Refinements for open automata

Rabéa[1][1111−2222−3333−4444], Quentin[1,2][1111−2222−3333−4444], Ludo[2][0000−1111−2222−3333], and Eric[3][2222−−3333−4444−5555]

[1] blabla lncs@fff
http://www.springer.com/gp/computer-science/lncs
[2] blibli
{abc,def}@fff

Abstract. Establishing equivalence and refinement relations between programs is an important mean for verifying their correctness. By establishing that the behaviours of a modified program simulate those of the source one, bisimulation relations formalise the desired relation between a specification and an implementation, two equivalent implementations, or a program and its optimised implementation. This article discusses a notion of refinement between *open automata*, which are symbolic behavioural models for communicating systems. Open automata may have *holes* modelling elements of their context, and can be composed by instantiation of the holes. This allows for a compositional approach for verification of their behaviour.
We define several variants of refinement relations between systems including either equal or different sets of holes, and show under which conditions these refinements are preserved by composition of open automata.

Keywords: Labelled transition systems · Refinement · Composition.

1 Introduction

Automata are convenient for specifying and verifying systems, but most automata definitions allow only the specifications of finite closed systems. Indeed those systems are the ones we can verify efficiently, but programming often consists in writing open systems that should be interfaced with others, and with potentially unbound behaviours (at least statically). We investigate in our works the possibility to define open symbolic systems, verify some of their properties, and use them to compose in a structured way more complex systems.

Refinement relations verify that an implementation satisfies some properties with respect to a specification. Refinement entails that one system can be considered as a more precise version of the specification, featuring all the specified behaviours with more concrete details. In this article we mostly rely on a simulation point of view, where all the specified behaviour must be followed by the refined system but additional behaviours may exist. However we also ensure that a whole scenario, made of several steps, of the specification can also

be simulated by the refined system, which is slightly richer than the traditional simulation relation.

The notion of refinement captures the relation between a specification and an implementation of the same component. Refinement entails that one system can be considered as a more precise version of the specification, featuring all the specified behaviours with more concrete details. It is usually defined as trace inclusion or simulation [15,14]; this ensures that all behaviours of the specification must belong to the behaviours of the implementation . This definition, which is based on systems whose behaviour is fully defined, is not well-suited for open systems, as it requires to reason also about unspecified behaviours.

Open automata were defined as a way to provide a semantics for open parameterized hierarchical LTSs used in verification tools and called *pNets*. An open automaton [12] is a classical automaton with variables and holes. Variables make automata symbolic and allow them to encode infinite state systems. Holes enable the composition of automata: an automaton with a hole is an operator that takes another automaton as parameter and reacts to the actions it emits; the composed automaton is a more precise automaton where the behaviour of one "process parameter" of the main automaton has been provided.

In previous works [4,16] a bisimulation relation was defined for open automata and open parameterized hierarchical LTSs. It has already good properties relatively to bisimulation, but refinement relations were not studied.

This article first introduces open automata that were defined in [12]. and additionaly defines the composition of open automata. Then we define a refinement relation for open automata that has the following characteristics:

- Classical simulation characterisation but also an additional criteria ensuring that refinement does not introduce deadlocks when following a trace from the simulated automaton.
- Good properties relatively to composition: we prove here that filling the same hole with the same automaton preserves the refinement relation.
- Ability to take into account both composition and transitivity: this is a challenge because composition changes the set of holes of the open automaton and refinement takes into account the actions of the holes.

The refinement relation is introduced in two steps. First we define a refinement that relates two automata with the same holes, which allows us to focus on the automaton aspect. Second we introduce a relation that relates two automata with different sets of holes, which allows us to take into account the open nature of open automata, and to deal with composition.

The following of this article is organised as follows. Section 2 recalls the definition of open automata and defines their composition. We then define a refinement relation for open automata, first only considering two automata with the same set of holes in Section 3 and generalize it to automata with a different set of holes in Section 4. Section 5 is dedicated to formalize and prove basic properties of refinement in ; here we prove that refinement is a preorder and we focus on one composition property. In Section 6 we review related works, and Section 7 concludes the paper.

2 Open Automata and their Composition

This section presents our notations and the principles of automata. Except for minor changes in the notations, compared to previous works [**?**] the only new contribution is the definition of a composition operator for open automata.

2.1 Preliminaries and notations

Families of values, or equivalently maps will be noted $x_i^{i\in I}$, $\{i \mapsto x_i \mid i \in I\}$, or $\{i \leftarrow x_i \mid i \in I\}$, depending on what is more convenient. Statements like $\exists c_j^{j\in J}$ defines both J and the mapping $j \mapsto c_j$. The disjoint union on sets is noted $\uplus$. Disjoint union is also used on maps. There are several ways of ensuring a union is disjoint, we will indifferently either suppose sets are disjoint or rename conflicting objects (useful for variables). In a formula, a quantifier followed by a finite set will be used as a shorthand for the quantification on every variable in the set: $\forall\{a_1,\ldots,a_n\},\exists\{b_1,\ldots,b_m\},P$ means $\forall a_1,\ldots,\forall a_n,\exists b_1,\ldots,\exists b_m,P$.

An expression algebra E is the disjoint union of terms, actions, and formulas $E = \mathcal{T} \uplus \mathcal{A} \uplus \mathcal{F}$. $\mathcal{T}$ and $\mathcal{A}$ are term algebras. The formulas $\mathcal{F}$ contain at least first order formulas and equality[3] over $\mathcal{T}$ and $\mathcal{A}$.

For $e \in E$, $vars(e)$ is the set of variables in e that are not bound by a binder. An expression is closed if $vars(e) = \varnothing$. The set $\mathcal{P}$ denotes values which is a subset of closed terms. $\mathcal{F}_V$ is the set of formulas f that only uses variables in V, i.e. the formulas such that $vars(f) \subseteq V$. The substitution in $e \in E$ of $x \in vars(e)$ by $t \in \mathcal{T}$, is denoted $e[t/x]$, and its generalisation to the parallel substitution of variables in V by $\psi : V \to \mathcal{T}$ is denoted $e\{\!\{\psi\}\!\}$.

We suppose given a decidable satisfiability relation on formulas; we denote $\vdash f$ the satisfiability over closed formulas. We will use two variants of the satisfiability relation:

- The satisfiability of a formula $f \in \mathcal{F}$ under some valuation $\sigma : V \to \mathcal{P}$ is defined as follows: $\sigma \vdash f \iff \vdash \exists vars(f\{\!\{\sigma\}\!\}), f\{\!\{\sigma\}\!\}$
- The satisfiability of a formula $f \in \mathcal{F}$ with some variable set V as context is defined as follows: $V \vdash f \iff \vdash \forall V, \exists(vars(f) \setminus V), f$

2.2 Open Automata

Open automata (abbreviated OA) are labelled transition systems with variables that can be used to compose other automata: they are made of transitions that are dependent on the actions of "holes", a composition operation consists in filling a hole with another automaton to obtain a more complex automaton. The variables makes the OA symbolic, and the holes allow for a partial definition of the behaviour.

[3] Equality does not need to be only syntactic.

Definition 1 (Open transition, Open automaton (OA)). *An* open automaton *is a tuple* $\langle S, s_0, V, \sigma_0, J, T\rangle$ *with* S *a set of states,* $s_0 \in S$ *the initial state,* V *the finite set of variable names,* $\sigma_0 : V \to \mathcal{P}$ *the initial valuation of variables,* J *the set of hole names and* T *the set of open transitions.*

An open transition *is a tuple* $\frac{\beta_j^{j\in J'}, g, \psi}{s \xrightarrow{\alpha} s'}$ *with* $s, s' \in S$ *the source and target states,* $\alpha \in \mathcal{A}$ *the produced action,* $J' \subseteq J$ *the holes involved in the transition,* $\beta_j \in \mathcal{A}$ *the actions of the holes,* $g \in \mathcal{F}$ *the guard and* $\psi : V \to \mathcal{T}$ *the variable assignments. To be well-formed, an open transition should use only variables of the automaton and variables appearing in the involved actions, formally:*

$$vars(g) \subseteq vars(\alpha) \cup \bigcup_{j\in J'} vars(\beta_j) \cup V$$

$$\forall v \in V.\, vars(\psi(v)) \subseteq vars(\alpha) \cup \bigcup_{j\in J'} vars(\beta_j) \cup V$$

Definition 2 (Out-transition, Transition variables). *Let* V *be the variable names of the considered automaton,* T *its transitions and* r *one of its states.* $\mathrm{OT}_T(r) \subset T$ *are called the out-transitions of the state* r*. When the transition set is clear from the context, it will be omitted. The local variables of a transition* $vars(t)$ *are all variables appearing in transition* t *except the variables of the automaton.*

$$\mathrm{OT}_T(r) = \left\{ \frac{\beta_j^{j\in J'}, g, \psi}{s \xrightarrow{\alpha} s'} \in T \;\middle|\; s = r \right\}$$

$$vars\left(\frac{\beta_j^{j\in J'}, g, \psi}{s \xrightarrow{\alpha} s'}\right) = \left(vars(\alpha) \cup \bigcup_{j\in J'} vars(\beta_j)\right) \setminus V$$

A pair consisting of a state and a valuation is called a *configuration.*

Example 1. As a running example, we consider a data base example inspired from [11]. The open automaton `database` modelling the behaviour of data base is depicted in Fig. 1. The `database` can be queried using an operation `qry` and updated using an operation `upd`. The associated automaton has a single hole: a counter (`cnt`) depicting the behaviour of the timer. When the `database` performs a query, it sets the new time limit of the counter component and the exposed action to the environment is an observable action `qry`. The `database` updates and switches when it agrees with the counter component the time is over, and the exposed action is `upd`. The open automaton `counter` depicting the behaviour of the timer is illustrated in Fig. 2.

Open automaton composition OA are partially specified automata, that partiality comes from the holes. A hole can be seen as a port in which we can plug an OA. The plugging operation is called composition. The composition of OA

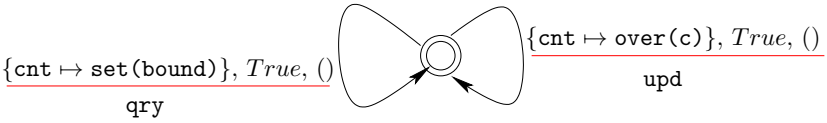


Fig. 1. Open Automaton. It exposes an unspecified counter in the "cnt" hole to count external tick actions. The system sets a counter when a query is enabled and updates when the time is over.

was already implicitly defined by the means of composition on pNets in previous work [12] but never formalised on OA[4]

Definition 3 (Composition of open automata). *The composition of* $A_c = \langle S_c, s_{0c}, V_c, \sigma_{0c}, J_c, T_c\rangle$ *in the hole* $k \in J_c$ *of* $A_p = \langle S_p, s_{0p}, V_p, \sigma_{0p}, J_p, T_p\rangle$ *is the OA defined as follows:*

$$A_c[A_p/k] ::= \langle S_c \times S_p, (s_{0c}, s_{0p}), V_c \uplus V_p, \sigma_{0c} \uplus \sigma_{0p}, J_p \uplus J_c \setminus \{k\}, T\rangle$$

with

$$T = \left\{ \frac{\beta_j^{j\in J'_p \uplus J'_c}, g_c \wedge g_p \wedge \alpha_p = \beta_k, \psi_c \uplus \psi_p}{(s_c, s_p) \xrightarrow{\alpha_c} (s'_c, s'_p)} \;\middle|\; \frac{\beta_j^{j\in J'_c \uplus \{k\}}, g_c, \psi_c}{s_c \xrightarrow{\alpha_c} s'_c} \in T_c, \frac{\beta_j^{j\in J'_p}, g_p, \psi_p}{s_p \xrightarrow{\alpha_p} s'_p} \in T_p \right\}$$
$$\cup \left\{ \frac{\beta_j^{j\in J'_c}, g_c, \psi_c}{(s_c, s_p) \xrightarrow{\alpha_c} (s'_c, s_p)} \;\middle|\; \frac{\beta_j^{j\in J'_c}, g_c, \psi_c}{s_c \xrightarrow{\alpha_c} s'_c} \in T_c, k \notin J'_c, s_p \in S_p \right\}$$

The first OA decides when the second can evolve by involving its hole in a transition: the action emitted when A_p makes a transition is synchronised with the action of the hole k in transitions of A_c.

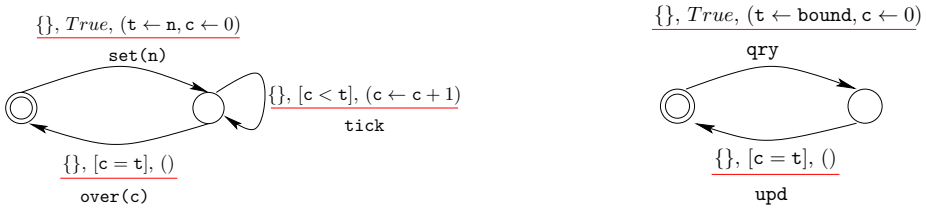


Fig. 2. (left) Open Automaton of `counter` process. Its role is to get a target, then count ticks until that target is reached, then emits an action with the elapsed time and restart. (right) Open Automaton of the composition `database`[`counter/cnt`].

In Fig. 2 (right) we present the composition `database`[`counter/cnt`] of the two open automata `database` and `counter`. Each state of the composite consists

[4] The definition of composition below is a direct translation of what happens with pNets composition without introducing pNets.

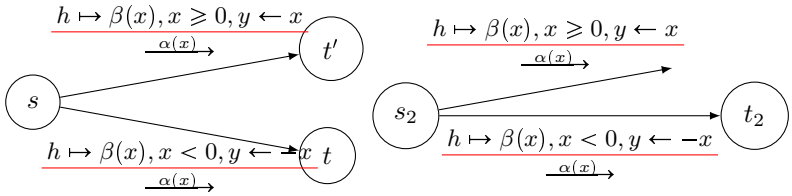
of a state of `database` together with a state of `counter`. The composed automaton takes over the same steps as the `database` automaton but it also includes new steps, indicating the change of states for the setting of timer.

Relations between open automata A relation (bisimulation, refinement, etc.) between OA requires to compare their states. To do so we will suppose that the variables of the two OA are disjoint (a renaming of variables may have to be applied before comparing OA states).

Definition 4 (Relation on OA states). *Suppose V_1 and V_2 are disjoint. A relation on states of $\langle S_1, s_{01}, V_1, \sigma_{01}, J_1, T_1\rangle$ and $\langle S_2, s_{02}, V_2, \sigma_{02}, J_2, T_2\rangle$ is a function $R : S_1 \times S_2 \to \mathcal{F}_{V_1 \uplus V_2}$.*

[TODO:C'est etrange, la formulation "the variables they refer to", comme il n'y a plus de variables d'etat... +1] The idea is that two states are related if the variables they refer to verify a certain formula. Additionally, we may check that initial states of the automata are related by checking that: $\sigma_{01} \uplus \sigma_{02} \vdash R(s_{01}, s_{02})$.

A bisimulation for open automata Bisimulation between OA was defined in [**?**]. Instead of recalling its definition we show below the principles of this bisimulation. **[TODO:Je ne comprends pas cette figure... elle est incomplete ? Ou alors il faut un peu d'explications...]**



3 Refinement Relations for automata with the same holes

Similarly to FH-bisimulation [4] we are interested in finding relations between states of two open automata that contain variables and holes. However here we want to build a refinement relation that also guarantees that no deadlock is introduced when refining the automaton (at least not among the scenarios originally planned in the specification). The refinement relation should be conditioned by the internal states of the automata. More precisely, a refinement relation characterizes when two states are related, and this characterisation is expressed as a predicate on the variables of the two automata. Thus, a refinement relation is a relation on OA states.

We rely on a classical notion of simulation inspired from [**?**]. The idea is that two states related by a given relation and states that if one state can do a transition, then the other can do a transition too. But to limit the creation of

deadlocks when simulating an automaton, we first define when a relation is preserved along identical transitions before characterising a first simple refinement relation to compare automata with identical holes.

Let us first explain the reasons why simulation can introduce allow undesired bwhaviours to emerge and how we want to limit them. The automaton that simulates a specification features any behaviour that contains the behaviour of the specification, or more precisely, for each state that simulates a state of the specification, the simulating automaton must at least feature the behaviour of the specification. In this definition, nothing prevents the implementation from adding non-determinism and non-deterministically reaching a state that is unrelated to the specification, this state could even be a deadlock even though we only followed transitions that exist in the specification. To avoid the appearance of such deadlocks, we will state that, when both the implementation and the specification do the same transition, they stay related. This ensures a form of continuity in the comparison in the sense that the refined system is constrained to follow the behaviour of the simulated system along a given scenario. In terms of traces this amounts to some form of deadlock reduction: for each trace followed by the simulated system if the refined system followed the beginning of the trace, it must be able to keep following the trace. Of course, the classical trace view is a simplified view compared to the guards, states and actions from the holes that exist in OA, but it gives the right intuition.

We thus define the following property that can be understood as follows. Take a specification automaton and its implementation. Consider two related states for the refinement relation. Suppose an identical transition exist in both automata (with same guards and same label), then either the reached states s_1' and s_2' are related by refinement, or there exists another transition of the specification that is also identical and leads to a state s_1'' that is related to s_2' in the refinement relation. Finally, we should take into account that 1) instead of equality, we should check if there is a state such that both transitions are feasible (the conjunction of guards is satisfiable), and that 2)instead of finding another transition of the specification we might find a family of transitions that cover all the cases of the transition of the implementation. This leads us to the following definition.

Definition 5 (Preservation along identical transitions).
Let $\langle S_1, s_{01}, V_1, \sigma_{01}, J_1, T_1\rangle$ *and* $\langle S_2, s_{02}, V_2, \sigma_{02}, J_2, T_2\rangle$ *be two OAs. A relation* $R : S_1 \times S_2 \to \mathcal{F}_{V_1 \uplus V_2}$ *between* $\langle S_1, s_{01}, V_1, \sigma_{01}, J_1, T_1\rangle$ *and* $\langle S_2, s_{02}, V_2, \sigma_{02}, J_2, T_2\rangle$

is preserved along identical transitions if it satisfies the following:

$$\forall (s_1,s_2)\in S_1\times S_2, \forall (t_1,t_2)=\left(\frac{\beta_{1j}^{j\in J_1'},g_1,\psi_1}{s_1\xrightarrow{\alpha_1}s_1'},\frac{\beta_{2j}^{j\in J_2'},g_2,\psi_2}{s_2\xrightarrow{\alpha_2}s_2'}\right)\in \mathrm{OT}(s_1)\times\mathrm{OT}(s_2)$$

$$\exists\left(t_{1x}=\frac{\beta_{1xj}^{j\in J_{1x}'},g_{1x},\psi_{1x}}{s_1\xrightarrow{\alpha_{1x}}s_{1x}'}\in \mathrm{OT}(s_1)\right)^{x\in X}, V_1\uplus V_2\uplus \biguplus_{x\in X} vars(t_{1x})\uplus vars(t_2)\uplus vars(t_1)\vdash$$

$$R(s_1,s_2)\wedge g_1\wedge g_2\wedge \alpha_1=\alpha_2 \implies$$

$$\bigwedge_{x\in X}\left(\alpha_1=\alpha_{1x}\wedge J_{1x}'=J_2'\wedge\bigwedge_{j\in J_2'}\beta_{2j}=\beta_{1xj}\wedge g_{1x}\wedge R\big(s_{1x}',s_2'\big)\{\!\{\psi_{1x}\uplus\psi_2\}\!\}\right)$$

We define here simulation between labelled transition systems to Hole-equal simulation between open automata that have exactly the same holes.

Definition 6 (Hole-equal simulation). *Consider two OAs* $\langle S_1,s_{01},V_1,\sigma_{01},J_1,T_1\rangle$ *and* $\langle S_2,s_{02},V_2,\sigma_{02},J_2,T_2\rangle$ *with* $J_1=J_2$*, the relation on configurations* $R: S_1\times S_2\to\mathcal{F}_{V_1\uplus V_2}$ *is a hole-equal simulation from* S_1 *to* S_2 *if the following conditions hold:*

(1) $\sigma_{01}\uplus\sigma_{02}\vdash R(s_{01},s_{02})$

(2) $\forall (s_1,s_2)\in S_1\times S_2$,

$$\forall t_1=\frac{\beta_{1j}^{j\in J_1'},g_1,\psi_1}{s_1\xrightarrow{\alpha_1}s_1'}\in \mathrm{OT}(s_1), \exists\left(t_{2x}=\frac{\beta_{2xj}^{j\in J_{2x}'},g_{2x},\psi_{2x}}{s_2\xrightarrow{\alpha_{2x}}s_{2x}'}\in \mathrm{OT}(s_2)\right)^{x\in X},$$
$$\big(\forall x\in X, J_{2x}'=J_1'\big)\wedge$$
$$V_1\uplus V_2\uplus vars(t_1)\vdash$$
$$\left(R(s_1,s_2)\wedge g_1 \implies \bigvee_{x\in X}\left(\begin{array}{c}\alpha_1=\alpha_{2x}\wedge\bigwedge_{j\in J_{2x}'}\beta_{1j}=\beta_{2xj}\wedge\\ g_{2x}\wedge R\big(s_1',s_{2x}'\big)\{\!\{\psi_1\uplus\psi_{2x}\}\!\}\end{array}\right)\right)$$

(3) R is is preserved along identical transitions of A_1 *and* A_2*.*

Note in this definition, instead of matching an instantiated transition (with all variables assigned) of the first automata to another instantiated transition of the second, it matches an open transition t_1 to a family of covering open transitions $t_{2x}^{x\in X}$.

Intuitively, this means that for every pair of related states (s_1,s_2) of the two automata, and for every transition of the first automaton from s_1, there is a set of matching transitions of the second automaton from s_2 such that the produced action match, the actions of the same holes and the successors are related after variable update. Our definition captures a simple kind of sub-classing of open automata with the same holes. It is stronger than a strict simulation since it matches a transition with a family of transitions. With such a relation we are able to check the refinement between two open automata with the same level of

abstraction but specified differently, for example, by duplicating states, removing transitions, reinforcing guards, modifying variables.

To illustrate the refinement simulation of open automata, consider a variation on the `database` automaton of Fig. 3. We suppose that on more concrete level, in addition to querying and updating the data base, a backup copy of it may also be made.

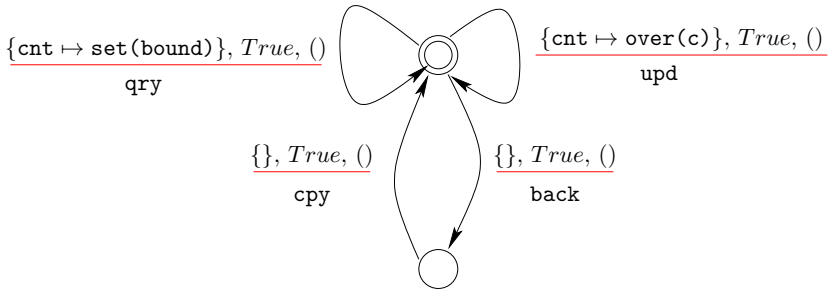


Fig. 3. The refined Open Automaton of the automaton of Fig.1. Two other actions, in addition to `qry` and `upd`, `back` representing a request to make a backup copy and `cpy` is used to reflect the actual operation of copying.

[TODO:The refinement relation also specifies that the refined process must follow the behaviour of the specification, not only at every compatible state but also along any trace.]

We will show in Section 5 that this refinement relation has good properties in terms of transitivity, compositionality, reflexivity, etc.

The refinement relation defined above is insufficient in the setting of composition which is the main advantage of the open automaton-based approach. Indeed, it should be possible to refine an automaton by filling its hole, providing a concrete view of a part of the application that was not specified originally. More generally, it should be possible to relate automata that do not have the same holes because composition is a crucial part of system specification. Thus, we believe composition should also be a form of refinement and call this feature *refinement through composition*. However, filling holes can result in a system with more or less holes than the original system because the plugged subsystem can contain itself many holes. Next section will define a more powerful refinement relation able to reason on automata with different sets of holes.

[TODO:add an example]

4 A Refinement Relation that Takes Holes into Account

[TODO:3.5 pages]

This section extends the preceding relation to automata where the set of holes is not the same. A simple use-case for this is filling a hole with a completely

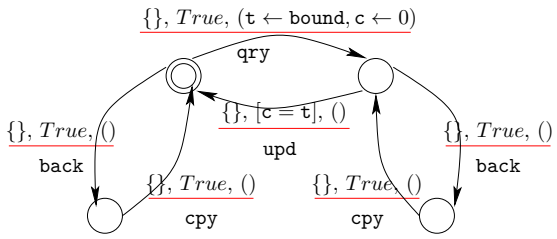


Fig. 4. The refined Open Automaton composed with the counter

defined automaton. In this case, we want to ensure that the automaton with a filled hole is a refinement of the other automata: actions of the identical holes will be taken into account the same way, and filling the hole partially reduces the behaviour of the automaton.

The major challenge in the design of this relation is to maintain a form of transitivity while being able to take into account the actions of some of the holes. A naive definition of refinement would ensure that the holes that are identical in the two open automata are taken into account in the simulation. Unfortunately considering all the common holes does not ensure transitivity of the simulation for the following reason. If A_1 simulates A_2 and A_2 simulates A_3, and one hole j appears in A_3 and in A_1 but not in A_2 then we have no guarantee on the way A_1 and A_3 take the actions of this his hole into account, thus a refinement between and A_1 and A_3 would require conditions involving actions of the hole j which cannot be ensured. The way we solve this issue is to remember in the simulation relation which holes have been compared. This makes the relation parameterized by a subset of the set of holes that belong to the two automata that we want to take into account. This way, in the example above, we would have no guarantee on actions the hole j by transitivity but can state a refinement relation with guarantees on the actions of the other holes.

In the following definition we add a parameter H which is the set of holes tracked by the refinement relation and adapt the definition by ignoring actions of the holes that are not in H.

Definition 7 (Open automata refinement). *For two open automata* $A_1 \coloneqq \langle S_1, s_{01}, J_1, V_1, \sigma_{01}, T_1\rangle$ *and* $A_2 \coloneqq \langle S_2, s_{02}, J_2, V_2, \sigma_{02}, T_2\rangle$*, A_1 is a refinement of A_2 tracking holes H, noted $A_1 \leqslant_H A_2$, with $H \subseteq J_1 \cap J_2$, if there is a relation $R : (S_1 \times S_2) \to \mathcal{F}_{V_1 \uplus V_2}$ such that:*

(1) $\sigma_{01} \uplus \sigma_{02} \vdash R(s_{01}, s_{02})$

(2) $\forall (s_1, s_2) \in S_1 \times S_2$,

$$\forall \frac{\beta_{1j}^{j \in J_1'}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s_1'} \in \mathrm{OT}(s_1), \exists \left(\frac{\beta_{2xj}^{j \in J_{2x}'}, g_{2x}, \psi_{2x}}{s_2 \xrightarrow{\alpha_{2x}} s_{2x}'} \in \mathrm{OT}(s_2) \right)^{x \in X},$$
$$\left(\forall x \in X, J_{2x}' \cap H = J_1' \cap H\right) \wedge$$
$$V_1 \uplus V_2 \uplus \mathit{vars}(t_1) \vdash$$
$$\left(R(s_1, s_2) \wedge g_1 \implies \bigvee_{x \in X} \left(\begin{array}{c} \alpha_1 = \alpha_{2x} \wedge \bigwedge_{j \in J_{2x}' \cap H} \beta_{1j} = \beta_{2xj} \wedge \\ g_{2x} \wedge R\left(s_1', s_{2x}'\right) \{\!\{ \psi_1 \uplus \psi_{2x} \}\!\} \end{array} \right) \right)$$

(3) R is preserved along identical transitions of A_1 and A_2.

Giving R and H is sufficient to characterise the refinement, so we call (R, H) a hole-tracking simulation of A_1 by A_2. Hole in H are called tracked holes.

Note that every action of the holes outside H is unconstrained in the related automata.

This definition leads to a notion of simulation as refinement the same as the one used in LOTOS [7] and presented under the name extension refinement. The extension consists in adding new functionalities by preserving existing ones. Indeed, in case there is no hole and no guard, i.e., in case the definitions of open automata coincide with those of automata, the relation allows new behaviour to be added in terms of new traces, but doesn't allow new deadlocks to be added, i.e., for traces of the specification (abstract automaton) the implementation (refined automaton) must not add deadlocks.

[TODO: A string $t \in \mathcal{A}^*$ is a trace of an automaton if there is some state s such that $s_0 \overset{t}{\twoheadrightarrow} s$. We denote the set of traces of an automaton A by $\mathcal{T}_A$.

Notion of Extension refinement. May be we need to introduce the notion of refusal of a trace $t \in \mathcal{T}_A$ is a set of actions such that all actions in the set can be refused after the automaton A has executed the trace t.

$$X \in \bot_A(t) \Leftrightarrow \{\exists s. s_0 \overset{t}{\twoheadrightarrow} s \wedge \forall a \in X. s \overset{a}{\not\rightarrow}\}$$

and replace deadlock reducing, under (3) by

$$\forall t \in \mathcal{T}_{A_1} \cap \mathcal{T}_{A_2},\ \bot_{A_2}(t) \subseteq \bot_{A_1}(t)$$

]

[TODO:thm: backward compatibility + simulation is equivalent to extension refinement in case there is no hole and no guard, i.e. in case the definitions of automata coincide]

Lemma 1 (Tracked holes). *By construction, if an automaton is the refinement of another one, it is also a refinement by tracking less holes.*

$$A_1 \leqslant_H A_2 \wedge H' \subseteq H \implies A_1 \leqslant_{H'} A_2$$

[TODO:example]

5 Properties of our Refinement Relations

[TODO:1.5 pages] We now state the properties of our refinement relations, proofs of the properties can be found in the appendices. We express these properties in terms of open automata refinement because hole-equal simulation is a particular case of Definition 7 when $J_1 = J_2 = H$, and thus most of the properties shown here are easy to adapt to hole-equal simulation.

The first crucial property of refinement is that it is a preorder on the set of open automata. This property enables stepwise refinement.

Theorem 1 (Refinement is a preorder). *Refinement is reflexive and transitive: it is a preorder on the set of open automata.*

The relation $\leqslant_H$ is reflexive, $A \leqslant_H A$, by taking $R(s_1, s_2) \mapsto \bigwedge_{v \in vars(s_1)} v = v$ to be $s_1 = s_2$ and checking the above conditions (Definition 7), we can see that the relation is indeed reflexive. Appendix A presents the proof of transitivity. It is done classically by identifying the relation between A_1 and A_3 that is a refinement. What is less classical is the definition of this relation because it is a boolean formula. For each couple of states s_1 and s_3 of A_1 and A_3 we build a a formula that defines the refinement relation. To do this, we take the disjunction of formulas relating s_1 and s_3, and passing by all states s_2 of A_2. More precisely, we define a relation of the following form:

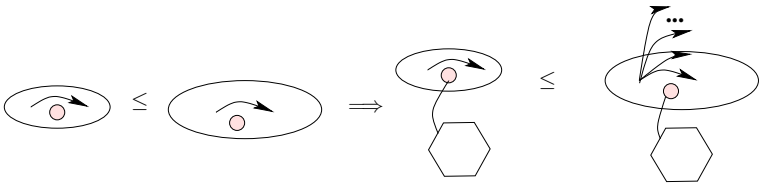
$$R_{13}(s_1, s_3) = \bigvee_{s_2 \in S_2} (R_{12}(s_1, s_2) \wedge R_{23}(s_2, s_3))$$

We then prove that this relation defines is a refinement, according to Definition 7.

The next two theorems state that refinement is compositional in the sense that it is sufficient to prove refinement for the composed automata separately to obtain a refinement relation. This can be split into theorems that can be trivially composed by transitivity.

Theorem 2 (Context refinement). *Let A_1, A_2 and A_3 be three open automata with $A_1 \leqslant_H A_2$. Let J_3 be the set of holes of A_3. We have:*

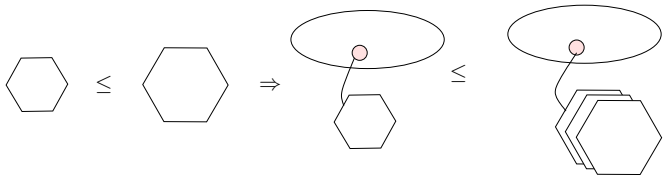
$$A_1[A_3/k] \leqslant_{J_3 \uplus H \setminus \{k\}} A_2[A_3/k]$$



Theorem 3 (Congruence). *Let A_1, A_2 and A_3 be three open automata with $A_2 \leqslant_H A_3$. We have:*

$$A_1[A_2/k] \leqslant_{J_1 \uplus H \setminus \{k\}} A_1[A_3/k]$$

[TODO:i did not put the global composition theorem, de we state it?]



6 Related Work

[TODO:1 page]

There are some works that have focused on the refinement of open systems. Defining refinement of open systems as trace inclusion is addressed as a notion of subtyping in type theory [10,6]. Such refinement is instead based on interface-oriented approach, it allows the expression more internal choices and less external choices. The refinement of open systems is also defined in terms of alternating simulation [3,2], which deals with game-based models. Alternating simulation that is originating from the game theory [1] allows the study of relation between individual components by viewing them as alternating transition systems. In particular, a refinement of game-based automata expresses that the refined component can offer more services (input actions) and fewer service demands (output actions). However, the composition of such automata may lead to illegal states, where one automaton issues an output that is not acceptable as input in the other one. The theory of alternating simulation provides an optimistic approach to compute compatibility between automata based on the fact that each automaton expects the other to provide legal inputs, i.e, two components can be composed if there is an environment where they can work together. As we shall see in this paper, our approach to design refinement has some commonalities with that of the above mentioned [2]: both are process-oriented approach even if they are not based on the same notion of simulation and they are based on optimistic approach to composition. For the composability, we shall see that we use the notion of comparability of holes (similar to the notion of compatibility), which is explicitly encompassed in the definition of composition.

Previous work on open automata focused on equivalence relations compatible with composition. In an article by Hou, Zechen and Madelaine [13], a computable bisimulation is introduced and proved equivalent to the previous bisimulation already introduced. In a more recent work by Ameur-Boulifa, Henrio and Madelaine [4], a weak version of the bisimulation on open automata is introduced. These works differ from ours because the relation introduced in this report is a refinement relation in the form of a simulation and not a bisimulation. Also we do not have results as strong as computability neither a weak version able to tackle silent actions.

[TODO:Some related work on other models than open automata introduce refinement relations. In a chapter by Bellegarde, Julliand

and Kouchnarenko [5], a simulation relation on transition systems is introduced. This simulation encompass action refinement, is able to deal with silent actions and is compatible with parallel composition. Here the refinement relation does not consider action refinement as valid but it should be done in future work. Also they check how LTL properties are preserved or combined using their refinement which we do not do. However their model is less expressive: the transition system model is less expressive than open automata and the parallel composition is less expressive than composition on open automata. In a later report by Kouchnarenko and Lanoix [?], the refinement relation they introduce is on LTS (labelled transition systems). Their relation additionally prevents deadlock and livelocks. The composition is also extended to synchronised composition which is more expressive. In our work we also deal with deadlocks but not with livelocks since the latter arise only with silent actions. This work is closer than the previous one to what we do here, still open automata are more expressive than LTS and composition is more general than synchronised composition.]

A refinement relation on models nearer to open automata is introduced in an article by Zhang, Meng and Lo [17]. In their article they work with transition systems with variable which makes the state space potentially infinite. This aspect is also present in open automata. They show how invariants are composed. By relation on these invariants they introduce several refinement relations. We could have done something similar for non-blocking composition reachability predicates which are introduced in Section **??**.

On the deadlock prevention aspect, an article by Dihego, Sampaio and Oliveira [8] present a refinement relation on process algebra (translated to LTS). This refinement relation is a special case of inheritance and prevents the introduction of deadlocks. Their refinement and inheritance are quite the opposite of our refinement in terms of new behaviours. They have channels, interfaces, inputs and outputs, which in the open automata model can be compared to action labels, holes and action data for both inputs and outputs. They have a rich composition as open automata but the introduction of deadlock is already prevented by a well chosen set of composing operations. Also their composition is slightly different than the one on open automata because they can cause loops by linking two channels of the same process, where in open automata the composition makes an oriented tree. In their model there is an explicit deadlock and a successful termination where in open automata there are no explicit termination. We define a deadlock as a configuration without possible transition and assume what is a deadlock when comparing the open automata. To define their refinement and inheritance relation they use trace and failure semantics, which are weaker than (bi)simulations [9] and could break with open automata composition.

7 Conclusion

In this article we investigated the notion of refinement for a symbolic and open model: open automata. Open automata are convenient to model parallel systems that are parameterised both by the use of variables and by the possibility to compose automata. The refinement relation first relies on a simulation relation between the specification and its refinement, but it also specifies that the refined process must follow the behaviour of the specification, not only at every compatible state but also along any trace. We finally showed that refinement is a preorder that is preserved when filling a hole.

In the future we will focus on the other composition property that considers the preservation of refinement when automata are placed in the same context, namely filling the hole of the same automaton by an automaton and its refinement. This property should require some restriction on the composed processes. Finally, we will investigate under which condition, the composition operation produces a refinement in the sense that filling a hole produces a refined process.

References

1. de Alfaro, L.: Game models for open systems. In: Dershowitz, N. (ed.) Verification: Theory and Practice, Essays Dedicated to Zohar Manna on the Occasion of His 64th Birthday. Lecture Notes in Computer Science, vol. 2772, pp. 269–289. Springer (2003). https://doi.org/10.1007/978-3-540-39910-0_12, https://doi.org/10.1007/978-3-540-39910-0_12
2. de Alfaro, L., Henzinger, T.A.: Interface automata. In: Tjoa, A.M., Gruhn, V. (eds.) Proceedings of the 8th European Software Engineering Conference held jointly with 9th ACM SIGSOFT International Symposium on Foundations of Software Engineering 2001, Vienna, Austria, September 10-14, 2001. pp. 109–120. ACM (2001). https://doi.org/10.1145/503209.503226, https://doi.org/10.1145/503209.503226
3. Alur, R., Henzinger, T.A., Kupferman, O., Vardi, M.Y.: Alternating refinement relations. In: Sangiorgi, D., de Simone, R. (eds.) CONCUR'98 Concurrency Theory. pp. 163–178. Springer Berlin Heidelberg, Berlin, Heidelberg (1998)
4. Ameur-Boulifa, R., Henrio, L., Madelaine, E.: Compositional equivalences based on open pnets. CoRR **abs/2007.10770** (2020), https://arxiv.org/abs/2007.10770
5. Bellegarde, F., Julliand, J., Kouchnarenko, O.: Ready-simulation is not ready to express a modular refinement relation. In: Maibaum, T. (ed.) Fundamental Approaches to Software Engineering. pp. 266–283. Springer Berlin Heidelberg, Berlin, Heidelberg (2000)
6. Bravetti, M., Zavattaro, G.: Asynchronous session subtyping as communicating automata refinement. Softw. Syst. Model. **20**(2), 311–333 (2021). https://doi.org/10.1007/s10270-020-00838-x, https://doi.org/10.1007/s10270-020-00838-x
7. Brinksma, E., Scollo, G., Steenbergen, C.: LOTOS specifications, their implementations and their tests. In: Bochmann, G., Sarikaya, B. (eds.) Protocol Specification, Testing and Verification VI, pp. 349–360. North Holland (1987)
8. Dihego, J., Sampaio, A., Oliveira, M.: A refinement checking based strategy for component-based systems evolution. Journal of Systems and Software **167**,

110598 (2020). https://doi.org/https://doi.org/10.1016/j.jss.2020.110598, https://www.sciencedirect.com/science/article/pii/S0164121220300765
9. Eshuis, R., Fokkinga, M.M.: Comparing refinements for failure and bisimulation semantics. Fundam. Inf. **52**(4), 297–321 (Apr 2002)
10. Gay, S.J., Hole, M.: Subtyping for session types in the pi calculus. Acta Informatica **42**(2-3), 191–225 (2005). https://doi.org/10.1007/s00236-005-0177-z, https://doi.org/10.1007/s00236-005-0177-z
11. Gorrieri, R., Rensink, A.: Action refinement. In: Bergstra, J.A., Ponse, A., Smolka, S.A. (eds.) Handbook of Process Algebra, pp. 1047–1147. North-Holland / Elsevier (2001). https://doi.org/10.1016/b978-044482830-9/50034-5, https://doi.org/10.1016/b978-044482830-9/50034-5
12. Henrio, L., Madelaine, E., Zhang, M.: A theory for the composition of concurrent processes. In: Albert, E., Lanese, I. (eds.) Formal Techniques for Distributed Objects, Components, and Systems. pp. 175–194. Springer International Publishing, Cham (2016)
13. Hou, Z., Madelaine, E.: Symbolic bisimulation for open and parameterized systems. In: Proceedings of the 2020 ACM SIGPLAN Workshop on Partial Evaluation and Program Manipulation. pp. 14–26. PEPM 2020, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3372884.3373161, https://doi.org/10.1145/3372884.3373161
14. Kouchnarenko, O., Lanoix, A.: How to verify and exploit a refinement of component-based systems. In: Virbitskaite, I., Voronkov, A. (eds.) Perspectives of Systems Informatics. pp. 297–309. Springer Berlin Heidelberg, Berlin, Heidelberg (2007)
15. Milner, R.: A Calculus of Communicating Systems, Lecture Notes in Computer Science, vol. 92. Springer (1980). https://doi.org/10.1007/3-540-10235-3, https://doi.org/10.1007/3-540-10235-3
16. Wang, B., Madelaine, E., Zhang, M.: Symbolic Weak Equivalences: Extension, Algorithms, and Minimization - Extended version. Research Report RR-9389, Inria & Université Cote d'Azur, CNRS, I3S, Sophia Antipolis, France ; East China Normal University (Shanghai) (Jan 2021), https://hal.inria.fr/hal-03126313
17. Zhang, L., Meng, Q., Lo, K.: Compositional abstraction refinement for component-based systems. Journal of Applied Mathematics **2014**, 1–12 (2014). https://doi.org/10.1155/2014/703098, https://doi.org/10.1155/2014/703098

A Proof of Transitivity for Refinement

If $A_1 \leqslant_H A_2$ and $A_2 \leqslant_{H'} A_3$, then $A_1 \leqslant_{H \cap H'} A_3$.

Proof. If $A_1 \leqslant_H A_2$ then there is R_{12} a relation between states of A_1 and of A_2; If $A_2 \leqslant_{H'} A_3$ then there is R_{23} a relation between states of A_2 and of A_3. We build a relation between states of A_1 and of A_3 as follows: for each pair of states s_1, s_3, for each state s_2 such that R_{12} relates s_1 and s_2, and R_{23} relates s_2 and s_3. Let R_{13} be the relation:

$$R_{13}(s_1, s_3) = \bigvee_{s_2 \in S_2} (R_{12}(s_1, s_2) \wedge R_{23}(s_2, s_3))$$

We will show that $A_1 \leqslant_{H \cap H'} A_3$ by exhibiting R_{13} as a hole-tracking simulation of A_1 by A_3.

We have to prove that the relation R_{13} satisfies the three conditions of the definition of a refinement of open automata.

1. Firstly, we have to R_{13} satisfies initial configurations:

$$\sigma_{01} \uplus \sigma_{03} \vdash R_{13}(s_{01}, s_{03})$$

By knowing that substitutions only have an effect on the variables of the open automaton they belong to, they also produce terms containing only variables of the open automaton they belong to. We have:

$$\begin{aligned}
(\sigma_{01} \uplus \sigma_{02} \vdash R_{12}(s_{01}, s_{02})) \wedge (\sigma_{02} \uplus \sigma_{03} \vdash R_{23}(s_{02}, s_{03})) &\implies \\
R_{12}(s_{01}, s_{02})\{\!\!\{\sigma_{01} \uplus \sigma_{02}\}\!\!\} \wedge R_{23}(s_{02}, s_{03})\{\!\!\{\sigma_{02} \uplus \sigma_{03}\}\!\!\} &\implies \\
R_{12}(s_{01}, s_{02})\{\!\!\{\sigma_{01} \uplus \sigma_{02} \uplus \sigma_{03}\}\!\!\} \wedge R_{23}(s_{02}, s_{03})\{\!\!\{\sigma_{01} \uplus \sigma_{02} \uplus \sigma_{03}\}\!\!\} &\implies \\
\underbrace{R_{12}(s_{01}, s_{02}) \wedge R_{23}(s_{02}, s_{03})}_{\implies R_{13}(s_{01}, s_{03})} \{\!\!\{\sigma_{01} \uplus \sigma_{02} \uplus \sigma_{03}\}\!\!\} &
\end{aligned}$$

Since σ_{02} has no effect on variables of s_{01} and s_{03} thus we get the expected result.

2. Secondly, we need to prove that for any open transition t_1 in T_1 originating from s_1:

$$\frac{\beta_{1j}^{j \in J_1'}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s_1'} \in \mathrm{OT}(s_1)$$

there exists an indexed family of OTs originating from s_3:

$$V_1 \uplus V_3 \uplus \mathit{vars}(t_1) \vdash \left(R_{13}(s_1, s_3) \wedge g_1 \implies \bigvee_{z \in Z} \left(\begin{array}{c} \alpha_1 = \alpha_{3z} \wedge \bigwedge\limits_{j \in J_{3z}' \cap (H \cap H')} \beta_{1j} = \beta_{3jz} \wedge \\ g_{3z} \wedge R_{12}\big(s_1', s_{3z}'\big)\{\!\!\{\psi_1 \uplus \psi_{3z}\}\!\!\} \end{array} \right) \right)$$

Consider $(s_1, s_3) \in R_{13}$ then there is a set of states $(s_{2p})^{p \in P}$ of A_2 relating s_1 and s_3:

$$R_{13}(s_1, s_3) = \bigvee_{p \in P} (R_{12}(s_1, s_{2p}) \wedge R_{23}(s_{2p}, s_3))$$

Let's consider any $s_{2p} \in (s_{2p})^{p \in P}$. We have on one side, for any open transition t_1 in T_1 originating from s_1:

$$\frac{\beta_{1j}^{j \in J_1'}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s_1'} \in \mathrm{OT}(s_1)$$

there exists an indexed family of OTs originating from s_{2p}:

$$\left(\frac{\beta_{2pxj}^{j \in J_{2px}'}, g_{2px}, \psi_{2px}}{s_{2p} \xrightarrow{\alpha_{2px}} s_{2px}'} \in \mathrm{OT}(s_{2p}) \right)^{x \in X_p}$$

such that $\forall x \in X, J'_{2px} \cap H = J'_1 \cap H$ and

$$V_1 \uplus V_2 \uplus vars(t_1) \vdash$$
$$\left(R_{12}(s_1, s_{2p}) \wedge g_1 \implies \bigvee_{x \in X_p} \begin{pmatrix} \alpha_1 = \alpha_{2px} \wedge \bigwedge\limits_{j \in J'_{2px} \cap H} \beta_{1j} = \beta_{2pxj} \wedge \\ g_{2px} \wedge R_{12}\big(s'_1, s'_{2px}\big) \{\!\{ \psi_1 \uplus \psi_{2px} \}\!\} \end{pmatrix} \right)$$

Adding to both sides of the implication the predicate $R_{23}(s_{2p}, s_3)$ we get:

$$V_1 \uplus V_2 \uplus V_3 \uplus vars(t_1) \vdash$$
$$R_{12}(s_1, s_{2p}) \wedge R_{23}(s_{2p}, s_3) \wedge g_1 \implies$$
$$\bigvee_{x \in X} \begin{pmatrix} \alpha_1 = \alpha_{2px} \wedge \bigwedge\limits_{j \in J'_{2px} \cap H} \beta_{1j} = \beta_{2pxj} \wedge \\ g_{2px} \wedge R_{12}\big(s'_1, s'_{2px}\big) \{\!\{ \psi_1 \uplus \psi_{2px} \}\!\} \wedge R_{23}(s_{2p}, s_3) \end{pmatrix} \quad (*)$$

On the other side, according the relation between A_2 and A_3 we have for any open transition t_{2px} in T_2 originating from s_{2p}:

$$\frac{\beta_{2pxj}^{j \in J'_{2px}}, g_{2px}, \psi_{2px}}{s_{2p} \xrightarrow{\alpha_{2px}} s'_{2px}} \in \mathrm{OT}(s_{2p})$$

there exists an indexed family of OTs originating from s_3:

$$\left(\frac{\beta_{3pxyj}^{j \in J'_{3pxy}}, g_{3pxy}, \psi_{3pxy}}{s_3 \xrightarrow{\alpha_{3pxy}} s'_{3pxy}} \in \mathrm{OT}(s_3) \right)^{y \in Y}$$

such that $\forall y \in Y, J'_{2px} \cap H' = J'_{3pxy} \cap H'$ and

$$V_2 \uplus V_3 \uplus vars(t_{2px}) \vdash$$
$$R_{23}(s_{2p}, s_3) \wedge g_{2px} \implies$$
$$\bigvee_{y \in Y} \begin{pmatrix} \alpha_{2px} = \alpha_{3pxy} \wedge \bigwedge\limits_{j \in J'_{3pxy} \cap H'} \beta_{2pxj} = \beta_{3pxyj} \wedge \\ g_{3pxy} \wedge R_{23}\big(s'_{2px}, s'_{3xy}\big) \{\!\{ \psi_{2px} \uplus \psi_{3pxy} \}\!\} \end{pmatrix} \quad (**)$$

From the two previous cases: $\forall x \in X, J'_{2px} \cap H = J'_1 \cap H$ and $\forall y \in Y, J'_{2px} \cap H' = J'_{3pxy} \cap H'$ we conclude: $\forall x \in X, \forall y \in Y, J'_1 \cap (H \cap H') = J'_{3pxy} \cap (H \cap H')$

In addition, by combining formula $(**)$ and $(*)$, we get:

$$V_1 \uplus V_2 \uplus V_3 \uplus vars(t_1) \uplus vars(t_{2px}) \vdash$$

$$R_{12}(s_1, s_{2p}) \wedge R_{23}(s_{2p}, s_3) \wedge g_1 \implies$$

$$\bigvee_{x \in X} \left(\begin{array}{l} \alpha_1 = \alpha_{2px} \wedge \bigwedge\limits_{j \in J'_{2px} \cap H} \beta_{1j} = \beta_{2pxj} \wedge R_{12}\big(s'_1, s'_{2px}\big) \{\!\!\{ \psi_1 \uplus \psi_{2px} \}\!\!\} \\ \wedge \bigvee\limits_{y \in Y} \left(\begin{array}{l} \alpha_{2px} = \alpha_{3pxy} \wedge \bigwedge\limits_{j \in J'_{3pxy} \cap H'} \beta_{2pxj} = \beta_{3pxyj} \wedge \\ g_{3pxy} \wedge R_{23}\big(s'_{2px}, s'_{3pxy}\big) \{\!\!\{ \psi_{2px} \uplus \psi_{3pxy} \}\!\!\} \end{array} \right) \end{array} \right)$$

With the rearrangement of the formula, we get:

$$V_1 \uplus V_2 \uplus V_3 \uplus vars(t_1) \uplus vars(t_{2px}) \vdash$$

$$R_{12}(s_1, s_{2p}) \wedge R_{23}(s_{2p}, s_3) \wedge g_1 \implies$$

$$\bigvee_{x \in X} \bigvee_{y \in Y} \left(\begin{array}{l} \alpha_1 = \alpha_{3pxy} \wedge \bigwedge\limits_{j \in J'_{3pxy} \cap (H \cap H')} \beta_{1j} = \beta_{3jpxy} \wedge g_{3pxy} \wedge \\ R_{12}\big(s'_1, s'_{2px}\big) \{\!\!\{ \psi_1 \uplus \psi_{2px} \}\!\!\} \wedge R_{23}\big(s'_{2px}, s'_{3pxy}\big) \{\!\!\{ \psi_{2px} \uplus \psi_{3pxy} \}\!\!\} \end{array} \right)$$

Because of the domain of the substitutions of the relations, the formula can be re-written:

$$V_1 \uplus V_2 \uplus V_3 \uplus vars(t_1) \uplus vars(t_{2px}) \vdash$$

$$R_{12}(s_1, s_{2p}) \wedge R_{23}(s_{2p}, s_3) \wedge g_1 \implies$$

$$\bigvee_{x \in X} \bigvee_{y \in Y} \left(\begin{array}{l} \alpha_1 = \alpha_{3pxy} \wedge \bigwedge\limits_{j \in J'_{3pxy} \cap (H \cap H')} \beta_{1j} = \beta_{3jpxy} \wedge g_{3pxy} \wedge \\ \big(R_{12}\big(s'_1, s'_{2px}\big) \wedge R_{23}\big(s'_{2px}, s'_{3pxy}\big)\big) \{\!\!\{ \psi_1 \uplus \psi_{2px} \uplus \psi_{3pxy} \}\!\!\} \end{array} \right)$$

The formula is valid for all $s_{2p} \in (s_{2p})^{p \in P}$. To build R_{13} we need to rely on the disjunction of all possible paths to relate s_1 and s_3, which leads to $R_{13}(s_1, s_3) = \bigvee\limits_{p \in P} (R_{12}(s_1, s_{2p}) \wedge R_{23}(s_{2p}, s_3))$.

$$V_1 \uplus V_2 \uplus V_3 \uplus vars(t_1) \uplus \biguplus_{p \in P} vars(t_{2px}) \vdash$$

$$\bigvee_{p \in P} (R_{12}(s_1, s_{2p}) \wedge R_{23}(s_{2p}, s_3) \wedge g_1) \implies$$

$$\bigvee_{p \in P} \bigvee_{x \in X} \bigvee_{y \in Y} \left(\begin{array}{l} \alpha_1 = \alpha_{3pxy} \wedge \bigwedge\limits_{j \in J'_{3pxy} \cap (H \cap H')} \beta_{1j} = \beta_{3jpxy} \wedge g_{3pxy} \wedge \\ \underbrace{R_{12}\big(s'_1, s'_{2px}\big) \wedge R_{23}\big(s'_{2px}, s'_{3pxy}\big)}_{\implies R_{13}\left(s'_1, s'_{3pxy}\right)} \{\!\!\{ \psi_1 \uplus \psi_{2px} \uplus \psi_{3pxy} \}\!\!\} \end{array} \right)$$

Indeed, since ψ_{2px} has no effect on variables of A_1 and A_3 we have:

$$\begin{aligned} R_{12}\big(s'_1, s'_{2px}\big) \wedge R_{23}\big(s'_{2px}, s'_{3pxy}\big)\{\!\{\psi_1 \uplus \psi_{2px} \uplus \psi_{3pxy}\}\!\} \\ \Longrightarrow R_{13}\big(s'_1, s'_{3pxy}\big)\{\!\{\psi_1 \uplus \psi_{2px} \uplus \psi_{3pxy}\}\!\} \\ \Longrightarrow R_{13}\big(s'_1, s'_{3pxy}\big)\{\!\{\psi_1 \uplus \psi_{3pxy}\}\!\} \end{aligned}$$

This allows us to conclude that the family of open transitions of A_3 indexed over p, x, and y simulates the original open transition with note that ψ_{2px} has no effect on variables of A_1 and A_3:

$$V_1 \uplus V_3 \uplus vars(t_1) \vdash$$

$$R_{13}(s_1, s_3) \wedge g_1 \implies$$

$$\bigvee_{p\in P}\bigvee_{x\in X}\bigvee_{y\in Y}\left(\begin{array}{c}\alpha_1 = \alpha_{3pxy} \wedge \bigwedge\limits_{j\in J'_{3pxy}\cap(H\cap H')} \beta_{1j} = \beta_{3jpxy} \\ \wedge g_{3pxy} \wedge R_{13}\big(s'_1, s'_{3pxy}\big)\{\!\{\psi_1 \uplus \psi_{3pxy}\}\!\}\end{array}\right)$$

Overall, we have a family of open transitions $t_{pxy}^{p\in P, x\in X, y\in Y} \subseteq T_3$ that should simulate t_1. All combinations of elements in P, X, and Y provide a set Z of open transitions. This allows us to get the desired conclusion, that there is a set of open transitions indexed over Z:

$$V_1 \uplus V_3 \uplus vars(t_1) \vdash$$

$$\left(R_{13}(s_1, s_3) \wedge g_1 \implies \bigvee_{z\in Z}\left(\begin{array}{c}\alpha_1 = \alpha_{3z} \wedge \bigwedge\limits_{j\in J'_{3z}\cap(H\cap H')} \beta_{1j} = \beta_{3jz} \wedge \\ g_{3z} \wedge R_{13}\big(s'_1, s'_{3z}\big)\{\!\{\psi_1 \uplus \psi_{3z}\}\!\}\end{array}\right)\right)$$

3. Finally, we have to prove the following holds:

$$\forall (s_1, s_3) \in S_1 \times S_3, \forall (t_1, t_3) = \left(\frac{\beta_{1j}^{j\in J'_1}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s'_1}, \frac{\beta_{3j}^{j\in J'_3}, g_3, \psi_3}{s_3 \xrightarrow{\alpha_3} s'_3}\right) \in \mathrm{OT}(s_1) \times \mathrm{OT}(s_3)$$

$$\exists\left(t_{1x} = \frac{\beta_{1xj}^{j\in J'_{1x}}, g_{1x}, \psi_{1x}}{s_1 \xrightarrow{\alpha_{1x}} s'_{1x}} \in \mathrm{OT}(s_1)\right)^{x\in X}, V_1 \uplus V_3 \uplus \biguplus_{x\in X} vars(t_{1x}) \uplus vars(t_1) \uplus vars(t_3) \vdash$$

$$R_{13}(s_1, s_3) \wedge g_1 \wedge g_3 \wedge \alpha_1 = \alpha_3 \implies$$

$$\bigwedge_{x\in X}\left(\alpha_1 = \alpha_{1x} \wedge J'_{1x} = J'_3 \wedge \bigwedge_{j\in J'_3} \beta_{3j} = \beta_{1xj} \wedge g_{1x} \wedge R_{13}\big(s'_{1x}, s'_3\big)\{\!\{\psi_{1x} \uplus \psi_3\}\!\}\right)$$

Consider $(s_1, s_3) \in R_{13}$ then there is a set of states $(s_{2p})^{p\in P}$ of A_2 relating s_1 and s_3:

$$R_{13}(s_1, s_3) = \bigvee_{p\in P}\left(R_{12}(s_1, s_{2p}) \wedge R_{23}(s_{2p}, s_3)\right)$$

Let's consider any $s_{2p} \in (s_{2p})^{p \in P}$. According to the refinement relation between A_1 and A_2 and Definition 5 we have:

$$\forall (s_1, s_{2p}) \in S_1 \times S_2, \forall (t_1, t_{2p}) = \left(\frac{\beta_{1j}^{j \in J_1'}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s_1'}, \frac{\beta_{2pj}^{j \in J_{2p}'}, g_{2p}, \psi_{2p}}{s_{2p} \xrightarrow{\alpha_{2p}} s_{2p}'} \right) \in \mathrm{OT}(s_1) \times \mathrm{OT}(s_{2p})$$

$$\exists \left(t_{1x} = \frac{\beta_{1xj}^{j \in J_{1x}'}, g_{1x}, \psi_{1x}}{s_1 \xrightarrow{\alpha_{1x}} s_{1x}'} \in \mathrm{OT}(s_1) \right)^{x \in X}, V_1 \uplus V_2 \uplus \biguplus_{x \in X} vars(t_{1x}) \uplus vars(t_1) \uplus vars(t_{2p}) \vdash$$

$$R_{12}(s_1, s_{2p}) \wedge g_1 \wedge g_{2p} \wedge \alpha_1 = \alpha_{2p} \implies$$

$$\bigwedge_{x \in X} \left(\alpha_1 = \alpha_{1x} \wedge J_{1x}' = J_{2p}' \wedge \bigwedge_{j \in J_{2p}'} \beta_{2pj} = \beta_{1xj} \wedge g_{1x} \wedge R_{12}\left(s_{1x}', s_{2p}'\right) \{\!\{ \psi_{1x} \uplus \psi_{2p} \}\!\} \right)$$

Furthermore, according to the refinement relation between A_2 and A_3 and Definition 7 we have for any open transition t_{2px} in T_2 originating from s_{2p} the following:

$$\frac{\beta_{2pxj}^{j \in J_{2px}'}, g_{2px}, \psi_{2px}}{s_{2p} \xrightarrow{\alpha_{2px}} s_{2px}'} \in \mathrm{OT}(s_{2p})$$

there exists an indexed family of OTs originating from s_3:

$$\left(\frac{\beta_{3pyj}^{j \in J_{3py}'}, g_{3py}, \psi_{3py}}{s_3 \xrightarrow{\alpha_{3py}} s_{3py}'} \in \mathrm{OT}(s_3) \right)^{y \in Y}$$

such that $\forall y \in Y, J_{2p}' \cap H = J_{3py}' \cap H$ and

$$V_2 \uplus V_3 \uplus vars(t_{2p}) \vdash$$

$$R_{23}(s_{2p}, s_3) \wedge g_{2p} \implies \bigvee_{y \in Y} \left(\begin{array}{c} \alpha_{2p} = \alpha_{3py} \wedge \bigwedge_{j \in J_{3py}' \cap H} \beta_{2pj} = \beta_{3pyj} \wedge \\ g_{3py} \wedge R_{23}\left(s_{2p}', s_{3y}'\right) \{\!\{ \psi_{2p} \uplus \psi_{3py} \}\!\} \end{array} \right)$$

Based on the above two properties, we get:

$$R_{12}(s_1, s_{2p}) \wedge R_{23}(s_{2p}, s_3) \wedge g_1 \wedge g_{2p} \wedge \alpha_1 = \alpha_{2p} \implies$$

$$\bigwedge_{x \in X} \bigvee_{y \in Y} \left(\left(\begin{array}{c} \alpha_{2p} = \alpha_{3py} \wedge \bigwedge_{j \in J_{3py}' \cap H} \beta_{2pj} = \beta_{3pyj} \wedge \\ g_{3py} \wedge R_{23}\left(s_{2p}', s_{3y}'\right) \{\!\{ \psi_{2p} \uplus \psi_{3py} \}\!\} \end{array} \right) \wedge \left(\begin{array}{c} \alpha_1 = \alpha_{1x} \wedge J_{1x}' = J_{2p}' \wedge \bigwedge_{j \in J_{2p}'} \beta_{2pj} = \beta_{1xj} \wedge \\ g_{1x} \wedge R_{12}\left(s_{1x}', s_{2p}'\right) \{\!\{ \psi_{1x} \uplus \psi_{2p} \}\!\} \end{array} \right) \right)$$

By rearranging the terms in the formula we get:

$$R_{12}(s_1,s_{2p})\wedge R_{23}(s_{2p},s_3)\wedge g_1\wedge g_{2p}\wedge \bigvee_{y\in Y}(\alpha_1=\alpha_{2p}\wedge\alpha_{2p}=\alpha_{3py}\wedge g_{3py}) \implies$$

$$\bigwedge_{x\in X}\bigvee_{y\in Y}\begin{pmatrix}\alpha_1=\alpha_{1x}\wedge J'_{1x}=J'_{2p}\wedge\bigwedge_{j\in J'_{2p}}\beta_{2pj}=\beta_{1xj}\wedge\bigwedge_{j\in J'_{3py}\cap H}\beta_{2pj}=\beta_{3pyj}\\ g_{1x}\wedge R_{12}\big(s'_{1x},s'_{2p}\big)\wedge R_{23}\big(s'_{2p},s'_{3y}\big)\{\!\{\psi_{1x}\uplus\psi_{2p}\uplus\psi_{3py}\}\!\}\end{pmatrix}$$

The formula is valid for all $s_{2p}\in(s_{2p})^{p\in P}$. To build R_{13} we need to rely on the disjunction of all possible path to relate s_1 and s_3:

$$\bigvee_{p\in P}\left(R_{12}(s_1,s_{2p})\wedge R_{23}(s_{2p},s_3)\wedge g_1\wedge g_{2p}\wedge\bigvee_{y\in Y}(\alpha_1=\alpha_{3py}\wedge g_{3py})\right)\implies$$

$$\bigwedge_{x\in X}\bigvee_{p\in P}\bigvee_{y\in Y}\begin{pmatrix}\alpha_1=\alpha_{1x}\wedge J'_{1x}=J'_{2p}\wedge\bigwedge_{j\in J'_{2p}}\beta_{2pj}=\beta_{1xj}\wedge\bigwedge_{j\in J'_{3py}\cap H}\beta_{2pj}=\beta_{3pyj}\\ g_{1x}\wedge R_{12}\big(s'_{1x},s'_{2p}\big)\wedge R_{23}\big(s'_{2p},s'_{3y}\big)\{\!\{\psi_{1x}\uplus\psi_{2p}\uplus\psi_{3py}\}\!\}\end{pmatrix}$$

Firstly, according to the refinement relation between A_1 and A_2 we have:

$$\forall(s_1,s_2)\in S_1\times S_2,\forall(t_1,t_2)=\left(\frac{\beta_{1j}^{j\in J'_1},g_1,\psi_1}{s_1\xrightarrow{\alpha_1}s'_1},\frac{\beta_{2j}^{j\in J'_2},g_2,\psi_2}{s_2\xrightarrow{\alpha_2}s'_2}\right)\in \mathrm{OT}(s_1)\times\mathrm{OT}(s_2)$$

$$\exists\left(t_{1x}=\frac{\beta_{1xj}^{j\in J'_{1x}},g_{1x},\psi_{1x}}{s_1\xrightarrow{\alpha_{1x}}s'_{1x}}\in\mathrm{OT}(s_1)\right)^{x\in X},V_1\uplus V_2\uplus\biguplus_{x\in X}vars(t_{1x})\uplus vars(t_1)\uplus vars(t_2)\vdash$$

$$R_{12}(s_1,s_2)\wedge g_1\wedge g_2\wedge\alpha_1=\alpha_2\implies$$

$$\bigwedge_{x\in X}\left(\alpha_1=\alpha_{1x}\wedge J'_{1x}=J'_2\wedge\bigwedge_{j\in J'_2}\beta_{2j}=\beta_{1xj}\wedge g_{1x}\wedge R_{12}\big(s'_{1x},s'_2\big)\{\!\{\psi_{1x}\uplus\psi_2\}\!\}\right)$$

Yet according to the refinement relation between A_2 and A_3 and Definition 7 we have:

$$\forall(s_2,s_3)\in S_1\times S_3,\forall(t_2,t_3)=\left(\frac{\beta_{2j}^{j\in J'_2},g_2,\psi_2}{s_2\xrightarrow{\alpha_2}s'_2},\frac{\beta_{3j}^{j\in J'_3},g_3,\psi_3}{s_3\xrightarrow{\alpha_3}s'_3}\right)\in\mathrm{OT}(s_2)\times\mathrm{OT}(s_3)$$

$$\exists\left(t_{2x}=\frac{\beta_{2xj}^{j\in J'_{2x}},g_{2x},\psi_{2x}}{s_2\xrightarrow{\alpha_{2x}}s'_{2x}}\in\mathrm{OT}(s_2)\right)^{x\in X},V_2\uplus V_3\uplus\biguplus_{x\in X}vars(t_{2x})\uplus vars(t_3)\uplus vars(t_2)\vdash$$

$$R_{23}(s_2,s_3)\wedge g_2\wedge g_3\wedge\alpha_2=\alpha_3\implies$$

$$\bigwedge_{x\in X}\left(\alpha_2=\alpha_{2x}\wedge J'_{2x}=J'_3\wedge\bigwedge_{j\in J'_3}\beta_{3j}=\beta_{2xj}\wedge g_{2x}\wedge R_{23}\big(s'_{2x},s'_3\big)\{\!\{\psi_{2x}\uplus\psi_3\}\!\}\right)$$

Note that if either $R_{12}(s_1, s_2)$ or $R_{23}(s_2, s_3)$ is false the statement trivially holds. So for each $s_2 \in S_2$, we remark that s_2 allows to relate s_1 and s_3 if and only if $R_{12}(s_1, s_2) \wedge R_{23}(s_2, s_3)$. We pick a state s_2 relating both, we can then the following be inferred from the above formulas: $\forall (s_1, s_3) \in S_1 \times S_3$ and

$$\forall (t_1, t_3) = \left(\frac{\beta_{1j}^{j \in J'_1}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s'_1}, \frac{\beta_{3j}^{j \in J'_3}, g_3, \psi_3}{s_3 \xrightarrow{\alpha_3} s'_3} \right) \in \mathrm{OT}(s_1) \times \mathrm{OT}(s_3)$$

$$\exists \left(t_{1x} = \frac{\beta_{1xj}^{j \in J'_{1x}}, g_{1x}, \psi_{1x}}{s_1 \xrightarrow{\alpha_{1x}} s'_{1x}} \in \mathrm{OT}(s_1) \right)^{x \in X},$$

$$V_1 \uplus V_2 \uplus V_3 \uplus vars(t_1) \uplus vars(t_2) \uplus vars(t_3) \uplus \biguplus_{x \in X} vars(t_{1x}) \vdash$$

$$R_{12}(s_1, s_2) \wedge R_{23}(s_2, s_3) \wedge g_1 \wedge g_2 \wedge g_3 \wedge \alpha_1 = \alpha_2 \wedge \alpha_2 = \alpha_3 \implies$$

$$\bigwedge_{x \in X} \left(\alpha_1 = \alpha_{1x} \wedge J'_{1x} = J'_2 \wedge \bigwedge_{j \in J'_2} \beta_{2j} = \beta_{1xj} \wedge g_{1x} \wedge R_{12}(s'_{1x}, s'_2) \{\!\{ \psi_{1x} \uplus \psi_2 \}\!\} \right) \wedge$$

$$R_{23}(s'_2, s'_3) \{\!\{ \psi_2 \uplus \psi_3 \}\!\}$$

By rearranging, simplifying and removing the variables of A_2 that have not effect on A_1 et A_3, we get the desired result.

□

B Proof of Context Refinement (Theorem 2)

Suppose that $A_1 \leqslant_H A_2$, $k \in H$ and that $A_1[A_3/k]$ is non-blocking. We have:

$$A_1[A_3/k] \leqslant_{J_3 \uplus H \setminus \{k\}} A_2[A_3/k]$$

[TODO:The subtlety here is that the original relation implies that valuations in A3 are equal (ie the value for each variable are the same in both valuations, modulo renaming), after a transition we should obtain "equal" valuations because Post are deterministic]

Proof. Let us denote by A_{13} (resp. A_{23}) the open automaton resulting from $A_1[A_3/k]$ (resp. $A_2[A_3/k]$), to prove the theorem it is sufficient to prove that there exists a relation between states of the two open automata that satisfies the conditions of the Definition 7.

We denote $A_1 = \langle S_1, s_{01}, J_1, V_1, \sigma_{01}, T_1 \rangle$ and $A_2 \coloneqq \langle S_2, s_{02}, J_2, V_2, \sigma_{02}, T_2 \rangle$ and $A_3 = \langle S_3, s_{03}, J_3, V_3, \sigma_{03}, T_3 \rangle$. The proof requires to rename the variables of one instance of the two A_3 automata to avoid clashes in variable names (this is required by the definition of refinement). In practice we will use superscripts ^{1} and ^{2} to distinguish elements of the two instances of A_3.

Let R be the refinement relation relating states of A_1 and A_2. Let us denote with t^1 and t^2 the elements of A_1 and A_2 respectively. Consider any two states $s_{13} = \left(s_1, s_3^1\right)$ and $s_{23} = \left(s_2, s_3^2\right)$ (s_3^1 and s_3^2 are the same with renaming). We define a relation R' relating states of s_{13} and s_{23} as follows:

$$R'(s_{13}, s_{23}) = R(s_1, s_2) \wedge \bigwedge_{v_3 \in V_3} v_3^1 = v_3^2 \wedge s_3^1 = s_3^2$$

We want to prove that $(R', H \uplus J_3 \setminus \{k\})$ is a hole-tracking simulation of A_{13} and A_{23}. In the following we denote $H' = H \cup J_3 \setminus \{k\}$.

1. First, we have to prove the relation for initial states:

   $$\sigma_{013} \uplus \sigma_{023} \vdash R'(s_{013}, s_{023})$$

   with $\sigma_{013} = \sigma_{01} \uplus \sigma_{03}^1$, $\sigma_{023} = \sigma_{02} \uplus \sigma_{03}^2$, $s_{013} = (s_{01}, s_{03}^1)$, and $s_{023} = (s_{02}, s_{02}^2)$.
   By using the fact that R relates initial configurations of A_1 and A_2, we have: $(\sigma_{01} \uplus \sigma_{02} \vdash R(s_{01}, s_{02}))$.
   Considering that initial valuations σ_{03}^1 and σ_{03}^2 associate the same values to the "same" variables modulo renaming, so the following holds:
   $\left(\bigwedge_{v_3 \in V_3} v_3^1 = v_3^2 \wedge s_3^1 = s_3^2\right) \{\!\{\sigma_{03}^1 \uplus \sigma_{03}^2\}\!\}$.
   Additionally, because the domains of the substitution function are disjoint, the substitution function has an effect only on the related elements, we get:

   $$\begin{aligned}
   & (\sigma_{01} \uplus \sigma_{02} \vdash R(s_{01}, s_{02})) \\
   \Longrightarrow\ & R(s_{01}, s_{02})\{\!\{\sigma_{01} \uplus \sigma_{02}\}\!\} \wedge \left(\bigwedge_{v_3 \in V_3} v_3^1 = v_3^2 \wedge s_3^1 = s_3^2\right) \{\!\{\sigma_{03}^1 \uplus \sigma_{03}^2\}\!\} \\
   \Longrightarrow\ & \left(R(s_{01}, s_{02}) \wedge \bigwedge_{v_3 \in V_3} v_3^1 = v_3^2 \wedge s_3^1 = s_3^2\right) \{\!\{\sigma_{01} \uplus \sigma_{02} \uplus \sigma_{03}^1 \uplus \sigma_{03}^2 \uplus \sigma_{013}\}\!\} \\
   \Longrightarrow\ & \sigma_{013} \uplus \sigma_{023} \vdash R'(s_{013}, s_{023})
   \end{aligned}$$

2. Second, we need to prove for any OT t_{13} in T_{13} originating from s_{13}:

   $$\frac{\beta_{13j}^{j \in J'_{13}}, g_{13}, \psi_{13}}{s_{13} \xrightarrow{\alpha_{13}} s'_{13}} \in \mathrm{OT}(s_{13})$$

   there exists an indexed family t_{23x} of OTs originating from s_{23} that simulate it:

   $$\left(\frac{\beta_{23xj}^{j \in J'_{23x}}, g_{23x}, \psi_{23x}}{s_{23} \xrightarrow{\alpha_{23x}} s'_{23x}} \in \mathrm{OT}(s_{23})\right)^{x \in X}$$

such that $\left(\forall x \in X, J'_{23x} \cap H' = J'_{13} \cap H'\right)$ and

$$V_{13} \uplus V_{23} \uplus vars(t_{13}) \vdash R'(s_{13}, s_{23}) \wedge g_{13} \implies \bigvee_{x \in X} \begin{pmatrix} \alpha_{13} = \alpha_{23x} \wedge \bigwedge_{j \in J'_{23x} \cap H'} \beta_{13j} = \beta_{23xj} \wedge \\ g_{23x} \wedge R'\left(s'_{13}, s'_{23x}\right) \{\!\{ \psi_{13} \uplus \psi_{23x} \}\!\} \end{pmatrix}$$

Recall that by definition of composition and open automata refinement we have:

$$V_{13} = V_1 \uplus V_3^1 \text{ and } V_{23} = V_2 \uplus V_3^2$$
$$H' \subseteq J_3 \uplus (J_1 \cap J_2) \setminus \{k\} = (J_3 \uplus J_1 \setminus \{k\}) \cap (J_3 \uplus J_2 \setminus \{k\})$$

First of all, we have by hypothesis $A_1 \leqslant_H A_2$, then for any open transition t_1 in T_1 originating from s_1:

$$\frac{\beta_{1j}^{j \in J'_1}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s'_1} \in \mathrm{OT}(s_1)$$

there exists an indexed family of OTs originating from s_2:

$$\left(\frac{\beta_{2xj}^{j \in J'_{2x}}, g_{2x}, \psi_{2x}}{s_2 \xrightarrow{\alpha_{2x}} s'_{2x}} \in \mathrm{OT}(s_2) \right)^{x \in X}$$

such that $\forall x \in X, J'_{2x} \cap H = J'_1 \cap H$ and

$$V_1 \uplus V_2 \uplus vars(t_1) \vdash$$
$$\left(R(s_1, s_2) \wedge g_1 \implies \bigvee_{x \in X} \begin{pmatrix} \alpha_1 = \alpha_{2x} \wedge \bigwedge_{j \in J'_{2x} \cap H} \beta_{1j} = \beta_{2xj} \wedge \\ g_{2x} \wedge R\left(s'_1, s'_{2x}\right) \{\!\{ \psi_1 \uplus \psi_{2x} \}\!\} \end{pmatrix} \right) \qquad (*)$$

Consider any transition t_{13} in A_{13}. Based on the definition of composition t_{13} can be obtained from two different cases, we will consider the two cases separately.

First case: Both automata perform a transition. The transition t_{13} is obtained by the composition of transitions $t_1 = \frac{\beta_{1j}^{j \in J'_1}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s'_1} \in \mathrm{OT}(s_1)$ and

$$t_3^1 = \frac{(\beta_{3j}^1)^{j \in J_3^{1\prime}}, g_3^1, \psi_3^1}{s_3^1 \xrightarrow{\alpha_3^1} s_3^{1\prime}} \in \mathrm{OT}\left(s_3^1\right) \qquad \text{when } k \in J'_1$$

The result is:

$$t_{13} = \frac{\beta_{1j}^{j \in J'_1 \setminus \{k\}} \uplus (\beta_{3j}^1)^{j \in J_3^{1\prime}}, g_1 \wedge g_3^1 \wedge \alpha_3^1 = \beta_{1k}, \psi_1 \uplus \psi_3^1}{\left(s_1, s_3^1\right) \xrightarrow{\alpha_1} \left(s'_1, s_3^{1\prime}\right)} \qquad \text{where } k \in J'_1$$

We then obtain a family of OTs by the simulation of A_1 by A_2 (as stated above). By hypothesis we have $k \in H$, so in the case where $k \in J'_1$, we deduce that $k \in J'_{2x}$ we can then build a family of OTs $t_{23x}^{x \in X}$ with the same transitions of A_3 (up to renaming) as those used to build t_{13}.

$$t_{23x} = \left(\frac{\beta_{2xj}^{j \in J'_{2x} \setminus \{k\}} \uplus (\beta_{3j}^2)^{j \in J_3^{2'}}, g_{2x} \wedge g_3^2 \wedge \alpha_3^2 = \beta_{2xk}, \psi_{2x} \uplus \psi_3^2}{(s_2, s_3^2) \xrightarrow{\alpha_{2x}} \left(s'_{2x}, s_3^{2'}\right)} \right)^{x \in X}$$

Recall that in this case $k \in J'_1$, so $\forall x \in X$ we have

$$\begin{aligned}
J'_{23x} \cap H' &= ((J'_{2x} \setminus \{k\}) \uplus J_3^{2'}) \cap (H \uplus J_3 \setminus \{k\}) \\
&= ((J'_{2x} \cap (H \uplus J_3)) \uplus (J_3^{2'} \cap (H \uplus J_3))) \setminus \{k\} \\
&= ((J'_{2x} \cap H) \uplus (J_3^{2'} \cap J_3)) \setminus \{k\} \text{ since } J_3 \cap J'_{2x} = \varnothing \text{ and } H \cap J_3^{2'} = \varnothing \\
&= ((J'_{2x} \cap H) \uplus J_3^{2'}) \setminus \{k\} \text{ since } J_3^{2'} \subseteq J_3 \\
&= ((J'_1 \cap H) \uplus J_3^{1'}) \setminus \{k\} \quad \text{since } J_3^{1'} = J_3^{2'} \text{ and } J'_1 \cap H = J'_{2x} \cap H \\
&= ((J'_1 \cap H) \uplus (J_3^{1'} \cap J_3)) \setminus \{k\} \text{ since } J_3^{1'} \subseteq J_3 \\
&= (J'_1 \cap (J_3 \uplus H)) \uplus ((J_3^{1'} \cap (J_3 \uplus H)) \setminus \{k\} \text{ since } J_3 \cap J'_1 = \varnothing \text{ and } H \cap J_3^{1'} = \varnothing \\
&= \left((J'_1 \uplus J_3^{1'}) \setminus \{k\}\right) \cap ((J_3 \uplus H) \setminus \{k\}) \\
&= J'_{13} \cap H'
\end{aligned}$$

In this case the composition gives:

$$g_{13} \Leftrightarrow g_1 \wedge g_3^1 \wedge \alpha_3^1 = \beta_{1k} \text{ and } g_{23x} \Leftrightarrow g_{2x} \wedge g_3^2 \wedge \alpha_3^2 = \beta_{2xk}$$

As $k \in H$ we have $\beta_{1k} = \beta_{2xk}$ then we deduce:

$$g_3^1 \wedge \alpha_3^1 = \beta_{1k} \Leftrightarrow g_3^2 \wedge \alpha_3^2 = \beta_{2xk}$$

The proof of the rest is based on the following facts:

(a) Because composition doesn't change the resulting actions, nor their variables, we can extend the valuation context of the variables to cover the variables of the transition t_3. By construction of t_{13} and t_{23} we have $\alpha_{13} = \alpha_1$ and $\alpha_{23x} = \alpha_{2x}$. So we deduce: $\alpha_1 = \alpha_{2x} \Rightarrow \alpha_{13} = \alpha_{23x}$.

(b) By composition we have also:
$\beta_{13j}^{j \in J'_{13}} = \beta_{1j}^{j \in J'_1 \setminus \{k\}} \uplus (\beta_{3j}^1)^{j \in J_3^{1'}}$ and $\beta_{23xj}^{j \in J'_{23}} = \beta_{2xj}^{j \in J'_{2x} \setminus \{k\}} \uplus (\beta_{3j}^2)^{j \in J_3^{2'}}$
Therefore, we have for all $j \in J'_{13}$ (recall that $J'_{13} = J'_{23}$):

$$\beta_{13j} = \beta_{23xj} \Rightarrow (j \in J_1 \wedge \beta_{1j} = \beta_{2xj}) \vee (j \in J_3^1 \wedge \beta_{3j}^1 = \beta_{3j}^2)$$

(c) Considering β_{3j}^1 and β_{3j}^2 are the same (up to renaming) we have:
$V_3^1 \uplus V_3^2 \uplus vars(t_{13}) \vdash \bigwedge_{v_3 \in V_3} v_3^1 = v_3^2 \implies \bigwedge_{j \in J_3^{2'}} \beta_{3j}^1 = \beta_{3j}^2$

The disjunction with the following hypothesis:
$V_1 \uplus V_2 \uplus vars(t_1) \vdash \bigwedge_{j \in J'_{2x} \cap H} \beta_{1j} = \beta_{2xj}$ will give:

$$V_1 \uplus V_2 \uplus V_3^1 \uplus V_3^2 \uplus vars(t_{13}) \vdash \bigwedge_{v_3 \in V_3} v_3^1 = v_3^2 \implies \bigwedge_{j \in J'_{2x} \cap H} \beta_{1j} = \beta_{2xj} \vee \bigwedge_{j \in J_3^{2\prime}} \beta_{3j}^1 = \beta_{3j}^2$$

$$\Rightarrow V_{13} \uplus V_{23} \uplus vars(t_{13}) \vdash \bigwedge_{v_3 \in V_3} v_3^1 = v_3^2 \implies \bigwedge_{j \in (J'_{2x} \cap H) \uplus J_3^{2\prime}} \beta_{13j} = \beta_{23xj}$$

$$\Rightarrow V_{13} \uplus V_{23} \uplus vars(t_{13}) \vdash \bigwedge_{v_3 \in V_3} v_3^1 = v_3^2 \implies \bigwedge_{j \in \Rightarrow ((J'_{2x} \cap H) \uplus J_3^{2\prime}) \setminus \{k\}} \beta_{13j} = \beta_{23xj}$$

$$\Rightarrow V_{13} \uplus V_{23} \uplus vars(t_{13}) \vdash \bigwedge_{v_3 \in V_3} v_3^1 = v_3^2 \implies \bigwedge_{j \in (J'_{23x} \cap H')} \beta_{13j} = \beta_{23xj}$$

By the extension of the valuation context mentioned above in the formula $(*)$ and by using the statements resulting from the cases (a), (b) and (c), we get:

$$V_{13} \uplus V_{23} \uplus vars(t_{13}) \vdash$$

$$R(s_1, s_2) \wedge \big(g_1 \wedge g_3^1 \wedge \alpha_3^1 = \beta_{1k}\big) \wedge \bigwedge_{v_3 \in V_3} v_3^1 = v_3^2 \implies$$

$$\bigvee_{x \in X} \left(\begin{array}{c} \alpha_{13} = \alpha_{23x} \wedge \bigwedge_{j \in (J'_{23x} \cap H')} \beta_{13j} = \beta_{23xj} \\ \wedge\ g_{2x} \wedge R\big(s'_1, s'_{2x}\big) \{\!\{ \psi_1 \uplus \psi_{2x} \}\!\} \end{array} \right) \wedge g_3^2 \wedge \alpha_3^2 = \beta_{2xk}$$

That can be re-written as follows:

$$V_{13} \uplus V_{23} \uplus vars(t_{13}) \vdash$$

$$R(s_1, s_2) \wedge g_{13} \wedge \bigwedge_{v_3 \in V_3} v_3^1 = v_3^2 \implies \bigvee_{x \in X} \left(\begin{array}{c} \alpha_{13} = \alpha_{23x} \wedge \bigwedge_{j \in J'_{23x} \cap H'} \beta_{13j} = \beta_{23xj} \\ \wedge\ g_{23x} \wedge R\big(s'_1, s'_{2x}\big) \{\!\{ \psi_1 \uplus \psi_{2x} \}\!\} \end{array} \right)$$

Moreover, we have for any transition t_3 in A_3 relying s_3 and s'_3 the following:

$$V_{13} \uplus V_{23} \uplus vars(t_{13}) \vdash$$

$$\bigwedge_{v_3 \in V_3} v_3^1 = v_3^2 \wedge s_3^1 = s_3^2 \implies \bigwedge_{v_3 \in V_3} \psi_{13}(v_3^1) = \psi_{13}(v_3^2) \wedge s_3^{1\prime} = s_3^{2\prime}$$

From the two previous formulas, we get:

$$V_{13} \uplus V_{23} \uplus vars(t_{13}) \vdash$$
$$R(s_1, s_2) \wedge g_{13} \wedge \bigwedge_{v_3 \in V_3} v_3^1 = v_3^2 \wedge s_3^1 = s_3^2 \implies$$
$$\bigvee_{x \in X} \left(\begin{array}{l} \alpha_{13} = \alpha_{23x} \wedge \bigwedge_{j \in J'_{23x} \cap H'} \beta_{13j} = \beta_{23xj} \wedge g_{23x} \wedge \\ R\big(s'_1, s'_{2x}\big) \{\!\{\psi_1 \uplus \psi_{2x}\}\!\} \wedge \bigwedge_{v_3 \in V_3} \psi_{13}(v_3^1) = \psi_{13}(v_3^2) \wedge s_3^{1\prime} = s_3^{2\prime} \end{array} \right)$$

Because of the independence of the substitution domains, we simplify and get the expected formula:

$$V_{13} \uplus V_{23} \uplus vars(t_{13}) \vdash$$
$$R'(s_{13}, s_{23}) \wedge g_{13} \implies \bigvee_{x \in X} \left(\begin{array}{l} \alpha_{13} = \alpha_{23x} \wedge \bigwedge_{j \in J'_{23x} \cap H'} \beta_{13j} = \beta_{23xj} \\ \wedge\ g_{23x} \wedge R'\big(s'_{13}, s'_{23x}\big) \{\!\{\psi_{13} \uplus \psi_{23x}\}\!\} \end{array} \right)$$

Second case: Only the encompassing automaton performs a transition t_{13} is obtained by the transition $t_1 = \dfrac{\beta_{1j}^{j \in J'_1}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s'_1}$ alone with the state s_3^1 unchanged, if $k \notin J'_1$

$$t_{13} = \frac{\beta_{1j}^{j \in J'_1}, g_1, \psi_1}{\big(s_1, s_3^1\big) \xrightarrow{\alpha_1} \big(s'_1, s_3^1\big)}$$

We then obtain a family of OTs by the simulation of A_1 by A_2 (stated above). As $k \in H$ we have $k \notin J'_{2x}$ and we build:

$$t_{23x} = \left(\frac{\beta_{2xj}^{j \in J'_{2x}}, g_{2x}, \psi_{2x}}{\big(s_2, s_3^2\big) \xrightarrow{\alpha_{2x}} \big(s'_{2x}, s_3^2\big)} \right)^{x \in X}$$

These two cases define (depending on how t_{13} is built) the family t_{23x} of OTs we are looking for. Thus, for both cases we have to prove the following:

$\big(\forall x \in X, J'_{23x} \cap H' = J'_{13} \cap H'\big)$ and

$$V_{13} \uplus V_{23} \uplus vars(t_{13}) \vdash$$
$$R'(s_{13}, s_{23}) \wedge g_{13} \implies \bigvee_{x \in X} \left(\begin{array}{l} \alpha_{13} = \alpha_{23x} \wedge \bigwedge_{j \in J'_{23x} \cap H'} \beta_{13j} = \beta_{23xj} \\ \wedge\ g_{23x} \wedge R'\big(s'_{13}, s'_{23x}\big) \{\!\{\psi_{13} \uplus \psi_{23x}\}\!\} \end{array} \right)$$

Recall in this case $k \notin J'_1$, $\forall x \in X$ we have

$$\begin{aligned} J'_{23x} \cap H' &= J'_{2x} \cap (J_3 \uplus H \setminus \{k\}) \\ &= \left(J'_{2x} \cap H\right) \text{ since } J'_{2x} \cap J_3 = \varnothing \wedge k \notin J'_{2x} \\ &= \left(J'_1 \cap H\right) \text{ since } J'_1 \cap H = J'_{2x} \cap H \\ &= \left(J'_{13} \cap H'\right) \text{ since } J_3 \cap J'_1 = \varnothing \wedge k \notin J'_1 \end{aligned}$$

The proof of the rest of the formula follows the same steps as the previous case the only argument that changes is that by composition we obtain: $g_{13} \Leftrightarrow g_1$ and $g_{23x} \Leftrightarrow g_{2x}$.

3. The last formula we need to prove is that the relation is preserved along identical transitions, i.e. the following is satisfied (we define $V_{13} = V_1 \uplus V_3$ and $V_{23} = V_2 \uplus V_3$):

$$\forall (s_{13}, s_{23}) \in S_{13} \times S_{23}, \forall (t_{13}, t_{23}) = \left(\frac{\beta_{13j}^{j \in J'_{13}}, g_{13}, \psi_{13}}{s_{13} \xrightarrow{\alpha_{13}} s'_{13}}, \frac{\beta_{23j}^{j \in J'_{23}}, g_{23}, \psi_{23}}{s_{23} \xrightarrow{\alpha_{23}} s'_{23}} \right) \in \mathrm{OT}(s_{13}) \times \mathrm{OT}(s_{23})$$

$$\exists \left(t_{13x} = \frac{\beta_{13xj}^{j \in J'_{13x}}, g_{13x}, \psi_{13x}}{s_{13} \xrightarrow{\alpha_{13x}} s'_{13x}} \in \mathrm{OT}(s_{13}) \right)^{x \in X},$$

$$V_{13} \uplus V_{23} \uplus \biguplus_{x \in X} vars(t_{1x}) \uplus vars(t_{13}) \uplus vars(t_{23}) \vdash$$

$$R(s_{13}, s_{23}) \wedge g_{13} \wedge g_{23} \wedge \alpha_{13} = \alpha_{23} \implies$$

$$\bigwedge_{x \in X} \left(\alpha_{13} = \alpha_{13x} \wedge J'_{13x} = J'_{23} \wedge \bigwedge_{j \in J'_{23}} \beta_{23j} = \beta_{13xj} \wedge g_{13x} \wedge R\left(s'_{13x}, s'_{23}\right) \{\!\{ \psi_{13x} \uplus \psi_{23} \}\!\} \right)$$

We pick s_{13}, s'_{13a}, and two open transitions t_{13} and t_{23}. Where $s_{13} = (s_1, s_3)$ and $s_{23} = (s_2, s_3)$ by construction of R'.
We reason on the way the OT t_{23} has been obtained: either the hole k of A_2 is involved in the transition or not.

First case: Both A_2 and A_3 are involved in transition t_{23}. The transition t_{23} is obtained by the composition of transitions $t_2 = \frac{\beta_{2j}^{j \in J'_2}, g_2, \psi_2}{s_2 \xrightarrow{\alpha_2} s'_2} \in \mathrm{OT}(s_2)$ and

$$t_3^2 = \frac{(\beta_{3j}^2)^{j \in J_3^{2\prime}}, g_3^2, \psi_3^2}{s_3 \xrightarrow{\alpha_3^2} s_3^{2\prime}} \in \mathrm{OT}(s_3) \qquad \text{when } k \in J'_2$$

The result is:

$$t_{23} = \frac{\beta_{2j}^{j \in J'_2 \setminus \{k\}} \uplus (\beta_{3j}^2)^{j \in J_3^{2\prime}}, g_2 \wedge g_3^2 \wedge \alpha_3^2 = \beta_{2k}, \psi_2 \uplus \psi_3^2}{(s_2, s_3) \xrightarrow{\alpha_2} \left(s'_2, s_3^{2\prime}\right)} \qquad \text{where } k \in J'_2$$

Two cases are possible for t_{13}: either the hole k of A_1 is involved in the transition or not.

• if the hole k of A_1 is involved in the transition: The transition t_{13} is obtained by the composition of transitions $t_1 = \frac{\beta_{1j}^{j\in J_1'}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s_1'} \in \mathrm{OT}(s_1)$ and

$$t_3^1 = \frac{(\beta_{3j}^1)^{j\in J_3^{1\prime}}, g_3^1, \psi_3^1}{s_3 \xrightarrow{\alpha_3^1} s_3^{1\prime}} \qquad \text{when } k \in J_1'$$

The result is:

$$t_{13} = \frac{\beta_{1j}^{j\in J_1'\setminus\{k\}} \uplus (\beta_{3j}^1)^{j\in J_3^{1\prime}}, g_1 \wedge g_3^1 \wedge \alpha_3^1 = \beta_{1k}, \psi_1 \uplus \psi_3^1}{(s_1, s_3) \xrightarrow{\alpha_1} \left(s_1', s_3^{1\prime}\right)} \qquad \text{where } k \in J_1'$$

• if the hole k of A_1 is not involved in the transition: $t_1 = \frac{\beta_{1j}^{j\in J_1'}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s_1'}$ when $k \notin J_1'$ and

$$t_{13} = \frac{\beta_{1j}^{j\in J_1'}, g_1, \psi_1}{(s_1, s_3) \xrightarrow{\alpha_1} (s_1', s_3)}$$

In both cases note that $g_{13} \implies g_1$, with $(s_1, s_2) \in S_1 \times S_2$, Definition 5 applied to OTs t_1, t_2 gives the following property that we denote $(\star)$:

$$\exists\left(t_{1x} = \frac{\beta_{1xj}^{j\in J_{1x}'}, g_{1x}, \psi_{1x}}{s_1 \xrightarrow{\alpha_{1x}} s_{1x}'} \in \mathrm{OT}(s_1)\right)^{x\in X},$$

$$V_1 \uplus V_2 \uplus \biguplus_{x\in X} vars(t_{1x}) \uplus vars(t_1) \uplus vars(t_2) \vdash$$

$$R(s_1, s_2) \wedge g_1 \wedge g_2 \wedge \alpha_1 = \alpha_2 \implies$$

$$\bigwedge_{x\in X}\left(\alpha_1 = \alpha_{1x} \wedge J_{1x}' = J_2' \wedge \bigwedge_{j\in J_2'} \beta_{2j} = \beta_{1xj} \wedge g_{1x} \wedge R\left(s_{1x}', s_2'\right)\{\!\{\psi_{1x} \uplus \psi_2\}\!\}\right)$$

Consider one $t_{1x} = \frac{\beta_{1xj}^{j\in J_{1x}'}, g_{1x}, \psi_{1x}}{s_1 \xrightarrow{\alpha_{1x}} s_{1x}} \in \mathrm{OT}(s_1)$ to obtain a new open transition of A_{13} and compose it with t_3^2. We obtain

$$t_{13x}' = \frac{\beta_{1xj}^{j\in J_{1x}'\setminus\{k\}} \uplus (\beta_{3j}^2)^{j\in J_3^{2\prime}}, g_{1x} \wedge g_3^2 \wedge \alpha_3^2 = \beta_{1xk}, \psi_{1x} \uplus \psi_3^2}{(s_1, s_3) \xrightarrow{\alpha_{1x}} (s_{1x}, s_3)} \qquad \text{where } k \in J_{1x}'$$

This specifies the family $(t_{13x}')^{x\in X}$ that allows us to prove the initial goal, i.e. the relation is preserved along identical transitions. We derive from the definition of t_{23}

$$V_{13} \uplus V_{23} \uplus \biguplus_{x\in X} vars(t_{1x}) \uplus vars(t_{13}) \uplus vars(t_{23}) \vdash$$

$$R'(s_{13}, s_{23}) \wedge g_{13} \wedge g_{23} \wedge \alpha_{13} = \alpha_{23} \implies$$

$$R(s_1, s_2) \wedge g_1 \wedge g_2 \wedge g_3^2 \wedge \alpha_3^2 = \beta_{2k} \wedge \alpha_1 = \alpha_2 \wedge \bigwedge_{v_3\in V_3} v_3^1 = v_3^2 \wedge s_3^1 = s_3^2$$

Furthermore, from the property $(\star)$ and from the definition of t'_{13x}:

$$V_{13} \uplus V_{23} \uplus \biguplus_{x \in X} vars(t_{1x}) \uplus vars(t_{13}) \uplus vars(t_{23}) \vdash$$

$$R'(s_{13}, s_{23}) \wedge g_{13} \wedge g_{23} \wedge \alpha_{13} = \alpha_{23} \implies \bigwedge_{x \in X} \Big(\alpha_{13} = \alpha_1 = \alpha_{1x} \wedge J'_{1x} \uplus J'_3 = J'_2 \uplus J'_3 \wedge$$

$$\bigwedge_{j \in J'_2 \setminus \{k\}} \beta_{2j} = \beta_{1xj} \wedge \bigwedge_{j \in j \in J_3^{2\prime}} (\beta^1_{3j}) = (\beta^2_{3j}) \wedge g_{1x} \wedge R\big(s'_{1x}, s'_2\big) \{\!\{ \psi_{1x} \uplus \psi_2 \}\!\} \wedge \alpha^2_3 = \beta_{2k} = \beta_{1xk} \Big) \wedge g^2_3$$

$$\wedge \bigwedge_{v_3 \in V_3} v^1_3 = v^2_3 \wedge s^1_3 = s^2_3$$

Finally, because $R'(s'_{13x}, s'_{23}) \{\!\{ \psi_{1x} \uplus \psi_2 \}\!\} \implies R'(s'_{13x}, s'_{23}) \{\!\{ (\psi_{1x} \uplus \psi^2_3) \uplus (\psi_2 \uplus \psi^2_3) \}\!\}$

$$V_{13} \uplus V_{23} \uplus \biguplus_{x \in X} vars(t_{1x}) \uplus vars(t_{13}) \uplus vars(t_{23}) \vdash$$

$$R'(s_{13}, s_{23}) \wedge g_{13} \wedge g_{23} \wedge \alpha_{13} = \alpha_{23} \implies$$

$$\bigwedge_{x \in X} \left(\alpha_{13} = \alpha_{13x} \wedge J'_{13x} = J'_{23} \wedge \bigwedge_{j \in J'_{23}} \beta_{23j} = \beta_{13xj} \wedge g_{13x} \wedge R'\big(s'_{13x}, s'_{23}\big) \{\!\{ \psi_{13x} \uplus \psi_{23} \}\!\} \right)$$

This concludes for the current case.

Second case: Only A_2 is involved in transition t_{23}. t_{23} is obtained by the transition $t_2 = \dfrac{\beta_{2j}^{j \in J'_2}, g_2, \psi_2}{s_2 \xrightarrow{\alpha_2} s'_2}$ alone with the state s^2_3 unchanged, if $k \notin J'_2$

$$t_{23} = \frac{\beta_{2j}^{j \in J'_2}, g_2, \psi_2}{(s_2, s^2_3) \xrightarrow{\alpha_2} (s'_2, s^2_3)}$$

As in the previous case, two cases are possible for t_{13}: either the hole k of A_1 is involved in the transition or not.

- if the hole k of A_1 is involved in the transition: The transition t_{13} is obtained by the composition of transitions $t_1 = \dfrac{\beta_{1j}^{j \in J'_1}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s'_1} \in \mathrm{OT}(s_1)$ and

$$t^1_3 = \frac{(\beta^1_{3j})^{j \in J_3^{1\prime}}, g^1_3, \psi^1_3}{s_3 \xrightarrow{\alpha^1_3} s_3^{1\prime}} \qquad \text{when } k \in J'_1$$

The result is:

$$t_{13} = \frac{\beta_{1j}^{j \in J'_1 \setminus \{k\}} \uplus (\beta^1_{3j})^{j \in J_3^{1\prime}}, g_1 \wedge g^1_3 \wedge \alpha^1_3 = \beta_{1k}, \psi_1 \uplus \psi^1_3}{(s_1, s_3) \xrightarrow{\alpha_1} \left(s'_1, s_3^{1\prime}\right)} \qquad \text{where } k \in J'_1$$

- if the hole k of A_1 is not involved in the transition: $t_1 = \dfrac{\beta_{1j}^{j \in J'_1}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s'_1}$ when $k \notin J'_1$ and

$$t_{13} = \frac{\beta_{1j}^{j \in J'_1}, g_1, \psi_1}{(s_1, s_3) \xrightarrow{\alpha_1} (s'_1, s_3)}$$

In both cases note that $g_{13} \implies g_1$, with $(s_1, s_2) \in S_1 \times S_2$, Definition 5 applied to OTs t_1 and t_2 gives the following property that we denote $(\star)$:

$$\exists \left(t_{1x} = \frac{\beta_{1xj}^{j \in J'_{1x}}, g_{1x}, \psi_{1x}}{s_1 \xrightarrow{\alpha_{1x}} s'_{1x}} \in \mathrm{OT}(s_1) \right)^{x \in X},$$

$$V_1 \uplus V_2 \uplus \biguplus_{x \in X} vars(t_{1x}) \uplus vars(t_1) \uplus vars(t_2) \vdash$$

$$R(s_1, s_2) \wedge g_1 \wedge g_2 \wedge \alpha_1 = \alpha_2 \implies$$

$$\bigwedge_{x \in X} \left(\alpha_1 = \alpha_{1x} \wedge J'_{1x} = J'_2 \wedge \bigwedge_{j \in J'_2} \beta_{2j} = \beta_{1xj} \wedge g_{1x} \wedge R\big(s'_{1x}, s'_2\big) \{\!\{ \psi_{1x} \uplus \psi_2 \}\!\} \right)$$

Consider one $t_{1x} = \frac{\beta_{1xj}^{j \in J'_{1x}}, g_{1x}, \psi_{1x}}{s_1 \xrightarrow{\alpha_{1x}} s_{1x}} \in \mathrm{OT}(s_1)$ to obtain a new open transition of A_1 with the state s_3^2 unchanged, since $J'_{1x} = J'_2$ and $k \notin J'_2$.
We obtain

$$t_{13x} = \frac{\beta_{1xj}^{j \in J'_{1x}}, g_{1x}, \psi_{1x}}{(s_1, s_3^2) \xrightarrow{\alpha_{1x}} (s_{1x}, s_3^2)}$$

In the same way this specifies the family, $(t'_{13x})^{x \in X}$ that allows us to prove the initial goal, i.e. the relation is preserved along identical transitions. We derive from the definition of t_{23}

$$V_{13} \uplus V_{23} \uplus \biguplus_{x \in X} vars(t_{1x}) \uplus vars(t_{13}) \uplus vars(t_{23}) \vdash$$

$$R'(s_{13}, s_{23}) \wedge g_{13} \wedge g_{23} \wedge \alpha_{13} = \alpha_{23} \implies$$

$$R(s_1, s_2) \wedge g_1 \wedge g_2 \wedge \alpha_1 = \alpha_2 \wedge \bigwedge_{v_3 \in V_3} v_3^1 = v_3^2 \wedge s_3^1 = s_3^2$$

And from the property $(\star)$ and from the definition of t'_{13x}:

$$V_{13} \uplus V_{23} \uplus \biguplus_{x \in X} vars(t_{1x}) \uplus vars(t_{13}) \uplus vars(t_{23}) \vdash$$

$$R'(s_{13}, s_{23}) \wedge g_{13} \wedge g_{23} \wedge \alpha_{13} = \alpha_{23} \implies \bigwedge_{x \in X} \Big(\alpha_{13} = \alpha_1 = \alpha_{1x} \wedge J'_{1x} \uplus J'_3 = J'_2 \uplus J'_3 \wedge$$

$$\bigwedge_{j \in J'_2} \beta_{2j} = \beta_{1xj} \wedge \bigwedge_{j \in j \in J_3^{2\prime}} (\beta_{3j}^1) = (\beta_{3j}^2) \wedge g_{1x} \wedge R\big(s'_{1x}, s'_2\big) \{\!\{ \psi_{1x} \uplus \psi_2 \}\!\} \Big) \wedge \bigwedge_{v_3 \in V_3} v_3^1 = v_3^2 \wedge s_3^1 = s_3^2$$

Which gives directly $R'(s'_{13x}, s'_{23})$ as A_3 does not move (thus $\psi_{13x} = \psi_{1x}$ and $\psi_{23} = \psi_2$):

$$V_{13} \uplus V_{23} \uplus \biguplus_{x \in X} vars(t_{1x}) \uplus vars(t_{13}) \uplus vars(t_{23}) \vdash$$
$$R'(s_{13}, s_{23}) \wedge g_{13} \wedge g_{23} \wedge \alpha_{13} = \alpha_{23} \implies$$
$$\bigwedge_{x \in X} \left(\alpha_{13} = \alpha_{13x} \wedge J'_{13x} = J'_{23} \wedge \bigwedge_{j \in J'_{23}} \beta_{23j} = \beta_{13xj} \wedge g_{13x} \wedge R'\left(s'_{13x}, s'_{23}\right) \{\!\!\{ \psi_{13x} \uplus \psi_{23} \}\!\!\} \right)$$

This concludes for the second case and the proof of the theorem.