \alpha' \\ 1 & \text{if } \alpha' \leq \alpha_o \\ \epsilon_{\sigma,\alpha_o}(C_{\alpha}) \cdot \sigma(\alpha)(\eta_{(\mathcal{B},q,a)}) \cdot \eta_{(\mathcal{B},q,a)}(q') & \text{if } \alpha_o \leq \alpha \text{ and } \alpha' = \alpha^\frown q a q' \end{cases}$$

Again, the executions-matching implies that i) both $\tilde{\alpha}' \nleq \tilde{\alpha}_o$ and $\tilde{\alpha}_o \nleq \tilde{\alpha}' \iff$ both $\alpha' \nleq \alpha_o$ and $\alpha_o \nleq \alpha'$, ii) $\tilde{\alpha} \leq \tilde{\alpha}_o \iff \alpha \leq \alpha_o$ and iii) $\tilde{\alpha}_o \leq \tilde{\alpha} \iff \alpha_o \leq \alpha$. Moreover, by induction assumption $\epsilon_{\tilde{\sigma},\tilde{\alpha}_o}(C_{\tilde{\alpha}}) = \epsilon_{\sigma,\alpha_o}(C_{\alpha})$. Hence we only need to show that $\tilde{\sigma}(\tilde{\alpha})(\eta_{(\mathcal{A},\tilde{q},a)}) \cdot \eta_{(\mathcal{A},\tilde{q},a)}(\tilde{q}') = \sigma(\alpha)(\eta_{(\mathcal{B},q,a)}) \cdot \eta_{(\mathcal{B},q,a)}(q')$ (***). By definition of alter-ego schedulers, $\tilde{\sigma}(\tilde{\alpha})(\eta_{(\mathcal{A},\tilde{q},a)}) = \sigma(\alpha)(\eta_{(\mathcal{B},q,a)})$ (j). By definition of executions-matching, $\eta_{(\mathcal{A},\tilde{q},a)}(\tilde{q}') = \eta_{(\mathcal{B},q,a)}(q')$ (jj). Thus (j) and (jj) implies (***) which allows us to terminate the induction to obtain $\epsilon_{\tilde{\sigma},\tilde{\alpha}_o}(C_{\tilde{\alpha}'}) = \epsilon_{\sigma,\alpha_o}(C_{\alpha'})$.

Finally, let $sig = \widehat{sig}(\mathcal{A})(lstate(\tilde{\alpha}')) = \widehat{sig}(\mathcal{A})(lstate(\alpha'))$, then $\epsilon_{\tilde{\sigma},\tilde{\alpha}_o}(\tilde{\alpha}') = \epsilon_{\tilde{\sigma},\tilde{\alpha}_o}(C_{\tilde{\alpha}'}) \cdot (1 - \Sigma_{a \in sig} \tilde{\sigma}(\tilde{\alpha}')(a)) = \epsilon_{\sigma,\alpha_o}(C_{\alpha'}) \cdot (1 - \Sigma_{a \in sig} \sigma(\alpha')(a)) = \epsilon_{\sigma,\alpha_o}(\alpha')$, which ends the proof. ◀

We restate the previous theorem with continued executions-matching.

▶ **Theorem 84** (Continued executions-matching preserves general probabilistic distribution). *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA. Let $(\tilde{\mu}, \mu) \in Disc(Frags(\mathcal{A})) \times Disc(Frags(\mathcal{B}))$. Let (f, f^{tr}, f^{ex}) be an executions-matching from $(\mathcal{A}, \tilde{\mu})$ to $(\mathcal{B}, \mu)$. Let $((f, f^+), f^{tr,+}, f^{ex,+})$ be a continuation of (f, f^{tr}, f^{ex}). Let $(\tilde{\sigma}, \sigma) \in schedulers(\mathcal{A}) \times schedulers(\mathcal{B})$, s.t. $(\tilde{\sigma}, \sigma)$ are (f, f^{tr}, f^{ex})-*alter egos. *Let $(\tilde{\alpha}, \alpha) \in Frags^*(\mathcal{A}) \times Frags^*(\mathcal{B})$ s.t. $\alpha = f^{ex,+}(\tilde{\alpha})$. Then $\epsilon_{\tilde{\sigma},\tilde{\mu}}(C_{\tilde{\alpha}}) = \epsilon_{\sigma,\mu}(C_{\alpha})$.*

Proof. The proof is exactly the same than the one for theorem 83 ◀

Before dealing with composability of executions-matching, we prove two results about injectivity and surjectivity of executions-matching in next lemma 85 and 86.

▶ **Lemma 85** (Injectivity of executions-matching). *Let (f, f^{tr}, f^{ex}) be an executions-matching from $\mathcal{A}$ to $\mathcal{B}$ and $((f, f^+), f^{tr,+}, f^{ex,+})$ a continuation of (f, f^{tr}, f^{ex}).*

Let $\tilde{f}^{ex,+} : F''_{\mathcal{A}} \subseteq dom(f^{ex,+}) \rightarrow \tilde{F}_{\mathcal{B}} \subseteq range(f^{ex,+})$. Let $\tilde{f} : Q''_{\mathcal{A}} \subseteq dom(f) \rightarrow Q_{\mathcal{B}}$ be the restriction of f on a set $Q''_{\mathcal{A}} \subseteq dom(f)$.

1. *If i) $\forall \alpha \in F''_{\mathcal{A}}$, $fstate(\alpha) \in Q''_{\mathcal{A}}$ and ii) $\tilde{f}$ is injective, then $\tilde{f}^{ex,+}$ is injective.*
2. *(Corollary) if $F''_{\mathcal{A}} \subseteq Execs(\mathcal{A})$, $f^{ex,+}$ is injective.*

Proof. **1.** By induction on the size k of the prefix: Basis: By i) $fstate(\alpha), fstate(\alpha') \in Q''_{\mathcal{A}}$, by construction of $f^{ex,+}$, $f(fstate(\alpha)) = f(fstate(\alpha')) = fstate(\pi)$ and by ii) $fstate(\alpha) = fstate(\alpha')$ Induction. We assume the injectivity of $\tilde{f}^{ex,+}$ to be true for execution on size k and we show this is also true for size $k+1$. Let $\pi = s^0 b^1 s^1 ... s^k b^{k+1} s^{k+1} \in F''_{\mathcal{B}}$ Let $\alpha = q^0 a^1 q^1 ... q^k a^{k+1} q^{k+1}, \alpha' = q'^0 a'^1 q'^1 ... q'^k a'^{k+1} q'^{k+1} \in F''_{\mathcal{A}}$ s.t. $f(\alpha) = f(\alpha') = \pi$. By construction of $f^{ex,+}$, $\forall i \in [1,n]$, $b^i = a^i = a'^i$. By construction of $f^{ex,+}$, $f^{ex,+}(q'^0 a'^1 q'^1 ... q'^k) = f^{ex,+}(q^0 a^1 q^1 ... q^k) = s^0 a^1 s^1 ... s^k$. By induction assumption $q'^0 a'^1 q'^1 ... q'^k) = q^0 a^1 q^1 ... q^k$. By definition of execution, $s^{k+1} \in supp(\eta_{(\mathcal{B}, s^k, a^{k+1})})$. By equitable corresponding distribution, If $\eta_{(\mathcal{A}, q^k, a^{k+1})} \in dom(f^{tr})$, the restriction of f, $\tilde{f} : supp(\eta_{(\mathcal{A}, q^k, a^{k+1})}) \rightarrow supp(\eta_{(\mathcal{B}, s^k, a^{k+1})})$ is bijective and $\eta_{(\mathcal{A}, q^k, a^{k+1})} \in dom(f^{tr,+}) \setminus dom(f^{tr})$, the restriction of f^+, $\tilde{f}^+ : supp(\eta_{(\mathcal{A}, q^k, a^{k+1})}) \rightarrow supp(\eta_{(\mathcal{B}, s^k, a^{k+1})})$ is bijective so $q^{k+1} = q'^{k+1}$ which ends the proof.

2. We have $|start(\mathcal{A})| = 1$. Hence the restriction of f on $start(\mathcal{A})$ is necessarily injective (ii). Let $\alpha \in Execs(\mathcal{A})$. By definition of execution, $fstate(\alpha) \in start(\mathcal{A})$ (i). All the requirements of lemma 85, first item are met, which ends the proof.

◀

▶ **Lemma 86** (Surjectivity property preserved by continuation). *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA. Let (f, f^{tr}, f^{ex}) be an executions-matching from $\mathcal{A}$ to $\mathcal{B}$. Let $((f, f^+), f^{tr,+}, f^{ex,+})$ be the $(f^+, D''_{\mathcal{A}})$-continuation of (f, f^{tr}, f^{ex}) (where by definition $D''_{\mathcal{A}} \setminus dom(f^{tr})$ respect the properties of matched states preservation and extension of equitable corresponding distribution from definition 81). If the restriction $\tilde{f}^{ex} : E'_{\mathcal{A}} \subseteq Execs(\mathcal{A}) \rightarrow \tilde{E}_{\mathcal{B}} \subseteq Execs(\mathcal{B})$ is surjective, then $\tilde{f}^{ex,+} : E'^{,+}_{\mathcal{A}} = \{\alpha' = \alpha^\frown q_{\mathcal{A}}, a, q'_{\mathcal{A}} \in Execs(\mathcal{A}) | \alpha \in E_{\mathcal{A}}, (q_{\mathcal{A}}, a, \eta_{\mathcal{A}, q_{\mathcal{A}}, a}) \in dom(f^{tr,+})\} \rightarrow \tilde{E}^+_{\mathcal{B}} = \{\pi' = \pi^\frown q_{\mathcal{B}}, a, q'_{\mathcal{B}} \in Execs(\mathcal{B}) | \pi \in \tilde{E}_{\mathcal{B}}, \exists \alpha \in (f^{ex})^{-1}(\pi) \cap E'_{\mathcal{A}}, q_{\mathcal{A}} = lstate(\alpha), (q_{\mathcal{A}}, a, \eta_{\mathcal{A}, q_{\mathcal{A}}, a}) \in dom(f^{tr,+})\}$ is surjective.*

Proof. Let $\pi' \in \tilde{E}_{\mathcal{B}}$. We have $\pi' = \pi^\frown q_{\mathcal{B}}, a, q'_{\mathcal{B}} \in Execs(\mathcal{B})$ s.t. $\pi \in \tilde{E}_{\mathcal{B}}$ and $\exists \alpha \in (f^{ex})^{-1}(\pi) \cap E'_{\mathcal{A}}, q_{\mathcal{A}} = lstate(\alpha)$ and $(q_{\mathcal{A}}, a, \eta_{(\mathcal{A}, q_{\mathcal{A}}, a)}) \in dom(f^{tr,+})$. By $(q_{\mathcal{A}}, a, \eta_{\mathcal{A}, q_{\mathcal{A}}, a}) \in dom(f^{tr,+})$, if i) $(q_{\mathcal{A}}, a, \eta_{\mathcal{A}, q_{\mathcal{A}}, a}) \in dom(f^{tr,+}) \setminus dom(f^{tr})$ $\eta_{\mathcal{A}, q_{\mathcal{A}}, a} \overset{f^+}{\longleftrightarrow} \eta_{\mathcal{B}, q_{\mathcal{B}}, a}$ and if ii) $(q_{\mathcal{A}}, a, \eta_{\mathcal{A}, q_{\mathcal{A}}, a}) \in dom(f^{tr})$ $\eta_{\mathcal{A}, q_{\mathcal{A}}, a} \overset{f}{\longleftrightarrow} \eta_{\mathcal{B}, q_{\mathcal{B}}, a}$. In both cases, it exists $q'_{\mathcal{A}} \in supp(\eta_{\mathcal{A}, q_{\mathcal{A}}, a})$ s.t. $f^{ex,+}(\alpha' = \alpha^\frown q_{\mathcal{A}}, a, q'_{\mathcal{A}}) = \pi'$ with $\alpha' \in E'^{,+}_{\mathcal{A}}$.

◀

We finish this paragraph with the concept of semantic equivalence that describes a pair of PSIOA that differ only syntactically.

▶ **Definition 87** (semantic equivalence). *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA. We say that $\mathcal{A}$ and $\mathcal{B}$ are semantically-equivalent if it exists $f : Execs(\mathcal{A}) \rightarrow Execs(\mathcal{B})$ which is a complete bijective executions-matching from $\mathcal{A}$ to $\mathcal{B}$.*

Composability of executions-matching relationship

Now we are looking for composability of executions-matching. First we define natural extension of notions presented in previous paragraph for the automaton obtained after composition with another automaton $\mathcal{E}$.

▶ **Definition 88** ($\mathcal{E}$-extension). *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA. Let $\mathcal{E}$ be partially-compatible with both $\mathcal{A}$ and $\mathcal{B}$.*

1. *Let $Q'_{\mathcal{A}} \subset Q_{\mathcal{A}}$. We call $\mathcal{E}$-*extension *of $Q'_{\mathcal{A}}$ the set of states $Q'_{\mathcal{A}||\mathcal{E}} = \{q \in Q_{\mathcal{A}||\mathcal{E}} | q \restriction \mathcal{A} \in Q'_{\mathcal{A}}\}$*

2. *Let* $f : Q'_{\mathcal{A}} \subset Q_{\mathcal{A}} \to Q_{\mathcal{B}}$. *We call* $\mathcal{E}$*-extension of* f *the function* $g : Q'_{\mathcal{A}||\mathcal{E}} \to Q_{\mathcal{B}} \times Q_{\mathcal{E}}$ *s.t.* $\forall (q_{\mathcal{A}}, q_{\mathcal{E}}) \in Q'_{\mathcal{A}||\mathcal{E}}$, $g((q_{\mathcal{A}}, q_{\mathcal{E}})) = (f(q_{\mathcal{A}}), q_{\mathcal{E}}))$
3. *Let* $D'_{\mathcal{A}} \subset D_{\mathcal{A}}$ *a subset of transitions. We call* $\mathcal{E}$*-extension of* $D'_{\mathcal{A}}$ *the set* $D'_{\mathcal{A}||\mathcal{E}} = \{((q_{\mathcal{A}}, q_{\mathcal{E}}), a, \eta_{((\mathcal{A},\mathcal{E}),(q_{\mathcal{A}},q_{\mathcal{E}}),a)}) \in D_{\mathcal{A}||\mathcal{E}} | q_{\mathcal{A}} \in Q'_{\mathcal{A}}$ *and either* $(q_{\mathcal{A}}, a, \eta_{(\mathcal{A},q_{\mathcal{A}},a)}) \in D'_{\mathcal{A}}$ *or the action* a *is not enabled in* $q_{\mathcal{A}}\}$.

Now, we can start with the composability of states-matching.

▶ **Lemma 89** (Composability of states-matching)**.** *Let* $\mathcal{A}$ *and* $\mathcal{B}$ *be two PSIOA. Let* $\mathcal{E}$ *be partially-compatible with* $\mathcal{A}$ *and* $\mathcal{B}$. *Let* $f : Q'_{\mathcal{A}} \subset Q_{\mathcal{A}} \to Q_{\mathcal{B}}$ *be a states-matching. Let* g *be the* $\mathcal{E}$*-extension of* f.

If $range(g) \subset Q_{\mathcal{B}||\mathcal{E}}$, *then* g *is a states-matching from* $\mathcal{A}||\mathcal{E}$ *to* $\mathcal{B}||\mathcal{E}$.

Proof.
- Starting state preservation: if $(\bar{q}_{\mathcal{A}}, \bar{q}_{\mathcal{E}}) \in Q_{\mathcal{A}||\mathcal{E}}$ then $\bar{q}_{\mathcal{A}} \in Q'_{\mathcal{A}}$ which means $f(\bar{q}_{\mathcal{A}}) = \bar{q}_{\mathcal{B}}$, thus $g((\bar{q}_{\mathcal{A}}, \bar{q}_{\mathcal{E}})) = (\bar{q}_{\mathcal{B}}, \bar{q}_{\mathcal{E}})$.
- Signature preservation (modulo an hiding operation): $\forall ((q_{\mathcal{A}}, q_{\mathcal{E}}), (q_{\mathcal{B}}, q_{\mathcal{E}})) \in Q'_{\mathcal{A}||\mathcal{E}} \times Q_{\mathcal{B}||\mathcal{E}}$ with $(q_{\mathcal{B}}, q_{\mathcal{E}}) = g((q_{\mathcal{A}}, q_{\mathcal{E}}))$, we have $sig(\mathcal{A})(q_{\mathcal{A}}) = sig(\mathcal{B})(f(q_{\mathcal{A}})) = hide(sig(\mathcal{B})(q_{\mathcal{B}}), h(q_{\mathcal{B}}))$ with $h(q_{\mathcal{B}}) \subseteq out(\mathcal{B})(q_{\mathcal{B}})$.
  Since $\mathcal{A}$ and $\mathcal{E}$ are partially-compatible, $sig(\mathcal{A})(q_{\mathcal{A}}) = hide(sig(\mathcal{B})(q_{\mathcal{B}}), h(q_{\mathcal{B}}))$ is compatible with $sig(\mathcal{E})(q_{\mathcal{E}})$ which means a fortiori $sig(\mathcal{B})(q_{\mathcal{B}})$ is compatible with $sig(\mathcal{E})(q_{\mathcal{E}})$. Namely $\forall act \in h(q_{\mathcal{B}}), act \notin in(\mathcal{E})(q_{\mathcal{E}})$. Hence $sig((\mathcal{A},\mathcal{E}))((q_{\mathcal{A}}), q_{\mathcal{E}})) = hide(sig((\mathcal{B},\mathcal{E}))((q_{\mathcal{B}}, q_{\mathcal{E}})), h'((q_{\mathcal{B}}, q_{\mathcal{E}}))$ with $h'((q_{\mathcal{B}}, q_{\mathcal{E}})) = h(q_{\mathcal{B}}) \subseteq out(\mathcal{B})(q_{\mathcal{B}}) \subseteq out(\mathcal{B}||)((q_{\mathcal{B}}, q_{\mathcal{E}}))$ which ends the proof.

◀

The composability of states-matching is ensured under the condition $range(g) \subset Q_{\mathcal{B}||\mathcal{E}}$ where g is the $\mathcal{E}$-extension of the original states-matching $f : Q'_{\mathcal{A}} \subseteq Q_{\mathcal{A}} \to Q_{\mathcal{B}}$. In next lemma, we give a sufficient condition to ensure $range(g) \subset Q_{\mathcal{B}||\mathcal{E}}$. This is the one that we will use in practice.

▶ **Definition 90** (reachable-by and states of execution (recall))**.** *Let* $\mathcal{A}$ *be a PSIOA or a PCA. Let* $E'_{\mathcal{A}} \subseteq Execs(\mathcal{A})$. *We note* $reachable\text{-}by(E'_{\mathcal{A}}) = \{q \in Q_{\mathcal{A}} | \exists \alpha \in E'_{\mathcal{A}}, lstate(\alpha) = q\}$. *Let* $\alpha = q^0, a^1, q^1, ...a^n, q^n, ...$ *We note* $states(\alpha) = \bigcup_{i \in |\alpha|} q^i$.

▶ **Lemma 91** (A sufficient condition to obtain $range(g) \subset Q_{\mathcal{B}||\mathcal{E}}$)**.** *Let* $\mathcal{A}$ *and* $\mathcal{B}$ *be two PSIOA. Let* $\mathcal{E}$ *be partially-compatible with both* $\mathcal{A}$ *and* $\mathcal{B}$. *Let* $f : Q'_{\mathcal{A}} \subset Q_{\mathcal{A}} \to Q_{\mathcal{B}}$ *be a states-matching. Let* $Q'_{\mathcal{A}||\mathcal{E}}$ *be the* $\mathcal{E}$*-extension of* $Q'_{\mathcal{A}}$.

Let $Q''_{\mathcal{A}||\mathcal{E}} \subset Q'_{\mathcal{A}||\mathcal{E}}$ *the set of states reachable by an execution that counts only states in* $Q'_{\mathcal{A}||\mathcal{E}}$, *i.e.*
- $E''_{\mathcal{A}||\mathcal{E}} = \{\alpha \in Execs(\mathcal{A}||\mathcal{E}) | states(\alpha) \subseteq Q'_{\mathcal{A}||\mathcal{E}}\}$
- $Q''_{\mathcal{A}||\mathcal{E}} = reachable\text{-}by(E''_{\mathcal{A}||\mathcal{E}})$

Let f'' *the restriction of* f *to set* $Q''_{\mathcal{A}} = \{q_{\mathcal{A}} = ((q_{\mathcal{A}}, q_{\mathcal{E}}) \upharpoonright \mathcal{A}) | (q_{\mathcal{A}}, q_{\mathcal{E}}) \in Q''_{\mathcal{A}||\mathcal{E}}\}$.

Then the $\mathcal{E}$*-extension of* f'', *noted* g'' *verifies* $range(g'') \subset Q_{\mathcal{B}||\mathcal{E}}$.

Proof. By induction on the minimum size of an execution $\tilde{\alpha} = q^0 a^1 ... q^n$ with $q^* = q^n, \forall i \in [0, n], q^i \in Q'_{\mathcal{A}||\mathcal{E}}$. Basis ($|\alpha| = 0 \implies \alpha = \bar{q}_{\mathcal{A}}$): we consider $q^* = \bar{q}_{\mathcal{A}}$. We have $g((\bar{q}_{\mathcal{A}}, \bar{q}_{\mathcal{E}})) = (f(\bar{q}_{\mathcal{A}}), \bar{q}_{\mathcal{E}}) = (\bar{q}_{\mathcal{B}}, \bar{q}_{\mathcal{E}}) \in Q_{\mathcal{B}||\mathcal{E}}$.

We assume this is true for $\tilde{\alpha}$ with $lstate(\tilde{\alpha}) = q$ and we show this is also true for $\tilde{\alpha}' = \tilde{\alpha} \frown q a q'$. By induction hypothesis $q \in Q_{\mathcal{B}||\mathcal{E}}$. Since $q' \in Q_{\mathcal{A}||\mathcal{E}}$, $\mathcal{A}$ and $\mathcal{E}$ are compatible at state $(q'_{\mathcal{A}}, q'_{\mathcal{E}})$, that is $sig(\mathcal{A})(q'_{\mathcal{A}})$ and $sig(\mathcal{E})(q'_{\mathcal{E}})$ are compatible, which means that a fortiori, $(sig(\mathcal{B})(f''(q'_{\mathcal{A}}))$ and $sig(\mathcal{E})(q'_{\mathcal{E}})$ are compatible and so $\mathcal{B}$ and $\mathcal{E}$ are compatible at state $(f''(q'_{\mathcal{A}}), q'_{\mathcal{E}}) = g''(q')$. Hence $g''(q')$ is a reachable compatible state of $(\mathcal{B}, \mathcal{E})$ which means this is a state of $\mathcal{B}||\mathcal{E}$.

◀

Now, we can continue with the composability of transitions-matching.

▶ **Lemma 92** (Composability of eligibility for transitions-matching)**.** *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA. Let $\mathcal{E}$ be partially-compatible with $\mathcal{A}$ and $\mathcal{B}$. Let $f : Q'_{\mathcal{A}} \subset Q_{\mathcal{A}} \to Q_{\mathcal{B}}$ be a states-matching and $D'_{\mathcal{A}}$ a subset of transitions eligible to transitions-matching domain from f. Let g be the $\mathcal{E}$-extension of f and $D'_{\mathcal{A}||\mathcal{E}}$ the $\mathcal{E}$-extension of $D_{\mathcal{A}}$.*

If $range(g) \subset Q_{\mathcal{B}||\mathcal{E}}$, then $D'_{\mathcal{A}||\mathcal{E}}$ is eligible to transitions-matching domain from g.

Proof. Let $((q_{\mathcal{A}}, q_{\mathcal{E}}), a, \eta_{((\mathcal{A},\mathcal{E}),(q_{\mathcal{A}},q_{\mathcal{E}}),a)}) \in D'_{\mathcal{A}||\mathcal{E}}$.

By definition, $q_{\mathcal{A}} \in Q'_{\mathcal{A}}$ which means $(q_{\mathcal{A}}, q_{\mathcal{E}}) \in Q'_{\mathcal{A}||\mathcal{E}}$, so the matched states preservation is ensured. We still need to ensure the equitable corresponding distribution.

- Let $(q''_{\mathcal{A}}, q''_{\mathcal{E}}) \in supp(\eta_{((\mathcal{A},\mathcal{E}),(q_{\mathcal{A}},q_{\mathcal{E}}),a)})$. If $a \in \widehat{sig}(\mathcal{A})(q_{\mathcal{A}})$, then $q''_{\mathcal{A}} \in supp(\eta_{(\mathcal{A},q_{\mathcal{A}},a)})$ which means $q''_{\mathcal{A}} \in Q'_{\mathcal{A}}$ and hence $(q''_{\mathcal{A}}, q''_{\mathcal{E}}) \in Q'_{\mathcal{A}||\mathcal{E}}$. If $a \notin \widehat{sig}(\mathcal{A})$, $\eta_{(\mathcal{A},q_{\mathcal{A}},a)} = \delta_{q_{\mathcal{A}}}$, which means $q''_{\mathcal{A}} = q_{\mathcal{A}} \in Q'_{\mathcal{A}}$ and hence $(q''_{\mathcal{A}}, q''_{\mathcal{E}}) \in Q'_{\mathcal{A}||\mathcal{E}}$. Thus for every $(q''_{\mathcal{A}}, q''_{\mathcal{E}}) \in supp(\eta_{((\mathcal{A},\mathcal{E}),(q_{\mathcal{A}},q_{\mathcal{E}}),a)})$, $(q''_{\mathcal{A}}, q''_{\mathcal{E}}) \in Q'_{\mathcal{A}||\mathcal{E}}$.
- $\eta_{((\mathcal{A},\mathcal{E}),(q_{\mathcal{A}},q_{\mathcal{E}}),a)}((q''_{\mathcal{A}}, q''_{\mathcal{E}})) = \eta_{(\mathcal{A},q_{\mathcal{A}},a)} \otimes \eta_{(\mathcal{E},q_{\mathcal{E}},a)}(q''_{\mathcal{A}}, q''_{\mathcal{E}}) = \eta_{(\mathcal{A},q_{\mathcal{A}},a)}(q''_{\mathcal{A}}) \cdot \eta_{(\mathcal{E},q_{\mathcal{E}},a)}(q''_{\mathcal{E}}) = \eta_{(\mathcal{B},f(q_{\mathcal{A}}),a)}(f(q''_{\mathcal{A}})) \cdot \eta_{(\mathcal{E},q_{\mathcal{E}},a)}(q''_{\mathcal{E}}) = \eta_{(\mathcal{B},f(q_{\mathcal{A}}),a)} \otimes \eta_{(\mathcal{E},q_{\mathcal{E}},a)}(f(q''_{\mathcal{A}}), q''_{\mathcal{E}}) = \eta_{((\mathcal{B},\mathcal{E}),g(q_{\mathcal{A}},q_{\mathcal{E}}),a)}(g(q''_{\mathcal{A}}, q''_{\mathcal{E}}))$ which ends the proof of equitable corresponding distribution.

◀

▶ **Definition 93** ($\mathcal{E}$-extension of an execution-matching)**.** *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA. Let $\mathcal{E}$ be partially-compatible with both $\mathcal{A}$ and $\mathcal{B}$. Let (f, f^{tr}, f^{ex}) be an executions-matching from $\mathcal{A}$ to $\mathcal{B}$. Let g the $\mathcal{E}$-extension of f. If $range(g) \subset Q_{\mathcal{B}||\mathcal{E}}$, then*

1. *we call the $\mathcal{E}$-extension of f^{tr} the function $g^{tr} : (q, a, \eta_{(\mathcal{A}||\mathcal{E},q,a)}) \in D'_{\mathcal{A}||\mathcal{E}} \mapsto (g(q), a, \eta_{(\mathcal{B}||\mathcal{E},g(q),a)})$ where $D'_{\mathcal{A}||\mathcal{E}}$ is the $\mathcal{E}$-extension of the domain $dom(f^{tr})$ of f^{tr}.*
2. *we call the $\mathcal{E}$-extension of (f, f^{tr}, f^{ex}) the matching-execution (g, g^{tr}, g^{ex}) from $\mathcal{A}||\mathcal{E}$ to $\mathcal{B}||\mathcal{E}$ induced by g and $dom(g^{tr})$.*

Finally we can states the main result of this paragraph, i.e. theorem 94 of executions-matching composability.

▶ **Theorem 94** (Composability of executions-matching)**.** *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA. Let $\mathcal{E}$ be partially-compatible with both $\mathcal{A}$ and $\mathcal{B}$. Let (f, f^{tr}, f^{ex}) be an execution-matching from $\mathcal{A}$ to $\mathcal{B}$ where g represents the $\mathcal{E}$-extension of f. If $range(g) \subset Q_{\mathcal{B}||\mathcal{E}}$, then the $\mathcal{E}$-extension of (f, f^{tr}, f^{ex}) is a matching-execution (g, g^{tr}, g^{ex}) from $\mathcal{A}||\mathcal{E}$ to $\mathcal{B}||\mathcal{E}$ induced by g and $dom(g^{tr})$.*

Proof. We repeated the previous definition, while an executions-matching only need a states-matching g and a set $dom(g^{tr})$ of transitions eligible to transitions-matching domain from g which is provided by construction. ◀

Here we give some properties preserved by $\mathcal{E}$-extension of an executions-matching.

▶ **Lemma 95** (Some properties preserved by $\mathcal{E}$-extension of an executions-matching)**.** *Let $\mathcal{A}$ and $\mathcal{B}$ be PSIOA. Let (f, f^{tr}, f^{ex}) be an execution-matching from $\mathcal{A}$ to $\mathcal{B}$.*

1. *If f is bijective and f^{-1} is complete, then for every PSIOA $\mathcal{E}$ partially-compatible with $\mathcal{A}$, $\mathcal{E}$ is partially-compatible with $\mathcal{B}$.*
2. *Let $\mathcal{E}$ partially-compatible with both $\mathcal{A}$ and $\mathcal{B}$, let g be the $\mathcal{E}$-extension of f.*

a. *If f is bijective and f^{-1} is complete, then $range(g) = Q_{\mathcal{B}||\mathcal{E}}$ and so we can talk about the $\mathcal{E}$-extension of (f, f^{tr}, f^{ex})*

b. *If (f, f^{tr}) is a bijective complete transition-matching, (g, g^{tr}) is a bijective complete transition-matching. (And (f, f^{tr}, f^{ex}) and (g, g^{tr}, g^{ex}) are bijective complete execution-matching.)*

c. *If f is strong, then g is strong*

3. *Let $\mathcal{E}$ partially-compatible with both $\mathcal{A}$ and $\mathcal{B}$, let g be the $\mathcal{E}$-extension of f. Let assume $range(g) \subseteq Q_{\mathcal{B}||\mathcal{E}}$. Let (g, g^{tr}, g^{ex}) be the $\mathcal{E}$-extension of (f, f^{tr}, f^{ex})*

a. *If the restriction $\tilde{f}^{ex} : E'_{\mathcal{A}} \subseteq Execs(\mathcal{A}) \rightarrow \tilde{E}_{\mathcal{B}} \subseteq Execs(\mathcal{B})$ is surjective, then $\tilde{g}^{ex}$: $\{\alpha \in Execs(\mathcal{A}||\mathcal{E}) | \alpha \upharpoonright \mathcal{A} \in E'_{\mathcal{A}}\} \rightarrow \{\pi \in Execs(\mathcal{B}||\mathcal{E}) | \pi \upharpoonright \mathcal{B} \in \tilde{E}_{\mathcal{B}}\}$ is surjective*

b. *If f is strong, g is strong.*

Proof. **1.** We need to show that every pseudo-execution of $(\mathcal{B}, \mathcal{E})$ ends on a compatible state. Let $\pi = q^0 a^1 q^1 ... a^n q^n$ be a finite pseudo-execution of $(\mathcal{B}, \mathcal{E})$. We note $\alpha = (f^{-1}(q^0_{\mathcal{B}}), q^0_{\mathcal{E}}) a^1 (f^{-1}(q^1_{\mathcal{B}}), q^1_{\mathcal{E}}) ... a^n (f^{-1}(q^n_{\mathcal{B}}), q^n_{\mathcal{E}})$. The proof is in two steps. First, we show by induction that $\alpha = (f^{-1}(q^0_{\mathcal{B}}), q^0_{\mathcal{E}}) a^1 (f^{-1}(q^1_{\mathcal{B}}), q^1_{\mathcal{E}}) ... a^n (f^{-1}(q^n_{\mathcal{B}}), q^n_{\mathcal{E}})$ is an execution of $\mathcal{A}||\mathcal{E}$. Second, we deduce that it means $(f^{-1}(q^n_{\mathcal{B}}), q^n_{\mathcal{E}})$ is a compatible state of $(\mathcal{A}, \mathcal{E})$ which means that a fortiori, $(q^n_{\mathcal{B}}, q^n_{\mathcal{E}})$ is a compatible state of $(\mathcal{B}, \mathcal{E})$ which ends the proof.

- First, we show by induction that α is an execution of $\mathcal{A}||\mathcal{E}$. We have $(f^{-1}(\bar{q}_{\mathcal{B}}), \bar{q}_{\mathcal{E}}) = (\bar{q}_{\mathcal{A}}, \bar{q}_{\mathcal{E}})$ which ends the basis.
  Let assume $(f^{-1}(q^0_{\mathcal{B}}), q^0_{\mathcal{E}}) a^1 (f^{-1}(q^1_{\mathcal{B}}), q^1_{\mathcal{E}}) ... a^k (f^{-1}(q^k_{\mathcal{B}}), q^k_{\mathcal{E}})$ is an execution of $\mathcal{A}||\mathcal{E}$. Hence $(f^{-1}(q^k_{\mathcal{B}}), q^k_{\mathcal{E}})$ is a compatible state of $(\mathcal{A}, \mathcal{E})$ which means that a fortiori q^k is a compatible state of $(\mathcal{B}, \mathcal{E})$ because of signature preservation of f.
  For the same reason, $\widehat{sig}(\mathcal{B}||\mathcal{E})(q^k) = \widehat{sig}(\mathcal{A}, \mathcal{E})((f^{-1}(q^k_{\mathcal{B}}), q^k_{\mathcal{E}}))$, so $a^{k+1} \in \widehat{sig}(\mathcal{A}, \mathcal{E})((f^{-1}(q^k_{\mathcal{B}}), q^k_{\mathcal{E}}))$.
  Then we use the completeness of $(f^{-1}, (f^{tr})^{-1})$, to obtain the fact that either $\eta_{(\mathcal{B}, q^k_{\mathcal{B}}, a^{k+1})} \in dom((f^{tr})^{-1})$ or $a^{k+1} \notin \widehat{sig(\mathcal{B})}(q^k_{\mathcal{B}})$ (and we recall the convention that in this second case $\eta_{(\mathcal{B}, q^k_{\mathcal{B}}, a^{k+1})} = \delta_{q^k_{\mathcal{B}}}$). which means either $(f^{-1}(q^k_{\mathcal{B}}), a^{k+1}, \eta_{(\mathcal{A}, f^{-1}(q^k_{\mathcal{B}}), a^{k+1})})$ is a transition of $\mathcal{A}$ that ensures $\forall q'' \in supp(\eta_{(\mathcal{B}, q^k_{\mathcal{B}}, a^{k+1})}), f^{-1}(q'') \in supp(\eta_{(\mathcal{A}, f^{-1}(q^k_{\mathcal{B}}), a^{k+1})})$ or $a^{k+1} \notin \widehat{sig(\mathcal{A})}(f^{-1}(q^k_{\mathcal{B}}))$ (and we recall the convention that in this second case $\eta_{(\mathcal{A}, f^{-1}(q^k_{\mathcal{B}}), a^{k+1})} = \delta_{f^{-1}(q^k_{\mathcal{B}})}$). Thus for every $(q'', q''') \in supp(\eta_{(\mathcal{B}, \mathcal{E}), q^k, a^{k+1})})$, $(f^{-1}(q''), q''') = g^{-1}((q'', q''')) \in supp(\eta_{(\mathcal{A}, \mathcal{E}), g^{-1}(q^k), a^{k+1})})$ namely for $(q'', q''') = (q^{k+1}_{\mathcal{B}}, q^{k+1}_{\mathcal{E}})$. Hence, $(f^{-1}(q^{k+1}_{\mathcal{B}}), q^{k+1}_{\mathcal{E}})$ is reachable by $(\mathcal{A}, \mathcal{E})$ which means the alternating sequence $(f^{-1}(q^0_{\mathcal{B}}), q^0_{\mathcal{E}}) a^1 (f^{-1}(q^1_{\mathcal{B}}), q^1_{\mathcal{E}}) ... a^k (f^{-1}(q^k_{\mathcal{B}}), q^k_{\mathcal{E}}) a^k (f^{-1}(q^k_{\mathcal{B}}), q^k_{\mathcal{E}}) a^{k+1} (f^{-1}(q^{k+1}_{\mathcal{B}}), q^{k+1}_{\mathcal{E}})$ is an execution of $\mathcal{A}||\mathcal{E}$. Thus by induction α is an execution of $\mathcal{A}||\mathcal{E}$.
- Since $\mathcal{A}$ and $\mathcal{E}$ are partially-compatible $(f^{-1}(q^n_{\mathcal{B}}), q^n_{\mathcal{E}})$ is a state of $\mathcal{A}||\mathcal{E}$, so $(f^{-1}(q^n_{\mathcal{B}}), q^n_{\mathcal{E}})$ is a compatible state of $(\mathcal{A}, \mathcal{E})$ which means $(q^k_{\mathcal{B}}, q^k_{\mathcal{E}})$ is a fortiori a compatible state of $(\mathcal{B}, \mathcal{E})$. Hence every reachable state of $(\mathcal{B}, \mathcal{E})$ is compatible which means $\mathcal{B}$ and $\mathcal{E})$ are partially compatible which ends the proof.

2. a.
- Let $(q^n_{\mathcal{B}}, q^n_{\mathcal{E}}) \in Q_{\mathcal{B}||\mathcal{E}}$. This state is reachable, so we note $\pi = (q^0_{\mathcal{B}}, q^0_{\mathcal{E}}) a^1 (q^1_{\mathcal{B}}, q^1_{\mathcal{E}}) ... a^n (q^n_{\mathcal{B}}, q^n_{\mathcal{E}})$ the execution of $\mathcal{B}||\mathcal{E}$. Thereafter, we note $\alpha = (f^{-1}(q^0_{\mathcal{B}}), q^0_{\mathcal{E}}) a^1 (f^{-1}(q^1_{\mathcal{B}}), q^1_{\mathcal{E}}) ... a^n (f^{-1}(q^n_{\mathcal{B}}), q^n_{\mathcal{E}})$. We can show by induction that α is an execution of $\mathcal{A}||\mathcal{E}$. The proof is exactly the same than in 1.
  Hence α is an execution of $\mathcal{A}||\mathcal{E}$ which means $(f^{-1}(q^n_{\mathcal{B}}), q^n_{\mathcal{E}})$ is a state of $\mathcal{A}||\mathcal{E}$ and then $g((f^{-1}(q^n_{\mathcal{B}}), q^n_{\mathcal{E}})) = (q^n_{\mathcal{B}}, q^n_{\mathcal{E}})$ to finally prove that it exists q^* s.t. $g(q^*) = (q^n_{\mathcal{B}}, q^n_{\mathcal{E}})$ which means $states(\mathcal{B}||\mathcal{E}) \subseteq dom(g)$.

We can reuse the proof of 1. to show that if $q \in Q_{\mathcal{A}||\mathcal{E}}$, then $g(q) \in Q_{\mathcal{B}||\mathcal{E}}$ which means $dom(g) \subseteq Q_{\mathcal{B}||\mathcal{E}}$.
Hence $dom(g) = Q_{\mathcal{B}||\mathcal{E}}$.
- We can apply the previous lemma 92 to obtain the eligibility of $D_{\mathcal{A}||\mathcal{E}}$.

b. Let assume (f, f^{tr}) are bijective. The bijectivity of g is immediate $g(.,.) = (f(.), Id(.))$. The bijectivity of g^{tr} is also immediate since $g^{tr} : \eta_{(\mathcal{A},q_{\mathcal{A}},a)} \otimes \eta_{(\mathcal{E},q_{\mathcal{E}},a)} \rightarrow f^{tr}(\eta_{(\mathcal{A},q_{\mathcal{A}},a)}) \otimes \eta_{(\mathcal{E},q_{\mathcal{E}},a)}$ with f^{tr} bijective.

c. Immediate, since in this case $sig(\mathcal{A})(q_{\mathcal{A}}) = sig(\mathcal{B})(f(q_{\mathcal{A}}))$ implies $sig(\mathcal{A}||\mathcal{E})((q_{\mathcal{A}}, q_{\mathcal{E}})) = sig(\mathcal{B}||\mathcal{E})((f(q_{\mathcal{A}}), q_{\mathcal{E}}))$.

3. a. Let $\pi = ((q^0_{\mathcal{B}}, q^0_{\mathcal{E}}), a^1, (q^1_{\mathcal{B}}, q^1_{\mathcal{E}}), ..., a^n, (q^n_{\mathcal{B}}, q^n_{\mathcal{E}})) \in Execs(\mathcal{B}||\mathcal{E})$ with $\pi \upharpoonright \mathcal{B} = \hat{q}^0_{\mathcal{B}}, \hat{a}^1, \hat{q}^1_{\mathcal{B}}, ..., \hat{a}^m, \hat{q}^m_{\mathcal{B}} \in \tilde{E}_{\mathcal{B}}$, where the monotonic function $k : [0, n] \rightarrow [0, m]$, verifies $\forall i \in [0, n], k(i) \in [0, m], q^i_{\mathcal{B}} = \hat{q}^{k(i)}_{\mathcal{B}}$ By surjectivity of f^{ex} we have $\hat{\alpha} = \hat{q}^0_{\mathcal{A}}, \hat{a}^1, \hat{q}^1_{\mathcal{A}}, ..., \hat{a}^m, \hat{q}^m_{\mathcal{A}} \in E'_{\mathcal{A}}$ s.t. $f^{ex}(\hat{\alpha}) = \pi \upharpoonright \mathcal{B}$. We note $\alpha = (q^0_{\mathcal{A}}, q^0_{\mathcal{E}})a^1(q^1_{\mathcal{A}}, q^1_{\mathcal{E}})...a^n(q^n_{\mathcal{A}}, q^n_{\mathcal{E}})$ where $\forall i \in [0, n], q^i_{\mathcal{A}} = \hat{q}^{k(i)}_{\mathcal{A}}$. Hence, $\forall i \in [0, n], g((q^i_{\mathcal{A}}, q^i_{\mathcal{E}})) = (q^i_{\mathcal{B}}, q^i_{\mathcal{E}})$. Moreover, by signature preservation, $\forall i \in [0, n-1], a^{i+1} \in \widehat{sig}(\mathcal{A})(q^i_{\mathcal{A}}) \cup \widehat{sig}(\mathcal{E})(q^i_{\mathcal{E}})$. Furthermore, $\forall i \in [0, n-1]$. $(q^{i+1}_{\mathcal{A}}, q^{i+1}_{\mathcal{E}}) \in supp(\eta_{(\mathcal{A},q^i_{\mathcal{A}},a^i)} \otimes \eta_{(\mathcal{B},q^i_{\mathcal{B}},a^i)})$ since $(q^{i+1}_{\mathcal{B}}, q^{i+1}_{\mathcal{E}}) \in supp(\eta_{(\mathcal{B},q^i_{\mathcal{B}},a^i)} \otimes \eta_{(\mathcal{B},q^i_{\mathcal{B}},a^i)})$, $(q^i_{\mathcal{B}}, a^i, \eta_{(\mathcal{B},q^i_{\mathcal{B}},a^i)}) = f^{tr}(q^i_{\mathcal{A}}, a^i, \eta_{(\mathcal{A},q^i_{\mathcal{A}},a^i)})$ and $q^{i+1}_{\mathcal{B}} = f(q^{i+1}_{\mathcal{A}})$. Thus, $\alpha \in Execs(\mathcal{A}||\mathcal{E})$. Finally, by signature preservation of f, $\forall i \in [1, n]\widehat{sig}(\mathcal{A})(q_{\mathcal{A}}) = \widehat{sig}(\mathcal{B})(q_{\mathcal{B}})$, which lead us to $\alpha \upharpoonright \mathcal{A} = \hat{\alpha} \in E'_{\mathcal{A}}$. So for every $\pi \in Execs(\mathcal{B}||\mathcal{E})$ with $\pi \upharpoonright \mathcal{B} \in \tilde{E}_{\mathcal{B}}$, it exists $\alpha \in Execs(\mathcal{A}||\mathcal{E})$ with $\alpha \upharpoonright \mathcal{A} \in E'_{\mathcal{A}}$ which ends the proof.

b. Immediate by rules of composition of signature: $\forall (q_{\mathcal{A}}, q_{\mathcal{E}}) \in states(\mathcal{A}||\mathcal{E})$, $\forall (q_{\mathcal{B}}, q_{\mathcal{E}}) \in states(\mathcal{B}||\mathcal{E})$ if $sig(\mathcal{A})(q_{\mathcal{A}}) = sig(\mathcal{B})(q_{\mathcal{B}})$, then $sig(\mathcal{A}||\mathcal{E})(q_{\mathcal{A}}, q_{\mathcal{E}}) = sig(\mathcal{B}||\mathcal{E}))(q_{\mathcal{B}}, q_{\mathcal{E}})$.

◀

We are ready to states the composability of semantic equivalence.

▶ **Theorem 96** (composability of semantic equivalence)**.** *Let $\mathcal{A}$ and $\mathcal{B}$ be PSIOA semantically-equivalent. Then for every PSIOA $\mathcal{E}$:*
- *$\mathcal{E}$ is partially-compatible with $\mathcal{A} \iff \mathcal{E}$ is partially-compatible with $\mathcal{B}$*
- *if $\mathcal{E}$ is partially-compatible with both $\mathcal{A}$ and $\mathcal{B}$, then $\mathcal{A}||\mathcal{E}$ and $\mathcal{B}||\mathcal{E}$ are semantically-equivalent PSIOA.*

Proof.
- The first item ($\mathcal{E}$ is partially-compatible with $\mathcal{A} \iff \mathcal{E}$ is partially-compatible with $\mathcal{B}$) comes from lemma 95, first item.
- The second item (if $\mathcal{E}$ is partially-compatible with both $\mathcal{A}$ and $\mathcal{B}$, then $\mathcal{A}||\mathcal{E}$ and $\mathcal{B}||\mathcal{E}$ are semantically-equivalent PSIOA) comes from lemma 95, second item.

◀

A weak complete bijective transition-matching implies a weak complete bijective execution-matching which means the two automata are completely sementically equivalent modulo some hiding operation that implies that some PSIOA are partially-compatible with one of the automaton and not with the other and that the traces are not necessarily the same ones.

composition of continuation of executions-matching

Here we define $\mathcal{E}$-extension of continued executions-matching in the same way we defined $\mathcal{E}$-extension of executions-matching just before.

▶ **Definition 97** ($\mathcal{E}$-extension of continued executions-matching)**.** *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA. Let $\mathcal{E}$ be partially-compatible with both $\mathcal{A}$ and $\mathcal{B}$. Let (f, f^{tr}, f^{ex}) be an executions-matching from $\mathcal{A}$ to $\mathcal{B}$. Let $((f, f^+), f^{tr,+}, f^{ex,+})$ be the $(f^+, D''_{\mathcal{A}})$-continuation of (f, f^{tr}, f^{ex}) (where* *by definition $D''_{\mathcal{A}} \setminus dom(f^{tr})$ respect the properties of matched states preservation and extension of equitable corresponding distribution from definition 81). If the respective $\mathcal{E}$-extension of f and f^{+}, noted g and g^{+}, verifie $range(g) \cup range(g^{+}) \subseteq (\mathcal{B}||\mathcal{E})$, we define the $\mathcal{E}$-extension of $((f, f^{+}), f^{tr,+}, f^{ex,+})$ as $((g, g^{+}), g^{tr,+}, g^{ex,+})$, where*

- (g, g^{tr}, g^{ex}) *is the $\mathcal{E}$-extension of* (f, f^{tr}, f^{e})
- $g^{tr,+} : (q, a, \eta_{(\mathcal{A}||\mathcal{E}),q,a}) \in D''_{\mathcal{A}||\mathcal{E}} \mapsto (g(q), a, \eta_{(\mathcal{A}||\mathcal{E}),g(q),a})$ *where $D''_{\mathcal{A}||\mathcal{E}}$ is the $\mathcal{E}$-extension of $dom(f^{tr,+})$*
- $\forall \alpha' = \alpha^\frown q, a, q'$, *with* $\alpha' \in dom(g^{ex})$, *if* $(q, a, \eta_{(\mathcal{A}||\mathcal{E}),q,a}) \in dom(g^{tr})$ $g^{ex,+}(\alpha) = g^{ex}(\alpha)$ *and if* $(q, a, \eta_{(\mathcal{A}||\mathcal{E}),q,a}) \in dom(g^{tr,+}) \setminus dom(g^{tr})$ $g^{ex,+}(\alpha') = g^{ex}(\alpha)^\frown g(q), a, g^{+}(q)$

▶ **Lemma 98** (Commutativity of continuation and extension)**.** *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA. Let $\mathcal{E}$ be partially-compatible with both $\mathcal{A}$ and $\mathcal{B}$. Let (f, f^{tr}, f^{ex}) be an executions-matching from $\mathcal{A}$ to $\mathcal{B}$. Let $((f, f^{+}), f^{tr,+}, f^{ex,+})$ be the $(f^{+}, D''_{\mathcal{A}})$-continuation of (f, f^{tr}, f^{ex}) (where by definition $D''_{\mathcal{A}}$ respect the properties of matched states preservation and extension of equitable corresponding distribution from definition 81). Let*

- (g, g^{tr}, g^{ex}) *be the $\mathcal{E}$-extension of (f, f^{tr}, f^{e}) verifying $range(g) \subseteq Q_{\mathcal{B}||\mathcal{E}}$,*
- $D''^{,(c,e)}_{\mathcal{A}||\mathcal{E}}$ *the $\mathcal{E}$-extension of $dom(f^{tr,+})$, i.e.* $D''^{,(c,e)}_{\mathcal{A}||\mathcal{E}} = \{((q_{\mathcal{A}}, q_{\mathcal{E}}), a, \eta_{(\mathcal{A}||\mathcal{E},(q_{\mathcal{A}},q_{\mathcal{E}}),a)}) \in D_{\mathcal{A}||\mathcal{E}} | q_{\mathcal{A}} \in dom(f) \wedge [(q_{\mathcal{A}}, a, \eta_{(\mathcal{A},q_{\mathcal{A}},a)}) \in dom(f^{tr,+}) \vee a \notin \widehat{sig}(\mathcal{A})(q_{\mathcal{A}})]\}$.
- $g^{+}_{(c,e)}$ *be the $\mathcal{E}$-extension of f^{+}*

Then

1. $D''_{\mathcal{A}||\mathcal{E}} \setminus dom(g^{tr})$ *verifies matched states preservation and extension of equitable corresponding distribution.*
2. *the* $(g^{+}_{(c,e)}, (D''^{,(c,e)}_{\mathcal{A}||\mathcal{E}}))$*-continuation of* (g, g^{tr}, g^{ex}), *noted* $((g, g^{+}_{(c,e)}), g^{tr,+}_{(c,e)}, g^{ex,+}_{(c,e)})$ *is equal to the $\mathcal{E}$-extension of* $((f, f^{+}), f^{tr,+}, f^{ex,+})$, *noted* $((g, g^{+}_{(e,c)}), g^{tr,+}_{(e,c)}, g^{ex,+}_{(e,c)})$.

We show that the operation of continuation and extension are in fact commutative.

Proof. We start by showing $D''^{,(c,e)}_{\mathcal{A}||\mathcal{E}} \setminus dom(g^{tr})$ verifies matched states preservation and extension of equitable corresponding distribution. By definition 81 of $\mathcal{E}$-extension, $D''^{,(c,e)}_{\mathcal{A}||\mathcal{E}} = \{((q_{\mathcal{A}}, q_{\mathcal{E}}), a, \eta_{(\mathcal{A}||\mathcal{E},(q_{\mathcal{A}},q_{\mathcal{E}}),a)}) \in D_{\mathcal{A}||\mathcal{E}} | q_{\mathcal{A}} \in dom(f) \wedge [(q_{\mathcal{A}}, a, \eta_{(\mathcal{A},q_{\mathcal{A}},a)}) \in dom(f^{tr,+}) \vee a \notin \widehat{sig}(\mathcal{A})(q_{\mathcal{A}})]\}$, while $dom(g^{tr}) = \{((q_{\mathcal{A}}, q_{\mathcal{E}}), a, \eta_{(\mathcal{A}||\mathcal{E},(q_{\mathcal{A}},q_{\mathcal{E}}),a)}) \in D_{\mathcal{A}||\mathcal{E}} | q_{\mathcal{A}} \in dom(f) \wedge [(q_{\mathcal{A}}, a, \eta_{(\mathcal{A},q_{\mathcal{A}},a)}) \in dom(f^{tr}) \vee a \notin \widehat{sig}(\mathcal{A})(q_{\mathcal{A}})]\}$.

Thus $D''^{,(c,e)}_{\mathcal{A}||\mathcal{E}} \setminus dom(g^{tr}) = \{((q_{\mathcal{A}}, q_{\mathcal{E}}), a, \eta_{(\mathcal{A}||\mathcal{E},(q_{\mathcal{A}},q_{\mathcal{E}}),a)}) \in D_{\mathcal{A}||\mathcal{E}} | q_{\mathcal{A}} \in dom(f) \wedge [(q_{\mathcal{A}}, a, \eta_{(\mathcal{A},q_{\mathcal{A}},a)}) \in dom(f^{tr,+}) \setminus dom(f^{tr})]\}$ (*)

Let $tr = ((q_{\mathcal{A}}, q_{\mathcal{E}}), a, \eta_{(\mathcal{A}||\mathcal{E}),(q_{\mathcal{A}},q_{\mathcal{E}}),a}) \in D''^{,(c,e)}_{\mathcal{A}||\mathcal{E}} \setminus dom(g^{tr})$, then

- Matched states preservation: By (*) $q_{\mathcal{A}} \in dom(f)$ which leads immediately to $(q_{\mathcal{A}}, q_{\mathcal{E}}) \in dom(g)$
- Extension of equitable corresponding distribution: $\forall (q''_{\mathcal{A}}, q''_{\mathcal{E}}) \in supp(\eta_{(\mathcal{A}||\mathcal{E},(q_{\mathcal{A}},q_{\mathcal{E}}),a)})$, $(q''_{\mathcal{A}}, q''_{\mathcal{E}}) \in supp(\eta_{(\mathcal{A}q_{\mathcal{A}},a)} \otimes \eta_{(\mathcal{E},q_{\mathcal{E}},a)})$ with $\eta_{(\mathcal{A}q_{\mathcal{A}},a)} \in dom(f^{tr,+}) \setminus dom(f^{tr})$ by (*) which means $q''_{\mathcal{A}} \in dom(f^{+})$ and $\eta_{(\mathcal{A}q_{\mathcal{A}},a)}(q''_{\mathcal{A}}) = \eta_{(\mathcal{B}f(q_{\mathcal{A}}),a)}(f^{+}(q''_{\mathcal{A}}))$ and so $(q''_{\mathcal{A}}, q''_{\mathcal{E}}) \in dom(g^{+})$ and $\eta_{(\mathcal{A},q_{\mathcal{A}},a)} \otimes \eta_{(\mathcal{E},q_{\mathcal{E}},a)}(q''_{\mathcal{A}}, q''_{\mathcal{E}}) = \eta_{(\mathcal{A},q_{\mathcal{A}},a)}(q''_{\mathcal{A}}) \cdot \eta_{(\mathcal{E},q_{\mathcal{E}},a)}(q''_{\mathcal{E}}) = \eta_{(\mathcal{B},f(q_{\mathcal{A}}),a)}(f^{+}(q''_{\mathcal{A}})) \cdot \eta_{(\mathcal{E},q_{\mathcal{E}},a)}(q''_{\mathcal{E}}) = \eta_{(\mathcal{B},f(q_{\mathcal{A}}),a)} \otimes \eta_{(\mathcal{E},q_{\mathcal{E}},a)}(f^{+}(q''_{\mathcal{A}}), q''_{\mathcal{E}}) = \eta_{(\mathcal{B}||\mathcal{E},g(q_{\mathcal{A}},q_{\mathcal{E}}),a)}(g^{+}(q''_{\mathcal{A}}, q''_{\mathcal{E}}))$

We have shown that $D''^{,(c,e)}_{\mathcal{A}||\mathcal{E}} \setminus dom(g^{tr})$ verifies matched states preservation and extension of equitable corresponding distribution.

Now, we show the second point.

- By definition 81 of continuation, $g^{+}_{(c,e)} = g^{+}_{(e,c)}$.

- We prove $dom(g^{tr,+}_{(c,e)}) = dom(g^{tr,+}_{(e,c)}) = D''^{,(c,e)}_{\mathcal{A}||\mathcal{E}}$. By definition 81 of continuation, $dom(g^{tr,+}_{(e,c)}) = dom(g^{tr}) \cup D''^{,(c,e)}_{\mathcal{A}||\mathcal{E}} = \{((q_\mathcal{A}, q_\mathcal{E}), a, \eta_{(\mathcal{A}||\mathcal{E},(q_\mathcal{A},q_\mathcal{E}),a)}) \in D_{\mathcal{A}||\mathcal{E}} | q_\mathcal{A} \in dom(f) \wedge [(q_\mathcal{A}, a, \eta_{(\mathcal{A},q_\mathcal{A},a)}) \in dom(f^{tr}) \vee a \notin \widehat{sig}(\mathcal{A})(q_\mathcal{A})]\} \cup \{((q_\mathcal{A}, q_\mathcal{E}), a, \eta_{(\mathcal{A}||\mathcal{E},(q_\mathcal{A},q_\mathcal{E}),a)}) \in D_{\mathcal{A}||\mathcal{E}} | q_\mathcal{A} \in dom(f) \wedge [(q_\mathcal{A}, a, \eta_{(\mathcal{A},q_\mathcal{A},a)}) \in dom(f^{tr,+}) \vee a \notin \widehat{sig}(\mathcal{A})(q_\mathcal{A})]\} = \{((q_\mathcal{A}, q_\mathcal{E}), a, \eta_{(\mathcal{A}||\mathcal{E},(q_\mathcal{A},q_\mathcal{E}),a)}) \in D_{\mathcal{A}||\mathcal{E}} | q_\mathcal{A} \in dom(f) \wedge [(q_\mathcal{A}, a, \eta_{(\mathcal{A},q_\mathcal{A},a)}) \in dom(f^{tr,+}) \vee a \notin \widehat{sig}(\mathcal{A})(q_\mathcal{A})]\} = D''^{,(c,e)}_{\mathcal{A}||\mathcal{E}}$.
  Parrallely, by definition 93 of $\mathcal{E}$-extension, $dom(g^{tr,+}_{(c,e)}) = \{((q_\mathcal{A}, q_\mathcal{E}), a, \eta_{(\mathcal{A}||\mathcal{E},(q_\mathcal{A},q_\mathcal{E}),a)}) \in D_{\mathcal{A}||\mathcal{E}} | q_\mathcal{A} \in dom(f) \wedge [(q_\mathcal{A}, a, \eta_{(\mathcal{A},q_\mathcal{A},a)}) \in dom(f^{tr,+}) \vee a \notin \widehat{sig}(\mathcal{A})(q_\mathcal{A})]\} = D''^{,(c,e)}_{\mathcal{A}||\mathcal{E}}$. Thus $dom(g^{tr,+}_{(c,e)}) = dom(g^{tr,+}_{(e,c)}) = D''^{,(c,e)}_{\mathcal{A}||\mathcal{E}}$.
- We prove $g^{tr,+}_{(c,e)} = g^{tr,+}_{(e,c)}$ Let $((q_\mathcal{A}, q_\mathcal{E}), a, \eta_{(\mathcal{A}||\mathcal{E},(q_\mathcal{A},q_\mathcal{E}),a)}) \in D''_{\mathcal{A}||\mathcal{E}}$.
  By definition 93 of $\mathcal{E}$-extension, $g^{tr,+}_{(c,e)}(((q_\mathcal{A}, q_\mathcal{E}), a, \eta_{(\mathcal{A}||\mathcal{E},(q_\mathcal{A},q_\mathcal{E}),a)})) = (g(q_\mathcal{A}, q_\mathcal{E}), a, \eta_{(\mathcal{A}||\mathcal{E},g(q_\mathcal{A},q_\mathcal{E}),a)}))$, while by definition 81 of continuation, $g^{tr,+}_{(e,c)}(((q_\mathcal{A}, q_\mathcal{E}), a, \eta_{(\mathcal{A}||\mathcal{E},(q_\mathcal{A},q_\mathcal{E}),a)})) = (g(q_\mathcal{A}, q_\mathcal{E}), a, \eta_{(\mathcal{A}||\mathcal{E},g(q_\mathcal{A},q_\mathcal{E}),a)}))$.
  We can remark that properties of equitable corresponding distribution are not conflicting since $dom(g^{tr,+}_{c,e}) \setminus dom(g^{tr}) = dom(g^{tr,+}_{e,c}) \setminus dom(g^{tr})$.
- $g^{e,+}_{(e,c)}$ and $g^{e,+}_{(c,e)}$ are entirely defined by $((g, g^{+}_{(e,c)}), (g^{tr}, g^{tr,+}_{(e,c)}))$ and $((g, g^{+}_{(c,e)}), (g^{tr}, g^{tr,+}_{(c,e)}))$ that are equal.

◀

application for renaming and hiding

Before dealing with PCA-executions-matching, we state two intuitive theorems of executions-matching after renaming and hiding operations.

▶ **Theorem 99.** *(strong complete bijective execution-matching after renaming) Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA and $ren : Q_\mathcal{A} \rightarrow Q_\mathcal{B}$ s. t. $\mathcal{B} = ren(\mathcal{A})$. $(ren, ren^{tr}, ren^{ex})$ is a strong complete bijective execution-matching from $\mathcal{A}$ to $\mathcal{B}$ with $dom(ren^{tr}) = D_\mathcal{A}$.*

Proof. By definition *ren* ensures starting state preservation and strong signature preservation. By definition *ren* is a complete bijection, which implies matched state preservation. The equitable corresponding distribution is also ensured by definition of *ren*. Hence, all the properties are ensured ◀

▶ **Theorem 100.** *(weak complete bijective executions-matching after hiding) Let $\mathcal{A}$ be a PSIOA. Let h defined on $states(\mathcal{A})$, s.t. $\forall q \in Q_\mathcal{A}, h(q) \subseteq out(\mathcal{A})(q)$. Let $\mathcal{B} = hiding(\mathcal{A}, h)$. Let Id the identity function from $states(\mathcal{A})$ to $states(\mathcal{B}) = Q_\mathcal{A}$. Then (Id, Id^{tr}, Id^{ex}) is a weak complete bijective execution-matching from $\mathcal{A}$ to $\mathcal{B}$.*

Proof. By definition Id ensures starting state preservation and weak signature preservation. By definition Id is a complete bijection, which implies matched state preservation. The equitable corresponding distribution is also ensured by definition of *hiding*. Hence, all the properties are ensured ◀

10.2 PCA-matching execution

Here we extend the notion of executions-matching to PCA. In practice, we will build executions-matchings that preserve the sequence of configurations visited by concerned executions. Hence, the definition of PCA states-matching is slightly more restrictive to capture this notion of configuration equivalence (modulo action hiding operation), while the other definitions are exactly the same ones.

matching execution

▶ **Definition 101** (PCA states-matching). *Let X and Y be two PCA and let $f : Q'_X \subset Q_X \rightarrow Q_Y$ be a mapping s.t. :*

- *Starting state preservation: If $\bar{q}_X \in Q'_X$, then $f(\bar{q}_X) = \bar{q}_Y$.*
- *Configuration preservation (modulo hiding): $\forall (q, q') \in Q'_X \times Q_Y$, s.t. $q' = f(q)$, if $auts(config(X)(q)) = (\mathcal{A}_1, ..., \mathcal{A}_n)$, then $auts(config(Y)(q')) = (\mathcal{A}'_1, ..., \mathcal{A}'_n)$ where $\forall i \in [1 : n], \mathcal{A}_i = hide(\mathcal{A}'_i, h_i)$ with h_i defined on $states(\mathcal{A}'_i)$, s. t. $h_i(q_{\mathcal{A}'_i}) \subseteq out(\mathcal{A}'_i)(q_{\mathcal{A}'_i})$ (resp. s.t. $h_i(q_{\mathcal{A}'_i}) = \emptyset$, that is $\mathcal{A}_i = \mathcal{A}'_i$)*
- *Hiding preservation (modulo hiding): $\forall (q, q') \in Q'_X \times Q_Y$, s.t. $q' = f(q)$, hidden-actions$(X)(q)$ = hidden-actions$(Y)(q') \cup h^+(q')$ where h^+ defined on $states(Y)$, s. t. $h^+(q_Y) \subseteq out(Y)(q_Y)$ (resp. s.t. $h^+(q_Y) = \emptyset$, that is hidden-actions$(X)(q)$ = hidden-actions$(Y)(q')$)*
- *Creation preservation $\forall (q, q') \in Q'_X \times Q_Y$, s.t. $q' = f(q)$, $\forall a \in \widehat{sig}(X)(q) = \widehat{sig}(Y)(q')$, $created(X)(q)(a) = created(Y)(q')(a)$.*

then we say that f is a weak *(resp.* strong*)* PCA states-matching *from X to Y. If $Q'_X = Q_X$, then we say that f is a* complete *(weak or strong) PCA states-matching from X to Y.*

We naturally obtain that a PCA states-matching is a PSIOA states-matching:

▶ **Lemma 102** (A PCA states-matching is a PSIOA states-matching). *If f is a weak (resp. strong) PCA states-matching from X to Y, then f is a PSIOA states-matching from $psioa(X)$ to $psioa(Y)$ (in the sense of definition 75). (The converse is not necessarily true.)*

Proof. The signature preservation immediately comes from the configuration preservation and the hiding preservation. ◀

Now, all the definitions from definition 76 to definition 78 of previous subsections are the same that is:

▶ **Definition 103** (PCA transitions-matching and PCA executions-matching). *Let X and Y be two PCA and let $f : Q'_X \subset Q_X \rightarrow Q_Y$ be a PCA states-matching from X to Y.*

- *Let $D'_X \subseteq D_X$ be a subset of transitions, D'_X is* eligible to PCA transitions-matching domain *from f if it is eligible to PSIOA transitions-matching domain from f according to definition 76.*
- *Let $D'_X \subseteq D_X$ be a subset of transitions eligible to PCA transitions-matching domain from f. We define the* PCA transitions-matching *(f, f^{tr}) induced by the PCA states-matching f and the subset of transitions D'_X as the PSIOA transitions-matching induced by the PSIOA states-matching f and the subset of transitions D'_X according to definition 77.*
- *Let $f^{tr} : D'_X \subseteq D_X \rightarrow D_Y$ s.t. (f, f^{tr}) is a PCA transitions-matching, we define the PCA executions-matching (f, f^{tr}, f^{ex}) induced by (f, f^{tr}) (resp. by f and $dom(f^{tr})$) as the PSIOA executions-matching (f, f^{tr}, f^{ex}) induced by (f, f^{tr}) (resp. by f and $dom(f^{tr})$) according to definition 78. Furthermore, let $(\mu, \mu') \in Disc(Frags(X)) \times Disc(Frags(Y))$ s.t. for every $\alpha' \in supp(\mu)$, $\alpha' \in dom(f^{ex})$ and $\mu(\alpha) = \mu'(f^{ex}(\alpha'))$. then we say that (f, f^{tr}, f^{ex}) is a* PCA executions-matching *from (X, μ) to (Y, μ') according to definition 80.*
- *The (f^+, D''_X)-continuation of a PCA-executions-matching (f, f^{tr}, f^{ex}) is the (f^+, D''_X)-continuation of (f, f^{tr}, f^{ex}) in the according to definition 81.*

We restate the theorem 83 and 84 for PCA executions-matching:

▶ **Theorem 104** (PCA-execution-matching preserves probabilsitic distribution). *Let X and Y be two PCA $(\mu, \mu') \in Disc(Frags(X)) \times Disc(Frags(Y))$. Let (f, f^{tr}, f^{ex}) be a PCA executions-matching from (X, μ) to (Y, μ') . Let $(\tilde{\sigma}, \sigma) \in schedulers(\mathcal{A}) \times schedulers(\mathcal{B})$, s.t. $(\tilde{\sigma}, \sigma)$ are (f, f^{tr}, f^{ex})-*alter egos. *Let $(\alpha, \pi) \in dom(f^{ex}) \times Frags(Y)$.*
If $\pi = f^{ex}(\alpha)$, then $\epsilon_{\tilde{\sigma},\tilde{\mu}}(C_{\tilde{\alpha}}) = \epsilon_{\sigma,\mu}(C_{\alpha})$ and $\epsilon_{\tilde{\sigma},\tilde{\mu}}(\tilde{\alpha}) = \epsilon_{\sigma,\mu}(\alpha)$.

Proof. We just re-apply the theorem 83, since (f, f^{tr}, f^{ex}) is a PSIOA executions-matching from $(psioa(X), \mu)$ to $(psioa(Y), \mu')$. ◀

▶ **Theorem 105** (Continued PCA executions-matching preserves general probabilistic distribution). *Let X and Y be two PCA $(\mu, \mu') \in Disc(Frags(X)) \times Disc(Frags(Y))$. Let (f, f^{tr}, f^{ex}) be a PCA executions-matching from (X, μ) to (Y, μ') . Let $((f, f^{+}), f^{tr,+}, f^{ex,+})$ be a continuation of (f, f^{tr}, f^{ex}). Let $(\tilde{\sigma}, \sigma) \in schedulers(\mathcal{A}) \times schedulers(\mathcal{B})$, s.t. $(\tilde{\sigma}, \sigma)$ are (f, f^{tr}, f^{ex})-*alter egos. *Let $(\alpha, \pi) \in dom(f^{ex,+}) \times Frags(Y)$.*
If $\pi = f^{ex,+}(\alpha)$, then $\epsilon_{\tilde{\sigma},\tilde{\mu}}(C_{\tilde{\alpha}}) = \epsilon_{\sigma,\mu}(C_{\alpha})$.

Proof. We just re-apply the theorem, 84 since $((f, f^{+}), f^{tr,+}, f^{ex,+})$ is a continued PSIOA executions-matching from $(psioa(X), \mu)$ to $(psioa(Y), \mu')$. ◀

Composability of execution-matching relationship

Now we are looking for composability of PCA executions-matching. Here again the notions are the same than the ones for PSIOA excepting for states-matching and for partial-compatibility. Hence we only need to show that i) the $\mathcal{E}$-extension of a PCA states-matching is still a PCA states-matching (see lemma 106), ii) if $f : Q_X \to Q_Y$ is a bijective PCA states-matching and f^{-1} is complete, then for every PCA $\mathcal{E}$ partial-compatible with X, $\mathcal{E}$ is partial-compatible Y (see lemma 108).

▶ **Lemma 106** (Composability of PCA states-matching). *Let X and Y be two PCA. Let $\mathcal{E}$ be partially-compatible with both X and Y. Let $f : Q'_X \subset Q_X \to Q_Y$ be a PCA states-matching. Let g be the $\mathcal{E}$-extension of f.*
If $range(g) \subset Q_{Y||\mathcal{E}}$, then g is a PCA states-matching from $X||\mathcal{E}$ to $Y||\mathcal{E}$.

Proof.
- If $(\bar{q}_X, \bar{q}_{\mathcal{E}}) \in Q_{X||\mathcal{E}}$ then $\bar{q}_X \in Q'_X$ which means $f(\bar{q}_X) = \bar{q}_Y$, thus $g((\bar{q}_X, \bar{q}_{\mathcal{E}})) = (\bar{q}_{\mathcal{E}}, \bar{q}_{\mathcal{E}})$.
- $\forall ((q_X, q_{\mathcal{E}}), (q_Y, q_{\mathcal{E}})) \in Q'_{X||\mathcal{E}} \times Q_{Y||\mathcal{E}}$ with $(q_Y, q_{\mathcal{E}}) = g((q_X, q_{\mathcal{E}}))$, we have
  - Configuration preservation (modulo hiding): if $auts(config(X)(q_X)) = (\mathcal{A}_1, ..., \mathcal{A}_n)$, then $auts(config(Y)(q_Y)) = (\mathcal{A}'_1, ..., \mathcal{A}'_n)$ where $\forall i \in [1:n], \mathcal{A}_i = hide(\mathcal{A}'_i, h_i)$ with h_i defined on $states(\mathcal{A}'_i)$, s. t. $h_i(q_{\mathcal{A}'_i}) \subseteq out(\mathcal{A}'_i)(q_{\mathcal{A}'_i})$ (resp. s.t. $h_i(q_{\mathcal{A}'_i}) = \emptyset$, that is $\mathcal{A}_i = \mathcal{A}'_i$). Hence if $auts(config(X||\mathcal{E})((q_X, q_{\mathcal{E}})) = (\mathcal{A}_1, ..., \mathcal{A}_n, \mathcal{B}_1, ..., \mathcal{B}_m)$, then $auts(config(Y||\mathcal{E})((q_Y, q_{\mathcal{E}})) = (\mathcal{A}'_1, ..., \mathcal{A}'_n, \mathcal{B}_1, ..., \mathcal{B}_m)$ where $\forall i \in [1:n], \mathcal{A}_i = hide(\mathcal{A}'_i, h_i)$ with h_i defined on $states(\mathcal{A}'_i)$, s. t. $h_i(q_{\mathcal{A}'_i}) \subseteq out(\mathcal{A}'_i)(q_{\mathcal{A}'_i})$ (resp. s.t. $h_i(q_{\mathcal{A}'_i}) = \emptyset$, that is $\mathcal{A}_i = \mathcal{A}'_i$).
  - Hidding preservation (modulo hiding): $hidden\text{-}actions(X)(q_X) = hidden\text{-}actions(Y)(q_Y) \cup h^{+}(q_Y)$ where h^{+} defined on $states(Y)$, s. t. $h^{+}(q_Y) \subseteq out(Y)(q_Y)$. Hence $hidden\text{-}actions(X||\mathcal{E})((q_X, q_{\mathcal{E}})) = hidden\text{-}actions(X)(q_X) \cup hidden\text{-}actions(\mathcal{E})(q_{\mathcal{E}}) = hidden\text{-}actions(Y)(q_Y) \cup hidden\text{-}actions(\mathcal{E})(q_{\mathcal{E}}) \cup h^{+}(q_Y) = hidden\text{-}actions(Y||\mathcal{E})((q_Y, q_{\mathcal{E}})) \cup h^{+\prime}((q_Y, q_{\mathcal{E}}))$ where $h^{+\prime}$ defined on $states(Y||\mathcal{E})$, s. t. $h^{+\prime}((q_Y, q_{\mathcal{E}})) = h^{+}(q_Y) \subseteq out(Y)(q_Y) \subseteq out(Y||\mathcal{E})((q_Y, q_{\mathcal{E}}))$.
  - Creation preservation $\forall a \in \widehat{sig}(X)(q_X) = \widehat{sig}(Y)(q_Y)$, $created(X)(q_X)(a) = created(Y)(q_Y)(a)$. Hence $\forall a \in \widehat{sig}(X||\mathcal{E})((q_X, q_{\mathcal{E}})) = \widehat{sig}(Y||\mathcal{E})((q_Y, q_{\mathcal{E}}))$, either

- $*$ $a \in \widehat{sig}(X)(q_X) = \widehat{sig}(Y)(q_Y)$ but $a \notin \widehat{sig}(\mathcal{E})(q_\mathcal{E})$ and then $created(X||\mathcal{E})((q_X, q_\mathcal{E}))(a) = created(X)(q_X)(a) = created(Y)(q_Y) = created(Y||\mathcal{E})((q_Y, q_\mathcal{E}))(a)$
- $*$ or $a \notin \widehat{sig}(X)(q_X) = \widehat{sig}(Y)(q_Y)$ but $a \in \widehat{sig}(\mathcal{E})(q_\mathcal{E})$ and then $created(X||\mathcal{E})((q_X, q_\mathcal{E}))(a) = created(\mathcal{E})(q_\mathcal{E})(a) = created(Y||\mathcal{E})((q_Y, q_\mathcal{E}))(a)$
- $*$ or $a \in \widehat{sig}(X)(q_X) = \widehat{sig}(Y)(q_Y)$ and $a \in \widehat{sig}(\mathcal{E})(q_\mathcal{E})$ and then $created(X||\mathcal{E})((q_X, q_\mathcal{E}))(a) = created(X)(q_X)(a) \cup created(\mathcal{E})(q_\mathcal{E})(a) = created(Y)(q_Y) \cup created(\mathcal{E})(q_\mathcal{E})(a) = created(Y||\mathcal{E})((q_Y, q_\mathcal{E}))(a)$

Thus, $\forall a \in \widehat{sig}(X||\mathcal{E})((q_X, q_\mathcal{E})) = \widehat{sig}(Y||\mathcal{E})((q_Y, q_\mathcal{E}))$, $created(X||\mathcal{E})((q_X, q_\mathcal{E}))(a) = created(Y||\mathcal{E})((q_Y, q_\mathcal{E}))(a)$.

◀

We restate the theorem 94 of executions-matching composability.

▶ **Theorem 107** (Composability of PCA matching-execution). *Let X and Y be two PCA. Let $\mathcal{E}$ be partially-compatible with both X and Y. Let (f, f^{tr}, f^{ex}) be a PCA executions-matching from X to Y. Let g be the $\mathcal{E}$-extension of f. If $range(g) \subset Q_{Y||\mathcal{E}}$, then the $\mathcal{E}$-extension of (f, f^{tr}, f^{ex}) is a PCA executions-matching (g, g^{tr}, g^{ex}) from $X||\mathcal{E}$ to $Y||\mathcal{E}$ induced by g and $dom(g^{tr})$.*

Proof. This comes immediately from theorem 94. ◀

We extend the lemma 95 but we have to take a little precaution for the partial-compatibility since here the configurations have to be pairwise compatible, not only the signatures.

▶ **Lemma 108** (Some properties preserved by $\mathcal{E}$-extension of a PCA executions-matching). *Let X and Y be two PCA. Let (f, f^{tr}, f^{ex}) be a PCA executions-matching from X to Y.*

1. *If f is complete, then for every PSIOA $\mathcal{E}$ partially-compatible with X, $\mathcal{E}$ is partially-compatible with Y.*
2. *Let $\mathcal{E}$ partially-compatible with both X and Y, let g be the $\mathcal{E}$-extension of f.*

   a. *If f is bijective and f^{-1} is complete, then $range(g) = Q_{Y||\mathcal{E}}$ and so we can talk about the $\mathcal{E}$-extension of (f, f^{tr}, f^{ex})*
   b. *If (f, f^{tr}) is a bijective complete transition-matching, (g, g^{tr}) is a bijective complete transition-matching. (And (f, f^{tr}, f^{ex}) and (g, g^{tr}, g^{ex}) are bijective complete execution-matching.)*
   c. *If f is strong, then g is strong*

Proof. **1.** We need to show that every pseudo-execution of $(Y, \mathcal{E})$ ends on a compatible state. Let $\pi = q^0 a^1 q^1 ... a^n q^n$ be a finite pseudo-execution of $(Y, \mathcal{E})$. We note $\alpha = (f^{-1}(q^0_Y), q^0_\mathcal{E}) a^1 (f^{-1}(q^1_Y), q^1_\mathcal{E}) ... a^n (f^{-1}(q^n_Y), q^n_\mathcal{E})$. The proof is in two steps. First, we show by induction that $\alpha = (f^{-1}(q^0_Y), q^0_\mathcal{E}) a^1 (f^{-1}(q^1_Y), q^1_\mathcal{E}) ... a^n (f^{-1}(q^n_Y), q^n_\mathcal{E})$ is an execution of $X||\mathcal{E}$. Second, we deduce that it means $(f^{-1}(q^n_Y), q^n_\mathcal{E})$ is a compatible state of $(X, \mathcal{E})$ which means that a fortiori, $(q^n_Y, q^n_\mathcal{E})$ is a compatible state of $(Y, \mathcal{E})$ which ends the proof.

- First, we show by induction that α is an execution of $X||\mathcal{E}$. We have $(f^{-1}(\bar{q}_Y), \bar{q}_\mathcal{E}) = (\bar{q}_X, \bar{q}_\mathcal{E})$ which ends the basis.
  Let assume $(f^{-1}(q^0_Y), q^0_\mathcal{E}) a^1 (f^{-1}(q^1_Y), q^1_\mathcal{E}) ... a^k (f^{-1}(q^k_Y), q^k_\mathcal{E})$ is an execution of $X||\mathcal{E}$. Hence $(f^{-1}(q^k_Y), q^k_\mathcal{E})$ is a compatible state of $(X, \mathcal{E})$ which means that a fortiori q^k is a compatible state of $(Y, \mathcal{E})$ because of signature preservation of f.

For the same reason, $\widehat{sig}(Y,\mathcal{E})(q^k) = \widehat{sig}(X||\mathcal{E})((f^{-1}(q^k_Y), q^k_{\mathcal{E}}))$, so $a^{k+1} \in \widehat{sig}(X,\mathcal{E})((f^{-1}(q^k_Y), q^k_{\mathcal{E}}))$. Then we use the completeness of $(f^{-1}, (f^{tr})^{-1})$, to obtain the fact that either $\eta_{(Y,q^k_Y,a^{k+1})} \in dom((f^{tr})^{-1})$ or $a^{k+1} \notin \widehat{sig(Y)}(q^k_Y)$ (and we recall the convention that in this second case $\eta_{(Y,q^k_Y,a^{k+1})} = \delta_{q^k_Y}$). which means either $(f^{-1}(q^k_Y), a^{k+1}, \eta_{(X,f^{-1}(q^k_Y),a^{k+1})})$ is a transition of X that ensures $\forall q'' \in supp(\eta_{(Y,q^k_Y,a^{k+1})}), f^{-1}(q'') \in supp(\eta_{(X,f^{-1}(q^k_Y),a^{k+1})})$ or $a^{k+1} \notin \widehat{sig(X)}(f^{-1}(q^k_Y))$ (and we recall the convention that in this second case $\eta_{(X,f^{-1}(q^k_Y),a^{k+1})} = \delta_{f^{-1}(q^k_Y)}$). Thus for every $(q'', q''') \in supp(\eta_{(Y,\mathcal{E}),q^k,a^{k+1}})$, $(f^{-1}(q''), q''') = g^{-1}((q'', q''')) \in supp(\eta_{(X,\mathcal{E}),g^{-1}(q^k),a^{k+1}})$ namely for $(q'', q''') = (q^{k+1}_Y, q^{k+1}_{\mathcal{E}})$. Hence, $(f^{-1}(q^{k+1}_Y), q^{k+1}_{\mathcal{E}})$ is reachable by $(X,\mathcal{E})$ which means the alternating sequence $(f^{-1}(q^0_Y), q^0_{\mathcal{E}})a^1(f^{-1}(q^1_Y), q^1_{\mathcal{E}})...a^k(f^{-1}(q^k_Y), q^k_{\mathcal{E}})a^k(f^{-1}(q^k_Y), q^k_{\mathcal{E}})a^{k+1}(f^{-1}(q^{k+1}_Y), q^{k+1}_{\mathcal{E}})$ is an execution of $X||\mathcal{E}$. Thus by induction α is an execution of $X||\mathcal{E}$.

- Since X and $\mathcal{E}$ are partially-compatible $(f^{-1}(q^n_Y), q^n_{\mathcal{E}})$ is a state of $X||\mathcal{E}$, so $(f^{-1}(q^n_Y), q^n_{\mathcal{E}})$ is a compatible state of $(X,\mathcal{E})$ which means $(q^k_Y, q^k_{\mathcal{E}})$ is a fortiori a compatible state of $(Y,\mathcal{E})$. Hence every reachable state of $(Y,\mathcal{E})$ is compatible which means Y and $\mathcal{E}$ are partially compatible which ends the proof.

2. This comes immediately from lemma 95 since (f, f^{tr}, f^{ex}) is a PSIOA executions-matching from $psioa(X)$ to $psioa(Y)$ by construction.

◀

Finally, we restate the semantic-equivalence.

A strong complete bijective transitions-matching implies a strong complete bijective executions-matching which means the two automata are completely semantically equivalent.

▶ **Definition 109** (PCA semantic equivalence). *Let X an Y be two PCA. We say that X and Y are semantically-equivalent if it exists a complete bijective strong PCA executions-matching from X to Y*

▶ **Theorem 110** (composability of semantic equivalence). *Let X and Y be PCA semantically-equivalent. Then for every PSIOA $\mathcal{E}$:*

- *$\mathcal{E}$ is partially-compatible with $X \iff \mathcal{E}$ is partially-compatible with Y*
- *if $\mathcal{E}$ is an environment for both X and Y, then $X||\mathcal{E}$ and $Y||\mathcal{E}$ are PCA semantically-equivalent.*

Proof.
- The first item comes from lemma 108, first item
- The second item comes from lemma 108, second item

◀

A weak complete bijective PCA transitions-matching implies a weak complete bijective PCA executions-matching which means the two automata are completely semantically equivalent modulo some hiding operation that implies that some PSIOA are partially-compatible with one of the automaton and not with the other one and that the traces are not necessarily the same ones.

11 Projection

This section aims to formalise the idea of a PCA $X_{\mathcal{A}}$ considered without an internal PSIOA $\mathcal{A}$. This PCA will be noted $Y_{\mathcal{A}} = X_{\mathcal{A}} \setminus \{\mathcal{A}\}$. The reader can already take a look on the figures 23 and 24 to get an intuition on the desired result. This is an important step in our reasoning since we will be able to formalise in which sense $X_{\mathcal{A}}$ and $psioa(X_{\mathcal{A}} \setminus \{\mathcal{A}\})||\mathcal{A}$ are similar.

We first define some notions of projection on configurations on subsection 11.1. Then we define the notion of $\mathcal{A}$-fair PCA X in subsection 11.2, which will be a sufficient condition to ensure that $Y = X \setminus \{\mathcal{A}\}$ is still a PCA, namely that it ensures the constraints of top/down and bottom/up transition preservation, which is proved in the last subsection 11.3.

11.1 Projection on Configurations

In this subsection, we want to define formally $\eta' \in Disc(Q_{conf})$ that would be the result of $\eta \in Disc(Q_{conf})$ "deprived of an automaton $\mathcal{A}$". This is achieved in definition 116. This definition requires particular precautions and motivate the next sequence of definitions, from definition 111 to 116.

The next definition captures the idea of a state deprived of a PSIOA $\mathcal{A}$.

▶ **Definition 111** (State projection). *Let $\mathbf{A} = \{\mathcal{A}_1, ..., \mathcal{A}_n\}$ be a set of PSIOA compatible at state $q = (q_1, ..., q_n) \in Q_{\mathcal{A}_1} \times ... \times Q_{\mathcal{A}_n}$. Let $\mathbf{A}^s = \{\mathcal{A}_{s^1}, ..., \mathcal{A}_{s^n}\}$. We note :*

- *$q \setminus \{\mathcal{A}_k\} = (q_1, ..., q_{k-1}, q_{k+1}, ..., q_n)$ if $\mathcal{A}_k \in \mathbf{A}$ and $q \setminus \{\mathcal{A}_k\} = q$ otherwise.*
- *$q \setminus \mathbf{A}^s = (q \setminus \{\mathcal{A}_{s^n}\}) \setminus (\mathbf{A}^s \setminus \{\mathcal{A}_{s^n}\})$ (recursive extension of the previous item).*
- *$q \restriction \mathbf{A}^s = q \setminus (\mathbf{A} \setminus \mathbf{A}^s)$ if $\mathbf{A}^s \subset \mathbf{A}$ (recursive extension of the previous item). We can remark that $q \restriction \mathcal{A}_k = q_k$ if $\mathcal{A}_k \in \mathbf{A}$.*

Since, $\restriction$ can be defined with $\setminus$, the next sequence of definitions only handle $\setminus$, but can be adapted to support $\restriction$ in the obvious way.

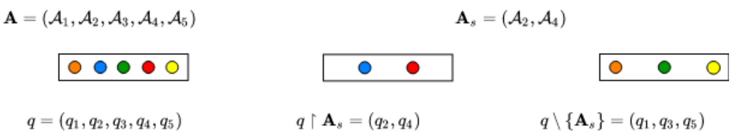


Figure 18 State projection

The next definition captures the idea of a family transition deprived of a PSIOA $\mathcal{A}$.

▶ **Definition 112** (Family transition projection). *(see figure 19 first for an intuition) Let $\mathbf{A}_1$ be a set of automata compatible at state $q_1 \in Q_{\mathbf{A}_1}$. Let $\mathbf{A}^s, \mathbf{A}_2 = \mathbf{A}_1 \setminus \mathbf{A}^s \neq \emptyset$. Let $q_2 = q_1 \setminus \mathbf{A}^s$. Let a be an action. We note $\eta_{(\mathbf{A}_1,q_1,a)} \setminus \mathbf{A}^s \triangleq \eta_{(\mathbf{A}_2,q_2,a)}$ with the convention $\eta_{(\mathbf{A}_i,q_i,a)} = \delta_{q_i}$ if $a \notin \widehat{sig}(\mathbf{A}_i)(q_i)$ for each $i \in \{1, 2\}$.*

▶ **Lemma 113** (family transition projection). *Let $\mathbf{A}_1$ be a set of automata compatible at state $q_1 \in Q_{\mathbf{A}_1}$. Let $\mathbf{A}^s, \mathbf{A}_2 = \mathbf{A}_1 \setminus \mathbf{A}^s \neq \emptyset$. Let $q_2 = q_1 \setminus \mathbf{A}^s$. Let a be an action. Let $\eta_1 = \eta_{(\mathbf{A}_1,q_1,a)}$ and $\eta_2 = \eta_1 \setminus \mathbf{A}^s$ with the convention $\eta_{(\mathbf{A}_1,q_1,a)} = \delta_{q_1}$ if $a \notin \widehat{sig}(\mathbf{A}_1)(q_1)$.*

Then $\forall q'_2 \in Q_{\mathbf{A}_2}$, $\eta_2(q'_2) = \sum_{q'_1 \in Q_{\mathbf{A}_1}, q'_1 \setminus \mathbf{A}^s = q'_2} \eta_1(q'_1)$

Proof. Comes from total probability law. If $\mathbf{A}^s \cap \mathbf{A}_1 = \emptyset$, $\mathbf{A}_2 = \mathbf{A}_1$, the result is immediate. Assume $\mathbf{A}^s \cap \mathbf{A}_1 \neq \emptyset$. Let $\mathbf{A}_3 = \mathbf{A} \setminus \mathbf{A}_2 = \mathbf{A} \setminus (\mathbf{A} \setminus \mathbf{A}^s) \neq \emptyset$. We note $q_3 = q_1 \setminus \mathbf{A}_2$, $\eta_3 = \eta_1 \setminus \mathbf{A}_2$ Then $\forall q'_1 \in Q_{\mathbf{A}_1}$, $\eta_1(q'_1) = \eta_2(q'_2) \otimes \eta_3(q'_3)$ with $q'_2 = q'_1 \restriction \mathbf{A}_2$ and $q'_3 = q'_1 \restriction \mathbf{A}_3$. Hence $\forall q'_2 \in Q_{\mathbf{A}_2}$, $\sum_{q'_1 \in Q_{\mathbf{A}_1}, q'_1 \setminus \mathbf{A}^s = q'_2} \eta_1(q'_1) = \sum_{q'_1 \in Q_{\mathbf{A}_1}, q'_1 \restriction \mathbf{A}_2 = q'_2} \eta_2(q'_2) \cdot \eta_3(q'_1 \restriction \mathbf{A}_3) = \eta_2(q'_2) \cdot \sum_{q'_3 \in Q_{\mathbf{A}_3}} \eta_3(q'_3) = \eta_2(q'_2)$, which ends the proof. ◀

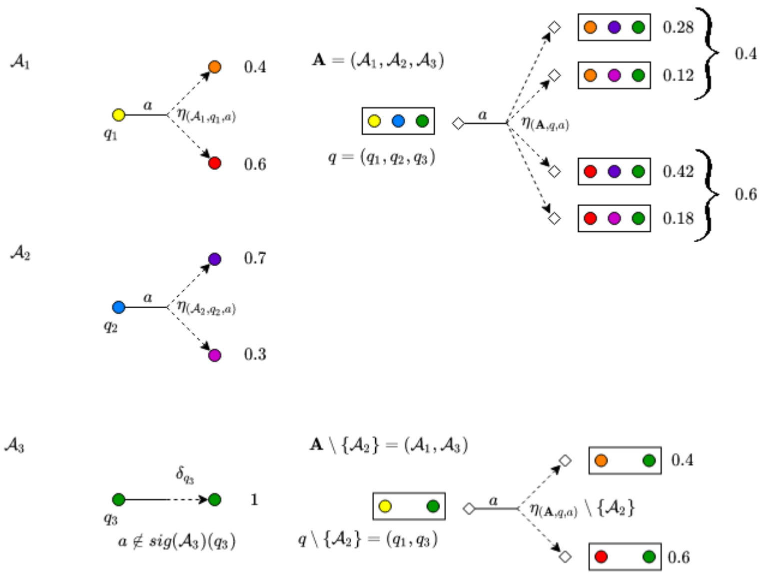


Figure 19 total probability law for family transition projection

Then we apply this notation to preserving distributions.

▶ **Definition 114** (preserving transition projection). *(see figure 20) Let* $\mathbf{A}$, $\mathbf{A}^s$, $\mathbf{A}_2 = \mathbf{A} \setminus \mathbf{A}^s$ *be set of automata,* $q \in Q_{\mathbf{A}}$, *and* a *be an action. Let* $\eta_p \in Disc(Q_{conf})$ *be the unique preserving distribution s.t.* $\eta_p \overset{TS}{\leftrightarrow} \eta_{(\mathbf{A},q,a)}$ *with the convention* $\eta_{(\mathbf{A},q,a)} = \delta_q$ *if* $a \notin \widehat{sig}(\mathbf{A})(q)$. *We note* $\eta_p \setminus \mathbf{A}^s$ *the unique preserving distribution s.t.* $(\eta_p \setminus \mathbf{A}^s) \overset{TS}{\leftrightarrow} (\eta_{(\mathbf{A},q,a)} \setminus \mathbf{A}^s)$ *if* $\mathbf{A}_2 \neq \emptyset$ *and* $\eta_p = \delta_{(\emptyset,\emptyset)}$ *otherwise.*

▶ **Lemma 115** (preserving transition projection). *Let* $\mathbf{A}^s$ *be finite sets of PSIOA. Let* a *be an action. For each* $i \in \{1,2\}$, *let* $C_i \in Q_{conf}$, $C_i \overset{a}{\rightharpoonup} \eta_p^i$ *if* $a \in \widehat{sig}(C_i)$ *and* $\eta_p^i = \delta_{C_i}$ *otherwise. Let* $\tilde{\eta}_p^2 = \eta_p^1 \setminus \mathbf{A}^s$. *Assume* $C_2 = C_1 \setminus \mathbf{A}^s$. *Then,*

- $\eta_p^2 = \tilde{\eta}_p^2$, *i.e.* $(C_1 \setminus \mathbf{A}^s) \overset{a}{\rightharpoonup} (\eta_p^1 \setminus \mathbf{A}^s)$.
- *For every* $C_2' \in Q_{conf}, \eta_p^2(C_2') = \Sigma_{(C_1' \in Q_{conf}, C_1' \setminus \mathbf{A}^s = C_2')} \eta_p^1(C_1')$

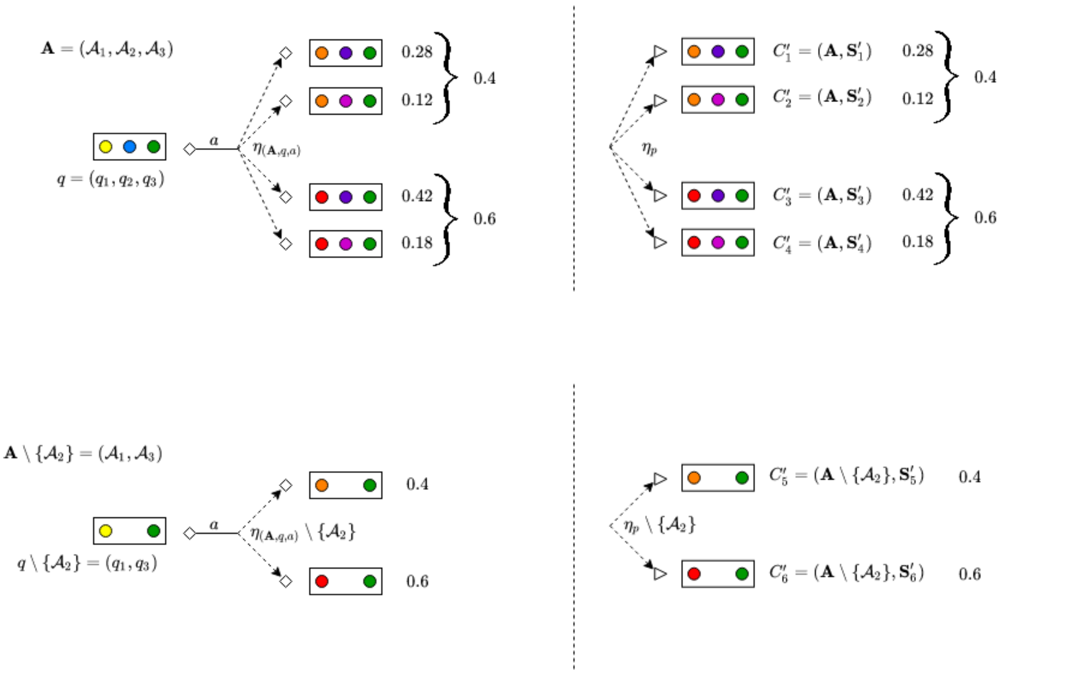
Proof.

- Immediate by definitions 18 and 114.
- For each $i \in \{1,2\}$, we note $\mathbf{A}_i = auts(C_i)$, $q_i = TS(C_i)$. By definition, we have $\eta_p^i \overset{TS}{\leftrightarrow} \eta_{(\mathbf{A}_i,q_i,a)}$ with the convention $\eta_{(\mathbf{A}_i,q_i,a)} = \delta_q$ if $a \notin \widehat{sig}(\mathbf{A}_i)(q_i)$. Finally, we apply lemma 113.

◀

Now we are able to define intrinsic transition deprived of a PSIOA $\mathcal{A}$.

▶ **Definition 116** (intrinsic transition projection). *(see figure 21) Let* $\mathbf{A}$, $\mathbf{A}^s$ *be finite sets of automata,* $q \in Q_{\mathbf{A}}$, *and* a *be an action. Let* $\eta_p \in Disc(Q_{conf})$ *be the unique preserving distribution s.t.* $\eta_p \overset{TS}{\leftrightarrow} \eta_{(\mathbf{A},q,a)}$ *with the convention* $\eta_{(\mathbf{A},q,a)} = \delta_q$ *if* $a \notin \widehat{sig}(\mathbf{A})(q)$. *Let* φ *be a finite set of PSIOA identifiers with* $aut(\varphi) \cap \mathbf{A} = \emptyset$. *Let* $\eta = reduce(\eta_p \uparrow \varphi)$. *We note* $\eta \setminus \mathbf{A}^s = reduce((\eta_p \setminus \mathbf{A}^s) \uparrow (\varphi \setminus \mathbf{A}^s))$.



■ **Figure 20** total probability law for preserving configuration

▶ **Lemma 117** (intrinsic transition projection). *Let $\mathbf{A}^s$ be finite sets of PSIOA. Let a be an action. For each $i \in \{1,2\}$, let φ_i be a finite set of PSIOA identifiers, let $C_i \in Q_{conf}$, $C_i \stackrel{a}{\Longrightarrow}_{\varphi_i} \eta^i$ if $a \in \widehat{sig}(C_i)$ and $\eta^i = \delta_{C_i}$ otherwise. Let $\tilde{\eta}^2 = \eta^1 \setminus \mathbf{A}^s$ and $\tilde{\varphi}^2 = \varphi^1 \setminus \mathbf{A}^s$. Assume $C_2 = C_1 \setminus \mathbf{A}^s$. Then,*

- *$\eta^2 = \tilde{\eta}^2$ and $\tilde{\varphi}_2 = \varphi_2$, i.e. $(C_1 \setminus \mathbf{A}^s) \stackrel{a}{\Longrightarrow}_{\varphi_1 \setminus \mathbf{A}^s} (\eta^1 \setminus \mathbf{A}^s)$.*
- *For every $C'_2 \in Q_{conf}, (\eta^2_p \uparrow \varphi_2)(C'_2) = \sum_{(C'_1 \in Q_{conf}, C'_1 \setminus \mathbf{A}^s = C'_2)} (\eta^1_p \uparrow \varphi_1)(C'_1)$*
- *For every $C'_2 \in Q_{conf}, \eta^2(C'_2) = \Sigma_{(C'_1 \in Q_{conf}, C'_1 \setminus \mathbf{A}^s = C'_2)} \eta^1(C'_1)$*

Proof.

- Immediate by definitions 18, 116 and lemma 115
- Let $C_3 = C_1 \setminus (auts(C_1) \setminus \mathbf{A}^s)$. We note $\varphi_3 = \varphi_1 \setminus \varphi_2$. By definition 18, for each $i \in \{1,2,3\}$, for each $C'_i \in Q_{conf}$, $(\eta^i_p \uparrow \varphi_i)(C'_i) = \delta_{C_{\varphi_i}}(C'_i \restriction \varphi_i) \cdot \eta^i_p(C'_i \setminus \varphi_i)$ with $auts(C_{\varphi_i}) = \varphi_i$ and $\forall \mathcal{A} \in \varphi_i, map(C_{\varphi_i})(\mathcal{A}) = \bar{q}_{\mathcal{A}}$. By previous lemma, for every $C''_2 \in Q_{conf}, \eta^1_p(C''_2) = \sum_{C''_1, C''_1 \setminus \mathbf{A}^s = C''_2} \eta^1_p(C''_1)$. Hence, $(\eta^2_p \uparrow \varphi_2)(C'_2) = \delta_{C_{\varphi_2}}(C'_2 \restriction \varphi_2) \cdot \sum_{C''_1, C''_1 \setminus \mathbf{A}^s = (C'_2 \setminus \varphi_2)} \eta^1_p(C''_1)$ and so $(\eta^2_p \uparrow \varphi_2)(C'_2) = \sum_{C''_1, C''_1 \setminus \mathbf{A}^s = (C'_2 \setminus \varphi_2)} \delta_{C_{\varphi_2}}(C'_2 \restriction \varphi_2) \cdot \eta^1_p(C''_1)$.
  We remark that the conjunction of $C''_1 \in supp(\eta^1_p)$, $C''_1 \setminus \mathbf{A}^s = (C'_2 \setminus \varphi_2)$ and $C'_2 \restriction \varphi_2 = C_{\varphi_2}$ implies $(C''_1 \cup C_{\varphi_3} \cup C_{\varphi_2}) \setminus \mathbf{A}_s = C'_2$.
  Thus, $(\eta^2_p \uparrow \varphi_2)(C'_2) = \sum_{C'''_1, C'''_1 \setminus \mathbf{A}^s = (C'_2)} \delta_{C_{\varphi_2}}(C'_2 \restriction \varphi_2) \cdot \delta_{C_{\varphi_3}}(C'''_1 \restriction \varphi_3) \cdot \eta^1_p(C'''_1 \setminus \varphi_1) = \sum_{C'''_1, C'''_1 \setminus \mathbf{A}^s = (C'_2)} \delta_{C_{\varphi_2}}(C'''_1 \restriction \varphi_2) \cdot \delta_{C_{\varphi_3}}(C'''_1 \restriction \varphi_3) \cdot \eta^1_p(C'''_1 \setminus \varphi_1) = \sum_{C'''_1, C'''_1 \setminus \mathbf{A}^s = C'_2} \delta_{C_{\varphi_1}}(C'''_1 \restriction \varphi_1) \cdot \eta^1_p(C'''_1 \setminus \varphi_1) = \sum_{C'''_1, C'''_1 \setminus \mathbf{A}^s = C'_2} (\eta^1_p \uparrow \varphi_1)$.
- By definition 18, for each $i \in \{1,2\}$, for each $C'_i \in Q_{conf}$, $\eta^i(C'_i) = \sum_{C''_i, reduce(C''_i) = C'_i} (\eta^i_p \uparrow \varphi_i)(C''_i)$. By previous lemma, for every $C''_2 \in Q_{conf}, \eta^1_p(C''_2) = \sum_{C'''_1, C'''_1 \setminus \mathbf{A}^s = C''_2} (\eta^1_p \uparrow \varphi_1)(C'''_1)$. Thus, $\eta^2(C'_2) = \sum_{C''_2, reduce(C''_2) = C'_2} (\sum_{C'''_1, C'''_1 \setminus \mathbf{A}^s = C''_2} (\eta^1_p \uparrow \varphi_1)(C'''_1))$ and so $\eta^2(C'_2) = \sum_{C'''_1, reduce(C'''_1 \setminus \mathbf{A}^s) = C''_2} (\eta^1_p \uparrow \varphi_1)(C'''_1) = \sum_{C'''_1, reduce(C'''_1) \setminus \mathbf{A}^s = C''_2} (\eta^1_p \uparrow$

$\varphi_1)(C'''_1)$.
Finally $\eta^2(C'_2) = \sum_{C'_1, C'_1 \setminus \mathbf{A}^s = C'_2} (\sum_{C''_1, reduce(C''_1) = C'_1} ((\eta^1_p \uparrow \varphi_1)(C''_1))) = \sum_{C'_1, C'_1 \setminus \mathbf{A}^s = C'_2} \eta^1(C'_1)$

◀

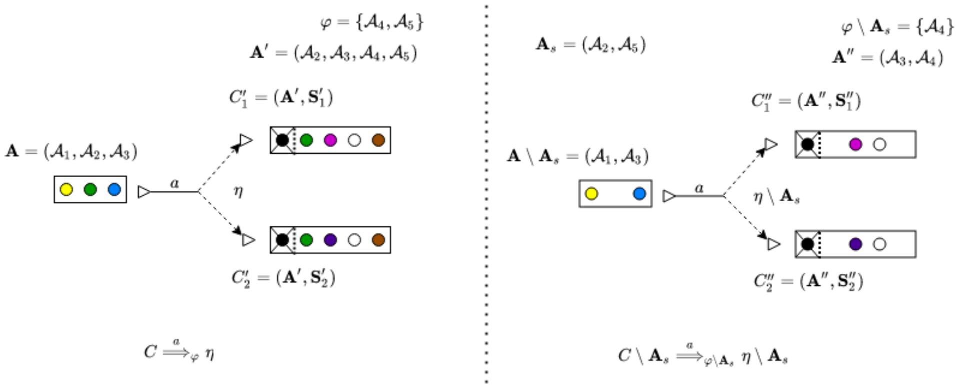


Figure 21 intrinsic transition projection

In next subsection, this lemma 117 will lead to lemma 119 which will be a key lemma to allow the constructive definition 120 of PCA deprived of a (sub) PSIOA.

11.2 $\mathcal{A}$-fairness assumption, motivated by our definition of PCA deprived from an internal PSIOA: $X \setminus \{\mathcal{A}\}$

Here we recall in definition 118 the definition 66 of a $\mathcal{A}$-fair PCA. Then we show lemma 119 (via lemma 117) that will be used to enable the constructive definition of $X \setminus \{\mathcal{A}\}$.

▶ **Definition 118** ($\mathcal{A}$-fair PCA (recall)). *Let $\mathcal{A} \in Autids$. Let X be a PCA. We say that X is $\mathcal{A}$-fair if it verifies the following constraints.*

- *(configuration-conflict-free) X is configuration-conflict-free, that is $\forall q, q' \in Q_X$, s.t. $qR_{conf}q'$ (i.e. $config(X)(q) = config(X)(q')$) then $q = q'$*
- *(no conflict for projection) $\forall q, q' \in Q_X$, s.t. $qR^{\setminus\{\mathcal{A}\}}_{conf}q'$ then $qR^{\setminus\{\mathcal{A}\}}_{strict}q'$. That is if $config(X)(q) \setminus \{\mathcal{A}\} = config(X)(q') \setminus \{\mathcal{A}\}$, then*
  - *$\forall a \in \widehat{sig}(X)(q) \cap \widehat{sig}(X)(q')$, $created(X)(q)(a) \setminus \{\mathcal{A}\} = created(X)(q')(a) \setminus \{\mathcal{A}\}$*
  - *$hidden\text{-}actions(X)(q) \setminus pot\text{-}out(X)(q)(\mathcal{A}) = hidden\text{-}actions(X)(q') \setminus pot\text{-}out(X)(q')(\mathcal{A})$ where for each $q'' \in Q_X$:*
    - *$pot\text{-}out(X)(q'')(\mathcal{A}) = \emptyset$ if $\mathcal{A} \notin auts(config(X)(q''))$, and*
    - *$pot\text{-}out(X)(q'')(\mathcal{A}) = out(\mathcal{A})(map(config(X)(q''))(\mathcal{A}))$ if $\mathcal{A} \in auts(config(X)(q''))$.*
- *(no exclusive creation by $\mathcal{A}$) $\forall q \in Q_X$, $\forall a \in \widehat{sig}(X)(q)$ $\mathcal{A}$-exclusive in q, $created(X)(q)(a) = \emptyset$ where $\mathcal{A}$-exclusive means $\forall \mathcal{B} \in auts(config(X)(q))$, $\mathcal{B} \neq \mathcal{A}$, $a \notin \widehat{sig}(\mathcal{B})(map(config(X)(q))(\mathcal{B}))$.*

A $\mathcal{A}$-fair PCA is a PCA s.t. we can deduce its current properties from its current configuration deprived of $\mathcal{A}$. This will allow the definition of $X \setminus \{\mathcal{A}\}$, where X is a PCA, to be well-defined.

Now we give the second key lemma (after lemma 117) to allow the definition 120 of PCA deprived of a (sub) PSIOA. Basically, this lemma states that if two states q_X and q_Y are strictly equivalent modulo the deprivation of a (sub) automaton P, noted $q_X R^{\setminus\{P\}}_{strict} q_Y$, then the intrinsic configurations issued from these states deprived of P are equal.

▶ **Lemma 119** (equality of intrinsic transition after deprivation of a sub-PSIOA). *Let X_1, X_2 be two PCA, $(q_1, q_2) \in Q_{X_1} \times Q_{X_2}$ s.t. $q_1 R^{\setminus\{P\}}_{strict} q_2$. Let a be an action. For each $i \in \{1,2\}$, we note $C_i \triangleq config(X)(q_i)$, $\varphi_i \triangleq created(X)(q_i)(a)$, η_i s.t. if $a \in \widehat{sig}(C_i)$, $C_i \overset{a}{\Longrightarrow}_{\varphi_i} \eta_i$ and $\eta_i = \delta_{C_i}$ otherwise. Then,*

- $C_0 \triangleq C_1 \setminus \{P\} = C_2 \setminus \{P\}$,
- $\varphi_0 \triangleq \varphi_1 \setminus \{P\} = \varphi_2 \setminus \{P\}$,
- $\eta \triangleq \eta_1 \setminus \{P\} = \eta_2 \setminus \{P\}$,
- *If $a \in \widehat{sig}(C_0)$, $C_0 \overset{a}{\Longrightarrow}_{\varphi} \eta_0$ and $\eta_0 = \delta_{C_0}$ otherwise.*

Proof. The two first items comes directly from definition of $R^{\setminus\{P\}}_{strict}$. By lemma 117, if $a \in \widehat{sig}(C_0)$, we have both $C_0 \overset{a}{\Longrightarrow}_{\varphi} \eta_1 \setminus \{P\}$ and $C_0 \overset{a}{\Longrightarrow}_{\varphi} \eta_2 \setminus \{P\}$, while if $a \notin \widehat{sig}(C_0)$, we have both $(\eta_1 \setminus \{P\}) = \delta_{C_0}$ and $(\eta_2 \setminus \{P\}) = \delta_{C_0}$. By uniqueness of intrinsic transition, we have $\eta_1 \setminus \{P\} = \eta_2 \setminus \{P\}$. ◀

▶ **Definition 120** ($X \setminus \{P\}$). *(see figure 22 for the constructive definition and figures 23 and 24 for the desired result.) Let $P \in Autids$. Let X be a P-fair PCA, with $psioa(X) = (Q_X, \bar{q_X}, sig(X), D_X)$. We note $X \setminus \{P\}$ the automaton Y equipped with the same attributes than a PCA (psioa, config, hidden-actions, created), $\mu^P_s : Q_X \rightarrow Q_Y$ and $\mu^P_d : D_X \setminus \{\eta_{(X,q_X,a)} \in D_X | a$ is P-exclusive in $q_X\} \rightarrow D_Y$ that respect systematically the following rules:*

- *P-deprivation: $\forall q_Y \in Q_Y$, $P \notin config(Y)(q_Y)$, $\forall a \in \widehat{sig}(Y)(q_Y)(a), P \notin created(Y)(q_Y)(a)$.*
- *μ^P_s-correspondence: $\forall (q_X, q_Y) \in Q_X \times Q_Y$ s.t. $\mu^P_s(q_X) = q_Y$, then $q_X R^{\setminus\{P\}}_{strict} q_Y$.*
- *μ^P_d-correspondence: $\forall ((q_Y, a, \eta_{(Y,q_Y,a_Y)}), (q_X, a, \eta_{(X,q_X,a_X)})) \in D_X \times D_Y$ s.t. $(q_Y, a, \eta_{(Y,q_Y,a_Y)}) = \mu^P_d(q_X, a, \eta_{(X,q_X,a_X)})$, then*
  - *$\mu^P_s(q_X) = q_Y$,*
  - *$a_X = a_Y$ and*
  - *$\forall q'_Y \in Q_Y$, $\eta_{(Y,q_Y,a)}(q'_Y) = \Sigma_{q'_X \in Q_X, \mu_s(q'_X) = q'_Y} \eta_{(X,q_X,a)}(q'_X)$.*

  and constructed (conjointly with the mapping μ^P_s and μ^P_d) as follows:
- *(Partitioning):*
  We partition Q_X in equivalence classes according to the equivalence relation $R^{\setminus\{P\}}_{conf}$ that is we obtain a partition $(C_j)_{j \in J \subset \mathbb{N}}$ s.t. $\forall j \in J$, $\forall q_X, q'_X \in C_j$, $q_X R^{\setminus\{P\}}_{conf} q'_X$ and by P-fair assumption, $q_X R^{\setminus\{P\}}_{strict} q'_X$
- *(Q_Y, $sig(Y)$ and μ^P_s):*
  $\forall j \in J$, we construct $q^j_Y \in Q_Y$ and conjointly extend μ^P_s s.t. $\forall q_X \in C_j$, $\mu^P_s(q_X) = q^j_Y$, verifying the P-deprivation-rule and μ^P_s-correspondence rule, that is
  - *$config(Y)(q^j_Y) = config(X)(q_X) \setminus \{P\}$,*
  - *$hidden\text{-}actions(Y)(q^j_Y) = hidden\text{-}actions(X)(q_X) \setminus pot\text{-}out(X)(q_X)(P)$,*
  - *$sig(Y)(q^j_Y) = hide(sig(config(Y)(q^j_Y)), hidden\text{-}actions(Y)(q^j_Y))$*
  - *$\forall a \in \widehat{sig}(Y)(q^j_Y)$, $created(Y)(q^j_Y)(a) = created(X)(q_X)(a) \setminus \{P\}$.*
  - *Furthermore $\bar{q}_Y = \mu^P_s(\bar{q}_X)$.*
- *(D_Y and μ^P_d):*
  $\forall q_Y \in Q_Y$, $\forall a \in \widehat{sig}(Y)(q_Y)$ (and so $\forall q_X \in (\mu^P_s)^{-1}(q_Y), a \in \widehat{sig}(X)(q_X)$) we construct $\eta_{(Y,q_Y,a)}$ and conjointly extend μ^P_d s.t. $\forall q_X \in (\mu^P_s)^{-1}(q_Y)$, $(q_Y, a, \eta_{(Y,q_Y,a_Y)}) = \mu^P_d(q_X, a, \eta_{(X,q_X,a_X)})$, verifies the μ^P_d-correspondence rule. We show this construction is possible:

  - *We note $C_Y = config(Y)(q_Y)$, $\varphi_Y = created(Y)(q_Y)(a)$, η_Y the unique element of $Disc(Q_{conf})$ s.t. $C_Y \overset{a}{\Longrightarrow}_{\varphi_Y} \eta_Y$. Let $(q^i_X)_{i \in I \subset \mathbb{N}} = (\mu^P_s)^{-1}(q_Y)$. For every $i \in I$,* *we note* $C^i_X = config(X)(q^i_X)$, $\varphi^i_X = created(X)(q^i_X)(a)$, η^i_X *the unique element of* $Disc(Q_{conf})$ *s.t.* $C^i_X \stackrel{a}{\Longrightarrow}_{\varphi^i_X} \eta^i_X$. *By lemma 119,* $\forall i \in I$, $C^i_X \setminus \{P\} = C_Y$, $\varphi^i \setminus \{P\} = \varphi_Y$ *and* $\eta^i_X \setminus \{P\} = \eta_Y$.

- *For every* $q^i_X \in (\mu^P_s)^{-1}(q_Y)$, *we partition* $supp(\eta_{(X,q^i_X,a)})$ *in equivalence classes according to the equivalence relation* $R^{\setminus \{P\}}_{conf}$ *that is we obtain a partition* $(C'_j)_{j \in J' \subset \mathbb{N}}$ *s.t.* $\forall j \in J'$, $\forall q'_X, q''_X \in C'_j$, $q'_X R^{\setminus \{P\}}_{conf} q''_X$ *and by* P*-fair assumption,* $q'_X R^{\setminus \{P\}}_{strict} q''_X$. *For each* $j \in J'$, *we extract an arbitrary* $q'_X \in C'_j$ *and* $q'_Y = \mu^P_s(q'_X)$. *We fix* $\eta_{(Y,q_Y,a)}(q'_Y) := \eta_Y(C'_Y)$ *with* $C'_Y = config(Y)(q'_Y)$.

$$
\begin{aligned}
\eta_Y(C'_Y) &= \sum_{C'_X, C'_Y = C'_X \setminus \{P\}} \eta^i_X(C'_X) && \textit{by lemma 117} \\
&= \sum_{q'_X, C'_Y = config(X)(q'_X) \setminus \{P\}} \eta_{(X,q^i_X,a)}(q'_X) && \textit{by bottom/up transition preservation} \\
&= \sum_{q'_X, q'_Y = \mu^P_s(q'_X)} \eta_{(X,q^i_X,a)}(q'_X) && \textit{By } \mu^P_s\textit{-correspondence}
\end{aligned}
$$

Thus, the μ^P_d*-correspondence constraint holds for all the possible* $q^i_X \in (\mu^P_s)^{-1}(q_Y)$.

In the remaining, if we consider a PCA X deprived of a PSIOA $\mathcal{A}$ we always implicitly assume that X is $\mathcal{A}$-fair.

11.3 $Y = X \setminus \{\mathcal{A}\}$ is a PCA if X is $\mathcal{A}$-fair

Here we prove a sequence of lemma to show that $Y = X \setminus \{P\}$ is indeed a PCA, by verifying all the constraints.

Prepare the top/down transition preservation

We show a useful lemma to show $Y = X \setminus \{\mathcal{A}\}$ verifies the constraint 2 of top/down transition preservation.

▶ **Lemma 121** (corresponding transition after projection). *Let* $\mathcal{A}$ *be a PSIOA. Let* X *be a* $\mathcal{A}$*-fair PCA and* $Y = X \setminus \mathcal{A}$. $((q_X, a, \eta_X), (q_Y, a, \eta_Y)) \in D_X \times D_Y$, *s.t.* $(q_Y, a, \eta_{(Y,q_Y,a)}) = \mu_d(q_X, a, \eta_{(X,q_X,a)})$. *For each* $K \in \{X, Y\}$, *we note* $C_K = config(K)(q_K)$, $\varphi_K = created(K)(q_K)(a)$.

Let η'_X *the unique element of* $Disc(Q_{conf})$ *s.t. x0)* $\eta_{(X,q_X,a)} \stackrel{c}{\leftrightarrow} \eta'_X$ *with x1)* $c = config(X)$ *and x2)* $C_X \stackrel{a}{\Longrightarrow}_{\varphi_X} \eta'_X$.

Let $\eta'_Y = \eta'_X \setminus \{\mathcal{A}\}$. *Then* η'_Y *verifies y0)* $\eta_{(Y,q_Y,a)} \stackrel{c'}{\leftrightarrow} \eta'_Y$ *with y1)* $c' = config(Y)(q_Y)$ *and y2)* $Config(Y)(q_Y) \stackrel{a}{\Longrightarrow}_{\varphi_Y} \eta'_Y$.

Proof. We note $(Q^X_i)_{i \in \mathcal{I}}$ the partition of $supp(\eta_{X,q_X,a})$ s.t. $\forall i \in \mathcal{I}, \forall q'_X, q''_X \in Q^X_i, q'_X R^{\setminus \{\mathcal{A}\}}_{conf} q''_X$. $\forall i \in \mathcal{I}$, we note $C^{\setminus \{\mathcal{A}\}}_i = config(q'_X) \setminus \{\mathcal{A}\}$ for an arbitrary element $q'_X \in Q^X_i$ and $C_i = \{C \in supp(\eta'_X) | C \setminus \mathcal{A} = C^{\setminus \{\mathcal{A}\}}_i\}$. Since x0) $\eta_{(X,q_X,a)} \stackrel{f}{\leftrightarrow} \eta'_X$ with x1) $f = config(X)(q_X)$, $(C_i)_{i \in \mathcal{I}}$ is a partition of $supp(\eta'_X)$.

For every $i \in \mathcal{I}$, we note $q^Y_i = \mu_s(q'_X)$ for an arbitrary element $q'_X \in Q^X_i$. By $\mu^{\mathcal{A}}_s$-correspondance, $config(q^Y_i) = C^{\setminus \{\mathcal{A}\}}_i = config(q'_X) \setminus \{\mathcal{A}\}$

By $\mu^{\mathcal{A}}_d$-correspondance,

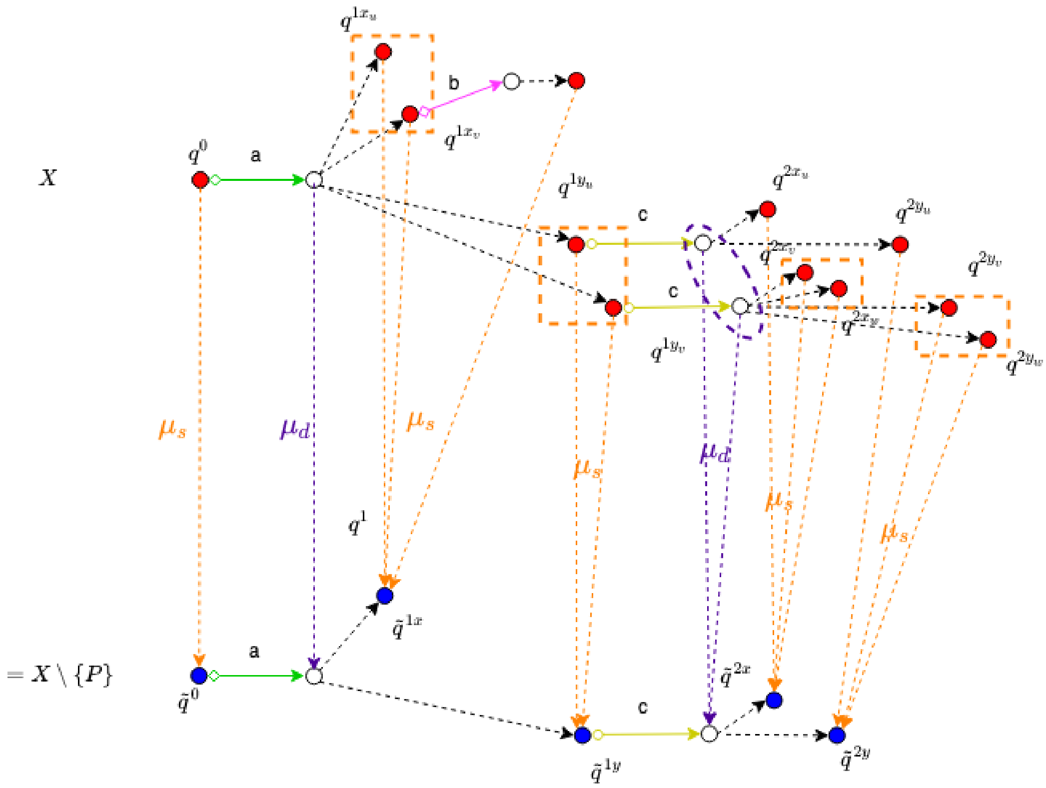
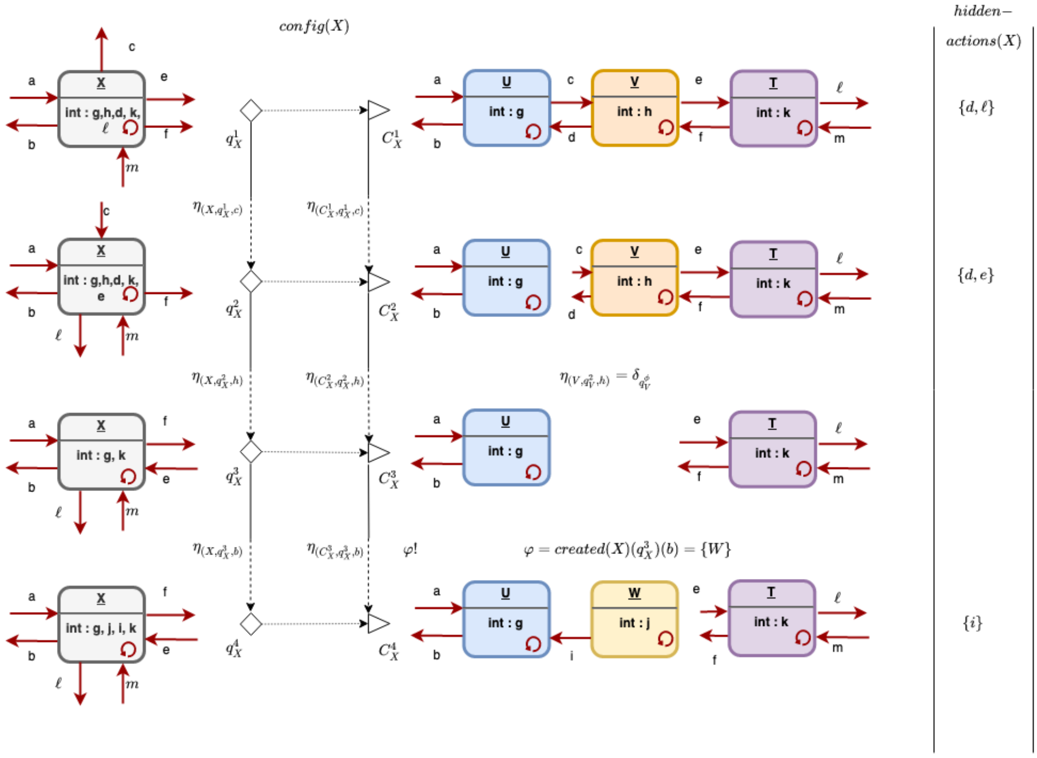


Figure 22 constructive definition of $Y = X \setminus \{P\}$. First we construct $\tilde{q}^0$ which is the initial state of Y. Then we partition $supp(\eta_{(X,q^0,a)}) = \{q^{1x_u}, q^{1x_v}\} \cup \{q^{1y_u}, q^{1y_v}\}$ s.t. $q^{1x_u} R^{\setminus\{P\}}_{conf} q^{1x_v}$ and $q^{1y_u} R^{\setminus\{P\}}_{conf} q^{1y_v}$. Thereafter we construct $\tilde{q^{1x}} = \mu_s(q^{1x_u}) = \mu_s(q^{1x_v})$ and $\tilde{q^{1y}} = \mu_s(q^{1y_u}) = \mu_s(q^{1y_v})$. Then, $\eta_{(Y,\tilde{q^0},a)}$ is defined s.t. $\eta_{(Y,\tilde{q^0},a)}(\tilde{q}^{1x}) = \eta_{(X,q^0,a)}(q^{1x_u}) + \eta_{(X,q^0,a)}(q^{1x_v})$ and $\eta_{(Y,\tilde{q^0},a)}(\tilde{q}^{1y}) = \eta_{(X,q^0,a)}(q^{1y_u}) + \eta_{(X,q^0,a)}(q^{1y_v})$. We perform another time this procedure. by partitioning $supp(\eta_{(X,q^{1y_u},a)}) = \{q^{2x_u}\} \cup \{q^{2y_u}\}$ or $supp(\eta_{(X,q^{1y_v},a)}) = \{q^{2x_v}, q^{2x_w}\} \cup \{q^{2y_v}, q^{2y_w}\}$ arbitrarily. Indeed the obtai,ed result is the same: (i) $q^{1y_u} R^{\setminus\{P\}}_{conf} q^{1y_v}$ since they are both pre-image of $\tilde{q}^{1y}$ by μ_s, which means (ii) $q^{1y_u} R^{\setminus\{P\}}_{strict} q^{1y_v}$ since X is assumed to be P-fair. If we note $C_u = config(X)(q^{1y_u})$, $C_v = config(X)(q^{1y_v})$, $\varphi_u = created(X)(q^{1y_u})(c)$, $\varphi_v = created(X)(q^{1y_v})(c)$, $C_u \overset{c}{\Longrightarrow}_{\varphi_u} \eta_u$ and $C_v \overset{c}{\Longrightarrow}_{\varphi_v} \eta_v$ we have j) $C_u \setminus \{P\} = C_v \setminus \{P\}$, jj) $C_u \setminus \{P\} \overset{c}{\Longrightarrow}_{\varphi_u \setminus \{P\}} \eta_u \setminus \{P\}$ and jjj) $C_v \setminus \{P\} \overset{c}{\Longrightarrow}_{\varphi_v \setminus \{P\}} \eta_v \setminus \{P\}$ which implies jv) $\eta_u \setminus \{P\} = \eta_v \setminus \{P\}$.

$$\eta_{(Y,q_Y,a)}(q'_Y) = \Sigma_{q'_X, \mu_s(q'_X) = q'_Y} \eta_{(X,q_X,a)}(q'_X)$$
$$= \Sigma_{i \in I} \Sigma_{q'_X \in Q^X_i, \mu_s(q'_X) = q'_Y} \eta_{(X,q_X,a)}(q'_X)$$

By assumption x0) and x1), $\eta_{(X,q_X,a)} \overset{c}{\leftrightarrow} \eta'_X$ with $c = config(X)$, thus

$$\eta_{Y,q_y,a}(q'_y) = \Sigma_{i \in \mathcal{I}} \Sigma_{q'_X \in Q^X_i, \mu_s(q'_X) = q'_Y} \eta'_X(config(X)(q'_X))$$
$$= \Sigma_{i \in \mathcal{I}} \Sigma_{C'_X \in C_i, C'_X \setminus \mathcal{A} = config(q'_Y)} \eta'_X(C'_X)$$
$$= \Sigma_{C'_X, C'_X \setminus \mathcal{A} = config(q'_Y)} \eta'_X(C'_X)$$



■ **Figure 23** Projection on PCA (part 1/2, the part 2/2 is in figure 24): the original PCA X

Therafter, we use the lemma 117 and get $\eta_{(Y,q_y,a)}(q'_Y) = \eta'_Y(config(Y)(q'_Y))$ with $\eta'_Y = \eta'_X \setminus \{\mathcal{A}\}$.

By definition of Y, $Config(Y)(q_Y = \mu_s(q_X)) = Config(X)(q_X) \setminus \{\mathcal{A}\}$. We can apply lemma 117. Since $a \in \widehat{sig}(config(X)(q_X) \setminus \{\mathcal{A}\})$, $Config(Y)(q_Y) \overset{a}{\Longrightarrow}_{\varphi_Y} \eta'_Y$ with $\eta'_Y = \eta'_X \setminus \{\mathcal{A}\}$ and $\varphi_Y = (\varphi_X \setminus \{\mathcal{A}\})$. By $\mu_s^{\mathcal{A}}$-correspondance, $created(Y)(q_Y)(a) = created(X)(q_X)(a) \setminus \{\mathcal{A}\}$, thus $\varphi_Y = created(Y)(q_Y)(a)$.

Finally the restriction of $config(Y)$ on $supp(\eta_{(Y,q_Y,a)})$ is a bijection. Indeed, we note $f_1 : q_Y \mapsto Q_i^X$ s.t. $\{q_Y\} = \mu_s(Q_i^X)$, $f_2 : Q_i^X \mapsto C_i$ $f_3 : C_i \mapsto C_i^{\setminus \mathcal{A}}$. By construction, f_1 and f_3 are bijection. By bijectivity of the restriction of $config(X)$ on $supp(\eta_{X,q_X,a})$, f_2 is a bijection too. Moreover, the restriction f' of $config(Y)$ on $supp(\eta_{Y,q_Y,a})$ is $f_1 \circ f_2 \circ f_3$ and hence this is a bijection too.

◀

Now we are able to demonstrate that the PCA set is closed under deprivation.

▶ **Theorem 122** ($X \setminus \{P\}$ is a PCA). *Let $P \in Autids$. Let X be a P-fair PCA, then $Y = X \setminus \{P\}$ is a PCA.*

Proof.
- (Constraint 1) By construction of Y, $\bar{q}_Y = \mu_s^P(\bar{q}_X)$ and by μ_s-correspondence rule, $config(Y)(\bar{q}_Y) = config(X)(\bar{q}_X) \setminus \{P\}$. Since the constraint 1 is respected by X, it is a fortiori respected by Y.
- (Constraint 2) Let $(q_Y, a, \eta_{(Y,q_Y,a)}) \in D_Y$. By construction of Y, we know it exists $(q_X, a, \eta_{(X,q_X,a)}) \in D_X$ with $\eta_{(Y,q_Y,a)} = \mu_d(\eta_{(X,q_X,a)})$ and $q_Y = \mu_s(q_X)$. Then, because of constraint 2 ensured by X, we obtain it exists a reduced configuration distribution $\eta'_X \in Disc(Q_{conf})$ s.t. x0) $\eta_{(X,q_X,a)} \overset{c}{\leftrightarrow} \eta'_X$ with x1) $c = config(X)$ and x2)

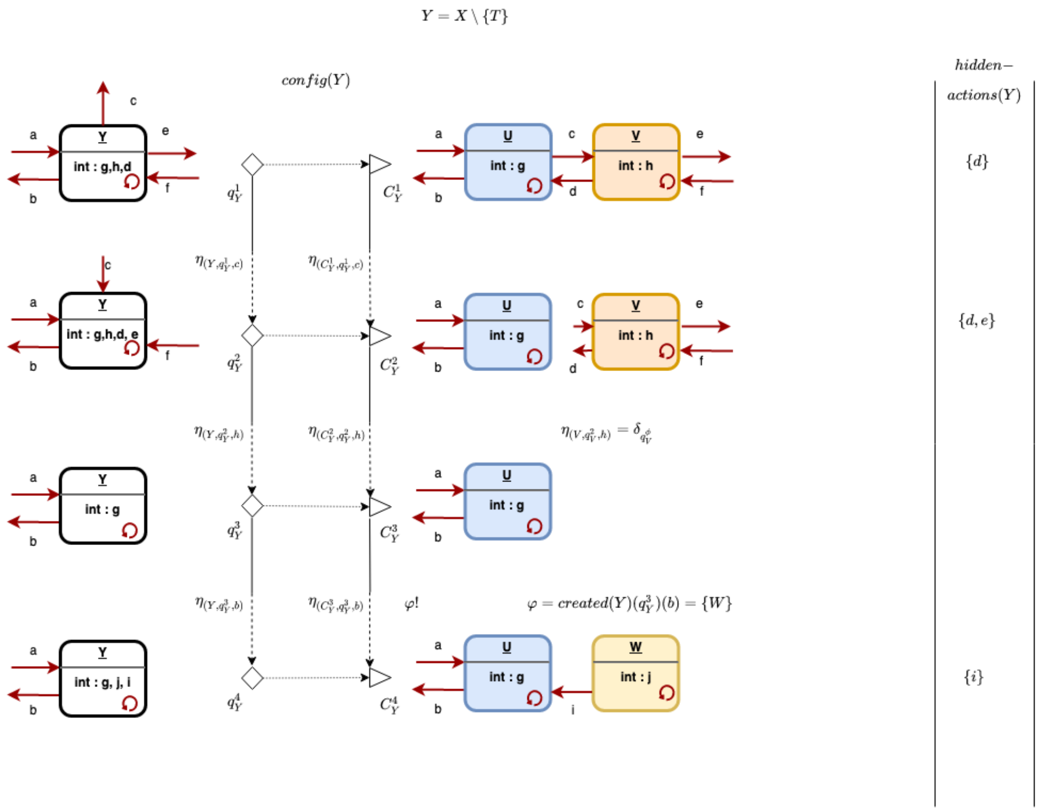


Figure 24 Projection on PCA (part 2/2, the part 1/2 is in figure 23): the PCA $Y = X \setminus \{T\}$

$Config(X)(q_X) \overset{a}{\Longrightarrow}_{\varphi_X} \eta'_X$ where $\varphi_X = created(X)(q_X)(a)$. We can apply lemma 121 to obtain that $\eta'_Y = \eta'_X \setminus \{P\}$ is a reduced configuration transition that verifies y0) $\eta_{(Y,q_Y,a)} \overset{c'}{\leftrightarrow} \eta'_Y$ with y1) $c' = config(Y)$ and y2) $config(Y)(q_Y) \overset{a}{\Longrightarrow}_{\varphi_Y} \eta'_Y$ where $\varphi_Y = \varphi_X \setminus \{P\} = created(Y)(q_Y)(a)$.
This terminates the proof of constraint 2.

- (Constraint 3) Let $q_Y \in Q_Y, C_Y = config(Y)(q_Y), a \in \widehat{sig}(C_Y), \varphi_Y = created(Y)(q_Y)(a)$, $\eta'_Y \in Disc(Q_{conf})$ s.t. $C_Y \overset{a}{\Longrightarrow} \eta'_Y$.
By construction of $Y = X \setminus \{P\}$, if $q_Y \in Q_Y$, $\exists q_X \in Q_X$, $\mu_s(q_X) = q_Y$, $C_X = config(X)(q_X)$, $C_X \setminus \{P\} = C_Y$. Necessarily, $a \in \widehat{sig}(C_X)$ and by construction of $Y = X \setminus \{P\}$, $\varphi_X \setminus \{P\} = \varphi_Y$ with $\varphi_X = created(X)(q_X)(a)$. We note η'_X verifying $C_X \overset{a}{\Longrightarrow}_{\varphi_X} \eta'_X$. By lemma 117, $\eta'_Y = \eta'_X \setminus \{\mathcal{A}\}$.
Because of constraint 3, it means $(q_X, a, \eta_{X,q_X,a}) \in D_X$ with x0) $\eta_{(X,q_X,a)} \overset{c}{\leftrightarrow} \eta'_X$ with x1) $c = config(X)$. Since $q_Y = \mu_s(q_Y)$ and $a \in \widehat{sig}(Y)(q_Y)$, the construction of D_Y implies $(q_Y, a, \eta_{(Y,q_Y,a)}) \in D_Y$ with $(q_Y, a, \eta_{(Y,q_Y,a)}) = \mu_d^P((q_X, a, \eta_{(X,q_X,a)}))$.
We can apply lemma 121 to obtain that η'_Y verifies y0) $\eta_{(Y,q_Y,a)} \overset{c'}{\leftrightarrow} \eta'_Y$ with y1) $c' = config(Y)$ and y2) $C_Y \overset{a}{\Longrightarrow}_{\varphi_Y} \eta'_Y$.
This terminates the proof of constraint 3.
- (Constraint 4) Verified by construction (We recall that $\forall (q_Y, q_X) \in Q_Y \times Q_X, q_Y = \mu_s^P(q_X), sig(Y)(q_Y) \triangleq hide(sig(config(Y)(q_Y), hidden\text{-}actions(Y)(q_Y))$ where $hidden\text{-}actions(Y)(q_Y) \triangleq hidden\text{-}actions(X)(q_X) \setminus pot\text{-}out(X)(q_X)(P)$.

◀

12 Reconstruction

In the previous section, we have shown that $Y = X \setminus \mathcal{A}$ is a PCA (as long as X is $\mathcal{A}$-fair). In this section we will

1. introduce the concept of simpleton wrapper $\tilde{\mathcal{A}}^{sw}$ that is a PCA that encapsulates $\mathcal{A}$.
2. prove that $X \setminus \{\mathcal{A}\}$ and $\tilde{\mathcal{A}}^{sw}$ are partially-compatible (see theorem 134)
3. There is a strong executions-matching from X to $(X \setminus \{\mathcal{A}\})||\tilde{\mathcal{A}}^{sw}$ in a restricted set of executions of X that do not create $\mathcal{A}$ (see theorem 140). Hence it is always possible to transfer a reasoning on X into a reasoning on $(X \setminus \{\mathcal{A}\})||\tilde{\mathcal{A}}^{sw}$ if no re-creation of $\mathcal{A}$ occurs.
4. The operation of projection/deprivation and composition are commutative (see theorem 145).

12.1 Simpleton wrapper : $\tilde{\mathcal{A}}^{sw}$

Here we introduce simpleton wrapper $\tilde{\mathcal{A}}^{sw}$, a PCA that only encapsulates $\tilde{\mathcal{A}}^{sw}$

▶ **Definition 123** (Simpleton wrapper). *(see figure 25) Let $\mathcal{A}$ be a PSIOA. We note $\tilde{\mathcal{A}}^{sw}$ the* simpleton wrapper *of $\mathcal{A}$ as the following PCA:*

- $psioa(\tilde{\mathcal{A}}^{sw}) = \mathcal{A}$
- $config(\tilde{\mathcal{A}}^{sw})(q^{\phi}_{\mathcal{A}}) = (\emptyset, \emptyset)$
- $\forall q \in Q_{\mathcal{A}}, q_{\mathcal{A}} \neq q^{\phi}_{\mathcal{A}}, config(\tilde{\mathcal{A}}^{sw})(q) = (\mathcal{A}, \{(\mathcal{A}, q)\})$
- $\forall q \in Q_{\mathcal{A}}, \forall a \in \widehat{sig}(\tilde{\mathcal{A}}^{sw})(q), created(\tilde{\mathcal{A}}^{sw})(q)(a) = \emptyset$
- $\forall q \in Q_{\mathcal{A}}, hidden\text{-}actions(\tilde{\mathcal{A}}^{sw})(q) = \emptyset$

We can remark that when $\tilde{\mathcal{A}}^{sw}$ enters in $q^{\phi}_{\tilde{\mathcal{A}}^{sw}} = q^{\phi}_{\mathcal{A}}$ where $\widehat{sig}(\tilde{\mathcal{A}}^{sw})(q^{\phi}_{\tilde{\mathcal{A}}^{sw}}) = \emptyset$, this matches the moment where $\mathcal{A}$ enters in $q^{\phi}_{\mathcal{A}}$ where $\widehat{sig}(\mathcal{A})(q^{\phi}_{\mathcal{A}}) = \emptyset$, s.t. the corresponding configuration is the empty one.

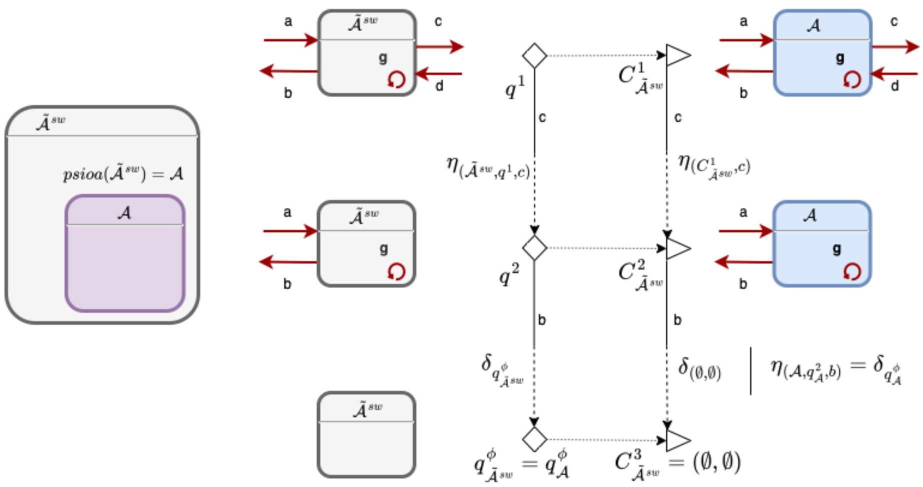


Figure 25 Simpleton wrapper

12.2 Partial-compatibility of $(X_{\mathcal{A}} \setminus \{\mathcal{A}\})$ and $\tilde{\mathcal{A}}^{sw}$

In this subsection, we show that $(X_{\mathcal{A}} \setminus \{\mathcal{A}\})$ and $\tilde{\mathcal{A}}^{sw}$ are partially-compatible and that $(X_{\mathcal{A}} \setminus \{\mathcal{A}\})||\tilde{\mathcal{A}}^{sw}$ mimics $X_{\mathcal{A}}$ as long as no creation of $\mathcal{A}$ occurs (see figure 26).

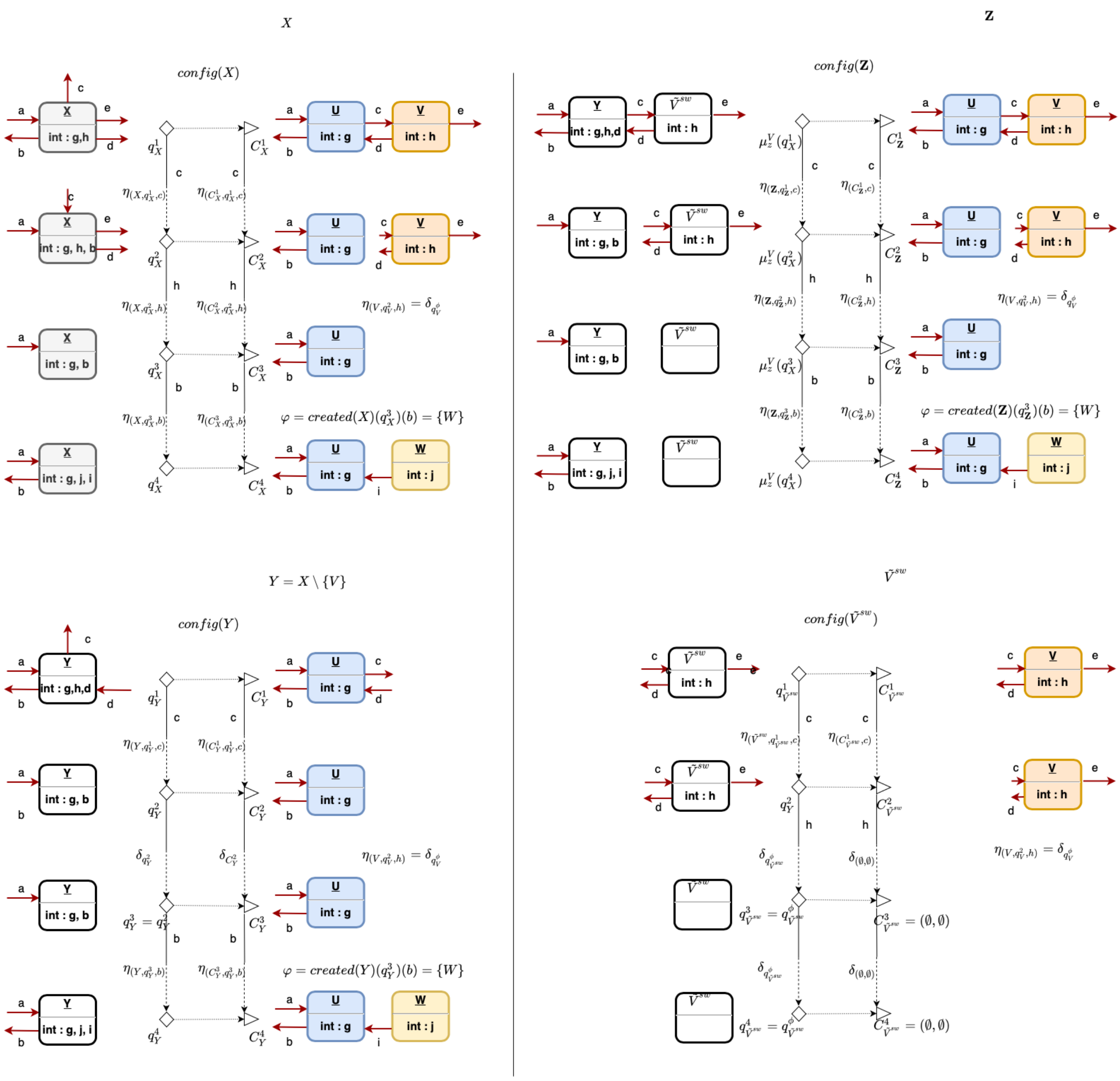


Figure 26 Reconstruction of a PCA via $\mathbf{Z} = (X, X \setminus \{V\})$

Map X and $(X \setminus \{\mathcal{A}\}, \tilde{\mathcal{A}}^{sw})$

We first introduce two functions to map X and $(X \setminus \{\mathcal{A}\}, \tilde{\mathcal{A}}^{sw})$.

▶ **Definition 124** ($\mu_z^{\mathcal{A}}$ and $\mu_e^{\mathcal{A}}$: mapping of reconstruction). *Let $\mathcal{A} \in Autids$, X be a $\mathcal{A}$-fair PCA, $Y = X \setminus \mathcal{A}$. Let $\tilde{\mathcal{A}}^{sw}$ be the simpleton wrapper of $\mathcal{A}$. Let $q_{\mathcal{A}}^{\phi} \in Q_{\mathcal{A}}$ the (assumed) unique state s.t. $\widehat{sig}(\mathcal{A})(q_{\mathcal{A}}^{\phi}) = \emptyset$. We note:*

- *The function $X.\mu_z^{\mathcal{A}} : Q_X \to Q_Y \times Q_{\tilde{\mathcal{A}}^{sw}}$ s.t. $\forall q_X \in Q_X$, $X.\mu_z^{\mathcal{A}}(q_X) = (X.\mu_s^{\mathcal{A}}(q_X), q_{\mathcal{A}})$ with $q_{\mathcal{A}} = map(config(X)(q_X))(\mathcal{A})$ if $\mathcal{A} \in (auts(config(X)(q_X)))$ and $q_{\mathcal{A}} = q_{\mathcal{A}}^{\phi}$ otherwise.*
- *The function $X.\mu_e^{\mathcal{A}}$ that maps any alternating sequence $\alpha_X = q_X^0, a^1, q_X^1, a^2...$ of states and actions of X, to $\mu_e^{\mathcal{A}}(\alpha_X)$ the alternating sequence $\alpha_Z = X.\mu_z^{\mathcal{A}}(q_X^0), a^1, X.\mu_z^{\mathcal{A}}(q_X^1), a^2, ...$*

The symbol $^{\mathcal{A}}$ and X. are omitted when this is clear in the context.

Now, we recall definition 67 of $\mathcal{A}$-conservative PCA, an additional condition to allow the compatibility between $X \setminus \mathcal{A}$ and $\tilde{\mathcal{A}}^{sw}$.

▶ **Definition 125** ($\mathcal{A}$-conservative PCA (recall)). *Let X be a PCA, $\mathcal{A} \in Autids$. We say that X is $\mathcal{A}$-conservative if it is $\mathcal{A}$-fair and for every state $q_X \in Q_X$, $C_X = (\mathbf{A}_X, \mathbf{S}_X) = config(X)(q_X)$ s.t. $\mathcal{A} \in \mathbf{A}_X$ and $\mathbf{S}_X(\mathcal{A}) \triangleq q_{\mathcal{A}}$, $hidden\text{-}actions(X)(q_X) = hidden\text{-}actions(X)(q_X) \setminus \widehat{ext}(\mathcal{A})(q_{\mathcal{A}})$.*

A $\mathcal{A}$-conservative PCA is a $\mathcal{A}$-fair PCA that does not hide any output action that could be an external action of $\mathcal{A}$.

Preservation of properties

Now we start a sequence of lemma (from lemma 126 to lemma 132) about properties preserved after reconstruction to eventually show in theorem 134 that $X \setminus \mathcal{A}$ and $\tilde{\mathcal{A}}^{sw}$ are partially-compatible.

The next lemma shows that reconstruction preserves signature compatibility.

▶ **Lemma 126** (preservation of signature compatibility of configurations). *Let $\mathcal{A} \in Autids$. Let X be a $\mathcal{A}$-conservative PCA, $Y = X \setminus \mathcal{A}$. Let $q_X \in Q_X$, $C_X = (\mathbf{A}_X, \mathbf{S}_X) = config(X)(q_X)$. Let $q_Y \in Q_Y, q_Y = \mu_s(q_X)$. Let $C_Y = (\mathbf{A}_Y, \mathbf{S}_Y) = config(Y)(q_Y)$.*

If $\mathcal{A} \in \mathbf{A}_X$ and $q_{\mathcal{A}} = \mathbf{S}_X(\mathcal{A})$, then $sig(C_Y)$ and $sig(\tilde{\mathcal{A}}^{sw})(q_{\mathcal{A}})$ are compatible and $sig(C_X) = sig(C_Y) \times sig(\tilde{\mathcal{A}}^{sw})(q_{\mathcal{A}})$.

If $\mathcal{A} \notin \mathbf{A}_X$, then $sig(C_Y)$ and $sig(\tilde{\mathcal{A}}^{sw})(q^{\phi}_{\mathcal{A}})$ are compatible and $sig(C_X) = sig(C_Y) \times sig(\tilde{\mathcal{A}}^{sw})(q^{\phi}_{\mathcal{A}})$.

Proof. Let $\mathcal{A} \in Autids$ Let X and $Y \setminus \{\mathcal{A}\}$ be PCA. Let $q_X \in Q_X$. Let $C_X = config(X)(q_X)$, $\mathbf{A}_X = auts(C_X)$ and $\mathbf{S}_X = map(C_X)$. Let $q_Y \in Q_Y, q_Y = \mu_s(q_X)$. Let $C_Y = config(Y)(q_Y)$, $\mathbf{A}_Y = auts(C_Y)$ and $\mathbf{S}_Y = map(C_Y)$. By definition of Y, $C_Y = C_X \setminus \{\mathcal{A}\}$.

Case 1: $\mathcal{A} \in \mathbf{A}_X$

Since X is a PCA, C_X is a compatible configuration, thus $((\mathbf{A}_Y, \mathbf{S}_Y) \cup (\mathcal{A}, q_{\mathcal{A}}))$ is a compatible configuration. Finally $sig(C_Y)$ and $sig(\mathcal{A})(q_{\mathcal{A}})$ are compatible with $sig(\mathcal{A})(q_{\mathcal{A}}) = sig(\tilde{\mathcal{A}}^{sw})(q^{\phi}_{\mathcal{A}})$.

By definition of intrinsinc attributes of a configuration, that are constructed with the attributes of the automaton issued from the composition of the family of automata of the configuration, we have $\mathbf{A}_X = \mathbf{A}_Y \cup \{\mathcal{A}\}$ and $sig(C_X) = sig(C_Y) \times sig(\mathcal{A})(q_{\mathcal{A}})$, that is $sig(C_X) = sig(C_Y) \times sig(\tilde{\mathcal{A}}^{sw})(q_{\mathcal{A}})$.

Case 2: $\mathcal{A} \notin \mathbf{A}_X$

Since X is a PCA, C_X is a compatible configuration, thus $C_Y = C_X$ is a compatible configuration. Finally $sig(C_Y)$ and $sig(\mathcal{A})(q^{\phi}_{\mathcal{A}}) = (\emptyset, \emptyset, \emptyset) = sig(\mathcal{A})(q_{\mathcal{A}}) = sig(\tilde{\mathcal{A}}^{sw})(q^{\phi}_{\mathcal{A}})$ are compatible.

By definition of intrinsinc attributes of a configuration, that are constructed with the attributes of the automaton issued from the composition of the family of automata of the configuration (here $\mathbf{A}_Y$ and $\mathbf{A}_X = \mathbf{A}_Y$), we have $sig(C_X) = sig(C_Y)$. Furthermore, $sig(\tilde{\mathcal{A}}^{sw})(q^{\phi}_{\mathcal{A}}) = sig(\mathcal{A})(q^{\phi}_{\mathcal{A}}) = (\emptyset, \emptyset, \emptyset)$. Thus $sig(C_X) = sig(C_Y) \times sig(\tilde{\mathcal{A}}^{sw})(q^{\phi}_{\mathcal{A}})$ ◀

The next lemma shows that reconstruction preserves signature.

▶ **Lemma 127** (preservation of signature). *Let $\mathcal{A} \in Autids$. Let X be a $\mathcal{A}$-conservative PCA, $\mathcal{A} \in Autids$, $Y = X \setminus \{\mathcal{A}\}$. For every $q_X \in Q_X$, we have $sig(X)(q_X) = sig(Y)(q_Y) \times sig(\tilde{\mathcal{A}}^{sw})(q_{\mathcal{A}})$ with $(q_Y, q_{\mathcal{A}}) = \mu^{\mathcal{A}}_z(q_X)$.*

Proof. The last lemma 126 tell us for every $q_X \in Q_X$, we have $sig(config(X)(q_X)) = sig(config(Y)(q_Y)) \times sig(\tilde{\mathcal{A}}^{sw})(q_{\mathcal{A}})$ with $(q_Y, q_{\mathcal{A}}) = \mu_z(q_X)$. Since X is $\mathcal{A}$-conservative, we have (*) $sig(X)(q_X) = hide(sig(config(X)(q_X)), \underline{acts})$ where $\underline{acts} \subseteq (out(X)(q_X) \setminus (ext(\mathcal{A})(q_{\mathcal{A}}))$. Hence $sig(Y)(q_Y) = hide(sig(config(Y)(q_Y)), \underline{acts})$. Since (**) $\underline{acts} \cap ext(\mathcal{A})(q_{\mathcal{A}}) = \emptyset$, $sig(Y)(q_Y)$ and $sig(\mathcal{A})(q_{\mathcal{A}})$ are also compatible. We have $sig(config(X)(q_X)) = sig(config(Y)(q_Y)) \times sig(\mathcal{A})(q_{\mathcal{A}}) = sig(config(Y)(q_Y)) \times sig(\tilde{\mathcal{A}}^{sw})(q_{\mathcal{A}})$ which gives because of (*) $hide(sig(config(X)(q_X)), \underline{acts}) = hide(sig(config(Y)(q_Y)), \underline{acts}) \times sig(\mathcal{A})(q_{\mathcal{A}})$, that is $sig(X)(q_X) = sig(Y)(q_Y) \times sig(\mathcal{A})(q_{\mathcal{A}}) = sig(Y)(q_Y) \times sig(\tilde{\mathcal{A}}^{sw})(q_{\mathcal{A}})$. ◀

The next lemma shows that reconstruction preserves partial-compatibility at any reachable state.

▶ **Lemma 128** (preservation of compatibility at any reachable state)**.** *Let $\mathcal{A} \in Autids$, X be a $\mathcal{A}$-conservative PCA, $Y = X \setminus \{\mathcal{A}\}$, $\mathbf{Z} = (Y, \tilde{\mathcal{A}}^{sw})$ Let $q_Z = (q_Y, \tilde{q}_{\tilde{\mathcal{A}}^{sw}}) \in Q_Y \times Q_{\tilde{\mathcal{A}}^{sw}}$ and $q_X \in Q_X$ s.t. $\mu_z^{\mathcal{A}}(q_X) = q_Z$. Then $psioa(Y)$ and $psioa(\tilde{\mathcal{A}}^{sw})$ are compatible. Moreover, by definition of $Y = X \setminus \{\mathcal{A}\}$ and $\tilde{\mathcal{A}}^{sw}$ being the simpleton wrapper of $\mathcal{A}$, the sub-automaton exclusivity and creation exclusivity of definition 21 are necessarily ensured. Hence,* $\mathbf{Z}$ *is compatible at state q_Z.*

Proof. Since X is a $\mathcal{A}$-conservative PCA, the previous lemma 127 ensures that $sig(Y)(q_Y)$ and $sig(\mathcal{A})(q_{\mathcal{A}}) = sig(\tilde{\mathcal{A}}^{sw})(q_{\mathcal{A}})$ are compatible, thus by definition $\mathbf{Z}$ is compatible at state q_Z. ◀

Here, we show that reconstruction preserves probabilistic distribution of corresponding transition, as long as no creation of the concerned automaton occurs.

▶ **Lemma 129** (homomorphic transition without creation)**.** *Let $\mathcal{A} \in Autids$, X be a $\mathcal{A}$-conservative PCA, $Y = X \setminus \{\mathcal{A}\}$, $\mathbf{Z} = (Y, \tilde{\mathcal{A}}^{sw})$. Let $q_Z = (q_Y, \tilde{q}_{\tilde{\mathcal{A}}^{sw}}) \in Q_Y \times Q_{\tilde{\mathcal{A}}^{sw}}$ and $q_X \in Q_X$ s.t. (i) $\mu_z(q_X) = q_Z$. Let $a \in sig(X)(q_X) = sig(Y)(q_Y) \times sig(\tilde{\mathcal{A}}^{sw})(\tilde{q}_{\tilde{\mathcal{A}}^{sw}})$, verifying (ii: No creation from $\mathcal{A}$) If a is $\mathcal{A}$-exclusive in state q_X,then $created(X)(q_X)(a) = \emptyset$,*

- *If $\mathcal{A}$ is not created by a, i.e. if either*
  - *$\mathcal{A} \in auts(config(X)(q_X))$, or*
  - *$\mathcal{A} \notin auts(config(X)(q_X))$ and $\mathcal{A} \notin created(X)(q_X)(a)$ (X does not create $\mathcal{A}$ with probability 1)*

  Then $\eta_{(X,q_X,a)} \overset{\mu_z}{\leftrightarrow} \eta_{(\mathbf{Z},q_Z,a)}$
- *If $\mathcal{A}$ is created by a i.e. $\mathcal{A} \notin auts(config(X)(q_X))$ and $\mathcal{A} \in created(X)(q_X)(a)$ (X creates $\mathcal{A}$ with probability 1)*

  Then $\eta_{(X,q_X,a)} \overset{f^{\phi}}{\leftrightarrow} \eta_{(\mathbf{Z},q_Z,a)}$ where $f^{\phi} : q'_X \in supp(\eta_{(X,q_X,a)}) \mapsto (X.\mu_s^{\mathcal{A}}(q'_X), q^{\phi}_{\tilde{\mathcal{A}}^{sw}})$.

Proof. By lemma 127, we have $sig(X)(q_X) = sig(Y)(q_Y) \times sig(\mathcal{A})(q_{\mathcal{A}}) = sig(Y)(q_Y) \times sig(\tilde{\mathcal{A}}^{sw})(\tilde{q}_{\tilde{\mathcal{A}}^{sw}} = q_{\mathcal{A}})$.

We note $C_X = (\mathbf{A}_X, \mathbf{S}_X) = config(X)(q_X)$, $C_Y = (\mathbf{A}_Y, \mathbf{S}_Y) = config(Y)(q_Y)$, $C_{\tilde{\mathcal{A}}^{sw}} = (\mathbf{A}_{\tilde{\mathcal{A}}^{sw}}, \mathbf{S}_{\tilde{\mathcal{A}}^{sw}}) = config(\tilde{\mathcal{A}}^{sw})(q_{\tilde{\mathcal{A}}^{sw}})$. By construction of μ_z, $C_X = C_Y \cup C_{\tilde{\mathcal{A}}^{sw}}$ with C_Y and $C_{\tilde{\mathcal{A}}^{sw}}$ compatible configuration (1).

We note $\varphi_X = created(X)(q_X)(a)$, $\varphi_Y = \varphi_X \setminus \{\mathcal{A}\}$, $\varphi_{\tilde{\mathcal{A}}^{sw}} = \emptyset$, $\varphi_Z = \varphi_X \cup \varphi_{\tilde{\mathcal{A}}^{sw}}$. If a is $\mathcal{A}$-exclusive in state q_X, then $\varphi_X = \varphi_Y = \emptyset$.

- If $\mathcal{A}$ is not created by a, then $\varphi_X = \varphi_Z$,
- If $\mathcal{A}$ is created by a, then $\varphi_X = \varphi_Z \cup \{\mathcal{A}\}$ and $\varphi_Z = \varphi_X \setminus \{\mathcal{A}\}$

Since X is a PCA and $(q_X, a, \eta_{(X,q_X,a)}) \in D_X$, the constraint 2 of top/down transition preservation says that there exists a unique reduced configuration distribution η'_X s.t. $\eta_{(X,q_X,a)} \overset{f^X}{\leftrightarrow} \eta'_X$ with $f^X = config(X)$ and $config(X)(q_X) \Longrightarrow_{\varphi_X} \eta'_X$ (2).

For Y (resp. $\tilde{\mathcal{A}}^{sw}$) we note $\eta_Y = \eta_{(Y,q_Y,a)}$ if $a \in \widehat{sig}(Y)(q_Y)$ and $\eta_Y = \delta_{q_Y}$ otherwise (resp. $\eta_{\tilde{\mathcal{A}}^{sw}} = \eta_{(\tilde{\mathcal{A}}^{sw},q_{\tilde{\mathcal{A}}^{sw}},a)}$ if $a \in \widehat{sig}(\tilde{\mathcal{A}}^{sw})(q_{\tilde{\mathcal{A}}^{sw}})$ and $\eta_{\tilde{\mathcal{A}}^{sw}} = \delta_{q_{\tilde{\mathcal{A}}^{sw}}}$ otherwise).

Since Y and $\tilde{\mathcal{A}}^{sw}$ are PCA, either because of the constraint 2 of top/down transition preservation or because a is not action of the signature, there exists a unique reduced configuration distribution η'_Y s.t. $\eta_Y \overset{f^Y}{\leftrightarrow} \eta'_Y$ with $f^Y = config(Y)$ and $config(Y)(q_Y) \Longrightarrow_{\varphi_Y} \eta'_Y$ (resp. $\eta'_{\tilde{\mathcal{A}}^{sw}}$ s.t. $\eta_{\tilde{\mathcal{A}}^{sw}} \overset{f^{\tilde{\mathcal{A}}^{sw}}}{\leftrightarrow} \eta'_{\tilde{\mathcal{A}}^{sw}}$ with $f^{\tilde{\mathcal{A}}^{sw}} = config(\tilde{\mathcal{A}}^{sw})$ and $config(\tilde{\mathcal{A}}^{sw})(q_{\tilde{\mathcal{A}}^{sw}}) \Longrightarrow_{\varphi_{\tilde{\mathcal{A}}^{sw}}} \eta'_{\tilde{\mathcal{A}}^{sw}}$) (3).

By construction $\forall (q'_Y, q'_{\tilde{\mathcal{A}}^{sw}}) \in Q_Y \times Q_{\tilde{\mathcal{A}}^{sw}}$, $constitution(Y)(q'_Y) \cap constitution(\tilde{\mathcal{A}}^{sw})(q'_{\tilde{\mathcal{A}}^{sw}}) = \emptyset$ (and so $auts(config(Y)(q'_Y)) \cap auts(config(\tilde{\mathcal{A}}^{sw})(q'_{\tilde{\mathcal{A}}^{sw}})) = \emptyset$) which means (**) $base(C_Y, a, \varphi_Y) \cap base(C_{\tilde{\mathcal{A}}^{sw}}, a, \varphi_{\tilde{\mathcal{A}}^{sw}}) = \emptyset$.

The conjonction of (1), (2), (3) and (**) allows us to apply the lemma 35. This means

- by item 1b of lemma 35: $merge((\eta'_{\tilde{\mathcal{A}}^{sw}}, \eta'_Y)) \overset{f^s}{\leftrightarrow} join((\eta'_{\tilde{\mathcal{A}}^{sw}}, \eta'_Y))$ with $f^s : C'_Z \mapsto (C'_Y, C'_{\tilde{\mathcal{A}}^{sw}})$ s.t. i) $C'_Z = C'_Y \cup C'_{\tilde{\mathcal{A}}^{sw}}$, ii) $\mathcal{A} \notin C'_Y$ and iii) $\forall \mathcal{B} \neq \mathcal{A}$, $\mathcal{B} \notin C'_{\tilde{\mathcal{A}}^{sw}}$ (4)
- by item 1d of lemma 35: $C_X \overset{a}{\Longrightarrow}_{\varphi_Z} merge((\eta'_{\tilde{\mathcal{A}}^{sw}}, \eta'_Y))$ (5)

Furthermore $\eta_{\mathbf{Z},q_Z,a} = \eta_Y \otimes \eta_{\tilde{\mathcal{A}}^{sw}}$. So by (3), $\eta_{\mathbf{Z},q_Z,a} \overset{f^Z}{\longleftrightarrow} join((\eta'_{\tilde{\mathcal{A}}^{sw}}, \eta'_Y))$ (***) with $f^Z : q'_Z = (q'_Y, q'_{\tilde{\mathcal{A}}^{sw}}) \mapsto (config(Y)(q'_{\tilde{\mathcal{A}}^{sw}}), config(\tilde{\mathcal{A}}^{sw})(q'_{\tilde{\mathcal{A}}^{sw}}))$.

Now we deal have to separate the treatment of the two cases:

- If $\mathcal{A}$ is not created by a, since $\varphi_Z = \varphi_X$, because of (5) and (2), $merge((\eta'_{\tilde{\mathcal{A}}^{sw}}, \eta'_Y)) = \eta'_X$ and because of (2) $\eta_{(X,q_X,a)} \overset{f^X}{\leftrightarrow} merge((\eta'_{\tilde{\mathcal{A}}^{sw}}, \eta'_Y))$ (6). Because of (6) and (4), $\eta_{(X,q_X,a)} \overset{g}{\longleftrightarrow} join((\eta'_{\tilde{\mathcal{A}}^{sw}}, \eta'_Y))$ with $g = f^s \circ f^X$.

  Hence, if $\mathcal{A}$ is not created by a $\eta_{(X,q_X,a)} \overset{h}{\longleftrightarrow} \eta_{(\mathbf{Z},q_Z,a)}$ with $h = (f^Z)^{-1} \circ f^s \circ f^X = \mu_z$ which ends the proof for this case.
- If $\mathcal{A}$ is created by a, we have both
  - $C_X \overset{a}{\Longrightarrow}_{\varphi_Z} merge((\eta'_{\tilde{\mathcal{A}}^{sw}}, \eta'_Y))$
  - $C_X \overset{a}{\Longrightarrow}_{\varphi_Z \cup \{\mathcal{A}\}} \eta'_X$

  which means $C_X \overset{a}{\rightharpoonup} \eta'_p$ with
  - $merge((\eta'_{\tilde{\mathcal{A}}^{sw}}, \eta'_Y))$ generated by η'_p and φ_Z and
  - η'_X generated by η'_p and $\varphi_Z \cup \{\mathcal{A}\}$.

  Thus $\eta'_X \overset{g^\phi}{\longleftrightarrow} merge((\eta'_{\tilde{\mathcal{A}}^{sw}}, \eta'_Y))$ with $g^\phi : C'_X = C'_Y \cup \bar{C}_{\tilde{\mathcal{A}}^{sw}} \mapsto C'_Y$. where $\bar{C}_{\tilde{\mathcal{A}}^{sw}}(\{\mathcal{A}\}, \mathbf{S}'_{\tilde{\mathcal{A}}^{sw}} : \mathcal{A} \mapsto \bar{q}_{\tilde{\mathcal{A}}^{sw}})$.

  To summerize, we have:
  - $\eta_{(X,q_X,a)} \overset{f^X}{\longleftrightarrow} \eta'_X$
  - $\eta'_X \overset{g^\phi}{\longleftrightarrow} merge((\eta'_{\tilde{\mathcal{A}}^{sw}}, \eta'_Y))$
  - $merge((\eta'_{\tilde{\mathcal{A}}^{sw}}, \eta'_Y)) \overset{f^s}{\longleftrightarrow} join((\eta'_{\tilde{\mathcal{A}}^{sw}}, \eta'_Y))$
  - $\eta_{(\mathbf{Z},q_z,a)} \overset{f^Z}{\longleftrightarrow} join((\eta'_{\tilde{\mathcal{A}}^{sw}}, \eta'_Y))$

  Hence $\eta_{(X,q_X,a)} \overset{h}{\longleftrightarrow} \eta_{(\mathbf{Z},q_Z,a)}$ with $f^\phi = (f^Z)^{-1} \circ f^s \circ g^\phi \circ f^X$, i.e. $f^\phi : q'_X \in supp(\eta_{(X,q_X,a)}) \mapsto (X.\mu_s^{\mathcal{A}}(q'_X), q^\phi_{\tilde{\mathcal{A}}^{sw}})$, which ends the proof for this case.

◀

The second case where $\mathcal{A}$ is created will not be used before section 14.

We take advantage of the lemma 132 used for theorem 134 to introduce the notion of twin PCA and extends directly the lemma 132 and theorem 134 to twin PCA.

▶ **Definition 130** ($X_{\bar{q}_X \to \bar{q}'_X}$)**.** *Let* $X = (Q_X, \bar{q}_X, sig(X), D_X)$ *be a PSIOA and* $\bar{q}'_X \in reachable(X)$*. We note* $X_{\bar{q}_X \to \bar{q}'_X}$ *the PSIOA* $X' = (Q_X, \bar{q}'_X, sig(X), D_X)$*.*

Two PCA X and X' are $\mathcal{A}$-twin if they differ only by their start state where one of them corresponds to $\mathcal{A}$-creation.

▶ **Definition 131** ($\mathcal{A}$-twin)**.** *Let* $\mathcal{A} \in Autids$*. Let* X, X' *be PCA. We say that* $X' = X_{\bar{q}_X \to \bar{q}_{X'}}$ *is a* $\mathcal{A}$*-twin of* X *if it differs from* X *at most only by its start states* $\bar{q}_{X'}$ *reachable by* X *s.t. either* $X' = X$ *or* $\mathcal{A} \in config(X')(\bar{q}_{X'})$ *and* $map(config(X')(\bar{q}_{X'}))(\mathcal{A}) = \bar{q}_{\mathcal{A}}$*. If* X' *is a* $\mathcal{A}$*-twin of* X *and* $Y = X \setminus \{\mathcal{A}\}$ *and* $Y' = X' \setminus \{\mathcal{A}\}$*, we slightly abuse the notation and say that* Y' *is a* $\mathcal{A}$*-twin of* Y'*.*

▶ **Lemma 132** (partial surjectivity 1)**.** *Let* $\mathcal{A} \in Autids$*. Let* X *be a PCA* $\mathcal{A}$*-conservative and* X' *a* $\mathcal{A}$*-twin of* X*. Let* $Y' = X' \setminus \{\mathcal{A}\}$*. Let* Y' *be a* $\mathcal{A}$*-twin of* Y*. Let* $\mathbf{Z}' = (Y', \tilde{\mathcal{A}}^{sw})$*.*

Let $\alpha = q^0, a^1, ..., a^k, q^k$ *be a pseudo execution of* $\mathbf{Z}'$*. Let assume the presence of* $\mathcal{A}$ *in* α*, i.e.* $\forall s \in [0, k-1]$*,* $q^s_{\tilde{\mathcal{A}}^{sw}} \neq q^{\phi}_{\mathcal{A}}$ *.*

Then $\exists \tilde{\alpha} \in Execs(X')$*, s.t.* $X'.\mu^{\mathcal{A}}_e(\tilde{\alpha}) = \alpha$*.*

Proof. By induction on each prefix $\alpha^s = q^0, a^1, ..., a^s, q^s$ with $s \leq k$.

Basis: case 1) $\mathcal{A} \in config(X')(\bar{q}_{X'})$: We have $\mu_z(\bar{q}_{X'}) = (\bar{q}_{Y'}, \bar{q}_{\mathcal{A}})$. Hence $\mu_e(\bar{q}_{X'}) = (\bar{q}_{Y'}, \bar{q}_{\mathcal{A}})$.

case 2) $\mathcal{A} \notin config(X')(\bar{q}_{X'})$, (necessarily $X = X'$): $\mu_z(\bar{q}_{X'}) = (\bar{q}_{Y'}, q^{\phi}_{\mathcal{A}})$. Hence $\mu_e(\bar{q}_{X'}) = (\bar{q}_{Y'}, q^{\phi}_{\mathcal{A}})$.

Induction: we assume this is true for s and we show it implies this true for $s+1$. We note $\tilde{\alpha}_s$ s.t. $\mu_e(\tilde{\alpha}^s) = \alpha^s$. We also note $\tilde{q}^s = lstate(\tilde{\alpha}^s)$ and we have by induction assumption $\mu_z(\tilde{q}^s) = q^s = (q^s_Y, q^s_{\mathcal{A}})$. Because of preservation of signature compatibility, $sig(X)(\tilde{q}^s)) = sig(Y)(q^s_Y)) \times sig(\mathcal{A})(q^s_{\mathcal{A}}))$. Hence $a^{s+1} \in sig(X)(\tilde{q}^s)$. Thereafter, by construction of $X \setminus \{\mathcal{A}\}$ there exists $\tilde{q}^{s+1}$ s.t. $q^{s+1} = \mu^{\mathcal{A}}_z(\tilde{q}^{s+1})$. Finally, since no creation of and from $\mathcal{A}$ occurs by assumption of presence of $\mathcal{A}$, we can use lemma 129 of homomorphic transition which give $\eta_{(X, \tilde{q}^s, a^{s+1})} \overset{\mu_z}{\leftrightarrow} \eta_{(\mathbf{Z}, q^s, a^{s+1})}$ which means $\tilde{q}^{s+1} \in supp(\eta_{(X, \tilde{q}^s, a^{s+1})})$ which ends the induction and so the proof. ◀

Before using lemma 132 and 128 to demonstrate theorem 134 of partial compatibility after reconstruction, we take the opportunity to extend lemma 132:

▶ **Lemma 133** (partial surjectivity 2)**.** *Let* $\mathcal{A} \in Autids$*. Let* X *be a PCA* $\mathcal{A}$*-conservative. Let* $Y = X \setminus \mathcal{A}$*. Let* Y' *be a* $\mathcal{A}$*-twin of* Y*. Let* $\mathcal{Z} = Y' || \tilde{\mathcal{A}}^{sw}$*.*

Let $\alpha = q^0, a^1, ..., a^k, q^k$ *be a an execution of* $\mathcal{Z}$*. Let assume (a)* $q^s_{\tilde{\mathcal{A}}^{sw}} \neq q^{\phi}_{\mathcal{A}}$ *for every* $s \in [0, k^*]$ *(b)* $q^s_{\tilde{\mathcal{A}}^{sw}} = q^{\phi}_{\tilde{\mathcal{A}}^{sw}}$ *for every* $s \in [k^*+1, k]$ *(c) for every* $s \in [k^*+1, k-1]$*, for every* $\tilde{q}^s$*, s.t.* $\mu_z(\tilde{q}^s) = q^s$*,* $\mathcal{A} \notin created(X)(\tilde{q}^s)(a^{s+1})$*. Then* $\exists \tilde{\alpha} \in Frags(X)$*, s.t.* $\mu_e(\tilde{\alpha}) = \alpha$*. If* $Y' = Y$*,* $\exists \tilde{\alpha} \in Execs(X)$*, s.t.* $\mu_e(\tilde{\alpha}) = \alpha$*.*

Proof. We already know this is true up to k^* because of lemma 132. We perform the same induction than the one of the previous lemma on partial surjectivity: We note $\tilde{\alpha}_s$ s.t. $\mu_e(\tilde{\alpha}^s) = \alpha^s$. We also note $\tilde{q}^s = lstate(\tilde{\alpha}^s)$ and we have by induction assumption $\mu_z(\tilde{q}^s) = q^s = (q^s_Y, q^s_{\mathcal{A}})$. Because of preservation of signature compatibility, $sig(X)(\tilde{q}^s)) = sig(Y)(q^s_Y)) \times sig(\mathcal{A})(q^s_{\mathcal{A}})$. Hence $a^{k+1} \in sig(X)(\tilde{q}^s)$. Now we use the assumption (c), that says that $\mathcal{A} \notin created(X)(\tilde{q}^s)(a^{s+1})$ to be able to apply preservation of transition since no creation of $\mathcal{A}$ can occurs. ◀

Now we can use lemma 132 and 128 to demonstrate theorem 134 of partial compatibility after reconstruction.

▶ **Theorem 134** (Partial-compatibility after resconstruction)**.** *Let $\mathcal{A} \in Autids$. Let X be a PCA $\mathcal{A}$-conservative s.t. $\forall q_X \in Q_X$, for every action a $\mathcal{A}$-exclusive in q_X, $created(X)(q_X)(a) = \emptyset$. Let X' ba a $\mathcal{A}$-twin of X and $Y' = X' \setminus \{\mathcal{A}\}$. Then Y' and $\tilde{\mathcal{A}}^{sw}$ are partially-compatible.*

Proof. Let $\mathbf{Z}' = (Y', \tilde{\mathcal{A}}^{sw})$. Let α be a pseudo-execution of $\mathbf{Z}'$ with Let $lstate(\alpha) = q_{\mathbf{Z}} = (q_{Y'}, q_{\tilde{\mathcal{A}}^{sw}})$. Case 1) $q_{\tilde{\mathcal{A}}^{sw}} = q^{\phi}_{\tilde{\mathcal{A}}^{sw}}$. The compatibility is immediate since $sig(\tilde{\mathcal{A}}^{sw})(q^{\phi}_{\tilde{\mathcal{A}}^{sw}}) = \emptyset$. Case 2) $q_{\tilde{\mathcal{A}}^{sw}} \neq q^{\phi}_{\tilde{\mathcal{A}}^{sw}}$. Since (*) $\mathcal{A}$ cannot be re-created after destruction by neither Y or $\tilde{\mathcal{A}}^{sw}$ and (**) $\forall q_X \in Q_X$, for every action a $\mathcal{A}$-exclusive in q_X, $created(X)(q_X)(a) = \emptyset$ we can use the previous lemma 132 to show $\exists\ \tilde{\alpha} \in Execs(X')$, s.t. $\mu_e(\tilde{\alpha}) = \alpha$. Thus, $lstate(\alpha) = \mu_z(lstate(\tilde{\alpha}))$ which means $\mathbf{Z}'$ is partially-compatible at $lstate(\alpha)$ by lemma 128. Hence $\mathbf{Z}$ is partially-compatible at every reachable state, which means Y' and $\tilde{\mathcal{A}}^{sw}$ are partially-compatible. We can legitimately note $\mathcal{Z}' = Y'||\tilde{\mathcal{A}}^{sw}$. ◀

Since $\mathbf{Z}' = (Y', \tilde{\mathcal{A}}^{sw})$ is partially-compatible, we can legitimately note $\mathcal{Z}' = Y'||\tilde{\mathcal{A}}^{sw}$, which will be the standard notation in the remaining.

12.3 Execution-matching from X to $X \setminus \{\mathcal{A}\}||\tilde{\mathcal{A}}^{sw}$

In this subsection, we show in theorem 140 that $X.\mu_e^{\mathcal{A}}$ is a (incomplete) PCA executions-matching from X to $(X \setminus \{\mathcal{A}\})||\tilde{\mathcal{A}}^{sw}$ in a restricted set of executions of X that do not create $\mathcal{A}$.

We start by defining the restricted set of executions of X that do not create $\mathcal{A}$ with definitions 135 and 136.

▶ **Definition 135** (execution without creation)**.** *Let $\mathcal{A}$ be a PSIOA. Let X be a PCA , we note execs-without-creation$(X)(\mathcal{A})$ the set of executions of X without creation of $\mathcal{A}$, i.e. execs-without-creation$(X)(\mathcal{A}) = \{\alpha = q^0 a^1 q^1 ... a^k q^k \in Execs(X) | \forall i \in [0, |\alpha|], \mathcal{A} \notin auts(config(X)(q^i)) \Longrightarrow \mathcal{A} \notin auts(config(X)(q^{i+1}))\}$.*

▶ **Definition 136** (reachable-by)**.** *Let X be a PSIOA or a PCA. Let $Execs'_X \subseteq Execs(X)$. We note reachable-by$(Execs'_X)$ the set of states of X reachable by an execution of $Execs'_X$, i.e. reachable-by$(Execs'_X) = \{q \in Q_X | \exists \alpha \in Execs'_X, lstate(\alpha) = q\}$*

The next 2 lemma show that reconstruction preserves configuration and signature. They will be sufficient to show that the restriction of $\mu_e^{\mathcal{A}}$ on *reachable-by*(*execs-without-creation*$(X)(\mathcal{A})$) is a PCA executions-matching.

▶ **Lemma 137** (μ_z configuration preservation)**.** *Let $\mathcal{A} \in Autids$. Let X be a $\mathcal{A}$-conservatiee PCA, $Y = X \setminus \mathcal{A}$, $Z = Y||\mathcal{A}^{sw}$. Let $q_X \in Q_X, q_Z = (q_Y, q_{\tilde{\mathcal{A}}^{sw}}) \in Q_Z$ s.t. $\mu_z(q_X) = q_Z$. Then $config(X)(q_X) = config(Z)(q_Z)$.*

Proof. By definition of composition of PCA, $config(Z)(q_Z) = config(Y)(q_Y) \cup config(\tilde{\mathcal{A}}^{sw})(q_{\tilde{\mathcal{A}}^{sw}})$. (*)

Also, by $\mu_z^{\mathcal{A}}$-correspondence, $config(X)(q_X) \setminus \mathcal{A} = config(Y)(q_Y)$ (**).

We deal with the two cases $\widehat{sig}(\tilde{A}^{sw})(q_{\tilde{\mathcal{A}}^{sw}}) = \emptyset$ or $\widehat{sig}(\tilde{A}^{sw})(q_{\tilde{\mathcal{A}}^{sw}}) \neq \emptyset$

- If $\widehat{sig}(\tilde{A}^{sw})(q_{\tilde{\mathcal{A}}^{sw}}) = \emptyset$, then $\mathcal{A} \notin aut(config(X)(q_X))$ which means, that $config(X)(q_X) = config(X)(q_X) \setminus \mathcal{A}$ (1). Furthermore, $config(\tilde{\mathcal{A}}^{sw})(q_{\tilde{\mathcal{A}}^{sw}}) = (\emptyset, \emptyset)$ (2) .Because of (**) and (1), $config(X)(q_X) = config(Y)(q_Y)$ and because of (*) and (2), $config(X)(q_X) = config(Z)(q_Z)$.

- If $\widehat{sig}(\tilde{\mathcal{A}}^{sw})(q_{\tilde{\mathcal{A}}^{sw}}) \neq \emptyset$, then $\mathcal{A} \in aut(config(X)(q_X))$. We note $C_{\mathcal{A}} = config(\tilde{\mathcal{A}}^{sw})(q_{\tilde{\mathcal{A}}^{sw}}) = (\{\mathcal{A}\}, \mathbf{S} : \mathcal{A} \mapsto map(config(X)(q_X))(\mathcal{A}))$. By (*), $config(Z)(q_Z) = config(Y)(q_Y) \cup C_{\mathcal{A}}$ and by (**) $config(Y)(q_Y) \cup C_{\mathcal{A}} = config(X)(q_X) \setminus \mathcal{A} \cup C_{\mathcal{A}} = config(X)(q_X)$. Hence, $config(X)(q_X) = config(Z)(q_Z)$

Thus in all cases, $config(X)(q_X) = config(Z)(q_Z)$ which ends the proof.

◀

▶ **Lemma 138** (μ_z signature-preservation)**.** *Let $\mathcal{A} \in Autids$. Let X be a $\mathcal{A}$-conservatiee PCA, $Y = X \setminus \mathcal{A}$, $Z = Y||\mathcal{A}^{sw}$. Let $q_X \in Q_X, q_Z = (q_Y, q_{\tilde{\mathcal{A}}^{sw}}) \in Q_Z$ s.t. $\mu_z(q_X) = q_Z$. Then $sig(X)(q_X) = sig(Z)(q_Z)$.*

Proof. By lemma 127 of preservation of signature $sig(X)(q_X) = sig(Y)(q_Y) \times sig(\tilde{\mathcal{A}}^{sw})(q_{\tilde{\mathcal{A}}^{sw}})$. By definition of composition of PCA, $sig(Z)(q_Z) = sig(Y)(q_Y) \times sig(\tilde{\mathcal{A}}^{sw})(q_{\tilde{\mathcal{A}}^{sw}})$ which ends the proof. ◀

Now we can states our strong PCA executions-matching:

▶ **Definition 139.** *Let $\mathcal{A}$ be a PSIOA. Let X be a $\mathcal{A}$-conservative PCA. Let $Y = X \setminus \{\mathcal{A}\}$ and $Z = Y||\tilde{\mathcal{A}}^{sw}$.*

We define $(X.\tilde{\mu}_z^{\mathcal{A}}, X.\tilde{\mu}_{tr}^{\mathcal{A}}, X.\tilde{\mu}_e^{\mathcal{A}})$ (noted $(\tilde{\mu}_z^{\mathcal{A}}, \tilde{\mu}_{tr}^{\mathcal{A}}, \tilde{\mu}_e^{\mathcal{A}})$ when it is clear in the context) as follows:

- *$\tilde{\mu}_z^{\mathcal{A}}$ the restriction of $\mu_z^{\mathcal{A}}$ on reachable-by(execs-without-creation$(X)(\mathcal{A})$).*
- *$f^{tr} : (q_X, a, \eta_{(X,q_X,a)}) \in D'_X \mapsto (\tilde{\mu}_z^{\mathcal{A}}(q_X), a, \eta_{(Z,\tilde{\mu}_z^{\mathcal{A}}(q_X),a)})$ where $D'_X = \{(q_X, a, \eta_{(X,q_X,a)}) \in D_X | q_X \in reachable\text{-}by(execs\text{-}without\text{-}creation(X)(\mathcal{A})), (\mathcal{A} \notin auts(config(X)(q_X) \Longrightarrow \mathcal{A} \notin created(X)(q_X)(a))\}$.*
- *$\tilde{\mu}_e^{\mathcal{A}}$ the restriction of $\mu_e^{\mathcal{A}}$ on execs-without-creation$(X)(\mathcal{A})$.*

▶ **Theorem 140** (execution-matching after reconstruction)**.** *Let $\mathcal{A}$ be a PSIOA. Let X be a $\mathcal{A}$-conservative PCA. Let $Y = X \setminus \{\mathcal{A}\}$. The triplet $(\tilde{\mu}_z^{\mathcal{A}}, \tilde{\mu}_{tr}^{\mathcal{A}}, \tilde{\mu}_e^{\mathcal{A}})$ is a strong PCA executions-matching from X to $Y||\tilde{\mathcal{A}}^{sw}$ if $\mathcal{A} \in auts(config(X_{\mathcal{A}})(start(X_{\mathcal{A}})))$ and from X to $Y||\tilde{\mathcal{A}}^{sw}_{\bar{q}_{\tilde{\mathcal{A}}^{sw}} \rightarrow q^{\phi}_{\tilde{\mathcal{A}}^{sw}}}$ otherwise.*

Proof. We note $Z = Y||\tilde{\mathcal{A}}^{sw}$ and $Z^{\phi} = Y||\tilde{\mathcal{A}}^{sw}_{\bar{q}_{\tilde{\mathcal{A}}^{sw}} \rightarrow q^{\phi}_{\tilde{\mathcal{A}}^{sw}}}$

- $\tilde{\mu}_z^{\mathcal{A}}$ is a strong PCA-state-matching since
  - starting state preservation is ensured by construction:
    - $\mathcal{A} \in auts(config(X_{\mathcal{A}})(start(X_{\mathcal{A}})))$: $\tilde{\mu}_z^{\mathcal{A}}(\bar{q}_X) = \bar{q}_Z$
    - $\mathcal{A} \notin auts(config(X_{\mathcal{A}})(start(X_{\mathcal{A}})))$:$\tilde{\mu}_z^{\mathcal{A}}(\bar{q}_X) = \bar{q}_{Z^{\phi}}$
  - signature preservation is ensured $\forall (q_X, q_Z) \in Q_X \times Q_Z$ s.t. $q_Z = \tilde{\mu}_z^{\mathcal{A}}(q_X)$, $sig(X)(q_X) = sig(Z)(q_Z)$ by lemma 138 of signature preservation of μ_z.
- $D'_X \triangleq dom(\tilde{\mu}_{tr}^{\mathcal{A}})$ is eligible to PCA transition-matching (and thus $(\tilde{\mu}_z^{\mathcal{A}}, \tilde{\mu}_{tr}^{\mathcal{A}})$ is a strong PCA-transition-matching) since
  - matched state preservation is ensured: $\forall \eta_{(X,q_X,a)} \in D'_X$, $q_X \in dom(\tilde{\mu}_z^{\mathcal{A}})$ by construction of D'_X
  - equitable corresponding distribution is ensured: $\forall \eta_{(X,q_X,a)} \in D'_X, \forall q'' \in supp(\eta_{(X,q_X,a)})$, $\eta_{(X,q_X,a)}(q'') = \eta_{(Z,\tilde{\mu}_z^{\mathcal{A}}(q_X),a)}(\tilde{\mu}_z^{\mathcal{A}}(q''))$ by lemma 129 of homomorphic transition.
- $(\tilde{\mu}_z^{\mathcal{A}}, \tilde{\mu}_{tr}^{\mathcal{A}}, \tilde{\mu}_e^{\mathcal{A}})$ is the PCA-execution-matching induced by $(\tilde{\mu}_z^{\mathcal{A}}, \tilde{\mu}_{tr}^{\mathcal{A}})$. and correctly verifies:
  - For each state q in an execution in *execs-without-creation*$(X)(\mathcal{A})$, $q \in dom(\tilde{\mu}_z^{\mathcal{A}})$.

Then, the triplet $(\tilde{\mu}_z^{\mathcal{A}}, \tilde{\mu}_{tr}^{\mathcal{A}}, \tilde{\mu}_e^{\mathcal{A}})$ is a strong PCA-execution-matching from X to Z if $\mathcal{A} \in auts(config(X_{\mathcal{A}})(start(X_{\mathcal{A}})))$: $\tilde{\mu}_z^{\mathcal{A}}(\bar{q}_X) = \bar{q}_Z$ and from X to Z^{ϕ} otherwise.

◀

extension and continuation of $(\tilde{\mu}_z^{\mathcal{A}}, \tilde{\mu}_{tr}^{\mathcal{A}}, \tilde{\mu}_e^{\mathcal{A}})$

Now, we continue the executions-matching $(\tilde{\mu}_z^{\mathcal{A}}, \tilde{\mu}_{tr}^{\mathcal{A}}, \tilde{\mu}_e^{\mathcal{A}})$ to deal with $\mathcal{A}$ creation at very last action.

▶ **Definition 141** (Preparing continuation of PCA executions-matching from X to Z)**.** *Let $\mathcal{A}$ be a PSIOA. Let X be a $\mathcal{A}$-conservative PCA. We define*

- $execs\text{-}with\text{-}only\text{-}one\text{-}creation\text{-}at\text{-}last\text{-}action(X)(\mathcal{A}) = \{\alpha' = \alpha^\frown q, a, q' \in Execs(X) | \alpha \in execs\text{-}without\text{-}creation(X)(\mathcal{A}) \wedge \alpha' \notin execs\text{-}without\text{-}creation(X)(\mathcal{A})\}$.
- $\tilde{\mu}_z^{\mathcal{A},+} : q_X \in reachable\text{-}by(execs\text{-}with\text{-}only\text{-}one\text{-}creation\text{-}at\text{-}last\text{-}action(X)(\mathcal{A})) \mapsto (\tilde{\mu}_s^{\mathcal{A}}(q_{Y_\mathcal{A}}), q_{\mathcal{A}}^{\phi})$.
- $\tilde{\mu}_{tr}^{\mathcal{A},+} : (q_X, a, \eta_{(X,q_X,a)}) \in dom(\tilde{\mu}_{tr}^{\mathcal{A}}) \cup D''_X \mapsto (\tilde{\mu}_z^{\mathcal{A}}(q_X), a, \eta_{(X,\tilde{\mu}_z^{\mathcal{A}}(q_X),a)})$ *where* $D''_X = \{(q_X, a, \eta_{(X,q_X,a)}) \in D_X | q_X \in reachable\text{-}by(execs\text{-}without\text{-}creation\text{-}at\text{-}last\text{-}action(X)(\mathcal{A})) \wedge \mathcal{A} \notin auts(config(X)(q_X)) \wedge \mathcal{A} \in created(X)(q_X)(a)\}$

We show that $dom(\tilde{\mu}_{tr}^{\mathcal{A},+}) \setminus dom(\tilde{\mu}_{tr}^{\mathcal{A}})$ verifies the equitable corresponding property of definition 81.

▶ **Lemma 142** (Continuation of PCA transitions-matching from X to Z)**.** *Let $\mathcal{A}$ be a PSIOA. Let X be a $\mathcal{A}$-conservative PCA. Let $Y = X \setminus \{\mathcal{A}\}$ and $Z = Y||\tilde{\mathcal{A}}^{sw}$.*

$\forall (q_X, a, \eta_{(X,q_X,a)}) \in dom(\tilde{\mu}_{tr}^{\mathcal{A},+}) \setminus dom(\tilde{\mu}_{tr}^{\mathcal{A}})$, $\forall q'_X \in supp(\eta_{(X,q_X,a)})$, $\eta_{(X,q_X,a)}(q'_X) = \eta_{(Z,\tilde{\mu}_z^{\mathcal{A}}(q_X),a)}(\tilde{\mu}_z^{\mathcal{A},+}(q'_X))$

Proof. By configuration preservation, $Conf = config(X)(q_X) = config(Z)(\tilde{\mu}_z^{\mathcal{A}}(q_X))$. We have $Conf \stackrel{a}{\rightharpoonup} \eta_{(Conf,a),p}$. Moreover, by μ_s-correspondence rule, $\varphi_X \setminus \{\mathcal{A}\} = \varphi_Z$, with $\varphi_X = created(X)(q_X)(a)$ and $\varphi_Z = created(Z)(\tilde{\mu}_z^{\mathcal{A}}(q_X))(a)$.

Hence $Conf \stackrel{a}{\Longrightarrow}_{\varphi_X} \eta'_X$ with η'_X generated by φ_X and $\eta_{(Conf,a),p}$, while $Conf \stackrel{a}{\Longrightarrow}_{\varphi_Z} \eta'_Z$ with η'_Z generated by φ_Z and $\eta_{(Conf,a),p}$.

Since $\mathcal{A}$ is created, for every $Conf'_Z = (\mathbf{A}'_Z, \mathbf{S}'_Z)$ with $\mathcal{A} \notin \mathbf{A}_Z$, for every $Conf'_X = (\mathbf{A}'_X, \mathbf{S}'_X)$ with $\mathbf{A}'_X = \mathbf{A}'_Z \cup \{\mathcal{A}\}$ where $\mathbf{S}'_X(\mathcal{A}) = \bar{q}_{\mathcal{A}}$ and $\mathbf{S}'_X$ agrees with $\mathbf{S}'_Z$ on $\mathbf{A}'_Z$, $\eta'_Z(Conf'_Z) = \eta'_X(Conf'_X)$, while $\eta'_X(Conf''_X) = 0$ for every $Conf''_X = (\mathbf{A}''_X, \mathbf{S}''_X)$ s. t either $\mathcal{A} \notin \mathbf{A}''_X$ or $\mathcal{A} \in \mathbf{A}''_X$ but $\mathbf{S}''_X(\mathcal{A}) \neq \bar{q}_{\mathcal{A}}$. So $\eta_{(Z,\tilde{\mu}_z^{\mathcal{A}}(q_X),a)}(\tilde{\mu}_z^{\mathcal{A},+}(q'_X)) = \eta'_Z(config(Z)(\tilde{\mu}_z^{\mathcal{A},+}(q'_X))) = \eta'_X((config(X)(q'_X))) = \eta_{(X,q_X,a)}(q'_X)$ which ends the proof.

◀

Since $dom(\tilde{\mu}_{tr}^{\mathcal{A},+}) \setminus dom(\tilde{\mu}_{tr}^{\mathcal{A}})$ verifies the equitable corresponding property of definition 81, we can define a continuation of $(\tilde{\mu}_z^{\mathcal{A}}, \tilde{\mu}_{tr}^{\mathcal{A}}, \tilde{\mu}_e^{\mathcal{A}})$ that deal with $\mathcal{A}$-creation at very last action.

▶ **Definition 143** (Continuation of PCA executions-matching from X to Z)**.** *Let $\mathcal{A}$ be a PSIOA. Let X be a $\mathcal{A}$-conservative PCA. Let $Y = X \setminus \{\mathcal{A}\}$ and $Z = Y||\tilde{\mathcal{A}}^{sw}$. Let $D''_X = dom(\tilde{\mu}_z^{\mathcal{A},+}) \setminus dom(\tilde{\mu}_z^{\mathcal{A}})$. Since $\forall (q_X, a, \eta_{(X,q_X,a)}) \in D''_X$, $\forall q'_X \in supp(\eta_{(X,q_X,a)})$, $\eta_{(X,q_X,a)}(q'_X) = \eta_{Z,\tilde{\mu}_z^{\mathcal{A}}(q_X),a)}(\tilde{\mu}_z^{\mathcal{A},+}(q'_X))$ by previous lemma 142, we can define:*

$((\tilde{\mu}_z^{\mathcal{A}}, \tilde{\mu}_z^{\mathcal{A},+}), \tilde{\mu}_{tr}^{\mathcal{A},+}, \tilde{\mu}_e^{\mathcal{A},+}))$ the $(\tilde{\mu}_z^{\mathcal{A},+}, D''_X)$-continuation of $(\tilde{\mu}_z^{\mathcal{A}}, \tilde{\mu}_{tr}^{\mathcal{A}}, \tilde{\mu}_e^{\mathcal{A}})$.

We terminate this subsection by showing the $\mathcal{E}$-extension of our continued PCA executions-matching is always well-defined.

▶ **Theorem 144** (extension of continued executions-matching after reconstruction)**.** *Let $\mathcal{A}$ be a PSIOA. Let X be a $\mathcal{A}$-conservative PCA. Let $Y = X \setminus \{\mathcal{A}\}$ and $Z = Y||\tilde{\mathcal{A}}^{sw}$. Let $\tilde{\mathcal{E}}$ partially-compatible with both X and Z. The $\tilde{\mathcal{E}}$-extension of $((X.\tilde{\mu}_z^{\mathcal{A}}, X.\tilde{\mu}_z^{\mathcal{A},+}), X.\tilde{\mu}_{tr}^{\mathcal{A}}, X.\tilde{\mu}_e^{\mathcal{A}})$, noted $(((\tilde{\mathcal{E}}||X).\tilde{\mu}_z^{\mathcal{A}}, (\tilde{\mathcal{E}}||X).\tilde{\mu}_z^{\mathcal{A},+}), (\tilde{\mathcal{E}}||X).\tilde{\mu}_{tr}^{\mathcal{A}}, (\tilde{\mathcal{E}}||X).\tilde{\mu}_e^{\mathcal{A}})$, is a strong continued PCA executions-matching from $\tilde{\mathcal{E}}||X$ to $\tilde{\mathcal{E}}||Z$.*

Proof. By definition of $\tilde{\mu}_z^{\mathcal{A},+}$ and $\tilde{\mu}_z^{\mathcal{A}}$, we have

- $\tilde{E}_{\tilde{\mathcal{E}}||X} = execs\text{-}without\text{-}creation(\tilde{\mathcal{E}}||X)(\mathcal{A})$
- $\tilde{E}^{+}_{\tilde{\mathcal{E}}||X} = execs\text{-}with\text{-}only\text{-}one\text{-}creation\text{-}at\text{-}last\text{-}action(\tilde{\mathcal{E}}||X)(\mathcal{A})$
- $\tilde{E}_{X} = execs\text{-}without\text{-}creation(X)(\mathcal{A})$
- $\tilde{E}^{+}_{X} = execs\text{-}with\text{-}only\text{-}one\text{-}creation\text{-}at\text{-}last\text{-}action(X)(\mathcal{A})$
- $\tilde{Q}_{\tilde{\mathcal{E}}||X} = reachable\text{-}by(\tilde{E}_{\tilde{\mathcal{E}}||X})$
- $\tilde{Q}^{+}_{\tilde{\mathcal{E}}||X} = reachable\text{-}by(\tilde{E}^{+}_{\tilde{\mathcal{E}}||X})$
- $\tilde{Q}_{X} = reachable\text{-}by(\tilde{E}_{X})$
- $\tilde{Q}^{+}_{X} = reachable\text{-}by(\tilde{E}^{+}_{X})$
- $dom((\tilde{\mathcal{E}}||X).\tilde{\mu}^{\mathcal{A},+}_{z}) = \tilde{Q}^{+}_{\tilde{\mathcal{E}}||X}$
- $dom((\tilde{\mathcal{E}}||X).\tilde{\mu}^{\mathcal{A}}_{z}) = \tilde{Q}_{\tilde{\mathcal{E}}||X}$
- $dom(X.\tilde{\mu}^{\mathcal{A},+}_{z}) = \tilde{Q}^{+}_{X}$
- $dom(X.\tilde{\mu}^{\mathcal{A}}_{z}) = \tilde{Q}_{X}$

This allow us to apply lemma 91 of "sufficient conditions to obtain range inclusion" to both $(\tilde{\mathcal{E}}||X).\tilde{\mu}^{\mathcal{A},+}_{z}$ and $(\tilde{\mathcal{E}}||X).\tilde{\mu}^{\mathcal{A}}_{z}$ which gives $range((\tilde{\mathcal{E}}||X).\tilde{\mu}^{\mathcal{A},+}_{z}) \subseteq Q_{\tilde{\mathcal{E}}||Z}$ and $range((\tilde{\mathcal{E}}||X).\tilde{\mu}^{\mathcal{A}}_{z}) \subseteq Q_{\tilde{\mathcal{E}}||Z}$ which allows us to apply lemma 98.

The lemma 108 implies that the resulting executions-matching is a strong one.

◀

12.4 Composition and projection are commutative

This section aims to show in theorem 145 that operation of projection/deprivation and composition are commutative.

▶ **Theorem 145** ($(X||\mathcal{E}) \setminus \{\mathcal{A}\}$ and $(X \setminus \{\mathcal{A}\})||\mathcal{E}$ are semantically equivalent)**.** *Let $\mathcal{A}$ be a PSIOA. Let X be a $\mathcal{A}$-fair PCA partially-compatible with $\mathcal{E}$ that never counts $\mathcal{A}$ in its constitution with both $X, \mathcal{E}$ and $X||\mathcal{E}$ configuration-conflict-free. The PCA $(X||\mathcal{E}) \setminus \{\mathcal{A}\}$ and $(X \setminus \{\mathcal{A}\})||\mathcal{E}$ are semantically equivalent.*

Proof. We note $W = X||\mathcal{E}$, $U = (X||\mathcal{E}) \setminus \{\mathcal{A}\}$, $V = (X \setminus \{\mathcal{A}\})||\mathcal{E}$, $\mu^{X,\mathcal{A}}_{s} = X.\mu^{\mathcal{A}}_{s}$, $\mu^{W,\mathcal{A}}_{s} = W.\mu^{\mathcal{A}}_{s}$. To stay simple, we note Id the identity function on any domain, that is we note Id for both $Id_{\mathcal{E}} : q_{\mathcal{E}} \in Q_{\mathcal{E}} \mapsto q_{\mathcal{E}}$ and $Id_{U} : q_{U} \in Q_{U} \mapsto q_{U}$.

The plan of the proof is the following one:

- We will construct two functions, $iso_{UV} : Q_{U} \to Q_{V}$ and $iso_{VU} : Q_{V} \to Q_{U}$, s.t. $iso_{UV}(q_{U})$ is the unique element of $(\mu^{X,\mathcal{A}}_{s}, Id)((\mu^{W,\mathcal{A}}_{s})^{-1}(q_{U}))$ and $iso_{VU}((q_{Y}, q_{\mathcal{E}}))$ is the unique element of $\mu^{W,\mathcal{A}}_{s}((\mu^{X,\mathcal{A}}_{s}, Id)^{-1}((q_{Y}, q_{\mathcal{E}})))$.
- Then we will show that iso_{UV} and iso_{VU} are two bijections s.t. $iso_{VU} = iso^{-1}_{UV}$.
- Thereafter we will show that for every $(q_{U}, q_{V}), (q'_{U}, q'_{V}) \in (states(U) \times Q_{V})$, s.t. $q_{V} = iso_{UV}(q_{U})$ and $q'_{V} = iso_{UV}(q'_{U})$, then $q_{U} R_{strict} q_{V}$, $q'_{U} R_{strict} q'_{V}$ and for every $a \in \widehat{sig}(U)(q_{U}) = \widehat{sig}(V)(q_{V})$, $\eta_{(U,q_{U},a)}(q'_{U}) = \eta_{(V,q_{V},a)}(q'_{V})$.
- Finally, it will allow us to construct a strong complete bijective execution-matching induced by iso_{UV} and D_{U} (the set of discrete transitions of U) in bijection with a strong complete bijective execution-matching induced by iso_{VU} and D_{V} (the set of discrete transitions of V) .

First, we show that for every $q_{W} = (q_{X}, q_{\mathcal{E}}) \in reachable(W) \subset Q_{X} \times Q_{\mathcal{E}}$, the state $q_{V} \triangleq (\mu^{X,\mathcal{A}}_{s}, Id)(q_{W}) = (\mu^{X,\mathcal{A}}_{s}(q_{X}), q_{\mathcal{E}})$ is an element of $reachable(V)$ (*). We proceed by induction. Basis: $(\mu^{X,\mathcal{A}}_{s}(\bar{q}_{X}), \bar{q}_{\mathcal{E}})$ is the initial state of V. Induction: Let $q_{W} \triangleq (q_{X}, q_{\mathcal{E}}), q'_{W} \triangleq$

$(q'_X, q'_\mathcal{E}) \in reachable(W), q_V \in reachable(V), a \in \widehat{sig}(W)(q_W)$ s.t. $q'_W \in supp(\eta_{(W,q_W,a)})$, $q_V = (\mu_s^{X,\mathcal{A}}, Id)(q_W)$, and $q'_V = (\mu_s^{X,\mathcal{A}}, Id)(q'_W)$. There is two cases:

case 1) a is $\mathcal{A}$-exclusive in q_W. In this case $q_W R^{\setminus\{\mathcal{A}\}} q'_W$, which means $q'_V = q_V$ and ends the proof

case 2) $a \in \widehat{sig}(V)(q_V) \cap \widehat{sig}(W)(q_W)$

We need to show that $q'_V \in supp(\eta_{(V,q_V,a)})$. This is easy to show. Indeed, $q'_W \in supp(\eta_{(W,q_W,a)})$ means $(q'_X, q'_\mathcal{E}) \in supp(\eta_{(X,q_X,a)} \otimes \eta_{(\mathcal{E},q_\mathcal{E},a)})$ (with the convention $\eta_{(X,q_X,a)} = \delta_{q_X}$ if $a \notin \widehat{sig}(X)(q_X)$) and $\eta_{(\mathcal{E},q_\mathcal{E},a)} = \delta_{q_\mathcal{E}}$ if $a \notin \widehat{sig}(\mathcal{E})(q_\mathcal{E})$)) which means $q'_X \in supp(\eta_{(X,q_X,a)})$ and $q'_\mathcal{E} \in supp(\eta_{(\mathcal{E},q_\mathcal{E},a)})$. So $\mu_s^{X,\mathcal{A}}(q'_X) \in supp(\eta_{(Y,\mu_s^{X,\mathcal{A}}(q_X),a)})$ which means $(\mu_s^{X,\mathcal{A}}(q'_X), q'_\mathcal{E}) \in supp(\eta_{(Y,\mu_s^{X,\mathcal{A}}(q_X),a)} \otimes \eta_{(\mathcal{E},q_\mathcal{E},a)})$, that is $(\mu_s^{X,\mathcal{A}}(q'_X), q'_\mathcal{E}) \in supp(\eta_{((Y,\mathcal{E}),(\mu_s^{X,\mathcal{A}}(q_X),q_\mathcal{E}),a)}) \eta_{(\mathcal{E},q_\mathcal{E},a)})$ and thus $q'_V \in supp(\eta_{(V,q_V,a)})$ so $q'_V \in reachable(V)$.

Second, we show that for every $q_V \triangleq (q_Y, q_\mathcal{E}) \in reachable(V)$, $\exists\ q_W \triangleq (q_X, q_\mathcal{E}) \in reachable(W)$ s.t. $q_V = (\mu_s^{X,\mathcal{A}}, Id)(q_W)$ (**). The reasoning is the same, we proceed by induction. The basis is performed with start state correspondance as before. Induction: Let $q_V \triangleq (q_Y, q_\mathcal{E}), q'_V \triangleq (q'_Y, q'_\mathcal{E}) \in reachable(V), q_W \in reachable(W), a \in \widehat{sig}(V)(q_V) \cap \widehat{sig}(W)(q_W)$ s.t. $q'_V \in supp(\eta_{(V,q_V,a)})$ with $q_V = (\mu_s^{X,\mathcal{A}}, Id)(q_W)$.

We need to show that $\exists\ q'_W \in supp(\eta_{(W,q_W,a)})$ s.t. $q'_V = (\mu_s^{X,\mathcal{A}}, Id)(q'_W)$. This is easy to show because of $\mu_d^{X,\mathcal{A}}$-correspondance. For every $q'_V \triangleq (q'_Y, q_\mathcal{E}) \in supp(\eta_{(V,(q_Y,q_\mathcal{E}),a)})$, $q'_Y \in supp(\eta_{(Y,q_Y,a)})$. Because of $\mu_d^{X,\mathcal{A}}$-correspondance, $\exists\ q'_X \in supp(\eta_{(X,q_X,a)})$ with $q'_Y = \mu_s^{X,\mathcal{A}}(q'_X)$, thus $\exists\ q'_W = (q'_X, q'_\mathcal{E}) \in supp(\eta_{(W,(q_X,q_\mathcal{E}),a)})$ s.t. $q'_V = (\mu_s^{X,\mathcal{A}}(q'_X), q'_\mathcal{E})$ which ends the proof of this second point.

Now we can construct iso_{UV} and iso_{VU}.

- iso_{UV}: for every $q_U \in Q_U$, $(\mu_s^{W,\mathcal{A}})^{-1}(q_U) \neq \emptyset$ by construction of U and for every $q_W \triangleq (q_X, q_\mathcal{E}), q'_W \triangleq (q'_X, q'_\mathcal{E}) \in (\mu_s^{W,\mathcal{A}})^{-1}(q_U)$, $q_W R_{strict}^{\setminus\{\mathcal{A}\}} q'_W$
  [...],
  which means for every $q_W \triangleq (q_X, q_\mathcal{E}), q'_W \triangleq (q'_X, q'_\mathcal{E}) \in (\mu_s^{W,\mathcal{A}})^{-1}(q_U)$, $(\mu_s^{X,\mathcal{A}}, Id)((q_X, q_\mathcal{E})) = (\mu_s^{X,\mathcal{A}}, Id)((q'_X, q'_\mathcal{E}))$ and so $(\mu_s^{X,\mathcal{A}}, Id)((\mu_s^{W,\mathcal{A}})^{-1}(q_U)) = \{q_V\}$ where $q_V \triangleq iso_{UV}(q_U) \in Q_V$ by (*).
- iso_{VU}: for every $q_V \triangleq (q_Y, q_\mathcal{E}) \in Q_V$, $(\mu_s^{X,\mathcal{A}}, Id)^{-1}(q_V) \neq \emptyset$ by (**). Furthermore for every $q_W \triangleq (q_X, q_\mathcal{E}), q'_W \triangleq (q'_X, q_\mathcal{E}) \in (\mu_s^{X,\mathcal{A}}, Id)^{-1}(q_V)$, $q_X R_{strict}^{\setminus\{\mathcal{A}\}} q'_X$, which means $q_W R_{strict}^{\setminus\{\mathcal{A}\}} q'_W$ and so $\mu_s^{W,\mathcal{A}}((\mu_s^{X,\mathcal{A}}, Id)^{-1}(q_V)) = \{q_U\}$ where $q_U \triangleq iso_{VU}(q_V) \in Q_U$

Now we can show that iso_{UV} is a bijection with $iso_{VU} = iso_{VU}^{-1}$.

- surjectivity of iso_{UV}: Let $q_V = (q_Y, q_\mathcal{E}) \in reachable(V)$, we will show that $\exists\ q_U \in reachable(U)$ s.t. $iso_{UV}(q_U) = q_V$. Indeed, we already know that (*) $\exists\ q_W = (q_X, q_\mathcal{E}) \in (\mu_s^{X,\mathcal{A}}, Id)^{-1}(q_V) \cap reachable(W)$. Let $q_U = \mu_s^{W,\mathcal{A}}(q_W)$. By construction of U, we have $q_U \in reachable(U)$ and $q_W \in (\mu_s^{W,\mathcal{A}})^{-1}(q_U)$ and $(\mu_s^{X,\mathcal{A}}, Id)(q_W) = q_V$ which means $iso_{UV}(q_U) = q_V$ and ends this item.
- injectivity of iso_{UV}: Let $q_V \in reachable(V)$, Let $q_U, q'_U \in reachable(U)$ s.t. $iso_{UV}(q_U) = iso_{UV}(q'_U)$ then $q_U = q'_U$. Again for every $q_W, q'_W \in (\mu_s^{X,\mathcal{A}}, Id)^{-1}(q_V)$, $q_W R_{strict}^{\setminus\mathcal{A}} q'_W$ and so $\mu_s^{W,\mathcal{A}}(q_W) = \mu_s^{W,\mathcal{A}}(q'_W)$. But for every $q_U, q'_U \in iso_{UV}^{-1}(q_V)$, $q_U, q'_U \in \mu_s^{W,\mathcal{A}}(\mu_s^{X,\mathcal{A}}, Id)^{-1}(q_V)$ which means $q_U = q'_U$.

Let (i) $q_V = iso_{UV}(q_U)$ or (ii) $q_U = iso_{UV}(q_V)$ we will show that in both (i) and (ii) $q_V R_{strict} q_U$. By definition, $\{q_V\} = (\mu_s^{X,\mathcal{A}}, Id)(\mu_s^{W,\mathcal{A}})^{-1}(q_U))$.

In case (i) we note q_W an arbitrary element of $(\mu_s^{W,\mathcal{A}})^{-1}(q_U) \neq \emptyset$, while in case (ii) we note q_W an arbitrary element of $(\mu_s^{X,\mathcal{A}}, Id)^{-1}(q_V) \neq \emptyset$. In both cases, we have 1a) $config(W)(q_W) \setminus \{\mathcal{A}\} = config(U)(q_U)$ and 1b) $config(W)(q_W) \setminus \{\mathcal{A}\} = config(V)(q_V)$, which means 1c) $config(U)(q_U) = config(V)(q_V)$. Then we have 2a) $hidden\text{-}actions(W)(q_W) \setminus pot\text{-}out(W)(q_W)(\mathcal{A}) = hidden\text{-}actions(U)(q_U) \setminus pot\text{-}out(W)(q_W)(\mathcal{A}) = hidden\text{-}actions(U)(q_U)$ and 2b) $hidden\text{-}actions(W)(q_W) \setminus pot\text{-}out(W)(q_W)(\mathcal{A}) = hidden\text{-}actions(V)(q_V) \setminus pot\text{-}out(W)(q_W)(\mathcal{A}) = hidden\text{-}actions(V)(q_V)$, which means 2c) $hidden\text{-}actions(U)(q_U) = hidden\text{-}actions(V)(q_V)$. Thereafter we have 3a) for every action $a \in \widehat{sig}(W)(q_W) \cap \widehat{sig}(U)(q_U), created(W)(q_W)(a) \setminus \{\mathcal{A}\} = created(U)(q_U)(a) \setminus \{\mathcal{A}\} = created(U)(q_U)(a)$ and 3b) for every action $a \in \widehat{sig}(W)(q_W) \cap \widehat{sig}(V)(q_V), created(W)(q_W)(a) \setminus \{\mathcal{A}\} = created(V)(q_V)(a) \setminus \{\mathcal{A}\} = created(V)(q_V)(a)$ which means 3c) for every action $a \in \widehat{sig}(U)(q_U) = \widehat{sig}(V)(q_V), created(U)(q_U)(a) = created(V)(q_V)(a)$. The conjonction of 3a), 3b) and 3c) lead us to $q_V R_{strict} q_U$.

Now we can show that iso_{UV} is the reverse function of iso_{VU}: Let $(q_U, q_V) \in reachable(U) \times reachable(V)$ s.t. $q_V = iso_{UV}(q_U)$. We need to show that $iso_{VU}(q_V) = q_U$. The point is that $\exists! \ q'_U \triangleq iso_{VU}(q_V)$ and we have $q_V R_{strict} q_U$ and $q_V R_{strict} q'_U$ which means $q_U R_{strict} q'_U$ and so $q_U = q'_U$ by assumption of configuration-conflict-free PCA. Hence $iso_{UV} = iso_{VU}^{-1}$.

The last point is to show that that for every $(q_U, q_V), (q'_U, q'_V) \in reachable(U) \times reachable(V)$, s.t. $q_V = iso_{UV}(q_U)$ and $q'_V = iso_{UV}(q'_U)$, then $q_U R_{strict} q_V$, $q'_U R_{strict} q'_V$ and for every $a \in \widehat{sig}(U)(q_U) = \widehat{sig}(V)(q_V)$, $\eta_{(U,q_U,a)}(q'_U) = \eta_{(V,q_V,a)}(q'_V)$.

For every $a \in \widehat{sig}(U)(q_U) = \widehat{sig}(V)(q_V)$ we have a unique η s.t. $C \overset{a}{\Longrightarrow}_{\varphi} \eta$ with $C = config(U)(q_U) = config(V)(q_V)$ and $\varphi = created(U)(q_U)(a) = created(V)(q_V)(a)$. Hence for every configuration $C' \in supp(\eta)$, $\exists! \ (q'_U, q'_V) \in reachable(U) \times reachable(V)$ s.t. $C' = config(U)(q'_U) = config(V)(q'_V)$. Hence $iso_{UV}(q'_U) = q'_V$ and furthermore $\eta_{(U,q_U,a)}(q'_U) = \eta_{(V,q_V,a)}(q'_V) = \eta(C)$.

Everything is ready to construct the PCA-execution-matching, which is (j) the PCA-execution-matching induced by iso_{UV} and D_U (the set of discrete transition of U) and (jj) the PCA-execution-matching induced by iso_{VU} and D_V (the set of discrete transition of V)

◀

13 PCA corresponding w.r.t. PSIOA $\mathcal{A}$, $\mathcal{B}$

In the previous section we have shown that $X_{\mathcal{A}}||\mathcal{E}$ and $\tilde{\mathcal{A}}^{sw}||(X_{\mathcal{A}} \setminus \{\mathcal{A}\}||\mathcal{E})$ are linked by a strong PCA executions-matching as long as $\mathcal{A}$ is not re-created by $X_{\mathcal{A}}$. This also means that the probability distribution of $X_{\mathcal{A}}||\mathcal{E}$ is preserved by $\tilde{\mathcal{A}}^{sw}||(X \setminus \{\mathcal{A}\}||\mathcal{E})$, as long as $\mathcal{A}$ is not re-created by $X_{\mathcal{A}}$. We can have the same reasoning to obtain a strong PCA executions-matching from $X_{\mathcal{B}}||\mathcal{E}$ and $\tilde{\mathcal{B}}^{sw}||(X_{\mathcal{B}} \setminus \{\mathcal{B}\}||\mathcal{E})$.

In this section we take an interest in PCA $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ that differ only on the fact that $\mathcal{B}$ supplants $\mathcal{A}$ in $X_{\mathcal{B}}$. Hence, we recall the definitions of section 9. Then, we show that under slight assumptions, $X_{\mathcal{A}} \setminus \{\mathcal{A}\}$ and $X_{\mathcal{B}} \setminus \{\mathcal{B}\}$ are semantically equivalent (see theorem 160).

Combined with the result of previous section we will realise that we can obtain a strong PCA executions-matching from (*) $X_{\mathcal{A}}||\mathcal{E}$ to $\tilde{\mathcal{A}}^{sw}||(Y||\mathcal{E})$ and (**) from $X_{\mathcal{B}}||\mathcal{E}$ to $\tilde{\mathcal{B}}^{sw}||(Y||\mathcal{E})$ where Y is semantically equivalent to both $X_{\mathcal{B}} \setminus \{\mathcal{B}\}$ and $X_{\mathcal{A}} \setminus \{\mathcal{A}\}$. Hence if $\mathcal{E}' = \mathcal{E}||Y$ cannot distinguish $\tilde{\mathcal{A}}^{sw}$ from $\tilde{\mathcal{B}}^{sw}$, we will be able to show that $\mathcal{E}$ cannot distinguish $X_{\mathcal{A}}$ from $X_{\mathcal{B}}$ which will be the subject of sections 14 to finally prove the monotonicity of p-implementation.

$\lhd_{\mathcal{A}\mathcal{B}}$-correspondence between two configurations

We formalise the idea that two configurations are the same excepting the fact that the automaton $\mathcal{B}$ supplants $\mathcal{A}$ but with the same external signature. The next definition comes from [2].

▶ **Definition 146** ($\triangleleft_{\mathcal{AB}}$-corresponding configurations)**.** *(see figure 27) Let $\Phi \subseteq Autids$, and $\mathcal{A}, \mathcal{B}$ be PSIOA identifiers. Then we define $\Phi[\mathcal{B}/\mathcal{A}] = (\Phi \setminus \mathcal{A}) \cup \{\mathcal{B}\}$ if $\mathcal{A} \in \Phi$, and $\Phi[\mathcal{B}/\mathcal{A}] = \Phi$ if $\mathcal{A} \notin \Phi$. Let C, D be configurations. We define $C \triangleleft_{\mathcal{AB}} D$ iff (1) $auts(D) = auts(C)[\mathcal{B}/\mathcal{A}]$, (2) for every $\mathcal{A}' \notin auts(C) \setminus \{\mathcal{A}\} : map(D)(\mathcal{A}') = map(C)(\mathcal{A}')$, and (3) $ext(\mathcal{A})(s) = ext(\mathcal{B})(t)$ where $s = map(C)(\mathcal{A}), t = map(D)(\mathcal{B})$. That is, in $\triangleleft_{\mathcal{AB}}$-corresponding configurations, the SIOA other than $\mathcal{A}, \mathcal{B}$ must be the same, and must be in the same state. $\mathcal{A}$ and $\mathcal{B}$ must have the same external signature. In the sequel, when we write $\Psi = \Phi[\mathcal{B}/\mathcal{A}]$, we always assume that $\mathcal{B} \notin \Phi$ and $\mathcal{A} \notin \Psi$.*

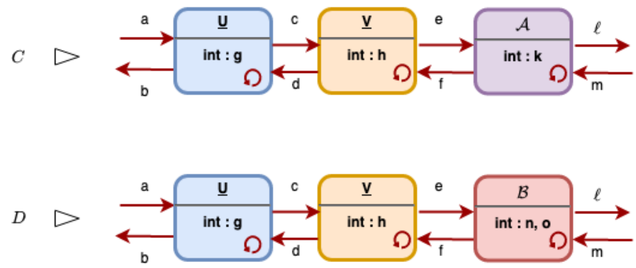


Figure 27 $\triangleleft_{\mathcal{AB}}$ corresponding-configuration

Next lemma states that $\triangleleft_{\mathcal{AB}}$-corresponding configurations have the same external signature, which is quite intuitive when we see the figure 27.

▶ **Proposition 147.** *Let C, D be configurations such that $C \triangleleft_{\mathcal{AB}} D$. Then $ext(C) = ext(D)$.*

Proof. The proof is in [2], section 6, p. 38. We write the proof here to be complete:

If $\mathcal{A} \notin C$ then $C = D$ by definition , and we are done. Now suppose that $\mathcal{A} \in C$, so that $C = (\mathbf{A} \cup \{\mathcal{A}\}, \mathbf{S})$ for some set $\mathbf{A}$ of PSIOA identifiers s.t. $\mathcal{A} \notin \mathbf{A}$, and let $s = \mathbf{S}(\mathcal{A})$. Then, by definition 16 of attributes of configuration, $out(C) = (\bigcup_{\mathcal{A}_i \in \mathbf{A}} out(\mathcal{A}_i)(\mathbf{S}(\mathcal{A}_i))) \cup out(\mathcal{A})(s)$. From $C \triangleleft_{\mathcal{AB}} D$ and definition , we have $D = (\mathbf{A} \cup \{\mathcal{B}\}, \mathbf{S}')$, where $\mathbf{S}'$ agrees with $\mathbf{S}$ on all $\mathcal{A}_i \in \mathbf{A}$, and $t = \mathbf{S}'(\mathcal{B})$ such that $ext(\mathcal{A})(s) = ext(\mathcal{B})(t)$. Hence $out(\mathcal{A})(s) = out(\mathcal{B})(t)$ and $in(\mathcal{A})(s) = in(\mathcal{B})(t)$. By definition 16 of configuration attributes, $out(D) = (\bigcup_{\mathcal{A}_i \in \mathbf{A}} out(\mathcal{A}_i)(\mathbf{S}'(\mathcal{A}_i))) \cup out(\mathcal{B})(t)$. Finally, $out(C) = out(D)$ since $\mathbf{S}'$ agrees with $\mathbf{S}$ on all $\mathcal{A} \in \mathbf{A}$ and $out(\mathcal{A})(s) = out(\mathcal{B})(t)$. We establish $in(C) = in(D)$ in the same manner, and omit the repetitive details. Hence $ext(C) = ext(D)$. ◀

▶ Remark 148. It is possible to have two configurations C, D s.t. $C \triangleleft_{\mathcal{AA}} D$. That would mean that C and D only differ on the state of $\mathcal{A}$ (s or t) that has even the same external signature in both cases $ext(\mathcal{A})(s) = ext(\mathcal{A})(t)$, while we would potentially have $int(\mathcal{A})(s) \neq int(\mathcal{A})(t)$.

The next lemma states that $\triangleleft_{\mathcal{AB}}$-corresponding configurations are equals if we omit the automata $\mathcal{A}$ and $\mathcal{B}$.

▶ **Lemma 149** (Same configuration)**.** *Let $\mathcal{A}, \mathcal{B} \in Autids$. Let $X_{\mathcal{A}}$, $X_{\mathcal{B}}$ be $\mathcal{A}$-fair and $\mathcal{B}$-fair PCA respectively, where $X_{\mathcal{A}}$ never contains $\mathcal{B}$ and $X_{\mathcal{B}}$ never contains $\mathcal{A}$. Let $Y_{\mathcal{A}} = X_{\mathcal{A}} \setminus \{\mathcal{A}\}$, $Y_{\mathcal{B}} = X_{\mathcal{B}} \setminus \{\mathcal{B}\}$. Let $(x_a, x_b) \in Q_{X_{\mathcal{A}}} \times Q_{X_{\mathcal{B}}}$ s.t. $config(X_{\mathcal{A}})(x_a) \triangleleft_{AB} config(X_{\mathcal{B}})(x_b)$. Let $y_a = X_{\mathcal{A}}.\mu_s^{\mathcal{A}}(x_a)$, $y_b = X_{\mathcal{A}}.\mu_s^{\mathcal{A}}(x_b)$*

Then $config(Y_{\mathcal{A}})(y_a) = config(Y_{\mathcal{B}})(y_b)$.

Proof. By projection, we have $config(Y_{\mathcal{A}})(y_a) \triangleleft_{AB} config(Y_{\mathcal{B}})(y_b)$ with each configuration that does not contain $\mathcal{A}$ nor $\mathcal{B}$, thus for $config(Y_{\mathcal{A}})(y_a)$ and $config(Y_{\mathcal{B}})(y_b)$ contain the same set of automata ids (rule (1) of $\triangleleft_{AB}$) and map each automaton of this set to the same state (rule (2) of $\triangleleft_{AB}$). ◀

same comportment of two PCA modulo $\mathcal{A}$, $\mathcal{B}$

In this paragraph we formalise the fact that two PCA have the same comportment, excepting for $\mathcal{B}$ that supplants $\mathcal{A}$.

First, we formalise the fact that two PCA create some PSIOA in the same manner, excepting for $\mathcal{B}$ that supplants $\mathcal{A}$. Here again, this definition comes from [2].

▶ **Definition 150** (Creation corresponding configuration automata). *Let X, Y be configuration automata and $\mathcal{A}, \mathcal{B}$ be PSIOA. We say that X, Y are creation-corresponding w.r.t. $\mathcal{A}, \mathcal{B}$ iff*
1. *X never creates $\mathcal{B}$ and Y never creates $\mathcal{A}$.*
2. *$\forall (\alpha, \pi) \in Execs^*(X) \times Execs^*(Y)$ s.t $trace_{\mathcal{A}}(\alpha) = trace_{\mathcal{B}}(\pi)$, for $x = lstate(\alpha), y = lstate(\pi)$, we have Then $\forall a \in \widehat{sig}(X)(x) \cap \widehat{sig}(Y)(y) : created(Y)(y)(a) = created(X)(x)(a)[\mathcal{B}/\mathcal{A}]$.*

Naturally $[\mathcal{B}/\mathcal{A}]$-corresponding sets of created automata are deprived of $\mathcal{A}$ and $\mathcal{B}$ respectively, they becomes equal, which is formalised in next lemma.

▶ **Lemma 151** (Same creation after projection). *Let $\mathcal{A}, \mathcal{B} \in Autids$. Let $X_{\mathcal{A}}$, $X_{\mathcal{B}}$ be $\mathcal{A}$-fair and $\mathcal{B}$-fair PCA respectively, where $X_{\mathcal{A}}$ never contains $\mathcal{B}$ and $X_{\mathcal{B}}$ never contains $\mathcal{A}$ ($\mathcal{B} \notin UA(X_{\mathcal{A}})$ and $\mathcal{A} \notin UA(X_{\mathcal{B}})$). Let $Y_{\mathcal{A}} = X_{\mathcal{A}} \setminus \{\mathcal{A}\}$, $Y_{\mathcal{B}} = X_{\mathcal{B}} \setminus \{\mathcal{B}\}$. Let $(x_a, x_b) \in Q_{X_{\mathcal{A}}} \times Q_{X_{\mathcal{B}}}$ and $act \in sig(X_{\mathcal{A}})(x_a) \cap sig(X_{\mathcal{B}})(x_b)$ s.t. $created(X_{\mathcal{B}})(x_b)(act) = created(X_{\mathcal{A}})(x_a)(act)[\mathcal{B}/\mathcal{A}]$. Let $y_a = X_{\mathcal{A}}.\mu_s^{\mathcal{A}}(x_a)$, $y_b = X_{\mathcal{B}}.\mu_s^{\mathcal{B}}(x_b)$*

Then $created(Y_{\mathcal{B}})(x_b)(act) = created(Y_{\mathcal{A}})(x_a)(act)$

Proof. By definition of PCA projection, we have $created(Y_{\mathcal{B}})(x_b)(act) = (created(X_{\mathcal{B}})(x_b)(act)) \setminus \mathcal{B} = (created(X_{\mathcal{A}})(x_a)(act)[\mathcal{B}/\mathcal{A}]) \setminus \mathcal{B} = created(X_{\mathcal{A}})(x_a)(act) \setminus \mathcal{A} = created(Y_{\mathcal{A}})(x_a)(act)$. ◀

Second, we formalise the fact that two PCA hide their actions in the same manner. The definition is strongly inspired by [2].

▶ **Definition 152** (Hiding corresponding configuration automata). *Let X, Y be configuration automata and $\mathcal{A}, \mathcal{B}$ be PSIOA. We say that X, Y are hiding-corresponding w.r.t. $\mathcal{A}, \mathcal{B}$ iff*
1. *X never creates $\mathcal{B}$ and Y never creates $\mathcal{A}$.*
2. *$\forall (\alpha, \pi) \in Execs^*(X) \times Execs^*(Y)$ s.t $trace_{\mathcal{A}}(\alpha) = trace_{\mathcal{B}}(\pi)$, for $x = lstate(\alpha), y = lstate(\pi)$, we have $hidden\text{-}actions(Y)(y) = hidden\text{-}actions(X)(x)$.*

Naturally if hidden actions of $\lhd_{\mathcal{AB}}$-corresponding states are equal, it remains true after respective deprivation of $\mathcal{A}$ and $\mathcal{B}$ which is formalised in next lemma.

▶ **Lemma 153** (Same hidden-actions after projection). *Let $\mathcal{A}, \mathcal{B} \in Autids$. Let $X_{\mathcal{A}}$, $X_{\mathcal{B}}$ be $\mathcal{A}$-fair and $\mathcal{B}$-fair PCA respectively, where $X_{\mathcal{A}}$ never contains $\mathcal{B}$ and $X_{\mathcal{B}}$ never contains $\mathcal{A}$ ($\mathcal{B} \notin UA(X_{\mathcal{A}})$ and $\mathcal{A} \notin UA(X_{\mathcal{B}})$). Let $Y_{\mathcal{A}} = X_{\mathcal{A}} \setminus \{\mathcal{A}\}$, $Y_{\mathcal{B}} = X_{\mathcal{B}} \setminus \{\mathcal{B}\}$. Let $(x_a, x_b) \in Q_{X_{\mathcal{A}}} \times Q_{X_{\mathcal{B}}}$, $y_a = X_{\mathcal{A}}.\mu_s^{\mathcal{A}}(x_a)$, $y_b = X_{\mathcal{B}}.\mu_s^{\mathcal{B}}(x_b)$ s.t.*
- *$x_a R_{conf}^{\setminus\{\mathcal{A}\}} x_b$, i.e. $y_a R_{conf} y_b$*
- *$hidden\text{-}actions(X_{\mathcal{B}})(x_b) = hidden\text{-}actions(X_{\mathcal{A}})(x_a)$*

Then $hidden\text{-}actions(Y_{\mathcal{B}})(y_b) = hidden\text{-}actions(Y_{\mathcal{A}})(y_a)$

Proof. We note $C_{X_{\mathcal{A}}} = config(X_{\mathcal{A}})(x_a)$, $C_{X_{\mathcal{B}}} = config(X_{\mathcal{B}})(x_b)$, $C_{Y_{\mathcal{A}}} = config(Y_{\mathcal{A}})(y_a)$, $C_{Y_{\mathcal{B}}} = config(Y_{\mathcal{B}})(y_b)$. By assumption, $C_{X_{\mathcal{A}}} \setminus \{\mathcal{A}\} = C_{Y_{\mathcal{A}}} = C_{Y_{\mathcal{B}}} = C_{X_{\mathcal{B}}} \setminus \{\mathcal{B}\}$.

We note $h_{X_{\mathcal{A}}} = hidden\text{-}actions(X_{\mathcal{A}})(x_a)$, $h_{X_{\mathcal{B}}} = hidden\text{-}actions(X_{\mathcal{B}})(x_b)$, $h_{Y_{\mathcal{A}}} = hidden\text{-}actions(Y_{\mathcal{A}})(y_a)$, $h_{Y_{\mathcal{B}}} = hidden\text{-}actions(Y_{\mathcal{B}})(y_b)$. By assumption, $h_{X_{\mathcal{A}}} = h_{X_{\mathcal{B}}}$, while by construction, $h_{Y_{\mathcal{A}}} = h_{X_{\mathcal{A}}} \setminus pot\text{-}out(X_{\mathcal{A}})(\mathcal{A})$ and $h_{Y_{\mathcal{B}}} = h_{X_{\mathcal{B}}} \setminus pot\text{-}out(X_{\mathcal{B}})(\mathcal{B})$.

Case 1: $pot\text{-}out(X_{\mathcal{A}})(\mathcal{A})(x_a) = pot\text{-}out(X_{\mathcal{B}})(\mathcal{B})(x_b)$, the result is immediate, Case 2: $pot\text{-}out(X_{\mathcal{A}})(\mathcal{A})(x_a) \cap h_{X_{\mathcal{A}}} = pot\text{-}out(X_{\mathcal{B}})(\mathcal{B})(x_b) \cap h_{X_{\mathcal{B}}} = \emptyset$, the result is immediate.

Case 3: Without loss of generality, we assume $\underline{act} = pot\text{-}out(X_{\mathcal{A}})(\mathcal{A})(x_a) \cap h_{X_{\mathcal{A}}} \neq \emptyset$. For every $\mathcal{C} \in auts(C_{Y_{\mathcal{B}}})$, $\mathcal{C} \in auts(C_{Y_{\mathcal{A}}})$ since $C_{Y_{\mathcal{A}}} = C_{Y_{\mathcal{B}}}$ and $\mathcal{C} \in auts(C_{X_{\mathcal{A}}})$ since $C_{Y_{\mathcal{A}}} = C_{X_{\mathcal{A}}} \setminus \{\mathcal{A}\}$. By compatibility of $C_{X_{\mathcal{A}}}$, $pot\text{-}out(X_{\mathcal{A}})(\mathcal{A})(x_a) \cap pot\text{-}out(X_{\mathcal{A}})(\mathcal{C})(x_a) = \emptyset$.

Case 3a) $\mathcal{B} \notin auts(C_{X_{\mathcal{B}}})$, which means both i) $\underline{act} \subset h_{X_{\mathcal{B}}}$, ii) $\underline{act} \cap out(C_{X_{\mathcal{B}}}) = \emptyset$ and iii) $h_{X_{\mathcal{B}}} \subset out(C_{X_{\mathcal{B}}})$ which is impossible. Thus we only consider

Case 3b) $\mathcal{B} \in auts(C_{X_{\mathcal{B}}})$. Since j) for every $\mathcal{C} \in auts(C_{Y_{\mathcal{B}}})$, $pot\text{-}out(X_{\mathcal{A}})(\mathcal{A})(x_a) \cap pot\text{-}out(X_{\mathcal{A}})(\mathcal{C})(x_a) = \emptyset$ and jj) $h_{X_{\mathcal{B}}} \subset out(C_{X_{\mathcal{B}}})$, we have $\underline{act} \subset pot\text{-}out(X_{\mathcal{B}})(\mathcal{B})(x_b)$.

For symmetrical reason, we have both $pot\text{-}out(X_{\mathcal{A}})(\mathcal{A})(x_a) \cap h_{X_{\mathcal{A}}} \subset pot\text{-}out(X_{\mathcal{B}})(\mathcal{B})(x_b)$ and $pot\text{-}out(X_{\mathcal{B}})(\mathcal{B})(x_b) \cap h_{X_{\mathcal{B}}} \subset pot\text{-}out(X_{\mathcal{A}})(\mathcal{A})(x_A)$, which means $h_{X_{\mathcal{A}}} \setminus pot\text{-}out(X_{\mathcal{B}})(\mathcal{B})(x_b) = h_{X_{\mathcal{B}}} \setminus pot\text{-}out(X_{\mathcal{B}})(\mathcal{B})(x_b)$ and ends the proof

◀

Now we are ready to define corresponding PCA w.r.t. PSIOA $\mathcal{A}$, $\mathcal{B}$, that is two PCA $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ that differ only on the fact that B supplants $\mathcal{A}$ in $X_{\mathcal{B}}$. Some additional assumptions are added to ensure monotonicity later. This definition is still inspired by definitions of [2].

▶ **Definition 154** (corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$). *Let $\mathcal{A}, \mathcal{B} \in Autids$, $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ be PCA we say that $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ are corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$, if they verify:*

- $config(X_{\mathcal{A}})(\bar{q}_{X_{\mathcal{A}}}) \lhd_{AB} config(X_{\mathcal{B}})(\bar{q}_{X_{\mathcal{B}}})$.
- *$X_{\mathcal{A}}$ never contains $\mathcal{B}$ ($\mathcal{B} \notin UA(X_{\mathcal{A}})$), while $X_{\mathcal{B}}$ never contains $\mathcal{A}$ ($\mathcal{A} \notin UA(X_{\mathcal{B}})$).*
- *$X_{\mathcal{A}}, X_{\mathcal{B}}$ are creation-corresponding w.r.t. $\mathcal{A}, \mathcal{B}$.*
- *$X_{\mathcal{A}}, X_{\mathcal{B}}$ are hiding-corresponding w.r.t. $\mathcal{A}, \mathcal{B}$.*
- *$X_{\mathcal{A}}$ (resp. $X_{\mathcal{B}}$) is a $\mathcal{A}$-conservative (resp. $\mathcal{B}$-conservative) PCA.*
- *(No exclusive creation from $\mathcal{A}$ and $\mathcal{B}$)*
  - *$\forall q_{X_{\mathcal{A}}} \in Q_{X_{\mathcal{A}}}$, for every action act $\mathcal{A}$-exclusive, $created(X_{\mathcal{A}})(q_{X_{\mathcal{A}}})(act) = \emptyset$ and similarly*
  - *$\forall q_{X_{\mathcal{B}}} \in Q_{X_{\mathcal{B}}}$, for every action act' $\mathcal{B}$-exclusive, $created(X_{\mathcal{B}})(q_{X_{\mathcal{B}}})(act') = \emptyset$*

equivalent transitions to obtain semantic equivalence after projection

In this last paragraph of the section, we show that if two PCA $X_{\mathcal{A}}$ $X_{\mathcal{B}}$ are corresponding w.r.t. $\mathcal{A}$ and $\mathcal{B}$, then there respective projection $Y_{\mathcal{A}} = X_{\mathcal{A}} \setminus \{\mathcal{A}\}$ and $Y_{\mathcal{B}} = X_{\mathcal{B}} \setminus \{\mathcal{B}\}$ are semantically equivalents. To do so, we use notions of equivalent transitions. the idea is to recursively show that any corresponding executions of $Y_{\mathcal{A}}$ and $Y_{\mathcal{B}}$ lead to strictly equivalent transitions to finally build the complete bijective PCA executions-matching from $Y_{\mathcal{A}}$ to $Y_{\mathcal{B}}$.

We start by defining equivalent transitions.

▶ **Definition 155** (configuration-equivalence and strict-equivalence between two distributions). *Let K, K' be PCA and $(\eta, \eta') \in Disc(states(K)) \times Disc(states(K'))$.*

- *We say that η and η' are* config-equivalent*, noted $\eta \overset{f}{\underset{conf}{\longleftrightarrow}} \eta'$, if there exists $f : Q_K \longrightarrow Q_{K'}$ s.t. $\eta \overset{f}{\longleftrightarrow} \eta'$ with $\forall q'' \in supp(\eta)$, $q'' R_{conf} f(q'')$.*
- *If additionally, $\forall q'' \in supp(\eta)$, $q'' R_{strict} f(q'')$, then we say that η and η' are* strictly-equivalent*, noted $\eta \overset{f}{\underset{strict}{\longleftrightarrow}} \eta'$.*

Basically, equivalent transitions are transitions where the states with non-zero probability to be reached are mapped by a bijective function that preserves i) measure of probability and ii) configuration. A stricter version preserves also iii) future created automata and hidden-actions.

The next lemma states that if we take two corresponding transitions from strict equivalent states, then we obtain configuration equivalent transitions.

▶ **Lemma 156.** *(strictly-equivalent states implies config-equivalent transition) Let K, K' be PCA and $(q, q') \in Q_K \times Q_{K'}$ strictly-equivalent, i.e. $qR_{stricit}q'$. Let $a \in \widehat{sig}(K)(q) = \widehat{sig}(K')(q')$ and $((q, a, \eta_{(K,q,a)}), (q', a, \eta_{(K',q',a)})) \in D_K \times D_K$. Then $\eta_{(K,q,a)}$ and $\eta_{(K',q',a)}$ are config-equivalent, i.e. $\exists f : Q_K \to Q_{K'}$ s.t. $\eta \xleftrightarrow[conf]{f} \eta'$.*

Proof. This is the direct consequence of constraint 2 and 3 of definition 19 of PCA. We note $C = config(K)(q) = config(K')(q')$ and $\varphi = created(K)(q)(a) = created(K')(q')(a)$. By constraint 2, applied to K, there exists η s.t. $\eta_{(K,q,a)} \xleftrightarrow{f^K} \eta$ with $f^K = config(K)$ and $config(K)(q) \xRightarrow{a}_{created(K)(q)(a)} \eta$ By constraint 2, applied to K', there exists η' s.t. $\eta_{(K',q',a)} \xleftrightarrow{f^{K'}} \eta'$ with $f^{K'} = config(K')$ and $config(K')(q') \xRightarrow{a}_{created(K')(q')(a)} \eta'$.

Since $qR_{strict}q'$, $C \triangleq config(K)(q) = config(K')(q')$ and $\varphi \triangleq created(K)(q)(a) = created(K')(q')(a)$.

Hence $C \xRightarrow{a}_{\varphi} \eta$ and $C \xRightarrow{a}_{\varphi} \eta'$ which means $\eta = \eta'$.

So $\eta_{(K,q,a)} \xleftrightarrow{f} \eta_{(K',q',a)}$ with $\tilde{f} = (\tilde{f}^{K'})^{-1} \circ \tilde{f}^K$ where $\tilde{f}$ (resp. $\tilde{f}^{K'}$, resp. $\tilde{f}^K$) is the restriction of f (resp. $f^{K'}$, resp. f^K) on $supp(\eta_{(K,q,a)})$ (resp. $supp(\eta_{(K',q',a)})$, resp. $supp(\eta_{(K,q,a)})$).

Thus, for every $(\tilde{q}, \tilde{q}') \in supp(\eta_{(K,q,a)}) \times supp(\eta_{(K',q',a)})$ s.t. $\tilde{q}' = f(\tilde{q})$, $f^K(\tilde{q}) = f^{K'}(\tilde{q}')$, that is $config(K)(\tilde{q}) = config(K')(\tilde{q}')$, i.e. $\tilde{q}R_{conf}\tilde{q}'$.

Hence $\eta_{(K,q,a)} \xleftrightarrow[conf]{f} \eta_{(K',q',a)}$ which ends the proof.

◀

Now we start a sequence of lemma (from lemma 157 to lemma 159) to finally show in theorem 160 that if $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ are corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$ then $X_{\mathcal{A}} \setminus \{\mathcal{A}\}$ and $X_{\mathcal{B}} \setminus \{\mathcal{B}\}$ are semantically-equivalent.

The next lemma shows that we can always construct an execution $\tilde{\alpha}_X \in Execs(X)$ from an execution $\alpha_Y \in Execs(Y)$ with $Y = X \setminus \{\mathcal{A}\}$ that preserves the trace.

▶ **Lemma 157** ($Execs(X \setminus \{\mathcal{A}\})$ **can be obtained by** $Execs(X)$). *Let $\mathcal{A} \in Autids$, X a $\mathcal{A}$-fair PCA, $Y = X \setminus \{\mathcal{A}\}$.*

Let $\alpha_Y = q_Y^0, a^1, q_Y^1, ..., q_Y^n \in Execs(Y)$. Then there exists, $\tilde{\alpha}_X = \tilde{q}_X^0, a^1, \tilde{q}_X^1, ..., \tilde{q}_X^n \in Execs(X)$ s.t. $\forall i \in [0, n], q_Y^i = \mu_s^{\mathcal{A}}(\tilde{q}_X^i)$.

Proof. By induction on the size $s = |\alpha_Y^s|$ of prefix $\alpha_Y^s = q_Y^0, a^1, q_Y^1, ..., q_Y^s$.

Basis ($|\alpha_Y^s| = 0$): By definition 120, $\bar{q}_Y = X.\mu_s^{\mathcal{A}}(\bar{q}_X)$

Induction: let assume the proposition is true for prefix $\alpha_Y^s = q_Y^0, a^1, q_Y^1, ..., q_Y^s$ with $s < |\alpha_Y|$. We will show it is true for α_Y^{s+1}. We have $q_Y^s = X.\mu_s^{\mathcal{A}}(q_X^s)$. By construction of D_Y provided by definition 120, there exists $\eta_{(X,q_X^s,a^{s+1})} \in D_X$ s.t. $X.\mu_d^{\mathcal{A}}(\eta_{(X,q_X^s,a^{s+1})}) = \eta_{(Y,q_Y^s,a^{s+1})}$. By $X.\mu_d^{\mathcal{A}}$-correspondence of definition 120, $\eta_{(Y,q_Y^s,a^{s+1})}(q_Y^{s+1}) = \sum_{q'_X \in Q_X, \mu_s(q'_X) = q_Y^{s+1}} \eta_{(X,q_X^s,a^{s+1})}(q'_X)$. By definition of an execution, $q_Y^{s+1} \in supp(\eta_{(Y,q_Y^s,a^{s+1})})$, which means there exists $q_X^{s+1} \in Q_X$ s.t. 1) $\mu_s^{\mathcal{A}}(q_X^{s+1}) = q_Y^{s+1}$ and 2) $q_X^{s+1} \in supp(\eta_{(X,q_X^s,a^{s+1})})$. Thus, it exist $\tilde{\alpha}_X^{s+1} = \tilde{q}_X^0, a^1, \tilde{q}_X^1, ..., \tilde{q}_X^{s+1} \in Execs(X)$ s.t. $\forall i \in [0, s+1], q_Y^i = \mu_s^{\mathcal{A}}(\tilde{q}_X^i)$, which ends the induction and so the proof. ◀

The next lemma states that, after projection, two configuration-equivalent states obtain via executions with the same trace are strictly equivalent.

▶ **Lemma 158** (After projection, configuration-equivalence obtain after same trace implies strict equivalence)**.** *Let $X_\mathcal{A}$ and $X_\mathcal{B}$ be two PCA corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$. Let $Y_\mathcal{A} = X_\mathcal{A} \setminus \{\mathcal{A}\}$ and $Y_\mathcal{B} = X_\mathcal{B} \setminus \{\mathcal{B}\}$. Let $(\alpha_{Y_\mathcal{A}}, \pi_{Y_\mathcal{B}}) \in Execs(Y_\mathcal{A}) \times Execs(Y_\mathcal{B})$ with $lstate(\alpha_{Y_\mathcal{A}}) = q_{Y_\mathcal{A}}$ and $lstate(\pi_{Y_\mathcal{B}}) = q_{Y_\mathcal{B}}$. If*

- *$q_{Y_\mathcal{A}} R_{conf} q_{Y_\mathcal{B}}$ and*
- *$trace(\alpha_{Y_\mathcal{A}}) = trace(\pi_{Y_\mathcal{B}}) = \beta$,*

then $q_{Y_\mathcal{A}} R_{strict} q_{Y_\mathcal{B}}$

Proof. By lemma 157, $\exists (\tilde{\alpha}_{X_\mathcal{A}}, \tilde{\pi}_{X_\mathcal{B}}) \in Execs(X_\mathcal{A}) \times Execs(X_\mathcal{B})$ s.t. (i) $trace(\tilde{\alpha}_{X_\mathcal{A}}) = trace(\alpha_{Y_\mathcal{A}}) = trace(\pi_{Y_\mathcal{B}}) = trace(\tilde{\pi}_{X_\mathcal{B}})$ and (ii) $q_{Y_\mathcal{A}} = X_\mathcal{A}.\mu_s^\mathcal{A}(\tilde{q}_{X_\mathcal{A}})$ and $q_{Y_\mathcal{B}} = X_\mathcal{B}.\mu_s^\mathcal{B}(\tilde{q}_{X_\mathcal{B}})$ where $\tilde{q}_{X_\mathcal{B}} = lstate(\tilde{\pi}_{X_\mathcal{B}})$ and $\tilde{q}_{X_\mathcal{A}} = lstate(\tilde{\alpha}_{X_\mathcal{A}})$.

Since $trace(\tilde{\alpha}_{X_\mathcal{A}}) = trace(\tilde{\pi}_{X_\mathcal{B}})$, we have j) $hidden\text{-}actions(X_\mathcal{A})(\tilde{q}_{X_\mathcal{A}}) = hidden\text{-}actions(X_\mathcal{B})(\tilde{q}_{X_\mathcal{B}})$ by hiding-correspondence of definition 56 and jj) $\forall a \in \widehat{sig}(X_\mathcal{A})(\tilde{q}_{X_\mathcal{A}}) \cap \widehat{sig}(X_\mathcal{B})(\tilde{q}_{X_\mathcal{B}})$, $created(X_\mathcal{A})(\tilde{q}_{X_\mathcal{A}})(a) = created(X_\mathcal{B})(\tilde{q}_{X_\mathcal{B}})(a)$.

By lemma 153 we have (*) $hidden\text{-}actions(Y_\mathcal{A})(\tilde{q}_{Y_\mathcal{A}}) = hidden\text{-}actions(Y_\mathcal{B})(\tilde{q}_{Y_\mathcal{B}})$, and by lemma 151 we have (**) $\forall a \in \widehat{sig}(Y_\mathcal{A})(q_{Y_\mathcal{A}}) = \widehat{sig}(Y_\mathcal{B})(q_{Y_\mathcal{B}})$.

If we combine the definition $q_{Y_\mathcal{A}} R_{conf} q_{Y_\mathcal{B}}$ with (*) and (**), we obtain $q_{Y_\mathcal{A}} R_{strict} q_{Y_\mathcal{B}}$, which ends the proof.

◀

Finally, the next lemma states that, after projection, two configuration-equivalent states obtain via executions with the same trace lead necessarily to strictly equivalent transitions.

▶ **Lemma 159** (After projection, configuration-equivalence obtain after same trace implies strict equivalent transitions)**.** *Let $X_\mathcal{A}$ and $X_\mathcal{B}$ be two PCA corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$. Let $Y_\mathcal{A} = X_\mathcal{A} \setminus \{\mathcal{A}\}$ and $Y_\mathcal{B} = X_\mathcal{B} \setminus \{\mathcal{B}\}$. Let $(\alpha_{Y_\mathcal{A}}, \pi_{Y_\mathcal{B}}) \in Execs(Y_\mathcal{A}) \times Execs(Y_\mathcal{B})$ with $lstate(\alpha_{Y_\mathcal{A}}) = q_{Y_\mathcal{A}}$ and $lstate(\pi_{Y_\mathcal{B}}) = q_{Y_\mathcal{B}}$. If*

- *$q_{Y_\mathcal{A}} R_{conf} q_{Y_\mathcal{B}}$ and*
- *$trace(\alpha_{Y_\mathcal{A}}) = trace(\pi_{Y_\mathcal{B}}) = \beta$,*

then for every $a \in \widehat{sig}(Y_\mathcal{A})(q_{Y_\mathcal{A}}) = \widehat{sig}(Y_\mathcal{B})(q_{Y_\mathcal{B}})$, $\eta_{(Y_\mathcal{A}, q_{Y_\mathcal{A}}, a)}$ and $\eta_{(Y_\mathcal{B}, q_{Y_\mathcal{B}}, a)}$ are strictly equivalent, i.e. $\exists f : Q_K \rightarrow Q_{K'}$ s.t. $\eta \overset{f}{\underset{strict}{\longleftrightarrow}} \eta'$

Proof. By previous lemma 158, $q_{Y_\mathcal{A}}$ and $q_{Y_\mathcal{B}}$ are strictly equivalent. Thus by previous lemma 156, there exists f s.t. $\eta_{(Y_\mathcal{A}, q_{Y_\mathcal{A}}, a)} \overset{f}{\underset{conf}{\longleftrightarrow}} \eta_{(Y_\mathcal{B}, q_{Y_\mathcal{B}}, a)}$. Let two corresponding states $(q'_{Y_\mathcal{A}}, q'_{Y_\mathcal{B}}) \in supp(\eta_{(Y_\mathcal{A}, q_{Y_\mathcal{A}}, a)}) \times \eta_{(Y_\mathcal{B}, q_{Y_\mathcal{B}}, a)}$ s.t. $f(q'_{Y_\mathcal{A}}) = q'_{Y_\mathcal{B}}$. We have $q'_{Y_\mathcal{A}} R_{conf} q'_{Y_\mathcal{B}}$(*). Furthermore, since $q_{Y_\mathcal{A}} R_{strict} q_{Y_\mathcal{B}}$, $sig(Y_\mathcal{A})(q_{Y_\mathcal{A}}) = sig(Y_\mathcal{B})(q_{Y_\mathcal{B}})$, namely $ext(Y_\mathcal{A})(q_{Y_\mathcal{A}}) = ext(Y_\mathcal{B})(q_{Y_\mathcal{B}})$, which means $trace(\alpha_{Y_\mathcal{A}} \frown q_{Y_\mathcal{A}} a q'_{Y_\mathcal{A}}) = trace(\pi_{Y_\mathcal{B}} \frown q_{Y_\mathcal{B}} a q'_{Y_\mathcal{B}})$. So we can reapply previous lemma to obtain $q'_{Y_\mathcal{A}} R_{strict} q'_{Y_\mathcal{B}}$ which ends the proof.

◀

Now we can finally show that if $X_\mathcal{A}$ and $X_\mathcal{B}$ are corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$ then $X_\mathcal{A} \setminus \{\mathcal{A}\}$ and $X_\mathcal{B} \setminus \{\mathcal{B}\}$ are semantically-equivalent which was the main aim of this subsection.

▶ **Theorem 160** ($X_\mathcal{A}$ and $X_\mathcal{B}$ corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$ implies $X_\mathcal{A} \setminus \{\mathcal{A}\}$ and $X_\mathcal{B} \setminus \{\mathcal{B}\}$ semantically-equivalent)**.** *Let $X_\mathcal{A}$ and $X_\mathcal{B}$ be two PCA corresponding w.r.t. $\mathcal{A}$, $\mathcal{B}$. Let $Y_\mathcal{A} = X_\mathcal{A} \setminus \{\mathcal{A}\}$ and $Y_\mathcal{B} = X_\mathcal{B} \setminus \{\mathcal{B}\}$.*

The PCA $Y_\mathcal{A}$ and $Y_\mathcal{B}$ are semantically-equivalent.

Proof. We recursively construct a strong complete bijective PCA executions-matching $(f_s, f_s^{tran}, f_s^{ex})$ where $f_s : reachable_{\leq s}(Y_\mathcal{A}) \rightarrow reachable_{\leq s}(Y_\mathcal{B})$ and $f_s^{ex} : \{\alpha \in Execs(Y_\mathcal{A}) || \alpha| \leq s\} \rightarrow \{\pi \in Execs(Y_\mathcal{B}) || \pi| \leq s\}$ s.t. $f_s^{ex}(\alpha) = \pi$ implies $lstate(\alpha) R_{strict} lstate(\pi)$.

Basis: $s = 0$, $reachable_{\leq 0}(Y_\mathcal{A}) = \{\bar{q}_{X_\mathcal{A}}\}$, while $reachable_{\leq 0}(Y_\mathcal{B}) = \{\bar{q}_{X_\mathcal{B}}\}$.

By definition 69 of corresponding automata $config(X_\mathcal{A})(\bar{q}_{X_\mathcal{A}}) \lhd_{\mathcal{AB}} config(X_\mathcal{B})(\bar{q}_{X_\mathcal{B}})$, while $(\bar{q}_{Y_\mathcal{A}}, \bar{q}_{Y_\mathcal{B}}) = (X_\mathcal{A}.\mu_s^\mathcal{A}(\bar{q}_{X_\mathcal{A}}), X_\mathcal{B}.\mu_s^\mathcal{B}(\bar{q}_{X_\mathcal{B}}))$ by definition 120 of PCA projection, which gives $\bar{q}_{Y_\mathcal{A}} R_{conf} \bar{q}_{Y_\mathcal{B}}$ by lemma 149. Moreover $trace_{Y_\mathcal{A}}(\bar{q}_{Y_\mathcal{A}}) = trace_{Y_\mathcal{B}}(\bar{q}_{Y_\mathcal{B}}) = \lambda$ (λ denotes the empty sequence). Thus we can apply lemma 158 to obtain $\bar{q}_{Y_\mathcal{A}} R_{strict} \bar{q}_{Y_\mathcal{B}}$. We construct $f_0(\bar{q}_{Y_\mathcal{A}}) = \bar{q}_{Y_\mathcal{B}}$, $f_0^{ex}(\bar{q}_{Y_\mathcal{A}}) = \bar{q}_{Y_\mathcal{B}}$. Clearly f_0 is a bijection from $reachable_0(Y_\mathcal{A})$ to $reachable_0(Y_\mathcal{B})$, while f_0^{ex} is a bijection from $Execs_0(Y_\mathcal{A})$ to $Execs_0(Y_\mathcal{B})$

Induction: We assume the result to be true for an integer $s \in \mathbb{N}$ and we will show it is then true for $s+1$. Let $Execs_s(Y_\mathcal{A}) = \{\alpha \in Execs(Y_\mathcal{A}) || \alpha| = s\}$ and $Execs_s(Y_\mathcal{B}) = \{\pi \in Execs(Y_\mathcal{B}) || \pi| = s\}$.

We can build f_{s+1} (resp. f_{s+1}^{ex}) s.t. $\forall q \in reachable_{\leq s}(Y_\mathcal{A}), f_{s+1}(q) = f_s(q)$ (resp. s.t. $\forall \alpha \in Execs_{\leq s}(Y_\mathcal{A})$ $f_{s+1}^{ex}(\alpha) = f_s^{ex}(\alpha)$) and $\forall q^j_{Y_\mathcal{A}} \in reachable_{s+1}(Y_\mathcal{A}), f_{s+1}(q^*)$ (resp. $\forall \alpha^{a,j} \in Execs_s(Y_\mathcal{A})$, $f_{s+1}^{ex}(\alpha')$) is built as follows:

We note $\alpha^{a,j} = \alpha_{Y_\mathcal{A}}^\frown q_{Y_\mathcal{A}} a q^j_{Y_\mathcal{A}}$ ($q_{Y_\mathcal{A}} = lstate(\alpha_{Y_\mathcal{A}})$). We note $\pi_{Y_\mathcal{B}} = f_s^{ex}(\alpha_{Y_\mathcal{A}})$. By induction assumption, $q_{Y_\mathcal{A}} R_{strict} q_{Y_\mathcal{B}}$ with $q_{Y_\mathcal{A}} = lstate(\alpha_{Y_\mathcal{A}})$ and $q_{Y_\mathcal{B}} = lstate(\pi_{Y_\mathcal{B}})$. Hence $sig(Y_\mathcal{A})(q_{Y_\mathcal{A}}) = sig(Y_\mathcal{B})(q_{Y_\mathcal{B}})$ and by previous lemma 159, for every $a \in sig(Y_\mathcal{A})(q_{Y_\mathcal{A}}) = sig(Y_\mathcal{B})(q_{Y_\mathcal{B}})$, $\exists g_a^j$, $\eta_{(Y_\mathcal{A}, q_{Y_\mathcal{A}}, a)} \xleftrightarrow[strict]{g_a^j} \eta_{(Y_\mathcal{B}, q_{Y_\mathcal{B}}, a)}$.

Hence, we define $f_{s+1}^{ex} : \alpha^{a,j} = \alpha_{Y_\mathcal{A}}^\frown q_{Y_\mathcal{A}} a q^j_{Y_\mathcal{A}} \mapsto f_{s+1}^{ex}(\alpha_{Y_\mathcal{A}})^\frown f_s(q_{Y_\mathcal{A}}) a g_a^j(q^j_{Y_\mathcal{A}})$, while f_{s+1} is naturally defined via f_{s+1}^{ex}, i.e. for every $q^j_{Y_\mathcal{A}} \in reachable_{s+1}(Y_\mathcal{A})$, we note $\alpha^{a,j} \in Execs_{s+1}(Y_\mathcal{A})$ s.t. $lstate(\alpha^{a,j}) = q^j_{Y_\mathcal{A}}$ and $f_{s+1}(q^j_{Y_\mathcal{A}}) = g_a^j(q^j_{Y_\mathcal{A}}) = lstate(f_{s+1}^{ex}(\alpha^{a,j}))$.

We finally define $f^{ex} : q^0 a^1 ... a^n q^n ... \mapsto f_0(q^0) a^1 ... a^n f_n(q^n)$, $f : q \mapsto f_n(q)$ where $q = lstate(q^0 a^1 ... q^n)$ and $f^{tr} : (q, a, \eta_{(Y_\mathcal{A}, q, a)}) \mapsto (f(q), a, \eta_{(Y_\mathcal{B}, f(q), a)})$.

Clearly (f, f^{tr}, f^{ex}) is strong since for every pair $(q_{Y_\mathcal{A}}, q_{Y_\mathcal{B}})$, s.t. $f(q_{Y_\mathcal{A}}) = q_{Y_\mathcal{B}}$, $q_{Y_\mathcal{A}} R_{strict} q_{Y_\mathcal{B}}$.

Moreover, (f, f^{tr}, f^{ex}) is complete since $dom(f) = reachable(Y_\mathcal{A}) = Q_{Y_\mathcal{A}}$.

Finally, the bijectivity of f^{ex} is given by the inductive bijective construction.

Hence (f, f^{tr}, f^{ex}) is strong complete bijective PCA executions-matching from $Y_\mathcal{A}$ to $Y_\mathcal{B}$ which ends the proof.

◀

14 Top/Down corresponding classes

In previous section 13, we have shown in theorem 160 that if $X_\mathcal{A}$ and $X_\mathcal{B}$ are corresponding w.r.t. $\mathcal{A}$ and $\mathcal{B}$ (in the sense of definition 69), then $Y_\mathcal{A} = X_\mathcal{A} \setminus \{\mathcal{A}\}$ and $Y_\mathcal{B} = X_\mathcal{B} \setminus \{\mathcal{B}\}$ are semantically equivalent. We can note Y an arbitrary PCA semantically equivalent with both $Y_\mathcal{A}$ and $Y_\mathcal{B}$.

In section 12, we have shown in theorem 140 that for every PCA $\mathcal{E}$ environment of both $X_\mathcal{A}$ and $X_\mathcal{B}$, $X_\mathcal{A} || \mathcal{E}$ and $\tilde{\mathcal{A}}^{sw} || Y_\mathcal{A} || \mathcal{E}$ (resp. $X_\mathcal{B} || \mathcal{E}$ and $\tilde{\mathcal{B}}^{sw} || Y_\mathcal{B} || \mathcal{E}$) are linked by a PCA executions-matching

It is time to combine this two results to realise that for every PCA $\mathcal{E}$ environment of both $X_\mathcal{A}$ and $X_\mathcal{B}$, $X_\mathcal{A} || \mathcal{E}$ and $\tilde{\mathcal{A}}^{sw} || \mathcal{E}'$ (resp. $X_\mathcal{B} || \mathcal{E}$ and $\tilde{\mathcal{B}}^{sw} || \mathcal{E}'$) are linked by a PCA executions-matching where $\mathcal{E}' = \mathcal{E} || Y$.

Hence (*) if $\mathcal{E}'$ cannot distinguish $\tilde{\mathcal{A}}^{sw}$ from $\tilde{\mathcal{B}}^{sw}$, we will be able to show that $\mathcal{E}$ cannot distinguish $X_\mathcal{A}$ from $X_\mathcal{B}$.

In this section, we formalise (*) in theorem 191 of monotonicity of implementation relation. However, some assumptions are required to reduce the implementation of $X_{\mathcal{B}}$ by $X_{\mathcal{A}}$ into implementation of $\mathcal{B}$ by $\mathcal{A}$. These are all minor technical assumptions except for one: our implementation relation concerns only a particular subset of schedulers so-called *creation-oblivious*, i.e. in order to compute (potentially randomly) the next transition, they do not take into account the internal actions of a sub-automaton preceding its last destruction.

14.1 Creation-oblivious scheduler

Here we recall the definition of creation-oblivious scheduler (already introduced in subsection 9.4), that does not take into account previous internal actions of a particular sub-automaton to output its probability over transitions to trigger.

We start by defining *strict oblivious-schedulers* that output the same transition with the same probability for pair of execution fragments that differ only by prefixes in the same class of equivalence. This definition is inspired by the one provided in the thesis of Segala, but is more restrictive since we require a strict equality instead of a correlation (section 5.6.2 in [20]).

▶ **Definition 161** (strict oblivious scheduler (recall)). *Let W be a PCA or a PSIOA, let $\sigma \in schedulers(W)$ and let $\equiv$ be an equivalence relation on $Frags^*(W)$ verifying $\forall \alpha_1, \alpha_2 \in Frags^*(W)$ s.t. $\alpha_1 \equiv \alpha_2$, $lstate(\alpha_1) = lstate(\alpha_2)$. We say that σ is $(\equiv)$-strictly oblivious if $\forall \alpha_1, \alpha_2, \alpha_3 \in Frags^*(\tilde{W})$ s.t. 1) $\alpha_1 \equiv \alpha_2$ and 2) $fstate(\alpha_3) = lstate(\alpha_2) = lstate(\alpha_1)$, then $\sigma(\alpha_1 \frown \alpha_3) = \sigma(\alpha_2 \frown \alpha_3)$.*

Now we define the relation of equivalence that defines our subset of creation-oblivious schedulers. Intuitively, two executions fragments ending on $\mathcal{A}$ creation are in the same equivalence class if they differ only in terms of internal actions of $\mathcal{A}$.

▶ **Definition 162.** *($\tilde{\alpha} \equiv^{cr}_{\mathcal{A}} \tilde{\alpha}'$ (recall)). Let $\tilde{A}$ be a PSIOA, $\tilde{W}$ be a PCA, $\forall \tilde{\alpha}, \tilde{\alpha}' \in Frags^*(\tilde{W})$, we say $\tilde{\alpha} \equiv^{cr}_{\mathcal{A}} \tilde{\alpha}'$ iff:*

1. *$\tilde{\alpha}, \tilde{\alpha}'$ both ends on $\mathcal{A}$-creation.*
2. *$\tilde{\alpha}$ and $\tilde{\alpha}'$ differ only in the $\mathcal{A}$-exclusive actions and the states of $\mathcal{A}$, i.e. $\mu(\tilde{\alpha}) = \mu(\tilde{\alpha}')$ where $\mu(\tilde{\alpha} = \tilde{q}^0 a^1 \tilde{q}^1 ... a^n \tilde{q}^n) \in Frags^*(\tilde{W})$ is defined as follows:*
   - *remove the $\mathcal{A}$-exclusive actions*
   - *replace each state $\tilde{q}^i$ by its configuration $Config(\tilde{W})(\tilde{q}) = (\mathbf{A}^i, \mathbf{S}^i)$*
   - *replace each configuration $(\mathbf{A}^i, \mathbf{S}^i)$ by $(\mathbf{A}^i, \mathbf{S}^i) \setminus \{\mathcal{A}\}$*
   - *replace the (non-alternating) sequences of identical configurations (due to $\mathcal{A}$-exclusiveness of removed actions) by one unique configuration.*
3. *$lstate(\alpha_1) = lstate(\alpha_2)$*

We can remark that the items 3 can be deduced from 1 and 2 if X is configuration-conflict-free. We can also remark that if $\tilde{W}$ is a $\mathcal{A}$-conservative PCA, we can replace $\mu(\tilde{\alpha}) = \mu(\tilde{\alpha}')$, by $\mu^{\mathcal{A}}_e(\tilde{\alpha}) \upharpoonright (\tilde{W} \setminus \{\mathcal{A}\}) = \mu^{\mathcal{A}}_e(\tilde{\alpha}') \upharpoonright (\tilde{W} \setminus \{\mathcal{A}\})$ but we want to be as general as possible for next definition of *creation oblivious scheduler*:

▶ **Definition 163** (creation-oblivious scheduler). *Let $\mathcal{A}$ be a PSIOA, W be a PCA, $\sigma \in schedulers(W)$. We say that σ is $\mathcal{A}$-*creation oblivious *if it is $(\equiv^{cr}_{\mathcal{A}})$-strictly oblivious.*

We say that σ is creation-oblivious *if it is $\mathcal{A}$-creation oblivious for every sub-automaton $\mathcal{A}$ of W ($\mathcal{A} \in \bigcup_{q \in Q_W} auts(config(W)(q))$). We note $CrOB$ the function that maps every PCA W to the set of creation-oblivious schedulers of W. If W is not a PCA but a PSIOA, $CrOB(W) = schedulers(W)$.*

If σ is $\mathcal{A}$-creation oblivious, we can remark that $\forall \alpha, \alpha' \in Execs^*(W), \alpha \equiv^{cr}_{\mathcal{A}} \alpha'$, $\sigma_{|\alpha} = \sigma_{|\alpha'}$ in the sense of definition 164 stated immediately below.

▶ **Definition 164** (conditioned scheduler). *Let $\mathcal{A}$ be a PSIOA, $\sigma \in schedulers(\mathcal{A})$ and let $\alpha_1 \in Frags^*(\mathcal{A})$. We note $\sigma_{|\alpha_1} : \{\alpha_2 \in Frags^*(\mathcal{A}) | fstate(\alpha_2) = lstate(\alpha_1)\} \rightarrow SubDisc(D_{\mathcal{A}})$ the* sub-scheduler conditioned by σ and α_1 *that verifies $\forall \alpha_2 \in Frags^*(\mathcal{A}), fstate(\alpha_2) = lstate(\alpha_1)$, $\sigma_{|\alpha_1}(\alpha_2) = \sigma(\alpha_1^\frown \alpha_2)$.*

We take the opportunity to state a lemma of conditional probability that will be used later for lemma 190.

▶ **Lemma 165** (conditional measure law). *Let $\mathcal{A}$ be a PSIOA, $\sigma \in schedulers(\mathcal{A})$ and let $\alpha_1 \in Frags^*(\mathcal{A})$ and $\sigma_{|\alpha_1}$ the sub-scheduler conditioned by σ and α_1. Let $\alpha_o, \alpha_2 \in Frags^*(\mathcal{A}), fstate(\alpha_2) = lstate(\alpha_1) \triangleq q_{12}$. Then*

$$\epsilon_{\sigma,\alpha_o}(C_{\alpha_1^\frown \alpha_2}) =: \begin{cases} \epsilon_{\sigma,\alpha_o}(C_{\alpha_1}) \cdot \epsilon_{\sigma_{|\alpha_1},q_{12}}(C_{\alpha_2}) & \text{if } \alpha_1 \nleq \alpha_o \\ \epsilon_{\sigma_{|\alpha_1},\alpha'_o}(C_{\alpha_2}) & \text{if } \alpha_o = \alpha_1^\frown \alpha'_o \end{cases}$$

Proof. We note $\alpha_{12} = \alpha_1^\frown \alpha_2$.

1. $\alpha_1 \nleq \alpha_o$:
   a. $\alpha_1 \nleq \alpha_o$ and $\alpha_o \nleq \alpha_1$:
      This implies $\alpha_{12} \nleq \alpha_o$ and $\alpha_o \nleq \alpha_{12}$ thus $\epsilon_{\sigma,\alpha_o}(C_{\alpha_1^\frown \alpha_2}) = \epsilon_{\sigma,\alpha_o}(C_{\alpha_1}) = 0$ which ends the proof.
   b. $\alpha_o \leq \alpha_1$:
      This implies $\alpha_o \leq \alpha_{12}$ By induction on size s of α_2. Basis: $s = 0$, i.e. $\alpha_2 = lstate(\alpha_1) = q_{12}$. Thus, we meet the second case of definition of $\epsilon_{\sigma_{|\alpha_1},q_{12}}(C_{\alpha_2})$: $\alpha_2 \leq q_{12}$, which means $\epsilon_{\sigma_{|\alpha_1},q_{12}}(C_{\alpha_2}) = 1$ and terminates the basis. Induction: We assume the result to be true up to size $s \in \mathbb{N}$ and we want to show it is still true for size $s+1$. Let $\alpha_2 \in Frags^*(\mathcal{A}), fstate(\alpha_2) = lstate(\alpha_1) \triangleq q_{12}$ with $|\alpha_2| = s+1$. We note $\alpha_2 = \alpha_2'^\frown q'aq$ and $\alpha'_{12} = \alpha_1^\frown \alpha'_2$. We have $|\alpha'_2| = s$ and $\alpha_o \leq \alpha'_{12}$
      By definition we have $\epsilon_{\sigma_{|\alpha_1},q_{12}}(C_{\alpha_2}) = \epsilon_{\sigma_{|\alpha_1},q_{12}}(C_{\alpha'_2}) \cdot \sigma(\alpha'_2)(\eta_{(\mathcal{A},q',a)}) \cdot \eta_{(\mathcal{A},q',a)}(q)$.
      In Parallel, by definition: $\epsilon_{\sigma,\alpha_o}(C_{\alpha_{12}}) = \epsilon_{\sigma,\alpha_o}(C_{\alpha'_{12}}) \cdot \sigma(\alpha'_{12})(\eta_{(\mathcal{A},q',a)}) \cdot \eta_{(\mathcal{A},q',a)}(q)$ and by induction assumption, $\epsilon_{\sigma,\alpha_o}(C_{\alpha_{12}}) = \epsilon_{\sigma,\alpha_o}(C_{\alpha_1}) \cdot \epsilon_{\sigma_{|\alpha_1},q_{12}}(C_{\alpha'_2}) \cdot \sigma(\alpha'_{12})(\eta_{(\mathcal{A},q',a)}) \cdot \eta_{(\mathcal{A},q',a)}(q)$ and so $\epsilon_{\sigma,\alpha_o}(C_{\alpha_{12}}) = \epsilon_{\sigma,\alpha_o}(C_{\alpha_1}) \cdot \epsilon_{\sigma_{|\alpha_1},q_{12}}(C_{\alpha_2})$, which ends the induction and so the case.
2. $\alpha_o = \alpha_1^\frown \alpha'_o$. By definition, $\epsilon_{\sigma,\alpha_o}(C_{\alpha_1}) = 1$
   a. both $\alpha_{12} \nleq \alpha_o$ and $\alpha_o \nleq \alpha_{12}$. This implies $\alpha_2 \nleq \alpha'_o$ and $\alpha'_o \nleq \alpha_2$ Then, by definition, $\epsilon_{\sigma,\alpha_o}(C_{\alpha_{12}}) = \epsilon_{\sigma_{|\alpha_1},\alpha'_o}(C_{\alpha_2}) = 0$.
   b. $\alpha_{12} \leq \alpha_o$. This implies $\alpha_2 \leq \alpha'_o$. Then, by definition, $\epsilon_{\sigma,\alpha_o}(C_{\alpha_{12}}) = \epsilon_{\sigma_{|\alpha_1},\alpha'_o}(C_{\alpha_2}) = 1$
   c. $\alpha_o \leq \alpha_{12}$:
      We proceed by induction on size s of α_2.
      Basis: $s = 0$, i.e. $\alpha_2 = q_{12}$. Then by definition $\epsilon_{\sigma,\alpha_o}(C_{\alpha_{12}}) = \epsilon_{\sigma,\alpha_o}(C_{\alpha_1}) = 1$. Moreover $q_{12} \leq \alpha'_o$ which means $\epsilon_{\sigma_{|\alpha_1},\alpha'_o}(C_{\alpha_2}) = 1$, which ends the basis.
      Induction:
      We assume the result to be true up to size $s \in \mathbb{N}$ and we want to show it is still true for size $s+1$. Let $\alpha_2 \in Frags^*(\mathcal{A}), fstate(\alpha_2) = lstate(\alpha_1) \triangleq q_{12}$ with $|\alpha_2| = s+1$.
      We note $\alpha_2 = \alpha_2'^\frown q'aq$ and $\alpha'_{12} = \alpha_1^\frown \alpha'_2$. We have $|\alpha'_2| = s$ and $\alpha_o \leq \alpha'_{12}$.
      By definition we have $\epsilon_{\sigma_{|\alpha_1},\alpha'_o}(C_{\alpha_2}) = \epsilon_{\sigma_{|\alpha_1},\alpha'_o}(C_{\alpha'_2}) \cdot \sigma(\alpha'_2)(\eta_{(\mathcal{A},q',a)}) \cdot \eta_{(\mathcal{A},q',a)}(q)$.
      In Parallel, by definition: $\epsilon_{\sigma,\alpha_o}(C_{\alpha_{12}}) = \epsilon_{\sigma,\alpha_o}(C_{\alpha'_{12}}) \cdot \sigma(\alpha'_{12})(\eta_{(\mathcal{A},q',a)}) \cdot \eta_{(\mathcal{A},q',a)}(q)$ and by induction assumption, $\epsilon_{\sigma,\alpha_o}(C_{\alpha_{12}}) = \epsilon_{\sigma,\alpha_o}(C_{\alpha_1}) \cdot \epsilon_{\sigma_{|\alpha_1},\alpha'_o}(C_{\alpha'_2}) \cdot \sigma(\alpha'_{12})(\eta_{(\mathcal{A},q',a)}) \cdot$

$\eta_{(\mathcal{A},q',a)}(q)$ and so $\epsilon_{\sigma,\alpha_o}(C_{\alpha_1^\frown\alpha_2}) = \epsilon_{\sigma,\alpha_o}(C_{\alpha_1})\cdot\epsilon_{\sigma_{|\alpha_1},\alpha'_o}(C_{\alpha_2})$. Finally, since $\epsilon_{\sigma,\alpha_o}(C_{\alpha_1}) = 1$, we have $\epsilon_{\sigma,\alpha_o}(C_{\alpha_{12}}) = \epsilon_{\sigma_{|\alpha_1},\alpha'_o}(C_{\alpha_2})$ which ends the induction, the case and so the proof.

◀

We have formally defined our notion of creation-oblivious scheduler. This will be a key property to ensure lemma 187 that allows to reduce the measure of a class of comportment as a function of measures of classes of shorter comportment where no creation of $\mathcal{A}$ or $\mathcal{B}$ occurs excepting potentially at very last action. This reduction is more or less necessary to obtain monotonicity of implementation relation.

14.2 Tools: proxy function, creation-explicitness, classes

In this subsection we introduce some tools frequently used during our proof of monotonicity. Later, we will adopt a quite general approach to understand the key properties of a perception function to ensure monotonicity. All these properties will be met by environment projection function $proj_{(.,.)}$, but not by trace function.

First we introduce proxy function, which enables a generic reduction from automata $(\tilde{\mathcal{E}}||X_{\mathcal{A}})$ to automata $((\tilde{\mathcal{E}}||X_{\mathcal{A}} \setminus \{\mathcal{A}\})||\tilde{\mathcal{A}}^{sw})$

▶ **Definition 166** (proxy). *Let $\mathcal{A}$ be a PSIOA. Let $f_{(.,.)}$ be an insight function. The $\mathcal{A}$-proxy function of f, noted $f^{\mathcal{A},proxy}_{(.,.)}$, is the insight function s.t. for every $\mathcal{A}$-conservative PCA X, $\forall \tilde{\mathcal{E}} \in env(X)$, $\forall \tilde{\alpha} \in dom((\tilde{\mathcal{E}}||X).\mu_e^{\mathcal{A},+})$, $f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\tilde{\alpha}) = f_{((\tilde{\mathcal{E}}||X\setminus\{\mathcal{A}\}),\tilde{\mathcal{A}}^{sw})}(\mu_e^{\tilde{\mathcal{A}},+}(\tilde{\alpha}))$*

Second, we define ordinary function, as functions capturing the fact that an environment obtain the exact same insight from $X_{\mathcal{A}}$ or from $((X_{\mathcal{A}} \setminus \{\mathcal{A}\})||\tilde{\mathcal{A}}^{sw})$. Any reasonable insight function is ordinary.

▶ **Definition 167** (ordinary). *Let $f_{(.,.)}$ be an insight function. We say $f_{(.,.)}$ is ordinary if for every PSIOA $\mathcal{A}$, for every $\mathcal{A}$-conservative PCA X, $\forall \tilde{\mathcal{E}} \in env(X)$, $\forall \tilde{\alpha} \in dom((\tilde{\mathcal{E}}||X).\mu_e^{\mathcal{A},+})$, $f_{(\tilde{\mathcal{E}},X)}(\tilde{\alpha}) = f_{(\tilde{\mathcal{E}},((X\setminus\{\mathcal{A}\})||\tilde{\mathcal{A}}^{sw}))}(\mu_e^{\tilde{\mathcal{A}},+}(\tilde{\alpha}))$*

It is worthy to remark that for ordinary perception function, a common perception in the reduced world implies a common perception in the original world. This fact will be used in the proof of lemma 185 of partitioning.

▶ **Lemma 168** (ordinary perception function). *Let f be an ordinary perception function. Then for every PSIOA $\mathcal{A}$, for every $\mathcal{A}$-conservative PCA X, $\forall \tilde{\mathcal{E}} \in env(X)$, $\forall \tilde{\alpha}, \tilde{\alpha}' \in dom((\tilde{\mathcal{E}}||X).\mu_e^{\mathcal{A},+})$*

$$f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\tilde{\alpha}) = f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\tilde{\alpha}') \Longrightarrow f_{(\tilde{\mathcal{E}},X)}(\tilde{\alpha}) = f_{(\tilde{\mathcal{E}},X)}(\tilde{\alpha}')$$

Proof. By definition of proxy function, $f_{((\tilde{\mathcal{E}}||X\setminus\{\mathcal{A}\}),\tilde{\mathcal{A}}^{sw})}(\mu_e^{\tilde{\mathcal{A}},+}(\tilde{\alpha})) = f_{((\tilde{\mathcal{E}}||X\setminus\{\mathcal{A}\}),\tilde{\mathcal{A}}^{sw})}(\mu_e^{\tilde{\mathcal{A}},+}(\tilde{\alpha}'))$. By definition of perception function, $f_{(\tilde{\mathcal{E}},((X\setminus\{\mathcal{A}\})||\tilde{\mathcal{A}}^{sw}))}(\mu_e^{\tilde{\mathcal{A}},+}(\tilde{\alpha})) = f_{(\tilde{\mathcal{E}},((X\setminus\{\mathcal{A}\})||\tilde{\mathcal{A}}^{sw}))}(\mu_e^{\tilde{\mathcal{A}},+}(\tilde{\alpha}'))$. By definition of ordinary function, $f_{(\tilde{\mathcal{E}},X)}(\tilde{\alpha}) = f_{(\tilde{\mathcal{E}},X)}(\tilde{\alpha}')$. ◀

▶ **Proposition 169.** *The environment projection function $proj_{(.,.)}$ (i.e. for each automaton K, $\forall \mathcal{E} \in env(K)$, $proj_{(\mathcal{E},K)} : \alpha \in Execs(\mathcal{E}||K) \mapsto \alpha \upharpoonright \mathcal{E}$) and the trace functions are ordinary function.*

Proof. By definition ◀

Now, we introduce two new concepts. First, we introduce notion of creation-explicitness, that states that an automaton has a clear dedicated set of actions to create each sub-automaton. This property of creation-explicitness will clarify the condition to obtain surjectivity of $\tilde{\mu}_e^{\mathcal{A},+}$ since it suffices to consider this function with a restricted range where no action of $\mathit{creation\text{-}actions}(X)(\mathcal{A})$ appears before last action.

▶ **Definition 170** (creation-explicit PCA). *Let $\mathcal{A}$ be a PSIOA and X be a PCA. We say that X is $\mathcal{A}$-*creation-explicit *iff: there exists a set of actions, noted $\mathit{creation\text{-}actions}(X)(\mathcal{A})$, s.t. $\forall q_X \in Q_X$, $\forall a \in \widehat{sig}(X)(q_X)$, if we note $\mathbf{A}_X = auts(config(X)(q_X))$ and $\varphi_X = created(X)(q_X)(a)$, then $\mathcal{A} \notin \mathbf{A}_X \wedge \mathcal{A} \in \varphi_X \Longleftrightarrow a \in \mathit{creation\text{-}actions}(X)(\mathcal{A})$.*

Second, we define classes of equivalence of some executions that imply the exact same perception from the environment.

▶ **Definition 171** (class of equivalence). *Let f be an insight function. Let $\mathcal{A}$ be a PSIOA. Let $\mathcal{E} \in env(\mathcal{A})$. Let $\zeta \in \bigcup_{PSIOA\ \mathcal{B}, \mathcal{E} \in env(\mathcal{B})} range(f_{(\mathcal{E},\mathcal{B})})$. We note $Class(\mathcal{E}, \mathcal{A}, f, \zeta) = \{\alpha \in Execs(\mathcal{E}||\mathcal{A})\} | f_{(\mathcal{E},\mathcal{A})}(\alpha) = \zeta\}$.*

14.3 Homomorphism between simple classes

In this subsection, we exhibit the conditions such that $\tilde{\mu}_e^{\mathcal{A},+}$ is an homomorphism between the perception after reduction and the original perception. These conditions are met by projection function.

First, we state that $\tilde{\mu}_e^{\mathcal{A},+}$ is surjective if we consider a range constituted of executions that does not create $\mathcal{A}$ before very last action.

▶ **Lemma 172** (Partial surjectivity with explicit creation). *Let $\mathcal{A}$ be a PSIOA and X be a $\mathcal{A}$-conservative and $\mathcal{A}$-creation-explicit PCA. Let $\tilde{\mathcal{E}}$ be partially-compatible with X. Let $Y = X \setminus \{\mathcal{A}\}$. Let $\mathcal{E}_{\mathcal{A}} = \tilde{\mathcal{E}}||Y$. Let $(((\tilde{\mathcal{E}}||X).\tilde{\mu}_z^{\mathcal{A}}, (\tilde{\mathcal{E}}||X).\tilde{\mu}_z^{\mathcal{A},+}), (\tilde{\mathcal{E}}||X).\tilde{\mu}_{tr}^{\mathcal{A},+}, (\tilde{\mathcal{E}}||X).\tilde{\mu}_e^{\mathcal{A},+})$ the $\tilde{\mathcal{E}}$-extension of $((X.\tilde{\mu}_z^{\mathcal{A}}, X.\tilde{\mu}_z^{\mathcal{A},+}), X.\tilde{\mu}_{tr}^{\mathcal{A},+}, X.\tilde{\mu}_e^{\mathcal{A},+})$. Let $\alpha, \alpha' \in Execs(\mathcal{E}_{\mathcal{A}}||\tilde{\mathcal{A}}^{sw})$ s.t. creation-actions$(X)(\mathcal{A}) \cap actions(\alpha) = \emptyset$*

1) Then $\exists \tilde{\alpha} \in dom(\tilde{\mu}_e^{\mathcal{A}})$ s.t. $\tilde{\mu}_e^{\mathcal{A},+}(\tilde{\alpha}) = \tilde{\mu}_e^{\mathcal{A}}(\tilde{\alpha}) = \alpha$.

2) If $\alpha' = \alpha^\frown q, a_!, q'$ with $a_! \in \mathit{creation\text{-}actions}(X)(\mathcal{A})$, then $\exists \tilde{\alpha}' \in dom(\tilde{\mu}_e^{\mathcal{A},+})$ s.t. $\tilde{\mu}_e^{\mathcal{A},+}(\tilde{\alpha}') = \alpha'$.

Proof. We proof the results in the same order they are stated in the lemma:

1. We note $\alpha = q^0, a^1, ..., a^n, q^n ...$ and we proof the result by induction on the prefix size s. Basis: the result trivially holds for any execution α of size 0 by construction of $X \setminus \{\mathcal{A}\}$ that requires $X.\mu_s^{\mathcal{A}}(\bar{q}_X) = \bar{q}_{X \setminus \{\mathcal{A}\}}$. We assume the result holds up to prefix size s and we show it still holds for prefix size $s+1$. We note $\alpha_s = q^0, a^1, ..., a^s, q^s$ and $\tilde{\alpha}^s \in Execs(\tilde{\mathcal{E}}||X)$ s.t. $\tilde{\mu}_e^{\mathcal{A}}(\tilde{\alpha}_s) = \alpha_s$. By lemma 138 of signature preservation $a^{s+1} \in sig(\tilde{\mathcal{E}}||X)(\tilde{q}_s)$. Moreover, by assumption $a^{s+1} \notin \mathit{creation\text{-}actions}(X)(\mathcal{A})$ which means the application of lemma 129 of homomorphic transitions leads us to $\eta_{((\tilde{\mathcal{E}}||X),\tilde{q}^s,a^{s+1})} \overset{\mu_z^{\mathcal{A}}}{\longleftrightarrow} \eta_{((\mathcal{E}_{\mathcal{A}}||\tilde{\mathcal{A}}^{sw}),q^s,a^{s+1})}$. So there exists $\tilde{q}^{s+1} \in supp(\eta_{((\tilde{\mathcal{E}}||X),\tilde{q},a_!)})$ with $\mu_z^{\mathcal{A}}(\tilde{q}) = q$. So $\mu_e^{\mathcal{A}}(\tilde{\alpha}_s^\frown \tilde{q}^s a^{s+1} \tilde{q}^{s+1}) = \alpha_{s+1}$. This ends the induction and so the proof of 1. .
2. We apply 1. and note $\tilde{\alpha} \in Execs(\tilde{\mathcal{E}}||X)$ s.t. $\tilde{\mu}_e^{\mathcal{A}}(\tilde{\alpha}) = \alpha$. By lemma 138 of signature preservation $a_! \in sig(\tilde{\mathcal{E}}||X)(\tilde{q})$ with $\tilde{q} = lstate(\alpha)$. Moreover, by lemma 129 of homomorphic transition, $\eta_{(\tilde{\mathcal{E}}||X),\tilde{q},a_!} \overset{\mu_z^{\mathcal{A},+}}{\longleftrightarrow} \eta_{(\mathcal{E}_{\mathcal{A}}||\tilde{\mathcal{A}}^{sw}),q,a_!}$. So there exists $\tilde{q}' \in supp(\eta_{(\tilde{\mathcal{E}}||X),\tilde{q},a_!})$ with $\mu_z^{\mathcal{A},+}(\tilde{q}') = q'$. So $\mu_e^{\mathcal{A},+}(\tilde{\alpha}^\frown \tilde{q} a_! \tilde{q}') = \alpha'$ which ends the proof.

◀

Since we i) classify executions in some classes according to their projection on an environment and ii) are concerned by the actions of the execution that create $\mathcal{A}$, the next lemma will simplify this classification. It states that if the projection e of an execution $\alpha \in Execs(\mathcal{E}_{\mathcal{A}}||\tilde{\mathcal{A}}^{sw})$ on the environment $\mathcal{E}_{\mathcal{A}}$ ends by an action $a_! \in creation\text{-}actions(X)(\mathcal{A})$, then the execution necessarily ends by $a_!$ (without additional suffix).

Then we define Γ-delineated function f that verifies the fact that an execution α perceived in Γ through f implies α does not create $\mathcal{A}$ before very last action.

▶ **Definition 173** (delineated function)**.** *Let $\mathcal{A}$ be a PSIOA, X a $\mathcal{A}$-conservative PCA, $\mathcal{E} \in env(X)$, $Y = X \setminus \{\mathcal{A}\}$, $\mathcal{E}_{\mathcal{A}} = \mathcal{E}||Y$. Let $f_{(.,.)}$ be an insight function. Let $\Gamma \subseteq range(f_{(\mathcal{E}_{\mathcal{A}},\tilde{\mathcal{A}}^{sw})})$. We say that f is $(\Gamma, \tilde{\mathcal{E}}, X, \mathcal{A})$-delineated if $\forall \zeta \in \Gamma$, $\forall \alpha \in Execs(\mathcal{E}_{\mathcal{A}}||\tilde{\mathcal{A}}^{sw})$, $f_{(\mathcal{E}_{\mathcal{A}},\tilde{\mathcal{A}}^{sw})}(\alpha) = \zeta$, implies $\alpha \in rangef(\tilde{\mathcal{E}}||X).\mu_e^{\mathcal{A},+}$, i.e $\forall \alpha' < \alpha$, $actions(\alpha') \cap creation\text{-}actions(X)(\mathcal{A}) = \emptyset$.*

It is worthy to remark that if the projection e of an execution α does not contain actions dedicated to the creation of $\mathcal{A}$ before very last action, then α does not create $\mathcal{A}$ before very last action.

▶ **Lemma 174** (projection is a delineated function with explicit creation)**.** *Let $\mathcal{A}$ be a PSIOA, X a $\mathcal{A}$-conservative PCA, $\mathcal{E} \in env(X)$, $Y = X \setminus \{\mathcal{A}\}$, $\mathcal{E}_{\mathcal{A}} = \mathcal{E}||Y$. Let $\Gamma \triangleq \{e \in Execs(\mathcal{E}_{\mathcal{A}}) | \forall e' < e, actions(e') \cap creation\text{-}actions(X)(\mathcal{A}) = \emptyset\}$. The projection function $proj_{(.,.)}$ is $(\Gamma, \tilde{\mathcal{E}}, X, \mathcal{A})$-delineated.*

Proof. Let $\alpha \in Execs(\mathcal{E}_{\mathcal{A}}||\tilde{\mathcal{A}}^{sw})$, $(\alpha \upharpoonright \mathcal{E}_{\mathcal{A}}) = e' \in \Gamma$. Hence either $|e'| = 0$ or $e' = e^\frown q a_! q'$ with $actions(e') \cap creation\text{-}actions(X)(\mathcal{A}) = \emptyset$. If $actions(\alpha) \cap creation\text{-}actions(X)(\mathcal{A}) = \emptyset$, the result is immediate. Assume the opposite. We note $\alpha = \alpha^{1\frown} q_\ell^1, a_!, q_f^{2\frown} \alpha^2$ with $a_! \in creation\text{-}actions(X)(\mathcal{A})$.

We have $q_\ell^1 \upharpoonright \tilde{\mathcal{A}}^{sw} = q^\phi_{\tilde{\mathcal{A}}^{sw}}$. Indeed, let us assume the contrary: $q_\ell^1 \upharpoonright \tilde{\mathcal{A}}^{sw} \neq q^\phi_{\tilde{\mathcal{A}}^{sw}}$. Then $q \upharpoonright \tilde{\mathcal{A}}^{sw} \neq q^\phi_{\tilde{\mathcal{A}}^{sw}}$ for every state $q \in \alpha^1$. Since $creation\text{-}actions(X)(\mathcal{A}) \cap actions(e') = \emptyset$, $creation\text{-}actions(X)(\mathcal{A}) \cap actions(\alpha^1) = \emptyset$. Thus we apply lemma 172 of partial surjectivity with explicit creation to obtain, there exists $\tilde{\alpha}^1 \in Execs(\tilde{\mathcal{E}}||X)$ s.t. $\tilde{\mu}_e^{\mathcal{A},+}(\tilde{\alpha}^1) = \alpha^1$ with both $\mathcal{A} \in auts(config(X)(lstate(\tilde{\alpha}^1) \upharpoonright X))$ and $a_! \in creation\text{-}actions(X)(\mathcal{A}) \cap sig(X)(lstate(\tilde{\alpha}^1)) \upharpoonright X)$ which is impossible.

Since $q_\ell^1 \upharpoonright \tilde{\mathcal{A}}^{sw} = q^\phi_{\tilde{\mathcal{A}}^{sw}}$, $q \upharpoonright \tilde{\mathcal{A}}^{sw} = q^\phi_{\tilde{\mathcal{A}}^{sw}}$ for every state $q \in \alpha^2$. Hence, $\alpha^2 = q_f^2$ to respect $\alpha \upharpoonright \mathcal{E}_{\mathcal{A}} = e'$, which means $\alpha = \alpha^{1\frown} q_\ell^1, a_!, q_f^2$. Since $creation\text{-}actions(X)(\mathcal{A}) \cap actions(e) = \emptyset$, $creation\text{-}actions(X)(\mathcal{A}) \cap actions(\alpha^1) = \emptyset$, which ends the proof.

◀

Now, we can clarify when $\tilde{\mu}_e^{\mathcal{A},+}$ is a bijection between "top/down" corresponding classes of equivalence.

▶ **Lemma 175.** *($\tilde{\mu}_e^{\mathcal{A},+}$ is a bijection from $\tilde{\mathcal{C}}$ to $\mathcal{C}$). Let $\mathcal{A}$ be a PSIOA and X be a $\mathcal{A}$-conservative and $\mathcal{A}$-creation-explicit PCA. Let $\tilde{\mathcal{E}} \in env(X)$. Let $Y = X \setminus \{\mathcal{A}\}$. Let $\mathcal{E}_{\mathcal{A}} = \tilde{\mathcal{E}}||Y$. Let $(((\tilde{\mathcal{E}}||X).\tilde{\mu}_z^{\mathcal{A}}, (\tilde{\mathcal{E}}||X).\tilde{\mu}_z^{\mathcal{A},+}), (\tilde{\mathcal{E}}||X).\tilde{\mu}_{tr}^{\mathcal{A},+}, (\tilde{\mathcal{E}}||X).\tilde{\mu}_e^{\mathcal{A},+})$ the $\tilde{\mathcal{E}}$-extension of $((X.\tilde{\mu}_z^{\mathcal{A}}, X.\tilde{\mu}_z^{\mathcal{A},+}), X.\tilde{\mu}_{tr}^{\mathcal{A},+}, X.\tilde{\mu}_e^{\mathcal{A},+})$.*

Let f be an ordinary perception function, $(\Gamma, \tilde{\mathcal{E}}, X, \mathcal{A})$-delineated.

For every $\zeta \in \Gamma$, $(\tilde{\mathcal{E}}||X).\tilde{\mu}_e^{\mathcal{A},+}$ is a bijection from $\tilde{\mathcal{C}}$ to $\mathcal{C}$, where

- $\tilde{\mathcal{C}} = Class(\tilde{\mathcal{E}}, X, f^{\mathcal{A},proxy}, \zeta)$
- $\mathcal{C} = Class(\mathcal{E}_{\mathcal{A}}, \tilde{\mathcal{A}}^{sw}, f, \zeta)$

Proof. ■ Injectivity is immediate by lemma 85, item (2).

- Surjectivity: Let $\alpha \in \mathcal{C}$. By definition, $f_{(\mathcal{E}_{\mathcal{A}}, \tilde{\mathcal{A}}^{sw})}(\alpha) = \zeta \in \Gamma$. Since f is $(\Gamma, \tilde{\mathcal{E}}, X, \mathcal{A})$-delineated, then $\forall \alpha' < \alpha$, $(actions(\alpha') \cap \text{creation-actions}(X)(\mathcal{A}) = \emptyset$. Hence, we can apply lemma 172 of partial surjectivity with explicit creation

◀

Hence, we obtain an equiprobability of top/down corresponding cones equipped with alter-ego schedulers.

▶ **Lemma 176** (equiprobability of top/down corresponding cones). *Let $\mathcal{A}$ be a PSIOA and X be a $\mathcal{A}$-conservative and $\mathcal{A}$-creation-explicit PCA. Let $\tilde{\mathcal{E}} \in env(X)$. Let $Y = X \setminus \{\mathcal{A}\}$. Let $\mathcal{E}_{\mathcal{A}} = \tilde{\mathcal{E}}||Y$. Let $(((\tilde{\mathcal{E}}||X).\tilde{\mu}_z^{\mathcal{A}}, (\tilde{\mathcal{E}}||X).\tilde{\mu}_z^{\mathcal{A},+}), (\tilde{\mathcal{E}}||X).\tilde{\mu}_{tr}^{\mathcal{A},+}, (\tilde{\mathcal{E}}||X).\tilde{\mu}_e^{\mathcal{A},+})$ the $\tilde{\mathcal{E}}$-extension of $((X.\tilde{\mu}_z^{\mathcal{A}}, X.\tilde{\mu}_z^{\mathcal{A},+}), X.\tilde{\mu}_{tr}^{\mathcal{A},+}, X.\tilde{\mu}_e^{\mathcal{A},+})$.*

Let f be an ordinary perception function, $(\Gamma, \tilde{\mathcal{E}}, X, \mathcal{A})$-delineated. Let $\zeta \in \Gamma$, and

- $\tilde{\mathcal{C}} = Class(\tilde{\mathcal{E}}, X, f^{\mathcal{A},proxy}, \zeta)$
- $\mathcal{C} = Class(\mathcal{E}_{\mathcal{A}}, \tilde{\mathcal{A}}^{sw}, f, \zeta)$

Then for every $\tilde{\sigma} \in schedulers(\tilde{\mathcal{E}}||X)$, for σ $(((\tilde{\mathcal{E}}||X).\tilde{\mu}_z^{\mathcal{A}}, (\tilde{\mathcal{E}}||X).\tilde{\mu}_z^{\mathcal{A},+}), (\tilde{\mathcal{E}}||X).\tilde{\mu}_{tr}^{\mathcal{A},+}, (\tilde{\mathcal{E}}||X).\tilde{\mu}_e^{\mathcal{A},+})$-alter ego of $\tilde{\sigma}$,

$$\epsilon_{\tilde{\sigma},\delta_{\bar{q}_{(\tilde{\mathcal{E}}||X)}}}(C_{\tilde{\mathcal{C}}}) = \epsilon_{\sigma,\delta_{\bar{q}_{(\mathcal{E}_{\mathcal{A}}||\tilde{\mathcal{A}}^{sw})}}}(C_{\mathcal{C}})$$

Proof. By lemma 175, $\tilde{\mu}_e^{\mathcal{A},+}$ is a bijection from $\tilde{\mathcal{C}}$ to $\mathcal{C}$. We note $\{(\tilde{\alpha}_i, \alpha_i)\}_{i \in I} = \tilde{\mathcal{C}} \times \mathcal{C}$ the related pairs of executions s.t. $\tilde{\mu}_e^{\mathcal{A},+}(\tilde{\alpha}_i) = \alpha_i$. We obtain $\epsilon_{\tilde{\sigma},\delta_{\bar{q}_{(\tilde{\mathcal{E}}||X)}}}(C_{\tilde{\mathcal{C}}}) = \sum_{i \in I} \epsilon_{\tilde{\sigma},\delta_{\bar{q}_{(\tilde{\mathcal{E}}||X)}}}(C_{\tilde{\alpha}_i})$ and $\epsilon_{\sigma,\delta_{\bar{q}_{(\mathcal{E}_{\mathcal{A}}||\tilde{\mathcal{A}}^{sw})}}}(C_{\mathcal{C}}) = \sum_{i \in I} \epsilon_{\sigma,\delta_{\bar{q}_{(\mathcal{E}_{\mathcal{A}}||\tilde{\mathcal{A}}^{sw})}}}(C_{\alpha_i})$.

Thus it is enough to show that $\forall i \in I, \epsilon_{\tilde{\sigma},\delta_{\bar{q}_{(\tilde{\mathcal{E}}||X)}}}(C_{\tilde{\alpha}_i}) = \epsilon_{\sigma,\delta_{\bar{q}_{(\mathcal{E}_{\mathcal{A}}||\tilde{\mathcal{A}}^{sw})}}}(C_{\alpha_i})$ which is given by theorem 84 that can be applied since $\tilde{\mu}_e^{\mathcal{A},+}$ is a continued executions-matching by theorem 144.

◀

14.4 Decomposition, pasting-friendly functions

In last subsection, the dynamic creation/destruction of $\mathcal{A}$ has been discarded. It is time to generalise previous approach with dynamic creation/destruction of $\mathcal{A}$.

We first define some tools to describe the decomposition of an executions into segments whose last action is in in the dedicated set to create $\mathcal{A}$.

▶ **Definition 177.** *(n-building-vector for executions). Let α be an alternating sequence states and actions starting by state and finishing by a state if α is finite. Let $n \in \mathbb{N} \cup \{\infty\}$. A n-*building-vector *of α is a (potentially infinite) vector $\vec{\alpha} = (\alpha^1, ..., \alpha^i, ...)$ of $|\vec{\alpha}| = n$ alternating sequences of states and actions starting by state and finishing by a state (excepting potentially the last one if it is infinite) s.t. $\alpha^1 {}^\frown ... \alpha^{i-1} {}^\frown \alpha^i {}^\frown ... = \alpha$ (with $\forall i \in [1, |\vec{\alpha}| - 1], fstate(\alpha_{i+1}) = lstate(\alpha_i)$). We note Building-vectors(α, n) the set of n-building-vector of α and $\vec{\alpha} \overset{n}{:} \alpha$ to say $\vec{\alpha} \in$ Building-vectors(α, n). We note Building-vectors$(\alpha) = \bigcup_{n \in \mathbb{N} \cup \{\infty\}}$ Building-vectors(α, n) and $\vec{\alpha} : \alpha$ to say $\alpha \in$ Building-vectors(α). We note $\vec{\alpha}[i] = \alpha^i$ and $\vec{\alpha}[: i] = \alpha^1 {}^\frown ... {}^\frown \alpha^{i-1}$. If W is an automaton, $\alpha \in Execs(W)$, $\vec{\alpha} : \alpha$ and f a function with $dom(f) \subseteq Frags(W)$, we note $f(\vec{\alpha}) = [f(\vec{\alpha}[1]), ..., f(\vec{\alpha}[i]), ...]$.*

▶ **Definition 178.** *($\vec{\alpha} \underset{(X,\mathcal{A})}{:} \alpha$) Let W and X be two PCA s.t. X is $\mathcal{A}$-creation-explicit, $\alpha \in Frags(W)$. We note $\vec{\alpha} \underset{(X,\mathcal{A})}{:} \alpha$ (and $\vec{\alpha} \underset{\mathcal{A}}{:} \alpha$ when X is clear in the context) the (clearly unique) vector $\vec{\alpha} \in$ Building-vectors(α) of execution fragments s.t.*

1. $\forall i \in [1,n]$, $\forall \alpha' < \vec{\alpha}[i]$, $actions(\alpha') \cap creation\text{-}actions(X)(\mathcal{A}) = \emptyset$ *and*
2. $\forall i \in [1, n-1]$, $laction(\vec{\alpha}[i])) \in creation\text{-}actions(X)(\mathcal{A})$.

We write $\vec{\alpha} \overset{n}{\underset{(X,\mathcal{A})}{\vdots}}$ *or* $\vec{\alpha} \overset{n}{\underset{\mathcal{A}}{\vdots}}$ *to indicate that* $|\vec{\alpha}| = n$.

▶ **Definition 179.** *($\mathcal{A}$-decomposition) Let $\mathcal{A}$ be a PSIOA and X be a PCA. Let* $\alpha = q^0 a^1 ... a^n q^n ... \in Frags(X)$. *We say that*

- α *is a* $\mathcal{A}$-open-portion *iff* α *does not create* $\mathcal{A}$*, i.e.* $\forall i \in [1, |\alpha|] \mathcal{A} \notin auts(config(X)(q^{i-1})) \Longrightarrow \mathcal{A} \notin auts(config(X)(q^i))$.
- α *is a* $\mathcal{A}$-closed-portion *iff* α *does not create* $\mathcal{A}$ *excepting at very last last action, i.e.* $\forall i \in [1, |\alpha|] \mathcal{A} \notin auts(config(X)(q^{i-1})) \wedge \mathcal{A} \in auts(config(X)(q^i)) \Longleftrightarrow i = |\alpha|$.
- α *is a* $\mathcal{A}$-portion *of X if it is either a $\mathcal{A}$-open-portion or a $\mathcal{A}$-closed-portion.*

We call $\mathcal{A}$-decomposition of α, noted $\mathcal{A}$-decomposition(α), the unique vector $(\alpha^1, ..., \alpha^n, ...) \in$ *Building-vectors*(α) *s.t.*

- $\forall i \in [1, |\mathcal{A}\text{-}decomposition(\alpha)| - 1]$, α^i *is a $\mathcal{A}$-closed-portion of X and*
- *if* $|\mathcal{A}\text{-}decomposition(\alpha)| = n \in \mathbb{N}$, α^n *is a $\mathcal{A}$-portion of X.*

▶ **Lemma 180.** *($\vec{\alpha} \underset{(X,\mathcal{A})}{\vdots} \alpha$ means $\vec{\alpha} = \mathcal{A}$-decomposition$(\alpha)$). Let $\mathcal{A}$ be a PSIOA and X be a $\mathcal{A}$-creation-explicit PCA. Let $\alpha \in Frags(X)$. Let $\vec{\alpha} = \mathcal{A}$-decomposition$(\alpha)$. Then* $\vec{\alpha} \overset{n}{\underset{(X,\mathcal{A})}{\vdots}} \alpha$.

Proof. By definition, $\vec{\alpha} \in Building\text{-}vectors(\alpha)$. Still by definition, $\forall i \in [1, |\mathcal{A}\text{-}decomposition(\alpha)| - 1]$, α^i is a $\mathcal{A}$-closed-portion of X, i.e. α^i does not create $\mathcal{A}$ excepting at very last last action $laction(\alpha_i)$. By definition of creation-explicitness, the two item of definition 178 are verified for every $i \in [1, |\mathcal{A}\text{-}decomposition(\alpha)| - 1]$. Finally, by definition, if $|\mathcal{A}\text{-}decomposition(\alpha)| = n \in \mathbb{N}$, α^n is a $\mathcal{A}$-portion of X, i.e. α^n does not create $\mathcal{A}$ excepting potentially at very last last action if α^n is finite. Again, by definition of creation-explicitness, the first item of definition 178 is verified.

◀

Now, we introduce the crucial property, called *pasting-friendly*, required for a perception function f to ensure monotonicity of $\leq_0^{CrOb,f}$. This property allows to cut-paste a general class of equivalence into a composition of smaller classes of equivalence, without creation of $\mathcal{A}$ before very last action, where lemma 176 of equiprobability between top-down corresponding cones can be applied to each smaller class.

▶ **Definition 181** (pasting friendly). *Let $f_{(.,.)}$ be an insight function. We say that $f_{(.,.)}$ is* pasting-friendly *if for every PSIOA $\mathcal{A}$, for every $\mathcal{A}$-conservative and $\mathcal{A}$-creation-explicit PCA X,* $\forall \tilde{\mathcal{E}} \in env(X)$, $\forall \tilde{\zeta} \in \bigcup_{K, \tilde{\mathcal{E}} \in env(K)} range(f_{(\tilde{\mathcal{E}},K)})$, $\forall \vec{\zeta} \in \overrightarrow{proxy}(\tilde{\zeta})_{\tilde{\mathcal{E}},X,\mathcal{A}}$ *then*

1. $\forall \tilde{\alpha}, \tilde{\alpha}'$, $\vec{\alpha} = \mathcal{A}\text{-}decomposition(\tilde{\alpha})$, $\vec{\alpha}' = \mathcal{A}\text{-}decomposition(\tilde{\alpha}')$, $f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\vec{\alpha}) = f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\vec{\alpha}') \triangleq \vec{\zeta}$ *implies* $|\vec{\alpha}| = |\vec{\alpha}| = |\vec{\zeta}| \triangleq n \in \mathbb{N} \cup \{\infty\} \wedge \forall i \in [1, n-1]$, $lstate(\vec{\alpha}[i]) = lstate(\vec{\alpha}'[i]) \triangleq q_i^\ell$.
2. *We note* $\tilde{\mathcal{E}}^1 = \tilde{\mathcal{E}}$, $X^1 = X$, *and* $\forall i \in [2, n]$, *we note* $\tilde{\mathcal{E}}^i = \tilde{\mathcal{E}}_{\bar{q}_{\tilde{\mathcal{E}}} \rightarrow (q^\ell_{i-1} \upharpoonright \tilde{\mathcal{E}})}$ *(resp* $X^i = X_{\bar{q}_X \rightarrow (q^\ell_{i-1} \upharpoonright X)}$*).*
   $\forall j \in [1, n], \forall \alpha^j \in Execs((\tilde{\mathcal{E}}^j || X^j)), f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}}^j, X^j)}(\alpha^j) = \vec{\zeta}[j]$, *then*
   a. *for every* $\alpha'_j < \alpha_j$, $actions(\alpha'_j) \cap creation\text{-}actions(X)(\mathcal{A}) = \emptyset$ *and*
   b. *if* $j \in [1, n-1]$, $\alpha_j = \alpha'_j \frown a^j_! q^j_\ell$ *with* $a^j_! \in creation(X)(\mathcal{A})$

We state an intermediate lemma to show that projection on environment is pasting-friendly (see lemma 183).

▶ **Lemma 182** (chunks ending on creation). *Let $\mathcal{A}$ be a PSIOA, let X be a $\mathcal{A}$-conservative and $\mathcal{A}$-creation-explicit PCA and $\tilde{\mathcal{E}}$ partially-compatible with X. Let $\tilde{\alpha} \in Frags(\tilde{\mathcal{E}}||X)$ and $e \in Frags(\tilde{\mathcal{E}}||X \setminus \{\mathcal{A}\})$ s.t. $(\tilde{\mathcal{E}}||X).\mu_e^{\mathcal{A},+}(\tilde{\alpha}) \upharpoonright (\tilde{\mathcal{E}}||X \setminus \{\mathcal{A}\}) = e$.*

Then

- $laction(\tilde{\alpha}) = a_! \in creation\text{-}actions(X)(\mathcal{A}) \Longrightarrow laction(e) = a_! \in creation\text{-}actions(X)(\mathcal{A})$.
- *if* $\tilde{\alpha} \in dom(\tilde{\mu}_e^{\mathcal{A},+})$,
  $laction(\tilde{\alpha}) = a_! \in creation\text{-}actions(X)(\mathcal{A}) \Longleftarrow laction(e) = a_! \in creation\text{-}actions(X)(\mathcal{A})$.

Proof. We prove the two implications in the same order.

- $\Longrightarrow$) Let assume $a_! \triangleq laction(\tilde{\alpha}) \in creation\text{-}actions(X)(\mathcal{A})$. Since X is $\mathcal{A}$-creation-explicit, we have $\tilde{\alpha} = \tilde{\alpha}'^\frown q' a_! q$ with $\mathcal{A} \notin auts(config(X)(q'))$. Thus $laction(e) = a_! \in creation\text{-}actions(X)(\mathcal{A})$.
- $\Longleftarrow$) Let assume $a_! \triangleq laction(e) \in creation\text{-}actions(X)(\mathcal{A})$. Thus $a_! \in actions(\tilde{\alpha})$. Since X is $\mathcal{A}$-creation-explicit, it implies $\tilde{\alpha} = \tilde{\alpha}^{1\frown} q_\ell^1, a_!, q_f^{2\frown} \tilde{\alpha}^2$ where $\mathcal{A} \notin auts(config(X)(q_\ell^1))$ and $\mathcal{A} \in auts(config(X)(q_f^2))$. But $\tilde{\alpha} \in dom((\tilde{\mathcal{E}}||X).\tilde{\mu}_e^{\mathcal{A},+})$, so $\tilde{\alpha}^2 = q_f^2$ and hence $laction(\tilde{\alpha}) = a_! \in creation\text{-}actions(X)(\mathcal{A})$

◀

Now, we are ready to show that projection on environment is pasting-friendly.

▶ **Lemma 183.** *The projection function $proj(.,.)$ (for each automaton K, $\forall \mathcal{E} \in env(K)$, $proj_{(\mathcal{E},K)} : \alpha \in Execs(\mathcal{E}||K) \mapsto \alpha \upharpoonright \mathcal{E}$ is pasting friendly.*

Proof. **1.** Let $\mathcal{A}$ be a PSIOA, let X be a $\mathcal{A}$-conservative PCA, let $\tilde{\mathcal{E}} \in env(X)$, let $\mathcal{E}_{\mathcal{A}} = (\tilde{\mathcal{E}}||(X \setminus \{\mathcal{A}\}))$. We note $q_{\ell,i} = lstate(\vec{\alpha}[i])$ and $q'_{\ell,i} = lstate(\vec{\alpha}'[i])$, $C_{\ell,i} = (\mathbf{A}_{\ell,i}, \mathbf{S}_{\ell,i}) = config(\tilde{\mathcal{E}}||X)(q_{\ell,i})$ and $C'_{\ell,i} = (\mathbf{A}'_{\ell,i}, \mathbf{S}'_{\ell,i}) = config(\tilde{\mathcal{E}}||X)(q'_{\ell,i})$. Let $i \in [1, |\alpha| - 1]$. By construction of $\mathcal{A}$-decomposition, $\mathbf{S}_{\ell,i}(\mathcal{A}) = \mathbf{S}'_{\ell,i}(\mathcal{A}) = \bar{q}_{\mathcal{A}}$ (1). Moreover, $f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\vec{\alpha}) = f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\vec{\alpha}') \triangleq \vec{\zeta}$, i.e. $proj_{(\mathcal{E}_{\mathcal{A}},\tilde{\mathcal{A}}^{sw})}(\vec{\alpha}[i]) = proj_{(\mathcal{E}_{\mathcal{A}},\tilde{\mathcal{A}}^{sw})}(\vec{\alpha}'[i])$, which means $q_{\ell,i} \upharpoonright \mathcal{E}_{\mathcal{A}} = q'_{\ell,i} \upharpoonright \mathcal{E}_{\mathcal{A}}$. Hence, $\mathbf{A}_{\ell,i} \setminus \{\mathcal{A}\} = \mathbf{A}'_{\ell,i} \setminus \{\mathcal{A}\} \triangleq \mathbf{A}''_{\ell,i}$ and $\forall \mathcal{B} \in \mathbf{A}''_{\ell,i}$, $\mathbf{S}_{\ell,i}(\mathcal{B}) = \mathbf{S}'_{\ell,i}(\mathcal{B})$ (2). By (1) and (2), $C_{\ell,i} = C'_{\ell,i}$. Since X is configuration-conflict-free, $q_{\ell,i} = q'_{\ell,i}$.

2. Let $j \in [1, n]$, let $\alpha^j \in Execs((\tilde{\mathcal{E}}^j||X^j)), f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}}^j,X^j)}(\alpha^j) = \vec{\zeta}[j]$ Let $\tilde{\alpha} \in Execs(\tilde{\mathcal{E}}||X)$, $\vec{\alpha} = \mathcal{A}\text{-decomposition}(\tilde{\alpha})$, $\vec{\alpha} \in (proj^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)})^{-1}(\vec{\zeta})$.

a. Let us assume $j \in [1, n-1]$. By construction of $\mathcal{A}$-decomposition, We have $\vec{\alpha}[j] = \alpha_j^{*\frown}(a_!^j q_\ell^j)$ with $actions(\alpha_j^*) \cap \text{creation-actions}(X)(\mathcal{A}) = \emptyset$ and $a_!^j \in \text{creation-actions}(X)(\mathcal{A})$. By lemma 182, it implies, $\vec{\zeta}[j] = e_j^{*\frown}(a_!^j q_\ell^j \upharpoonright \tilde{\mathcal{E}})$ with $actions(e_j^*) \cap \text{creation-actions}(X)(\mathcal{A}) = \emptyset$ and $a_!^j \in \text{creation-actions}(X)(\mathcal{A})$. By lemma 182, it implies $\alpha_j = \alpha_j'^\frown(a_!^j(q_\ell^j))$ with $actions(\alpha'_j) \cap \text{creation-actions}(X)(\mathcal{A}) = \emptyset$ and $a_!^j \in \text{creation-actions}(X)(\mathcal{A})$ (*). Moreover, let us assume $n \in \mathbb{N}$. For every $\alpha_n^* < \vec{\alpha}[n]$, $actions(\alpha_n^*) \cap \text{creation-actions}(X)(\mathcal{A}) = \emptyset$, hence, for every $e_n^* < \vec{\zeta}[n]$, $actions(e_n^*) \cap \text{creation-actions}(X)(\mathcal{A}) = \emptyset$ and so for every $\alpha_n^* < \alpha_n$, $actions(\alpha_n^*) \cap \text{creation-actions}(X)(\mathcal{A}) = \emptyset$.

b. Assume $j \in [1, n-1]$. By previous item, $\alpha_j = \alpha_j'^\frown(a_!^j q_\ell^j)$ with $actions(\alpha'_j) \cap \text{creation-actions}(X)(\mathcal{A}) = \emptyset$ and $a_!^j \in \text{creation-actions}(X)(\mathcal{A})$ (*). Moreover, by construction, we have $proj^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)})(\alpha_j) = proj^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)})(\vec{\alpha}[j])$ (**). We can apply the exact same reasoning than in item 1.

◀

Before stating our first lemma 185 of decomposition, we define the set of vector proxies. This set contains all the explanations $\overrightarrow{\zeta}^k$, from reduction, of a perception $\tilde{\zeta}$.

▶ **Definition 184.** *($\overrightarrow{proxy}(\tilde{\zeta})$) Let $f_{(.,.)}$ be an insight function. Let $\mathcal{A}$ be a PSIOA, let X be a $\mathcal{A}$-conservative PCA, let $\tilde{\mathcal{E}} \in env(X)$, Let $\tilde{\zeta} \in \bigcup_{K, \tilde{\mathcal{E}} \in env(K)} range(f_{(\tilde{\mathcal{E}},K)})$. We note $\overrightarrow{proxy}(\tilde{\zeta})_{(\tilde{\mathcal{E}},X,\mathcal{A})} = \{ \overrightarrow{\zeta}^k | \exists \tilde{\alpha} \in f^{-1}_{(\tilde{\mathcal{E}},X)}(\tilde{\zeta}) \wedge f^{\mathcal{A},proxy}_{\tilde{\mathcal{E}},X}(\mathcal{A}\text{-}decomposition(\tilde{\alpha})) = \overrightarrow{\zeta}^k \}$.*

Now, we can partition executions with a common perception $\tilde{\zeta}$ into sub-set of classes with more details related to the reduction.

▶ **Lemma 185.** *Let f be an ordinary perception function pasting friendly. Let $\mathcal{A}$ be a PSIOA, let X be a $\mathcal{A}$-conservative PCA, let $\tilde{\mathcal{E}} \in env(X)$, Let $\tilde{\zeta} \in \bigcup_{K, \tilde{\mathcal{E}} \in env(K)} range(f_{(\tilde{\mathcal{E}},K)})$. Let $\mathcal{C}^{\tilde{\zeta}} = Class(\tilde{\mathcal{E}}, X, f, \tilde{\zeta})$.*

$\mathcal{C}^{\tilde{\zeta}} = \biguplus_{\overrightarrow{\zeta}^k \in \overrightarrow{proxy}(\tilde{\zeta})_{(\tilde{\mathcal{E}},X,\mathcal{A})}} \mathcal{C}^{\overrightarrow{\zeta}^k}$ with

$\mathcal{C}^{\overrightarrow{\zeta}^k} = Class(\tilde{\mathcal{E}}, X, f^{\mathcal{A},proxy} \circ \mathcal{A}\text{-}decomposition, \overrightarrow{\zeta}^k)$

Proof. The proof is immediate by construction, since $\mathcal{A}$-decomposition is unique.

- (equality) We first show the equality by double inclusion.
  - ($\subseteq$) Let $\tilde{\alpha} \in \mathcal{C}^{\tilde{\zeta}}$. We note $\overrightarrow{\alpha} = \mathcal{A}$-decomposition($\tilde{\alpha}$). By construction, we have $\overrightarrow{\alpha} \underset{\mathcal{A}}{:} \tilde{\alpha}$. We note $\overrightarrow{\zeta} = f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\overrightarrow{\alpha})$. Obviously, $\overrightarrow{\zeta} \in \overrightarrow{proxy}(\tilde{\zeta})_{(\tilde{\mathcal{E}},X,\mathcal{A})}$.
  - ($\supseteq$) Let $\overrightarrow{\zeta}^k \in \overrightarrow{proxy}(\tilde{\zeta})_{(\tilde{\mathcal{E}},X,\mathcal{A})}$, with $n \triangleq |\overrightarrow{\zeta}^k|$, let $\tilde{\alpha} \in \mathcal{C}^{\overrightarrow{\zeta}^k}$. We want to show that $\tilde{\alpha} \in \mathcal{C}^{\tilde{\zeta}}$.
    Let $\overrightarrow{\alpha} = \mathcal{A}$-decomposition($\tilde{\alpha}$) By definition of $\overrightarrow{proxy}(\tilde{\zeta})_{(\tilde{\mathcal{E}},X,\mathcal{A})}$, $\exists \tilde{\alpha}' \in f^{-1}_{(\tilde{\mathcal{E}},X)}(\tilde{\zeta})$ s.t. $f^{\mathcal{A},proxy}_{\tilde{\mathcal{E}},X}(\mathcal{A}\text{-decomposition}(\tilde{\alpha}')) = \overrightarrow{\zeta}^k$. Let fix such a $\tilde{\alpha}'$. Let $\overrightarrow{\alpha}' = \mathcal{A}$-decomposition($\tilde{\alpha}'$). By construction $f^{\mathcal{A},proxy}_{\tilde{\mathcal{E}},X}(\overrightarrow{\alpha}) = f^{\mathcal{A},proxy}_{\tilde{\mathcal{E}},X}(\overrightarrow{\alpha}')$. Moreover, f is assumed to be pasting friendly, which implies $\forall i \in [1,n]$, $f^{\mathcal{A},proxy}_{\tilde{\mathcal{E}}^i,X^i}(\overrightarrow{\alpha}[i]) = f^{\mathcal{A},proxy}_{\tilde{\mathcal{E}}^i,X^i}(\overrightarrow{\alpha}'[i])$ where $\tilde{\mathcal{E}}^i$ and X^i are defined as in definition 181 of pasting friendly functions. Since f is an ordinary perception function, we can apply lemma 168, which implies that $\forall i \in [1,n]$, $f_{\tilde{\mathcal{E}},X}(\overrightarrow{\alpha}[i]) = f_{\tilde{\mathcal{E}},X}(\overrightarrow{\alpha}'[i]))$ and so $f_{\tilde{\mathcal{E}},X}(\tilde{\alpha}) = f_{\tilde{\mathcal{E}},X}(\tilde{\alpha}') = \tilde{\zeta}$, that is $\tilde{\alpha} \in \mathcal{C}^{\tilde{\zeta}}$.
- (partitioning) We show that $\forall (\overrightarrow{\zeta}^k, \overrightarrow{\zeta}^\ell), \overrightarrow{\zeta}^k \neq \overrightarrow{\zeta}^\ell, \mathcal{C}^{\overrightarrow{\zeta}^k} \cap \mathcal{C}^{\overrightarrow{\zeta}^\ell} = \emptyset$. Let $(\tilde{\alpha}, \tilde{\alpha}') \in \mathcal{C}^{\overrightarrow{\zeta}^k} \times \mathcal{C}^{\overrightarrow{\zeta}^\ell}$. Let $\overrightarrow{\alpha} \underset{\mathcal{A}}{:} \alpha$ and $\overrightarrow{\alpha}' \underset{\mathcal{A}}{:} \alpha'$. We have $f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\overrightarrow{\alpha}) = \overrightarrow{\zeta}^k \neq \overrightarrow{\zeta}^\ell = f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\overrightarrow{\alpha}')$. Thus $\overrightarrow{\alpha} \neq \overrightarrow{\alpha}'$. By lemma 180, $\overrightarrow{\alpha} = \mathcal{A}$-decomposition($\tilde{\alpha}$) and $\overrightarrow{\alpha}' = \mathcal{A}$-decomposition($\tilde{\alpha}'$), and so $\tilde{\alpha} \neq \tilde{\alpha}'$. Hence, $\forall (\overrightarrow{\zeta}^k, \overrightarrow{\zeta}^\ell), \overrightarrow{\zeta}^k \neq \overrightarrow{\zeta}^\ell, \mathcal{C}^{\overrightarrow{\zeta}^k} \cap \mathcal{C}^{\overrightarrow{\zeta}^\ell} = \emptyset$.

◀

Then, we perform our decomposition of $\hat{\mathcal{C}}^{\overrightarrow{\zeta}} = Class(\tilde{\mathcal{E}}, X, f^{\mathcal{A},proxy} \circ \mathcal{A}\text{-decomposition}, \overrightarrow{\zeta}^k)$ into small chunks.

▶ **Lemma 186** (decomposition into simple classes). *Let $f_{(.,.)}$ be pasting friendly. Let $\mathcal{A}$ be a PSIOA, X be a $\mathcal{A}$-conservative and $\mathcal{A}$-creation-explicit a PCA and $\tilde{\mathcal{E}}$ partially-compatible with X. Let $\mathcal{E}_{\mathcal{A}} = \tilde{\mathcal{E}}||(X \setminus \{\mathcal{A}\})$. Let $\tilde{\zeta} \in \bigcup_{K,\tilde{\mathcal{E}} \in env(K)} range(f_{(\tilde{\mathcal{E}},K)})$. Let $n \in \mathbb{N} \cup \{\infty\}$, let $\vec{\zeta} \in proxy(\tilde{\zeta})_{(\tilde{\mathcal{E}},X,\mathcal{A})}$ with $|\vec{\zeta}| = n$. Let $\hat{\mathcal{C}}^{\vec{\zeta}} = Class(\tilde{\mathcal{E}}, X, f^{\mathcal{A},proxy} \circ \mathcal{A}\text{-}decomposition, \vec{\zeta}^{k})$.*

Then, $\hat{\mathcal{C}}^{\vec{\zeta}} = \bigotimes_{i}^{n} \hat{\mathcal{C}}^{\vec{\zeta}[i]}$ with

1. $\hat{\mathcal{C}}^{\vec{\zeta}[i]} = Class(\tilde{\mathcal{E}}^i, X^i, f^{\mathcal{A},proxy}, \vec{\zeta}[i])$
2. $\forall \alpha^i \in \hat{\mathcal{C}}^{\vec{\zeta}[i]}$ *if* $i \in [1, n-1]$, $\alpha_i = \alpha_i' {}^\frown a_!^i q_\ell^i$ *with* $a_!^i \in creation(X)(\mathcal{A})$ *and if* $n \in \mathbb{N}$ $\forall \alpha_n' < \alpha_n, actions(\alpha_n') \cap creation\text{-}actions(X)(\mathcal{A}) = \emptyset$ *(ensured by pasting friendship of f).*
3. $\forall i \in [1, n-1]$, *we note q_ℓ^{i-1} the unique last state of every execution of $\hat{\mathcal{C}}^{\vec{\zeta}[i]}$ (ensured by pasting friendship of f).*
4. $\tilde{\mathcal{E}}^1 = \tilde{\mathcal{E}}$ *and* $\forall i \in [2, n]$, $\tilde{\mathcal{E}}^i = \tilde{\mathcal{E}}_{\bar{q}_{\mathcal{E}} \rightarrow q_{\mathcal{E}}^i}$ *, (as per definition 130), with* $q_{\mathcal{E}}^i = q_\ell^{i-1} \upharpoonright \tilde{\mathcal{E}}$.
5. $X^1 = X$ *and* $\forall i \in [2, n]$, $X^i = X_{\bar{q}_X \rightarrow q_X^i}$ *(as per definition 130) with* $q_X^i = q_\ell^{i-1} \upharpoonright X$.
6. $\bigotimes_{i}^{n} \mathcal{C}^i = \mathcal{C}^1 \otimes \mathcal{C}^2 \otimes ... \otimes \mathcal{C}^n$
7. $\mathcal{C}^1 \otimes \mathcal{C}^2 = \{\alpha_1 {}^\frown \alpha_2 | \alpha_1 \in \mathcal{C}^1, \alpha_2 \in \mathcal{C}^2\}$ *(The concatenation is always defined by item 3)*

Proof. The properties are ensured by the fact f is pasting-friendly. We prove the equality by double inclusion.

- $\subseteq$) Let $\alpha \in \hat{\mathcal{C}}^{\vec{\zeta}}$, and. $\vec{\alpha} = \mathcal{A}\text{-}decomposition(\alpha)$, i.e. $f_{\tilde{\mathcal{E}},X}^{\mathcal{A},proxy}(\vec{\alpha}) = \vec{\zeta}$. By construction due to $\mathcal{A}$-decomposition, $\forall i \in [2, n]$, $fstate(\vec{\alpha}[i]) = lstate(\vec{\alpha}[i-1])$ where $\vec{\alpha}[i-1]$ ends on $\mathcal{A}$-creation (1). Moreover, since f is assumed to be pasting-friendly, each q_ℓ^i is well defined (2). By (1) and (2), $fstate(\vec{\alpha}[i]) = \bar{q}_{\tilde{\mathcal{E}}^i||X^i}$ where $\tilde{\mathcal{E}}^i$ and X^i are defined like in the lemma (3). By construction due to $\mathcal{A}$-decomposition, $\vec{\alpha}[i]$ does not create $\mathcal{A}$ before its very last action, i.e. $\forall \alpha_i' < \vec{\alpha}[i], actions(\alpha_i') \cap \text{creation-actions}(X)(\mathcal{A}) = \emptyset$ (4). Thus by (3) and (4), $\alpha \in \bigotimes_{i}^{n} \hat{\mathcal{C}}^{\vec{\zeta}[i]}$. Hence, $\hat{\mathcal{C}}^{\vec{\zeta}} \subseteq \bigotimes_{i}^{n} \hat{\mathcal{C}}^{\vec{\zeta}[i]}$
- $\supseteq$) Let $\alpha \in \bigotimes_{i}^{n} \hat{\mathcal{C}}^{\vec{\zeta}[i]}$ Let $\vec{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_i, ...) \in \hat{\mathcal{C}}^{\vec{\zeta}[1]} \times \hat{\mathcal{C}}^{\vec{\zeta}[2]} \times ... \times \hat{\mathcal{C}}^{\vec{\zeta}[i]} \times ...$, s.t. $\vec{\alpha} : \alpha$. By construction, $\forall i \in [1, n]$ $f_{(\tilde{\mathcal{E}}^i, X^i)}^{\mathcal{A},proxy}(\alpha_i) = \vec{\zeta}[i]$. Hence $f_{(\tilde{\mathcal{E}}^i, X^i)}^{\mathcal{A},proxy}(\vec{\alpha}) = \vec{\zeta}$. It remains to show that $\vec{\alpha} = \mathcal{A}$-decomposition$(\alpha)$, which comes immediately from item 2.

◀

A first trivial analysis of measure of big class of equivalence gives the following lemma

▶ **Lemma 187** (measure after partitioning and decomposition). *Let $\mathcal{A}$ be a PSIOA, X be a $\mathcal{A}$-conservative and $\mathcal{A}$-creation-explicit PCA and $\tilde{\mathcal{E}}$ partially-compatible with X. Let $\mathcal{E}_{\mathcal{A}} = \tilde{\mathcal{E}}||X \setminus \{\mathcal{A}\}$. Let $\tilde{\zeta} \in \bigcup_{K,\tilde{\mathcal{E}} \in env(K)} range(f_{(\tilde{\mathcal{E}},K)})$. Let $\tilde{\sigma} \in schedulers(\tilde{\mathcal{E}}||X)$.*

$\epsilon_{\tilde{\sigma}}(C_{\tilde{\mathcal{C}}\tilde{\zeta}}) = \sum_{\vec{\zeta} \in proxy(\tilde{\zeta})_{(\tilde{\mathcal{E}},X,\mathcal{A})}} \epsilon_{\tilde{\sigma}}(C_{\bigotimes_{i=1}^{|\vec{\zeta}|} \hat{\mathcal{C}}^{\vec{\zeta}[i]}})$.

Proof. Immediate by two previous lemma 185 and 186 ◀

14.5 Creation oblivious scheduler applied to decomposition

Now we want to transform the term $\epsilon_{\tilde{\sigma}}(C_{\bigotimes_{i=1}^{|\vec{\zeta}|} \hat{\mathcal{C}}^{\vec{\zeta}[i]}})$ as a function of some terms $\epsilon_{\tilde{\sigma}^i}(C_{\hat{\mathcal{C}}^{\vec{\zeta}[i]}})$ where $\tilde{\sigma}^i$ must be defined. The critical point is that the occurrence of these events might not be independent with (*) a perfect-information scheduler that chooses the measure of class $\hat{\mathcal{C}}^{\vec{\zeta}[i]}$ as a function of the concrete prefix in class $\hat{\mathcal{C}}^{\vec{\zeta}[j<i]}$. This observation enforced us to weaken the implementation definition to make it monotonic w.r.t. PSIOA creation by handling only creation-oblivious schedulers that cannot make the choice (*).

Here again, we exhibit a key property of a perception function to ensure monotonicity of implementation w.r.t. creation oblivious schedulers.

▶ **Definition 188** (creation oblivious function)**.** *Let $f_{(.,.)}$ be an insight function. f is said creation-oblivious, if for every PSIOA $\mathcal{A}$, for every $\mathcal{A}$-conservative and $\mathcal{A}$-creation-explicit PCA X, $\forall \tilde{\mathcal{E}} \in env(X)$, $\forall \tilde{\alpha}, \tilde{\alpha}' \in Execs(\tilde{\mathcal{E}}||X)$, $\tilde{\alpha}, \tilde{\alpha}'$ ends on $\mathcal{A}$-creation, then $f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\tilde{\alpha}) = f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\tilde{\alpha}')$ implies $\tilde{\alpha} \equiv^{cr}_{\mathcal{A}} \tilde{\alpha}'$.*

In that case, for every $\mathcal{A}$-creation-oblivious scheduler $\tilde{\sigma}$ of $\tilde{\mathcal{E}}||X$, we can note $\tilde{\sigma}|_{\mathcal{A},\zeta} = \tilde{\sigma}|_{\tilde{\alpha}}$ for any $\tilde{\alpha} \in Execs(\tilde{\mathcal{E}}||X)$ s.t. $f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\tilde{\alpha}) = \zeta$.

This property is naturally verified by environment projection function.

▶ **Lemma 189.** *Let $proj_{(.,.)}$ the environment projection function i.e. for each automaton K, $\forall \mathcal{E} \in env(K)$, $proj_{(\mathcal{E},K)} : \alpha \in Execs(\mathcal{E}||K) \mapsto \alpha \upharpoonright \mathcal{E}$. Then $proj_{(.,.)}$ is creation-oblivious.*

Proof. Let $\mathcal{A}$ be a PSIOA, let X be a $\mathcal{A}$-conservative and $\mathcal{A}$-creation-explicit PCA, let $\tilde{\mathcal{E}} \in env(X)$, let $\tilde{\alpha}, \tilde{\alpha}' \in Execs(\tilde{\mathcal{E}}||X)$, s.t. $\tilde{\alpha}, \tilde{\alpha}'$ ends on $\mathcal{A}$-creation and $proj^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\tilde{\alpha}) = proj^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\tilde{\alpha}')$. Then by definition, $(\tilde{\mathcal{E}}||X).\tilde{\mu}^{\mathcal{A}}_e(\alpha) \upharpoonright (\tilde{\mathcal{E}}||(X \setminus \{\mathcal{A}\}) = (\tilde{\mathcal{E}}||X).\tilde{\mu}^{\mathcal{A}}_e(\alpha') \upharpoonright (\tilde{\mathcal{E}}||(X \setminus \{\mathcal{A}\})$ which meets the definition of $\tilde{\alpha} \equiv^{cr}_{\mathcal{A}} \tilde{\alpha}'$. ◀

Finally, we can terminate our decomposition argument, assuming creation oblivious schedulers.

▶ **Lemma 190** (measure after decomposition for oblivious creation scheduler)**.** *Let $\mathcal{A}$ be a PSIOA, X be a $\mathcal{A}$-conservative, $\mathcal{A}$-creation-explicit PCA and $\tilde{\mathcal{E}}$ partially-compatible with X. Let f a creation-oblivious insight function.*

Let $\tilde{\zeta} \in \bigcup_{K,\tilde{\mathcal{E}} \in env(K)} range(f_{(\tilde{\mathcal{E}},K)})$. Let $n \in \mathbb{N} \cup \{\infty\}$, let $\vec{\zeta} \in \overrightarrow{proxy}(\tilde{\zeta})_{(\tilde{\mathcal{E}},X,\mathcal{A})}$ with $|\vec{\zeta}| = n$. Let $\tilde{\sigma} \in schedulers(\tilde{\mathcal{E}}||X)$ that is $\mathcal{A}$-creation-oblivious.

Then $\epsilon_{\tilde{\sigma}}(C_{\bigotimes_i^n \hat{\mathcal{C}}^{\vec{\zeta}[i]}}) = \prod_i^n \epsilon_{\tilde{\sigma}^i}(C_{\mathcal{C}^{\vec{\zeta}[i]}})$ with $\forall i \in [1,n], \tilde{\sigma}^i = oblivious_{\mathcal{A},\vec{\zeta}[:i]}(\tilde{\sigma})$.

Proof. We recall the remark of definition 163 of $\mathcal{A}$-creation-oblivious scheduler for a $\mathcal{A}$-conservative PCA that raises the fact that if an execution fragment $\tilde{\alpha} \in Frags^*((\tilde{\mathcal{E}}||X))$ verifying

i) $\tilde{\alpha}$ ends on $\mathcal{A}$-creation and ii) $f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\tilde{\alpha}) = \zeta$, then $\tilde{\sigma}_{|\mathcal{A},\zeta} = \tilde{\sigma}_{|\tilde{\alpha}}$, the sub-scheduler conditioned by $\tilde{\sigma}$ and $\tilde{\alpha}$ in the sense of definition 164. Then we simply apply lemma 165, which states that for every $\alpha = \alpha_x^\frown \alpha_y \in Frags^*(\tilde{\mathcal{E}}||X)$, for $\tilde{\sigma}_{|\alpha_x}$ the sub-scheduler conditioned by $\tilde{\sigma} \in schedulers(\tilde{\mathcal{E}}||X)$ and α_x (in the sense of definition 164), for $\epsilon_{\tilde{\sigma}}$ generated by $\tilde{\sigma}$, $\epsilon_{\tilde{\sigma}}(C_\alpha) = \epsilon_{\tilde{\sigma}}(C_{\alpha_x}) \cdot \epsilon_{\tilde{\sigma}_{|\alpha_x}}(C_{\alpha_y})$ with $\tilde{\sigma}_{|\alpha_x}(\alpha_z) = \tilde{\sigma}(\alpha_x^\frown \alpha_z)$ for every α_z with $fstate(\alpha_z) = lstate(\alpha_x)$.

For every $\alpha \in \bigotimes_i^n \hat{\mathcal{C}}^{\overrightarrow{\zeta}[i]}$, for $\overrightarrow{\alpha} = \mathcal{A}$-decomposition, $\epsilon_{\tilde{\sigma}}(C_\alpha) = \prod_i^n \epsilon_{\tilde{\sigma}_{|\overrightarrow{\alpha}[1:i-1]}}(C_{\overrightarrow{\alpha}[i]})$, with $\overrightarrow{\alpha}[1:i-1] = \alpha^1 \frown ... \frown \alpha^{i-1}$.

By $\mathcal{A}$-creation-oblivious property of $\tilde{\sigma}$ and creation-oblivious of f, $\prod_i^n \epsilon_{\tilde{\sigma}_{|\overrightarrow{\alpha}[1:i-1]}}(C_{\overrightarrow{\alpha}[i]}) = \prod_i^n \epsilon_{\tilde{\sigma}_{|\overrightarrow{\zeta}[1:i-1]}}(C_{\overrightarrow{\alpha}[i]})$ with $\overrightarrow{\zeta}[1:i-1] = f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X)}(\overrightarrow{\alpha}[1:i-1])$.

Hence, for every $i \in [1,n]$ we note $\tilde{\sigma}^i \in schedulers(\tilde{\mathcal{E}}^i||X^i)$ that matches $\tilde{\sigma}_{|\overrightarrow{\alpha}[1:i-1]}$ on $\mathcal{C}^{\zeta_j}$ for an arbitrary $\overrightarrow{\alpha}[1:i-1]$.

This leads us to: $\forall \alpha \in \bigotimes_i^n \hat{\mathcal{C}}^{\overrightarrow{\zeta}[i]}$, for $\overrightarrow{\alpha} \underset{(X,\mathcal{A})}{:} \alpha$, $\epsilon_{\tilde{\sigma}}(C_\alpha) = \prod_i^n \epsilon_{\tilde{\sigma}^i}(C_{\overrightarrow{\alpha}[i]})$

Thus $\epsilon_{\tilde{\sigma}}(C_{\bigotimes_i^n \hat{\mathcal{C}}^{\overrightarrow{\zeta}[i]}}) = \sum_{\overrightarrow{\alpha} \underset{(X,\mathcal{A})}{:} \alpha,\ \alpha \in \bigotimes_i^n \hat{\mathcal{C}}^{\overrightarrow{\zeta}[i]}} \prod_i^n \epsilon_{\tilde{\sigma}^i}(C_{\overrightarrow{\alpha}[i]})$ and by lemma 186,

$\epsilon_{\tilde{\sigma}}(C_{\bigotimes_i^n \hat{\mathcal{C}}^{\overrightarrow{\zeta}[i]}}) = \sum_{\alpha_1 \in \mathcal{C}^{\overrightarrow{\zeta}[1]}} ... \sum_{\alpha_i \in \mathcal{C}^{\overrightarrow{\zeta}[i]}} ... \prod_i^n \epsilon_{\tilde{\sigma}^i}(C_{\alpha_i}) = \prod_i^n \epsilon_{\tilde{\sigma}^i}(C_{\mathcal{C}^{\overrightarrow{\zeta}[i]}})$

◀

14.6 Monotonicity of implementation

We use the previous decomposition to state the monotonicity of implementation relationship.

▶ **Theorem 191** (monotonicity). *Let $\mathcal{A}$ and $\mathcal{B}$ be two PSIOA, let $X_\mathcal{A}$ be a $\mathcal{A}$-conservative and $\mathcal{A}$-creation-explicit PCA, let $X_\mathcal{B}$ be a $\mathcal{B}$-conservative and $\mathcal{B}$-creation-explicit PCA, s.t. $X_\mathcal{A}$ and $X_\mathcal{B}$ are corresponding w.r.t. $\mathcal{A},\mathcal{B}$ with creation-actions$(X_\mathcal{A})(\mathcal{A})$ = creation-actions$(X_\mathcal{B})(\mathcal{B}) \triangleq CrActs$.*

Let $S = CrOb$ the scheduler schema of creatio-oblivious scheduler. Let $f_{(.,.)} = proj_{(.,.)}$ the environment projection function i.e. for each automaton K, $\forall \mathcal{E} \in env(K)$, $f_{(\mathcal{E},K)} : \alpha \in Execs(\mathcal{E}||K) \mapsto \alpha \upharpoonright \mathcal{E}$.

If $\mathcal{A} \leq_0^{S,f} \mathcal{B}$, then $X_\mathcal{A} \leq_0^{S,f} X_\mathcal{B}$.

Proof. Let $\tilde{\mathcal{E}} \in env(X_\mathcal{A}) \cap env(X_\mathcal{B})$. Let $Y_\mathcal{A} = X_\mathcal{A} \setminus \{\mathcal{A}\}$, $Y_\mathcal{B} = X_\mathcal{B} \setminus \{\mathcal{B}\}$, $\mathcal{E}_\mathcal{A} = \tilde{\mathcal{E}}||Y_\mathcal{A}$, $\mathcal{E}_\mathcal{B} = \tilde{\mathcal{E}}||Y_\mathcal{B}$ and $\mathcal{E}$ an arbitrary PCA semantically equivalent to both $\mathcal{E}_\mathcal{A}$ and $\mathcal{E}_\mathcal{B}$ with $\mathcal{E} \in env(\tilde{A}^{sw}) \cap env(\tilde{\mathcal{B}}^{sw})$ by theorem 160. We note $\mu_{\mathcal{AC}}$ the (complete, strong and bijective) PCA executions-matching from $\mathcal{E}_\mathcal{A}$ to $\mathcal{E}$ and $\mu_{\mathcal{CB}}$ the (complete, strong and bijective) PCA executions-matching from $\mathcal{E}$ to $\mathcal{E}_\mathcal{B}$. We also note $\mu^\times_{\mathcal{AC}}$ the (complete, strong and bijective) PCA executions-matching from $\mathcal{E}_\mathcal{A}||\tilde{A}^{sw}$ to $\mathcal{E}||\tilde{A}^{sw}$ and $\mu^\times_{\mathcal{CB}}$ the (complete, strong and bijective) PCA executions-matching from $\mathcal{E}||\tilde{B}^{sw}$ to $\mathcal{E}_\mathcal{B}||\tilde{B}^{sw}$.

In the remaining we note $(\tilde{\mathcal{E}}||X_\mathcal{A})^{\downarrow\zeta}$ the automaton $(\tilde{\mathcal{E}}||X_\mathcal{A})_{\bar{q}_{(\tilde{\mathcal{E}}||X_\mathcal{A})} \rightarrow q}$ (as per definition 130) where q is the unique last state of every execution $\tilde{\alpha}$ s.t. $f^{proxy}_{(\tilde{\mathcal{E}},X_\mathcal{A})}(\tilde{\alpha}) = \zeta$. Respectively, we note $(\tilde{\mathcal{E}}||X_\mathcal{B})^{\downarrow\zeta}$ the automaton $(\tilde{\mathcal{E}}||X_\mathcal{B})_{\bar{q}_{(\tilde{\mathcal{E}}||X_\mathcal{B})} \rightarrow q}$ (as per definition 130) where q is the unique last state of every execution $\tilde{\pi}$ s.t. $f^{proxy}_{(\tilde{\mathcal{E}},X_\mathcal{B})}(\tilde{\pi}) = \zeta$. This notation is possible because f is pasting-friendly. Finally, $\forall e \in Execs(\tilde{\mathcal{E}})$, we note $\tilde{\mathcal{E}}^e = \tilde{\mathcal{E}}_{\bar{q}_{\tilde{\mathcal{E}}} \rightarrow lstate(e)}$.

Let $\tilde{\sigma} \in S(\tilde{\mathcal{E}}||X_\mathcal{A})$. We need to show there exists $\tilde{\sigma}' \in S(\tilde{\mathcal{E}}||X_\mathcal{B})$ s.t.

- $\forall \tilde{\zeta} \in range(f_{(\tilde{\mathcal{E}},X_\mathcal{A})}) \cup range(f_{(\tilde{\mathcal{E}},X_\mathcal{B})})$, $\epsilon_{\tilde{\sigma}}(C_{\tilde{\mathcal{C}}^{\tilde{\zeta}}_{X_\mathcal{A}}}) = \epsilon_{\tilde{\sigma}'}(C_{\tilde{\mathcal{C}}^{\tilde{\zeta}}_{X_\mathcal{B}}})$
- where $\tilde{\mathcal{C}}^{\tilde{\zeta}}_{X_\mathcal{A}} = Class(\tilde{\mathcal{E}}, X_\mathcal{A}, f, \tilde{\zeta})$ and $\tilde{\mathcal{C}}^{\tilde{\zeta}}_{X_\mathcal{B}} = Class(\tilde{\mathcal{E}}, X_\mathcal{B}), f, \tilde{\zeta})$.

Let $\tilde{\zeta} \in range(f_{(\tilde{\mathcal{E}},X_\mathcal{A})}) \cup range(f_{(\tilde{\mathcal{E}},X_\mathcal{B})})$. For every $\overrightarrow{\zeta} \in \overrightarrow{proxy}(\tilde{\zeta})_{(\tilde{\mathcal{E}},X_\mathcal{A},\mathcal{A})}$, $\forall i \in [1 : |\overrightarrow{\zeta}|]$, we note $\sigma_{|\mathcal{A},\overrightarrow{\zeta}[:i]}$ the $((\tilde{\mathcal{E}}||X_\mathcal{A})^{\downarrow\overrightarrow{\zeta}[:i]}).\tilde{\mu}^{\mathcal{A},+}_e$ alter-ego of $\tilde{\sigma}_{|\mathcal{A},\overrightarrow{\zeta}[:i]}$. For every $i \in [1 : |\overrightarrow{\zeta}|]$

$\tilde{\alpha}', \tilde{\alpha}'' \in (f^{\mathcal{A},proxy}_{(\tilde{\mathcal{E}},X_{\mathcal{A}})})^{-1}(\vec{\zeta}[: i])$, $lstate(\tilde{\alpha}') = lstate(\tilde{\alpha}'') \triangleq q^{i-1}_{\ell}$ since f is pasting-friendly. We note $\mathcal{E}^{(\vec{\zeta},i)} = \mathcal{E}_{\bar{q}_{\mathcal{E}} \rightarrow \mu_{\mathcal{AC}}(q^{i-1}_{\ell} \upharpoonright \mathcal{E}_{\mathcal{A}})}$

We note $\sigma^{c}_{|\mathcal{A}, \vec{\zeta}[:i]} \in schedulers(\mathcal{E}^{(\vec{\zeta},i)}||\tilde{\mathcal{A}}^{sw})$ the $\mu^{\times}_{\mathcal{AC}}$ alter-ego of $\sigma_{|\mathcal{A}, \vec{\zeta}[:i]}$.

(*) Since $\mathcal{A} \leq^{S,f}_{0} \mathcal{B}$, $\exists \sigma^{d}_{|\mathcal{B}, \vec{\zeta}[:i]} \in S(\mathcal{E}^{(\vec{\zeta},i)}||\tilde{\mathcal{B}}^{sw})$ balanced with $\sigma^{c}_{|\mathcal{A}, \vec{\zeta}[:i]}$, i.e.

- $\forall \zeta' \in range(f_{(\mathcal{E}^{i},\tilde{\mathcal{A}}^{sw})}) \cup range(f_{(\mathcal{E}^{i},\tilde{\mathcal{B}}^{sw})})$, $\sigma^{c}_{|\mathcal{A}, \vec{\zeta}[:i]}(C_{\check{\mathcal{C}}^{\zeta'}_{\mathcal{A}}}) = \sigma^{d}_{|\mathcal{B}, \vec{\zeta}[:i]}(C_{\check{\mathcal{C}}^{\zeta'}_{\mathcal{B}}})$
- where $\check{\mathcal{C}}^{\zeta'}_{\mathcal{A}} = Class(\mathcal{E}^{i}, \tilde{\mathcal{A}}^{sw}, f, \zeta')$ and $\check{\mathcal{C}}^{\zeta'}_{\mathcal{B}} = Class(\mathcal{E}^{i}, \tilde{\mathcal{B}}^{sw}, f, \zeta')$

We note $\sigma'_{|\mathcal{B}, \vec{\zeta}[:i]}$ the $\mu^{\times}_{\mathcal{CB}}$ alter-ego of $\sigma^{d}_{|\mathcal{B}, \vec{\zeta}[:i]}$.

We build $\tilde{\sigma}' \in S(\tilde{\mathcal{E}}||X_{\mathcal{B}})$ as follows:

For every $\tilde{\zeta} \in range(f_{(\tilde{\mathcal{E}},X_{\mathcal{B}})}) \setminus range(f_{(\tilde{\mathcal{E}},X_{\mathcal{A}})})$, $\forall \vec{\zeta} \in \overrightarrow{proxy}(\tilde{\zeta})_{(\tilde{\mathcal{E}},X_{\mathcal{B}},\mathcal{B})}$, $\forall i \in [1 : |\vec{\zeta}|]$, we require that $\tilde{\sigma}_{|\mathcal{B}, \vec{\zeta}[:i]}$ halts (i.e. $\forall \tilde{\alpha}', f^{\mathcal{B},proxy}_{(\tilde{\mathcal{E}},X_{\mathcal{B}})}(\tilde{\alpha}') = \vec{\zeta}[: i]$, $supp(\tilde{\sigma}_{|\mathcal{B}, \vec{\zeta}[:i]}(\tilde{\alpha}')) = \emptyset$).

For every $\tilde{\zeta} \in range(f_{(\tilde{\mathcal{E}},X_{\mathcal{A}})}) \cup range(f_{(\tilde{\mathcal{E}},X_{\mathcal{B}})})$, $\forall \vec{\zeta} \in \overrightarrow{proxy}(\tilde{\zeta})_{(\tilde{\mathcal{E}},X_{\mathcal{B}},\mathcal{B})}$, $\forall i \in [1 : |\vec{\zeta}|]$, we require that $\tilde{\sigma}_{|\mathcal{B}, \vec{\zeta}[:i]}$ and $\sigma'_{|\mathcal{B}, \vec{\zeta}[:i]}$ are $((\tilde{\mathcal{E}}||X_{\mathcal{B}})^{\downarrow \vec{\zeta}[:i]}).\tilde{\mu}^{\mathcal{B},+}_{e}$ alter-ego.

Let $\tilde{\zeta} \in range(f_{(\tilde{\mathcal{E}},X_{\mathcal{A}})}) \cup range(f_{(\tilde{\mathcal{E}},X_{\mathcal{B}})})$, let $\vec{\zeta} \in \overrightarrow{proxy}(\tilde{\zeta})_{(\tilde{\mathcal{E}},X_{\mathcal{B}},\mathcal{B})}$ For every $i \in [1 : |\vec{\zeta}|]$ $\tilde{\pi}', \tilde{\pi}'' \in (f^{\mathcal{B},proxy}_{(\tilde{\mathcal{E}},X_{\mathcal{B}})})^{-1}(\vec{\zeta}[: i])$, $lstate(\tilde{\pi}') = lstate(\tilde{\pi}') \triangleq q^{i-1}_{\ell}$ since f is pasting-friendly. We note $\mathcal{E}'^{(\vec{\zeta},i)} = \mathcal{E}_{\bar{q}_{\mathcal{E}} \rightarrow \mu_{\mathcal{BC}}(q^{i-1}_{\ell} \upharpoonright \mathcal{E}_{\mathcal{B}})}$. Moreover, $\mathcal{E}'^{(\vec{\zeta}',i)} = \mathcal{E}^{(\vec{\zeta},i)}$ for every pair $(\vec{\zeta}, \vec{\zeta}')$, s.t. $\mu^{\times}_{\mathcal{AC}}(\vec{\zeta}) = \mu^{\times}_{\mathcal{BC}}(\vec{\zeta}')$.

Now we show that $\tilde{\sigma}$ and $\tilde{\sigma}'$ are balanced:

Let $\tilde{\zeta} \in range(f_{(\tilde{\mathcal{E}},X_{\mathcal{A}})}) \cup range(f_{(\tilde{\mathcal{E}},X_{\mathcal{B}})})$, $(\tilde{\zeta} \in Execs(\tilde{\mathcal{E}}))$. Let

- $\tilde{\mathcal{C}}^{\tilde{\zeta}}_{\mathcal{A}} = Class(\tilde{\mathcal{E}}, X_{\mathcal{A}}, f, \tilde{\zeta})$ and
- $\tilde{\mathcal{C}}^{\tilde{\zeta}}_{\mathcal{B}} = Class(\tilde{\mathcal{E}}, X_{\mathcal{B}}, f, \tilde{\zeta})$

.

We need to show that $\epsilon_{\tilde{\sigma}}(C_{\tilde{\mathcal{C}}^{\tilde{\zeta}}_{\mathcal{A}}}) = \epsilon_{\tilde{\sigma}'}(C_{\tilde{\mathcal{C}}^{\tilde{\zeta}}_{\mathcal{B}}})$:

We apply lemma 187 to obtain:

- $\epsilon_{\tilde{\sigma}}(C_{\tilde{\mathcal{C}}^{\tilde{\zeta}}_{\mathcal{A}}}) = \sum_{\vec{\zeta}_a \in \overrightarrow{proxy}(\tilde{\zeta})_{(\tilde{\mathcal{E}},X_{\mathcal{A}},\mathcal{A})}} \epsilon_{\tilde{\sigma}}(C_{\bigotimes^{n}_{i} \hat{\mathcal{C}}^{\vec{\zeta}_a[i]}_{\mathcal{A}}})$.
- $\epsilon_{\tilde{\sigma}'}(C_{\tilde{\mathcal{C}}^{\tilde{\zeta}}_{\mathcal{B}}}) = \sum_{\vec{\zeta}_b \in \overrightarrow{proxy}(\tilde{\zeta})_{(\tilde{\mathcal{E}},X_{\mathcal{B}},\mathcal{B})}} \epsilon_{\tilde{\sigma}'}(C_{\bigotimes^{n}_{i} \hat{\mathcal{C}}^{\vec{\zeta}_b[i]}_{\mathcal{B}}})$.

Since $\mathcal{E}_{\mathcal{A}}$ and $\mathcal{E}_{\mathcal{B}}$ are semantically equivalent, the sets $\{\zeta^{a} \in Execs(\mathcal{E}_{\mathcal{A}}) | \zeta^{a} \upharpoonright \tilde{\mathcal{E}} = \tilde{\zeta}\}$ and $\{\zeta^{b} \in Execs(\mathcal{E}_{\mathcal{B}}) | \zeta^{b} \upharpoonright \tilde{\mathcal{E}} = \tilde{\zeta}\}$ are in bijection. Hence, it is enough to show that $\forall (\zeta^{ac}, \zeta^{bc}) \in Execs(\mathcal{E}_{\mathcal{A}}) \times Execs(\mathcal{E}_{\mathcal{B}})$ with $\zeta^{bc} = \mu_{\mathcal{AC}} \circ \mu_{\mathcal{CB}}(\zeta^{ac})$ and $\zeta^{bc} \upharpoonright \tilde{\mathcal{E}} = \zeta^{ac} \upharpoonright \tilde{\mathcal{E}} = \tilde{\zeta}$, for $\vec{\zeta}^{ac} \overset{n}{\underset{\mathcal{A}}{\vdots}} \zeta^{ac}$, $\vec{\zeta}^{bc} \overset{n}{\underset{\mathcal{A}}{\vdots}} \zeta^{bc}$, then $\epsilon_{\tilde{\sigma}}(C_{\bigotimes^{n}_{i} \hat{\mathcal{C}}^{\vec{\zeta}^{ac}[i]}_{\mathcal{A}}}) = \epsilon_{\tilde{\sigma}'}(C_{\bigotimes^{n}_{i} \hat{\mathcal{C}}^{\vec{\zeta}^{bc}[i]}_{\mathcal{B}}})$.

By definition, $\tilde{\sigma}$ is $\mathcal{A}$-creation-oblivious, and by construction, $\tilde{\sigma}'$ is $\mathcal{B}$-creation-oblivious. This allows us to apply lemma 190 to obtain:

- $\epsilon_{\tilde{\sigma}}(C_{\bigotimes^{n}_{i} \hat{\mathcal{C}}^{\vec{\zeta}^{ac}[i]}_{\mathcal{A}}}) = \prod^{n}_{i} \epsilon_{\tilde{\sigma}^{i}}(C_{\hat{\mathcal{C}}^{\vec{\zeta}^{ac}[i]}_{\mathcal{A}}})$ with $\forall i \in [1, n], \tilde{\sigma}^{i} = oblivious_{\mathcal{A}, \vec{\zeta}^{ac}[:i]}(\tilde{\sigma}) = \tilde{\sigma}_{|\mathcal{A}, \vec{\zeta}^{ac}[:i]}$.

- $\epsilon_{\tilde{\sigma}'}(C_{\bigotimes_i^n \hat{\mathcal{C}}_{\mathcal{B}}^{\vec{\zeta}^{bc}[i]}}) = \prod_i^n \epsilon_{\tilde{\sigma}'^i}(C_{\hat{\mathcal{C}}_{\mathcal{B}}^{\vec{\zeta}^{ac}[i]}})$ with $\forall i \in [1,n], \tilde{\sigma}'^i = oblivious_{\mathcal{B},\vec{\zeta}^{bc}[:i]}(\tilde{\sigma}') = \tilde{\sigma}_{|\mathcal{B},\vec{\zeta}^{bc}[:i]}$.
- where $\vec{z}[:i] = \vec{z}[1]^\frown ... ^\frown \vec{z}[i-1]$ for $\vec{z} \in \{\vec{\zeta}^{ac}, \vec{\zeta}^{bc}\}$
- $\hat{\mathcal{C}}_{\mathcal{A}}^{\vec{\zeta}^{ac}[i]} = Class((\tilde{\mathcal{E}}||X_{\mathcal{A}})^{\downarrow \vec{e}^{ac}[:i]}), f^{\mathcal{A},proxy}, \vec{\zeta}^{ac}[i])$
- $\hat{\mathcal{C}}_{\mathcal{B}}^{\vec{\zeta}^{bc}[i]} = Class((\tilde{\mathcal{E}}||X_{\mathcal{B}})^{\downarrow \vec{e}^{bc}[:i]}), f^{\mathcal{B},proxy}, \vec{\zeta}^{bc}[i])$

Thus it is enough to show that $\forall i \in [1,n]$, $\epsilon_{\tilde{\sigma}^i}(C_{\hat{\mathcal{C}}_{\mathcal{A}}^{\vec{\zeta}^{ac}[i]}}) = \epsilon_{\tilde{\sigma}'^i}(C_{\hat{\mathcal{C}}_{\mathcal{B}}^{\vec{\zeta}^{bc}[i]}})$. Let $i \in [1,n]$

By lemma 174 combined with lemma 176, we obtain:

- $\epsilon_{\tilde{\sigma}^i}(C_{\hat{\mathcal{C}}_{\mathcal{A}}^{\vec{\zeta}^{ac}[i]}}) = \epsilon_{\sigma_{|\mathcal{A},\vec{\zeta}^{ac}[:i]}}(\check{\mathcal{C}}^{(\vec{\zeta}^{ac}[i])}_{(\mathcal{E}_{\mathcal{A}},\mathcal{A})})$
- $\epsilon_{\tilde{\sigma}'^i}(C_{\hat{\mathcal{C}}_{\mathcal{B}}^{\vec{\zeta}^{bc}[i]}}) = \epsilon_{\sigma'_{|\mathcal{B},\vec{\zeta}^{bc}[:i]}}(\check{\mathcal{C}}^{(\vec{\zeta}^{bc}[i])}_{(\mathcal{E}_{\mathcal{B}},\mathcal{B})}$.

where:

- $\check{\mathcal{C}}^{(\vec{\zeta}^{ac}[i])}_{(\mathcal{E}_{\mathcal{A}},\mathcal{A})} = Class(\mathcal{E}_{\mathcal{A}}^{\vec{\zeta}^{ac}[:i]}, \tilde{\mathcal{A}}^{sw}, f, \vec{\zeta}^{ac}[i])$ and
- $\check{\mathcal{C}}^{(\vec{\zeta}^{bc}[i])}_{(\mathcal{E}_{\mathcal{B}},\mathcal{B})} = Class(\mathcal{E}_{\mathcal{B}}^{\vec{\zeta}^{bc}[:i]}, \tilde{\mathcal{B}}^{sw}, f, \vec{\zeta}^{bc}[i])$
- $\sigma_{|\mathcal{A},\vec{\zeta}^{ac}[:i]}$ is the $((\tilde{\mathcal{E}}||X_{\mathcal{A}})^{\downarrow \vec{\zeta}^{ac}[:i]}).\tilde{\mu}_e^{\mathcal{A},+}$ alter-ego of $\tilde{\sigma}^i$.
- $\sigma'_{|\mathcal{B},\vec{\zeta}^{bc}[:i]}$ is the $((\tilde{\mathcal{E}}||X_{\mathcal{B}})^{\downarrow \vec{\zeta}^{bc}[:i]}).\tilde{\mu}_e^{\mathcal{B},+}$ alter-ego of $\tilde{\sigma}'^i$.

.

Hence it is sufficient to show that $\epsilon_{\sigma_{|\mathcal{A},\vec{\zeta}^{ac}[:i]}}(C_{\check{\mathcal{C}}^{(\vec{\zeta}^{ac}[i])}_{(\mathcal{E}_{\mathcal{A}},\mathcal{A})}}) = \epsilon_{\sigma'_{|\mathcal{B},\vec{\zeta}^{bc}[:i]}}(C_{\check{\mathcal{C}}^{(\vec{\zeta}^{bc}[i])}_{(\mathcal{E}_{\mathcal{B}},\mathcal{B})}})$.

Finally, we find again our construction (*):

- $\epsilon_{\sigma_{|\mathcal{A},\vec{\zeta}^{ac}[:i]}}(C_{\check{\mathcal{C}}^{(\vec{\zeta}^{ac}[i])}_{(\mathcal{E}_{\mathcal{A}},\mathcal{A})}}) = \epsilon_{\sigma^c_{|\mathcal{A},\vec{\zeta}[:i]}}(C_{\check{\mathcal{C}}^{(\vec{\zeta}[i])}_{(\mathcal{E},\mathcal{A})}})$
- $\epsilon_{\sigma'_{|\mathcal{B},\vec{\zeta}^{bc}[:i]}}(C_{\check{\mathcal{C}}^{(\vec{\zeta}^{bc}[i])}_{(\mathcal{E}_{\mathcal{B}},\mathcal{B})}}) = \epsilon_{\sigma^d_{|\mathcal{B},\vec{\zeta}[:i]}}(C_{\check{\mathcal{C}}^{(\vec{\zeta}[i])}_{(\mathcal{E},\mathcal{B})}})$
- $\epsilon_{\sigma^c_{|\mathcal{A},\vec{\zeta}[:i]}}(C_{\check{\mathcal{C}}^{(\vec{\zeta}[i])}_{(\mathcal{E},\mathcal{A})}}) = \epsilon_{\sigma^d_{|\mathcal{B},\vec{\zeta}[:i]}}(C_{\check{\mathcal{C}}^{(\vec{\zeta}[i])}_{(\mathcal{E},\mathcal{B})}})$

where:

- $\vec{e}$ is the vector of $(Frags^*(\mathcal{E}))^n$ s.t. $\forall j \in [1:n]$, $\vec{\zeta}[j] = \mu_{\mathcal{AC}}(\vec{\zeta}^{ac}[j]) = \mu_{\mathcal{CB}}^{-1}(\vec{\zeta}^{bc}[j])$.
- $\check{\mathcal{C}}^{(\vec{\zeta}[i])}_{(\mathcal{E},\mathcal{A})} = Class(\mathcal{E}^{\vec{\zeta}[:i]}), \tilde{\mathcal{A}}^{sw}, f, \vec{\zeta}[i])$ and
- $\check{\mathcal{C}}^{(\vec{\zeta}[i])}_{(\mathcal{E},\mathcal{B})} = Class(\mathcal{E}^{\vec{\zeta}[:i]}), \tilde{\mathcal{B}}^{sw}, f, \vec{\zeta}[i])$

.

This leads us to $\epsilon_{\sigma_{|\mathcal{A},\vec{\zeta}^{ac}[:i]}}(C_{\check{\mathcal{C}}^{(\vec{\zeta}^{ac}[i])}_{(\mathcal{E}_{\mathcal{A}},\mathcal{A})}}) = \epsilon_{\sigma'_{|\mathcal{B},\vec{\beta}[:i],\vec{\zeta}^{bc}[:i]}}(C_{\check{\mathcal{C}}^{(\vec{\zeta}^{bc}[i])}_{(\mathcal{E}_{\mathcal{B}},\mathcal{B})}})$, which ends the proof.

◀

15 Task schedule

We have shown in previous section that $\leq_0^{CrOb,proj}$ was a monotonic relationship. In this section, we explain why, without cautious modifications, an easy to use off-line scheduler introduced by Canetti & al. [5], so-called task-scheduler, is not a priori creation-oblivious which surprisingly prevents us from obtaining monotonicity of the implementation relation w.r.t. PSIOA creation for this scheduler schema.

15.1 Discussion on adaptation of task-structure in dynamic setting

We adapt the task structure of [3] to dynamic setting. For any PSIOA $\mathcal{A} = (Q_\mathcal{A}, \bar{q}_\mathcal{A}, sig(\mathcal{A}), D_\mathcal{A})$, we note $acts(\mathcal{A}) = \bigcup_{q \in Q_\mathcal{A}} sig(\mathcal{A})(q)$, $UI(\mathcal{A}) = \bigcup_{q \in Q_\mathcal{A}} in(\mathcal{A})(q)$, $UO(\mathcal{A}) = \bigcup_{q \in Q_\mathcal{A}} out(\mathcal{A})(q)$, $UH(\mathcal{A}) = \bigcup_{q \in Q_\mathcal{A}} int(\mathcal{A})(q)$, $UL(\mathcal{A}) = \bigcup_{q \in Q_\mathcal{A}} \widehat{local}(\mathcal{A})(q)$, $UE(\mathcal{A}) = \bigcup_{q \in Q_\mathcal{A}} \widehat{ext}(\mathcal{A})(q)$.

In classic $PIOA$ formalism [20], if an action $a \in O_\mathcal{A} \cap I_\mathcal{B}$ is an output action for $\mathcal{A}$ and an input action for $\mathcal{B}$, then a is an output for $\mathcal{A}||\mathcal{B}$ and this does not depend on the current state of $\mathcal{A}||\mathcal{B}$.

In $PSIOA$, if an action $a \in UO(\mathcal{A}) \cap UI(\mathcal{B})$ is an output action for $\mathcal{A}$ at a certain state $q_\mathcal{A}$, without being an input action of $\mathcal{A}$ at any other state, while this is an input action for $\mathcal{B}$ at some state $q_\mathcal{B}$, without being an output action of $\mathcal{B}$ at another state, then it does not say that a will never be an input of $\mathcal{A}||\mathcal{B}$ at a certain state $q' = (q'_\mathcal{A}, q'_\mathcal{B})$ where $a \in in(\mathcal{B})(q'_\mathcal{B})$ but $a \notin out(\mathcal{A})(q'_\mathcal{A})$.

To summerize, if an action can clearly and definitely be an input or an ouput in PIOA formalism [20], this is not the case in PSIOA formalism where an action can be an input and becomes an output an vice-versa.

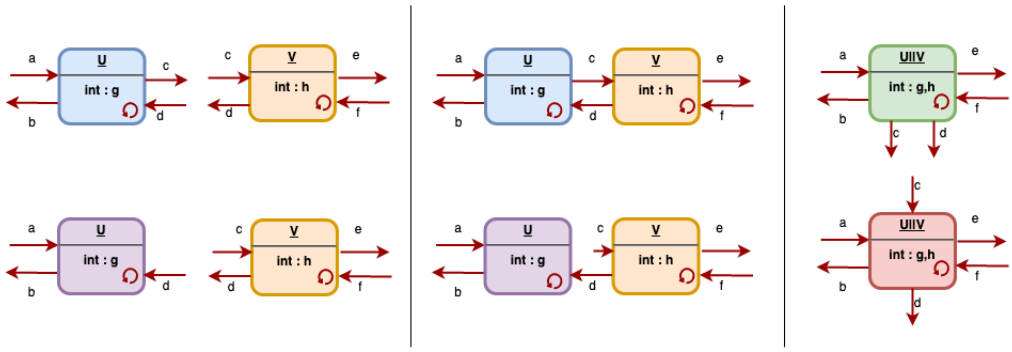


Figure 28 We represents the composition $W = U||V$ of two automata U and V. At two different states $q_W = (q_U, q_V)$ and $q'_W = (q'_U, q'_V)$ where $sig(U)(q'_U) = (in(U)(q_U), out(U)(q_U) \setminus \{c\}, int(U)(q'_U))$. The different states are represented with different colors. The action c is an output of W in q_W but an input of W' in q'_W.

In [3], a task-structure $\mathcal{R}_\mathcal{A}$ of a PIOA $\mathcal{A}$ is an equivalence class on local actions of $\mathcal{A}$ and a task-schedule is a sequence of tasks. The task-structure is assumed to ensure *next-action determinism*, that is for each state $q \in Q_\mathcal{A}$, for each task $T \in \mathcal{R}_\mathcal{A}$, there exists at most one (local) action $a \in T \cap local(\mathcal{A})(q)$ enabled in q. A task-schedule can hence "resolve the non-determinism", leading to a unique probabilistic measure on the executions. A nice property is that next-action determinism is preserved by composition if the task-structure $\mathcal{R}$ of the parallel composition of task-PIOA $(\mathcal{A}, \mathcal{R}_\mathcal{A})$ and $(\mathcal{B}, \mathcal{R}_\mathcal{B})$ is defined as $\mathcal{R} = \mathcal{R}_\mathcal{A} \cup \mathcal{R}_\mathcal{B}$

In PSIOA formalism, the preservation of well-formdness after composition is less obvious. If we assume that a task is a set of actions ensuring (local action determinism) (that is for each state $q \in Q_{\mathcal{A}}$, for each task $T \in \mathcal{R}_{\mathcal{A}}$, at most one local action $a \in T$ is enabled in q), this property will not be preserved by the composition. Indeed let imagine PISOA $\mathcal{A}$, $\mathcal{B}$, $(q_{\mathcal{A}}, q_{\mathcal{B}}) \in Q_{\mathcal{A}} \times Q_{\mathcal{B}}$ with $sig(\mathcal{A})(q_{\mathcal{A}}) = (\{a\}, \{b\}, \emptyset)$, $sig(\mathcal{B})(q_{\mathcal{B}}) = (\emptyset, \{a\}, \emptyset)$ and $T = \{a, b\}$ is a task of $\mathcal{A}$. Then $sig(\mathcal{A}||\mathcal{B})(q_{\mathcal{A}}, q_{\mathcal{B}}) = (\emptyset, \{a, b\}, \emptyset)$ and both a and b can be enabled.

This observation motivates an additional assumption, called *input partitioning*. We assume the existence of a set of "atomic entities" $Autids_0 \subset Autids$, s.t. for every $\mathcal{A} \in Autids_0$, every action $a \in acts(\mathcal{A})$, $a \in UI(\mathcal{A}) \implies a \notin UO(\mathcal{A})$. Since the vocation of an input a of $\mathcal{A}$ is to be triggered as an output action of a compatible automaton $\mathcal{B}$, this assumption is very conservative. Furthermore, in [2], the composition is defined for automata where all the states are compatible. Hence nothing is lost compared to the formalisation of [2]. Now, we can assume that, for every $\mathcal{A} \in Autids_0$, for every action $a \in UI(\mathcal{A})$, for every task T of $\mathcal{A}$, $a \notin T$.

This assumption is not preserved by the composition. Indeed, if a is an output of $\mathcal{A} \subset Autids_0$ and an input of $\mathcal{B} \subset Autids_0$, we can have a task $T = \{a\}$ of $\mathcal{A}$, that would become a task of $\mathcal{A}||\mathcal{B}$, where a can be an input of $\mathcal{A}||\mathcal{B}$. In fact we will assume both input partitioning for $Autids_0$ and local action determinism and we will show that local action determinism is ensured by any PSIOA or PCA built with atomic elements of $Autids_0$.

Another subtlety appears. In static setting, since the signature is unique and compatibility of $\mathcal{A}$ and $\mathcal{B}$ means $UL(\mathcal{A}) \cap UL(\mathcal{B}) = \emptyset$, there is no ambiguity in defining a subset of tasks $\underline{T}' = \{T_{k'}\}_{k' \in K'}$ among the ones of $\mathcal{A}||\mathcal{B}$ composed uniquely of tasks of $\mathcal{A}$ (or $\mathcal{B}$ symetrically). In dynamic setting if a task T is only a set of action labels, T could be a task for different automata (not a the same time). For example, T could be triggered by the $\mathcal{A}$ "contribution" of $\mathcal{A}||\mathcal{B}$ or by the $\mathcal{B}$ "contribution" of $\mathcal{A}||\mathcal{B}$ in alternative execution branches. The confusion can become much greater for a configuration automaton X (formalised in section 4) where each state points to a configuration of dynamic set $\mathbf{A}_X$ of automata (with their own current state). What if the scheduler proposes a task T to a configuration automaton X that goes successively into states q_X and q'_X pointing to configuration C_X and C'_X with different set of automata $\mathbf{A}_X$ and $\mathbf{A}'_X$ where $\mathcal{B} \in \mathbf{A}_X$ and is in its current state $q_{\mathcal{B}}$ and $\mathcal{B}' \in \mathbf{A}'_X$ and is in its current state $q_{\mathcal{B}'}$ with $\mathcal{B} \neq \mathcal{B}'$ but $\widehat{loc}(\mathcal{B})(q_{\mathcal{B}}) \cap \widehat{loc}(\mathcal{B}')(q_{\mathcal{B}'}) \cap T \neq \emptyset$? There are a lot of different ways to deal with this source of ambiguity. To solve it, we have two motivations:

- Reuse the notion of projection of a schedule on an environment as in [5]
- Obtain our theorem of monocity,. To do so, we need to avoid that a task T that was intented to be triggered by an automaton $\mathcal{A}$ in a certain execution branch α and ignored in another branch α' can be triggered by another automata $\mathcal{A}'$ in an execution branch $\tilde{\alpha}'$ with $trace(\alpha') = trace(\tilde{\alpha}')$ of a configuration automaton X that creates $\mathcal{A}'$ instead of $\mathcal{A}$.

The monocity theorem is based on the fact that $X_{\mathcal{A}}||\mathcal{E}$ mimics the behaviour of $\tilde{\mathcal{A}}^{sw}||\mathcal{E}''_{\mathcal{A}}$ with $\mathcal{E}''_{\mathcal{A}} = X_{\mathcal{A}} \setminus \{\mathcal{A}\}||\mathcal{E}$ where $\tilde{\mathcal{A}}^{sw}$ is the simpleton wrapper of $\mathcal{A}$ (formalised in definition 123) and $X_{\mathcal{A}} \setminus \{\mathcal{A}\}$ (formalised in definition 120) is the PCA $X_{\mathcal{A}}$ deprived of $\mathcal{A}$ at each configuration (see figures 29 and 30). If we examine the succession of reduced configurations (configuration without automata with empty signature) visited in $\tilde{\alpha} \in Execs(X_{\mathcal{A}}||\mathcal{E})$ and in corresponding $\alpha \in Execs(\mathcal{A}||\mathcal{E}''_{\mathcal{A}})$, $\alpha = \mu_e^{\mathcal{A}}(\tilde{\alpha})$, we obtain the same ones (see figure 31). Since our theorem takes advantage of the corresponding successions of configurations, it is natural to make appear the ids of $Autids_0$, representing the "atomic" entities among all the entities.

This formalism avoid the possibility for an atomic entity $\mathcal{A}$ to be a "member" of two different hierachy as it was already the case in [2] which is completely normal in IO automata formalism. However, contrary to [2], the notion of partial-compatibility does not prevent an automaton $\mathcal{A}$ to move from a configuration X to another configuration Y. Indeed we can

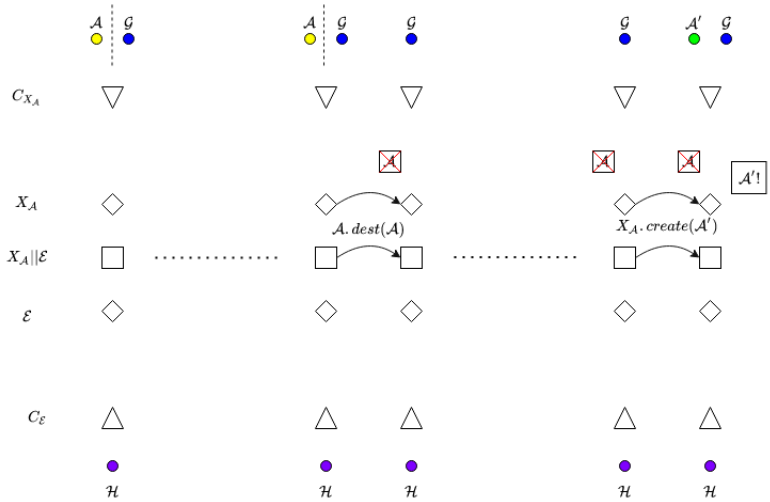


Figure 29 An example of an execution $\tilde{\alpha}$ of a probabilistic configuration automata (PCA) $X_\mathcal{A}||\mathcal{E}$. At first, $\mathcal{A}$ is a "member" (yellow dot) of $X_\mathcal{A}$, then it is destroyed and finally a clone $\mathcal{A}'$ is created (green dot) in $X_\mathcal{A}$. The formalism of [2] allows that $\mathcal{A}$ and $\mathcal{A}'$ are "member" of $X_\mathcal{A}$ in two different states as long as they cannot be member in the same state.

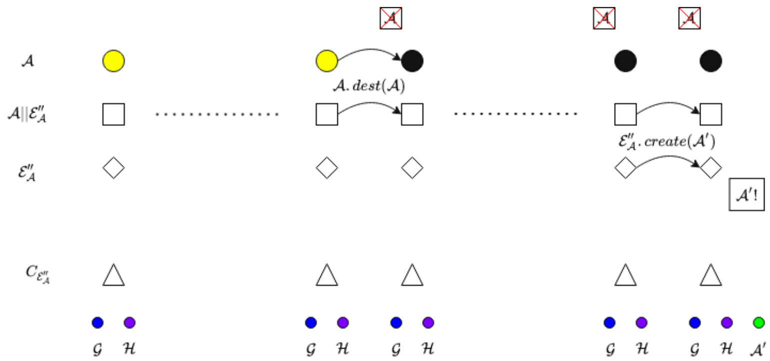


Figure 30 The corresponding execution α of $\mathcal{A}||\mathcal{E}''_\mathcal{A}$, noted $\alpha = \mu_e^\mathcal{A}(\tilde{\alpha})$. At first, $\mathcal{A}$ is "alive" (yellow dot), then it goes forever into a "zombie state" $q^\phi_\mathcal{A}$ (black dot) where $\widehat{sig}(\mathcal{A})(q^\phi_\mathcal{A}) = \emptyset$. Finally a clone $\mathcal{A}'$ is created (green dot) in $\mathcal{E}''_\mathcal{A}$. The formalism of [2] is not supposed to allow this composition since among all the states of $Q_\mathcal{A} \times Q_{\mathcal{E}''_\mathcal{A}}$, some of them are not compatible. However, it is possible to extend their formalism and define a partial-compatibility where all *reachable* states of $Q_\mathcal{A} \times Q_{\mathcal{E}''_\mathcal{A}}$ are compatible.

imagine X and Y that create and destroy $\mathcal{A}$ so that they are partially-compatible (while they cannot be compatible). Neverteless, this possibility will not be handled by our theorem of monocity, since $\mathcal{A}$, even in its zombie state, cannot be partially-compatible with a PCA $\mathcal{E}$ that creates $\mathcal{A}$. Here again, we do not lose any expressiveness compared to the original work of [2]. We can remark we are not dealing with a schedule *of a specific automaton* anymore, which differs from [5]. However the restriction of our definition to "static" setting, where each automaton is the composition of a finite set of automata in $Autids_0$, matches their definition. It will be the responsibility of the task-scheduler to chose a task-schedule $\rho = T_1, ..., T_k, ...$ that produces the probabilistic distribution that it wants.

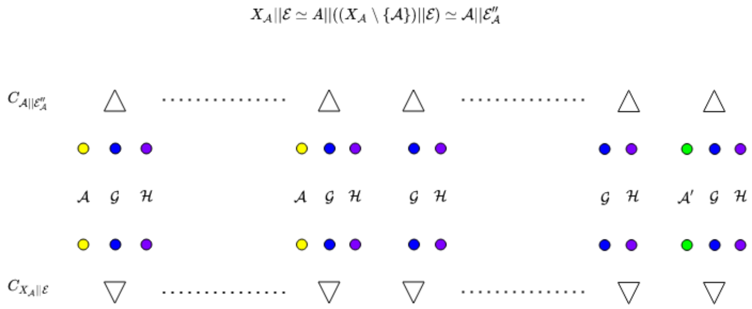


Figure 31 As long as no creation of $\mathcal{A}$ occurs, the executions $\tilde{\alpha} \in Execs(X_{\mathcal{A}}||\mathcal{E})$ and $\alpha \in Execs(\mathcal{A}||\mathcal{E}''_{\mathcal{A}})$ handle the same succession of reduced configurations.

According to our understanding, the fact that the set of tasks is not a set of equivalence classes for an equivalence relation is not crucial for the model.

15.2 task-schedule for dynamic setting

We formalise the scheduler schema of *task-schedulers* that is a schema of off-line schedulers.

We assume the existence of a subset $Autids_0 \subset Autids$ that represents the "atomic entities" of our formalism. Any automaton is the result of the composition of automata in $Autids_0$.

▶ **Definition 192** (Constitution). *For every PSIOA or PCA $\mathcal{A}$, we note*

$$constitution(\mathcal{A}) : \begin{cases} Q_{\mathcal{A}} & \rightarrow \quad \mathcal{P}(Autids_0) \text{ where } \mathcal{P}(Autids_0) \text{ denotes the power set of } Autids_0 \\ q & \mapsto \quad constitution(\mathcal{A})(q) \end{cases}$$

The function constitution is defined as follows:

- *for every PSIOA $\mathcal{A} \in Autids_0$, $\forall q \in Q_{\mathcal{A}}$, $constitution(\mathcal{A})(q) = \{\mathcal{A}\}$.*
- *for every finite set of partially-compatible PSIOA $\mathbf{A} = \{\mathcal{A}_1, ..., \mathcal{A}_n\} \in (Autids_0)^n$, $\forall q \in Q_{\mathbf{A}}$, $constitution(\mathcal{A}_1||...||\mathcal{A}_n)(q) = \mathbf{A}$.*
- *The constitution of a PCA is defined recursively through its configuration. For every PCA X, $\forall q \in Q_X$, if we note $(\mathbf{A}, \mathbf{S}) = config(X)(q)$, $constitution(X)(q) = \bigcup_{\mathcal{A} \in \mathbf{A}} constitution(\mathcal{A})(\mathbf{S}(\mathcal{A}))$.*

We can extend the principle of a partial function *map* (attached to a configuration) to the entire constitution of a PCA or PSIOA.

▶ **Definition 193** (hierarchy mapping S^H). *Let X be a PCA or a PSIOA. Let $q \in Q_X$ We note $\mathbf{S}^H(X)(q)$ [6] the function that maps any PSIOA $\mathcal{A}_i \in constitution(X)(q)$ to a state $q_{\mathcal{A}_i} \in Q_{\mathcal{A}_i}$ s.t.*

- *if $X = \mathcal{A}_i$, $q_{\mathcal{A}_i} = q$*
- *if $X = \mathcal{A}_1||...||\mathcal{A}_i||...||\mathcal{A}_n$ and $q = (q_1, ..., q_i, ..., q_n) \in Q_{\mathcal{A}_1}||...||\mathcal{A}_i||...||\mathcal{A}_n)$, $q_{\mathcal{A}_i} = q_i$*
- *if X is a PCA, $q_{\mathcal{A}_i} = S^H(Y)(q_Y)$ where Y is the unique member of $auts(config(X)(q))$ s.t. $\mathcal{A}_i \in constitution(Y)(q_Y)$ with $q_Y = map(config(X)(q))(Y)$*

Anticipating the definition of an enabled task, we extend the definition of task of [3] with an id of $Autids_0$.

[6] H stands for "hierarchy" and $\mathbf{S}$ refers to notation of mapping function of a configuration $(\mathbf{A}, \mathbf{S})$.

▶ **Definition 194** (Task). *A task T is a pair $(id, actions)$ where $id \in Autids_0$ and $actions \subset acts(aut(id))$ is a set of action labels. Let $T = (id, actions)$, we note $id(T) = id$ and $actions(T) = actions$.*

Now, we are ready to define notion of enabled task.

▶ **Definition 195** (Enabled task). *Let X be a PSIOA or a PCA. A task T is said* enabled *in state $q \in Q_X$ if*

- *$id(T) \in constitution(X)(q)$*
- *it exists a unique local action $a \in \widehat{loc}(\mathcal{A})(q_{\mathcal{A}_i}) \cap actions(T)$ enabled at state $S^H(X)(q)(\mathcal{A})$*[7].

All previous precautions allow us to define a task-schedule, which is a particular subclass of schedulers, avoiding the technical problems mentioned in previous subsection. We are not dealing with a task-schedule *of a specific automaton* anymore, which differs from [3]. However the restriction of our definition to "static" setting matches their definition.

▶ **Definition 196** (task-schedule). *A task-schedule $\rho = T_1, T_2, T_3, ...$ is a (finite or infinite) sequence of tasks.*

Since our task-schedule is defined, we are ready to solve the non-determinism and define a probability on the executions of a PSIOA. We use the measure of [3].

▶ **Definition 197.** *(task-based probability on executions: $apply_{\mathcal{A}}(\mu, \rho) : Frags(\mathcal{A}) \rightarrow [0,1]$) Let $\mathcal{A}$ be a PSIOA. Given $\mu \in Disc(Frags(\mathcal{A}))$ a discrete probability measure on the execution fragments and a task schedule ρ, $apply(\mu, \rho)$ is a probability measure on $Frags(\mathcal{A})$. It is defined recursively as follows.*

1. *$apply_{\mathcal{A}}(\mu, \lambda) := \mu$. Here λ denotes the empty sequence.*
2. *For every T and $\alpha \in Frags^*(\mathcal{A})$, $apply(\mu, T)(\alpha) := p_1(\alpha) + p_2(\alpha)$, where:*
   - $p_1(\alpha) = \begin{cases} \mu(\alpha')\eta_{(\mathcal{A},q',a)}(q) & \text{if } \alpha = \alpha'^\frown(a,q), q' = lstate(\alpha') \text{ and } a \text{ is triggered by } T \text{ enabled after } \alpha' \\ 0 & \text{otherwise} \end{cases}$
   - $p_2(\alpha) = \begin{cases} \mu(\alpha) & \text{if } T \text{ is not enabled after } \alpha \\ 0 & \text{otherwise} \end{cases}$
3. *3. If ρ is finite and of the form $\rho' T$, then $apply_{\mathcal{A}}(\mu, \rho) := apply_{\mathcal{A}}(apply_{\mathcal{A}}(\mu, \rho'), T)$.*
4. *4. If ρ is infinite, let ρ_i denote the length-i prefix of ρ and let pm_i be $apply_{\mathcal{A}}(\mu, \rho_i)$. Then $apply_{\mathcal{A}}(\mu, \rho) := \lim_{i \to \infty} pm_i$.*

▶ **Proposition 198.** *Let $\mathcal{A}$ be a PSIOA, For each measure μ on $Frags^*(\mathcal{A})$ and task schedule ρ, there is scheduler σ for $\mathcal{A}$ such that $apply(\mu, \rho)$ is the generalized probabilistic execution fragment $\epsilon_{\sigma,\mu}$.*

Proof. The result has been proven in [3], appendix B.4. ◀

15.3 Why a task-scheduler is not creation-oblivious ?

Let us imagine the following example. The class C^x is composed of two executions $\alpha^{x,1}$ and $\alpha^{x,2}$, the class C^y is composed of two executions $\alpha^{y,1}$ and $\alpha^{y,2}$ and the class C^z is composed of four executions $\alpha^{z,11} = \alpha^{x,1\frown}\alpha^{y,1}$, $\alpha^{z,12} = \alpha^{x,1\frown}\alpha^{y,2}$, $\alpha^{z,21} = \alpha^{x,2\frown}\alpha^{y,1}$, $\alpha^{z,22} = \alpha^{x,2\frown}\alpha^{y,2}$. Let $\rho = \rho^{1\frown}\rho^2$ be a task-schedule. We do not have $apply(., \rho)(C^z)) =$

[7] action enabling assumption implies that $a \in \widehat{sig}(\mathcal{A}_i)(S^H(X)(q)(\mathcal{A})) \implies a$ enabled at state $S^H(X)(q)(\mathcal{A})$ (i.e. $\exists \eta \in Disc(Q_{\mathcal{A}})$ s.t. $(S^H(X)(q)(\mathcal{A}), a, \eta) \in D_{\mathcal{A}}$)

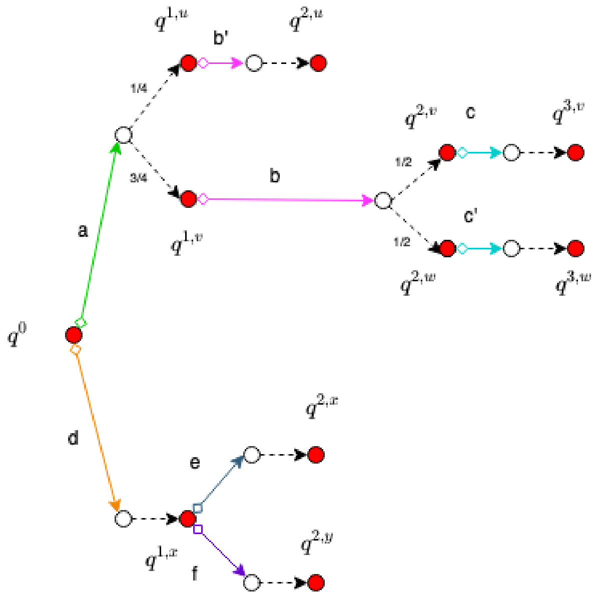


Figure 32 Non-deterministic execution: The scheduler allows us to solve the non-determinism, by triggering an action among the enabled one. We give an example with an automaton $\mathcal{A} = (Q_{\mathcal{A}}, \bar{q}_{\mathcal{A}} = q_0, sig(\mathcal{A}), D_{\mathcal{A}})$ and the tasks T_g, T_o, T_p, T_b (for green, orange, pink, blue) with the respective actions $\{a\}, \{d\}, \{b, b'\}, \{c, c'\}$, and the tasks T_{go}, T_{bo} with the respective actions $\{a, d\}, \{c, c', d\}$. At state q_0, $sig(\mathcal{A})(q_0) = (\emptyset, \{a\}, \{d\})$. Hence both a and d are enabled local action at q_0, which means both T_g and T_o are enabled at state q_0, but T_{go} is not enabled at state q_0 since it does not solve the non-determinism (a and d are enabled local action at q_0). At state q_1, T_p is enabled but neither T_o or T_b. We give some results: $apply(\delta_{q^0}, T_g)(q^0, a, q^{1,v}) = 1$
$apply(\delta_{q^0}, T_g T_p)(q^0, a, q^{1,v}, b, q^{2,w}) = apply(apply(\delta_{q^0}, T_g), T_p)(q^0, a, q^{1,v}, b, q^{2,w}) = 1/2$
$apply(\delta_{q^0}, T_g T_p T_b)(q^0, a, q^{1,v}, b, q^{2,w}, c, q^{3,w}) = apply(apply(\delta_{q^0}, T_g T_p), T_b)(q^0, a, q^{1,v}, b, q^{2,w}, c, q^{3,w}) = 3/8$
$apply(\delta_{q^0}, T_g T_p T_o T_b)(q^0, a, q^{1,v}, b, q^{2,w}, c, q^{3,w}) = 3/8$, since T_o is not enabled at state $q^{2,w}$.

$apply(., \rho^1)(C^x) \cdot apply(., \rho^2)(C^y)$! Indeed, the executions $\alpha^{x,1}$ and $\alpha^{x,2}$ can differ s.t. they do not ignore the same tasks. Typically, ρ^1 could be written $\rho^1 = \rho^{1,a} \frown \rho^{1,b}$ where the last action of $\alpha^{x,1}$ is triggered by the last task of $\rho^{1,a}$ and $\rho^{1,b}$ is "ignored by $\alpha^{x,1}$. The issue comes if both $apply(., \rho^2)(C^y) \neq \emptyset$ and $apply(., \rho^{1,b} \frown \rho^2)(C^y) \neq \emptyset$. The point is that C^z can be obtained with different cut-paste: cut-paste A: $\rho^{1,a}$ for C^x and $\rho^{1,b} \frown \rho^2$ for C^y ; cut-paste B: ρ^1 for C^x and ρ^2 for C^y.

There is room for finding the appropriate natural assumptions to obtain creation-obliviousness for task-schedules in future work.

16 Conclusion

We extended *dynamic I/O Automata* formalism of Attie & Lynch [2] to probabilistic settings in order to cope with emergent distributed systems such as peer-to-peer networks, robot networks, adhoc networks or blockchains. Our formalism includes operators for parallel composition, action hiding, action renaming, automaton creation and use a refined definition of probabilistic configuration automata in order to cope with dynamic actions. The key result of our framework is as follows: the implementation of probabilistic configuration automata is monotonic to automata creation and destruction. That is, if systems $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ differ only in that $X_{\mathcal{A}}$ dynamically creates and destroys automaton $\mathcal{A}$ instead of creating and destroying automaton $\mathcal{B}$ as $X_{\mathcal{B}}$ does, and if $\mathcal{A}$ implements $\mathcal{B}$ (in the sense they cannot be distinguished by any external observer), then $X_{\mathcal{A}}$ implements $X_{\mathcal{B}}$. This results is particularly interesting in the design and refinement of components and subsystems in isolation. In our construction we exhibit the need of considering only *creation-oblivious* schedulers in the implementation relation, i.e. a scheduler that, upon the (dynamic) creation of a sub-automaton $\mathcal{A}$, does not take into account the previous internal behaviours of $\mathcal{A}$ to output (randomly) a transition.

Interestingly and of independent interest, motivated by the monotonicity of execution w.r.t. to automata creation, we introduce new proof techniques to deduce certain properties of a system $X_{\mathcal{A}}$ from a sub-automaton $X_{\mathcal{A}}$ dynamically created and destroyed by $X_{\mathcal{A}}$. This proof technique is used to construct a homomorphism between the probabilistic spaces of automata executions. Then we expose such homomorphism from a system $X_{\mathcal{A}}$ to a new system resulting from the composition of $\mathcal{A}$ and $X_{\mathcal{A}} \setminus \{\mathcal{A}\}$. The latter corresponds intuitively to the system $X_{\mathcal{A}}$ deprived of $\mathcal{A}$. Furthermore, the homomorphism is used to show that under certain minor technical assumptions, if $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ differ only in the fact that $X_{\mathcal{A}}$ dynamically creates and destroys the automaton $\mathcal{A}$ instead of creating and destroying the automaton $\mathcal{B}$ as $X_{\mathcal{B}}$ does, then $X_{\mathcal{A}} \setminus \{\mathcal{A}\}$ and $X_{\mathcal{B}} \setminus \{\mathcal{B}\}$ are semantically equivalent, i.e. they only differ syntactically. The homomorphism is finally reused to establish the monotonicity of the implementation relation. Our technique can be used in extensions of our formalism with time and cryptography notions.

As future work we plan to extend the composable secure-emulation of Canetti et al. [5] to dynamic settings. This extension is necessary for formal verification of protocols combining probabilistic distributed systems and cryptography in dynamic settings (e.g. blockchains, secure distributed computation, cybersecure distributed protocols etc).

17 Glossary

$\mathcal{A}$	PSIOA with id $\mathcal{A}$
$(Q_{\mathcal{A}}, \mathcal{F}_{Q_{\mathcal{A}}})$	state space of $\mathcal{A}$
$\bar{q}_{\mathcal{A}}$	start state of $\mathcal{A}$
$D_{\mathcal{A}}$	discrete trandistions of $\mathcal{A}$
$steps(\mathcal{A})$	steps of $\mathcal{A}$
$sig(\mathcal{A})$	signature of $\mathcal{A}$, maps each state to a triplet
$\widehat{sig}(\mathcal{A})$	signature of $\mathcal{A}$, maps each state to the union of actions of the triplet $sig(\mathcal{A})$
$in(\mathcal{A})$	input actions of $\mathcal{A}$
$out(\mathcal{A})$	output actions of $\mathcal{A}$
$int(\mathcal{A})$	internal actions of $\mathcal{A}$
$ext(\mathcal{A})$	external actions of $\mathcal{A}$, maps each state $q \in Q_{\mathcal{A}}$ to the pair $(in(\mathcal{A})(q), out(\mathcal{A}))(q))$
$\widehat{ext}(\mathcal{A})$	external actions of $\mathcal{A}$, maps each state $q \in Q_{\mathcal{A}}$ to $in(\mathcal{A})(q) \cup out(\mathcal{A}))(q)$
$loc(\mathcal{A})$	local actions of $\mathcal{A}$, maps each state $q \in Q_{\mathcal{A}}$ to the pair $(out(\mathcal{A}))(q), int(\mathcal{A}))$
$\widehat{loc}(\mathcal{A})$	local actions of $\mathcal{A}$, maps each state $q \in Q_{\mathcal{A}}$ to $out(\mathcal{A}))(q) \cup int(\mathcal{A})$
$acts(\mathcal{A})$	universal set of actions of $\mathcal{A}$, i.e. $\bigcup_{q \in Q_{\mathcal{A}}} \widehat{sig}(\mathcal{A})$
$Execs(\mathcal{A})$	executions of $\mathcal{A}$
$Execs^*(\mathcal{A})$	finite executions of $\mathcal{A}$
$Execs^{\omega}(\mathcal{A})$	infinite executions of $\mathcal{A}$
$Frags(\mathcal{A})$	execution fragments of $\mathcal{A}$
$Frags^*(\mathcal{A})$	finite execution fragments of $\mathcal{A}$

$Frags^{\omega}(\mathcal{A})$	infinite execution fragments of $\mathcal{A}$
$Traces(\mathcal{A})$	traces of $\mathcal{A}$
$Traces^{*}(\mathcal{A})$	finite traces of $\mathcal{A}$
$Traces^{\omega}(\mathcal{A})$	infinite traces of $\mathcal{A}$
$Reachable(\mathcal{A})$	reachable states of $\mathcal{A}$
C_{α}	cone of executions with α as prefix
$trace_{\mathcal{A}}(\alpha)$	trace of execution α
$lstate(\alpha)$	last state of execution α
$fstate(\alpha)$	first state of execution α
$states(\alpha)$	set of states composing the execution α
$actions(\alpha)$	set of actions composing the execution α
$\upharpoonright$	projection for states, executions
$\leq_{\epsilon}^{S,f}$	implementation relation w.r.t. scheduler schema S, insight-function f, approximation ϵ
$\|\|$	parallel composition
$\times$	cardinal product, also used as operator of composition for signature
$\otimes$	product of measures or product of σ-algebra
Q_{conf}	set of configurations
$auts(C)$	automata of configuration C
$map(C)$	maps each automata of $auts(C)$ to its current state
$sig(C)$	signature of configuration C
$config(X)$	maps each state q to associated configurations of PCA X at state q
$created(X)(q)$	maps each action a to sub-automata created by X at state q through action a
$hidden\text{-}actions(X)$	maps each state q to hidden actions of PCA X at state q
ϵ_{σ}	measure of probability on $Execs(\mathcal{A})$ generated by scheduler σ
$env(\mathcal{A})$	set of environment of $\mathcal{A}$
$f\text{-}dist_{(\mathcal{E},\mathcal{A})}(\sigma)$	measure of probability on $f(Execs(\mathcal{E}\|\|\mathcal{A}))$ generated by scheduler σ for $\mathcal{E} \in env(\mathcal{A})$
$proj_{(.,.)}$	for each automaton K, $\forall \mathcal{E} \in env(K)$, $\forall \alpha \in Execs(\mathcal{E}\|\|K)$, $proj_{(\mathcal{E},K)}(\alpha) = \alpha \upharpoonright \mathcal{E}$
$\eta_1 \overset{c}{\leftrightarrow} \eta_2$	c is a preserving-measure bijection between distributions η_1 and η_2
$\Phi[\mathcal{B}/\mathcal{A}]$	same automata ids than in Φ, modulo $\mathcal{B}$ replacing $\mathcal{A}$
$C \lhd_{\mathcal{A}\mathcal{B}} C'$	C and C' are the same configurations modulo $\mathcal{B}$ replacing $\mathcal{A}$ in C'
$X \setminus \{\mathcal{A}\}$	PCA X deprived of $\mathcal{A}$
$qR_{conf}q'$	the states q and q' are associated to the same configuration
$qR_{conf}^{\setminus\{\mathcal{A}\}}q'$	the states q and q' are associated to configurations that are equal if we ignore $\mathcal{A}$
$qR_{strict}q'$	the states q and q' are associated to the same components of their PCA
$qR_{strict}^{\setminus\{\mathcal{A}\}}q'$	the states q and q' are associated to the same components of their PCA if we ignore $\mathcal{A}$
$pot\text{-}out(X)(\mathcal{A})(q)$	the (potential) output actions of $\mathcal{A}$ in $config(X)(q)$
$\tilde{\mathcal{A}}^{sw}$	simpleton wrapper of $\mathcal{A}$
$\alpha \equiv_{\mathcal{A}}^{cr} \alpha'$	α and α' differs only on internal states and internal actions of sub-automaton $\mathcal{A}$.

References

1 Edward A. Ashcroft. Proving assertions about parallel programs. *J. Comput. Syst. Sci.*, 10(1):110–135, 1975. `doi:10.1016/S0022-0000(75)80018-3`.

2 Paul C. Attie and Nancy A. Lynch. Dynamic input/output automata: A formal and compositional model for dynamic systems. *Inf. Comput.*, 249:28–75, 2016. `doi:10.1016/j.ic.2016.03.008`.

3 Ran Canetti, Ling Cheung, Dilsun Kaynar, Moses Liskov, Nancy Lynch, Olivier Pereira, and Roberto Segala. Task-Structured Probabilistic {I/O} Automata. *Journal of Computer and System Sciences*, 94:63—-97, 2018. URL: `https://doi.org/10.1016/j.jcss.2017.09.007`.

4 Ran Canetti, Ling Cheung, Dilsun Kirli Kaynar, Moses D. Liskov, Nancy A. Lynch, Olivier Pereira, and Roberto Segala. Using probabilistic I/O automata to analyze an oblivious transfer protocol. *IACR Cryptol. ePrint Arch.*, page 452, 2005. URL: http://eprint.iacr.org/2005/452.

5 Ran Canetti, Ling Cheung, Dilsun Kirli Kaynar, Nancy A. Lynch, and Olivier Pereira. Compositional security for task-pioas. In *20th IEEE Computer Security Foundations Symposium, CSF 2007, 6-8 July 2007, Venice, Italy*, pages 125–139. IEEE Computer Society, 2007. doi:10.1109/CSF.2007.15.

6 Jing Chen and Silvio Micali. Algorand: A secure and efficient distributed ledger. *Theor. Comput. Sci.*, 777:155–183, 2019. doi:10.1016/j.tcs.2019.02.001.

7 R. M. Dudley. *Real analysis and probability.* Wadsworth, Belmont, Calif, 1989.

8 C. A. R. Hoare. *Communicating Sequential Processes.* Prentice-Hall, 1985.

9 Richard M. Karp and Raymond E. Miller. Parallel program schemata. *J. Comput. Syst. Sci.*, 3(2):147–195, 1969. doi:10.1016/S0022-0000(69)80011-5.

10 Leslie Lamport. Proving the correctness of multiprocess programs. *IEEE Trans. Software Eng.*, 3(2):125–143, 1977. doi:10.1109/TSE.1977.229904.

11 Nancy Lynch, Michael Merritt, William Weihl, and Alan Fekete. A theory of atomic transactions. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 326 LNCS:41–71, 1988. doi:10.1007/3-540-50171-1_3.

12 Nancy A. Lynch and Mark R. Tuttle. Hierarchical correctness proofs for distributed algorithms. In Fred B. Schneider, editor, *Proceedings of the Sixth Annual ACM Symposium on Principles of Distributed Computing, Vancouver, British Columbia, Canada, August 10-12, 1987*, pages 137–151. ACM, 1987. doi:10.1145/41840.41852.

13 Nancy A. Lynch and Frits W. Vaandrager. Forward and backward simulations: I. untimed systems. *Inf. Comput.*, 121(2):214–233, 1995. doi:10.1006/inco.1995.1134.

14 Robin Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer, 1980. doi:10.1007/3-540-10235-3.

15 Sayan Mitra. A Verification Framework for Hybrid Systems. 2007. doi:10.1007/3-540-44936-1_6.

16 Rocco De Nicola and Roberto Segala. A process algebraic view of input/output automata. *Theor. Comput. Sci.*, 138(2):391–423, 1995. doi:10.1016/0304-3975(95)92307-J.

17 Susan S. Owicki and David Gries. An axiomatic proof technique for parallel programs I. *Acta Informatica*, 6:319–340, 1976. doi:10.1007/BF00268134.

18 C.A. Petri. *Kommunikation mit Automaten.* PhD thesis, Institut für instrumentelle Mathematik, Bonn, 1962.

19 Martin L. Puterman. *Markov decision processes: discrete stochastic dynamic programming.* Wiley series in probability and mathematical statistics. John Wiley & Sons, 1 edition, 1994.

20 Roberto Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems.* PhD thesis, Massachusettes Institute of technology, 1995.

21 Frits W. Vaandrager. On the relationship between process algebra and input/output automata. In *Proceedings of the Sixth Annual Symposium on Logic in Computer Science (LICS '91), Amsterdam, The Netherlands, July 15-18, 1991*, pages 387–398. IEEE Computer Society, 1991. doi:10.1109/LICS.1991.151662.

22 Kazuki Yoneyama. Formal modeling of random oracle programmability and verification of signature unforgeability using task-pioas. *Int. J. Inf. Sec.*, 17(1):43–66, 2018. doi:10.1007/s10207-016-0352-y.