

Refinement for open automata

Quentin CORRADI

March 22, 2021

1 Introduction

Open automata are used to give an interpretation for open pNets.

2 Notations

Throughout this paper, tuples might be denoted differently depending on what they represent. Quantification will be used loosely, any previously defined object in a quantification means that the object is constrained to have that previous value, and any undefined object is defined by the quantifier. For instance $\forall(x, y) \in \mathbb{Z}^2, \exists(x', y) \in \mathbb{R}^2, P$ means $\forall(x, y), \exists(x', y'), y = y' \wedge P$.

Family of values, or equivalently maps will be noted $\{i \mapsto x_i \mid i \in I\}$, $\{i \leftarrow x_i \mid i \in I\}$ or $x_i^{i \in I}$. The latter will only be used when unambiguous; for instance $(ax)^{x \in \mathbb{R}}$ represents a scaling function, $c^{i \in I}$ is a constant function over I , but $\{\alpha \mapsto 1, \beta \mapsto 2, \gamma \mapsto 3\}$ must be represented with one of the two first notation. The disjoint union of two maps $\varphi \in X^I$ and $\psi \in Y^J$ with $I \cap J = \emptyset$ is $\varphi \uplus \psi \in (X \cup Y)^{I \uplus J}$.

A function with two parameters $R \in X \times Y \rightarrow Z$ might be noted using the relational notation $[x \ R \ y \mid z]$ when it is relevant for noting $R(x, y) = z$ or equivalently $(x, y, z) \in R$.

3 Open Automaton

To define an open automaton we need some preliminary definitions.

Definition 1 (Expression algebra). An expression algebra E is a disjoint union $E := \mathcal{T} \uplus \mathcal{F}$ where \mathcal{T} is a term algebra and \mathcal{F} the formulae over \mathcal{T} .

The terms algebra is arbitrary. The formulae form at least a propositional logic over terms in \mathcal{T} and some relations, or equivalently they are at least first order logic formulae without quantifiers but with free variables allowed. An example of term algebra can be Peano integers (zero, variable and successor function), the formulae associated can use the relations equality, $sum(a, b, c) := a = b + c$ and $prod(a, b, c) := a = b \times c$.

Definition 2 (Free variables, Closed terms/formulae). $vars(e)$ is the set of unbound variables in $e \in E$. The closed terms are $\mathcal{T}_0 ::= \{t \in \mathcal{T} \mid vars(t) = \emptyset\}$; $\mathcal{T}_0 \subset \mathcal{T}$. The closed formulae are $\mathcal{F}_0 ::= \{f \in \mathcal{F} \mid vars(f) = \emptyset\}$; $\mathcal{F}_0 \subset \mathcal{F}$.

More generally a closed expression is an expression without unbound variable. If the expression algebra has no binding construction then closed terms are terms without variables.

Definition 3 (Interpretation, Values, Satisfaction, (Parallel) substitution). We assume that the following functions are given:

- The interpretations of closed terms, $\llbracket _ \rrbracket \in \mathcal{T}_0 \rightarrow \mathcal{P}$, with \mathcal{P} a set of interpreted value of terms;
- The satisfaction of closed formulae, $\vdash \subseteq \mathcal{F}_0 \rightarrow \{0, 1, ?\}$;
- The substitution in $e \in E$ of $x \in \text{vars}(e)$ by $t \in \mathcal{T}$, $e[t/x]$;
- The parallel substitution in $e \in E$ of variables in V by $\psi \in \mathcal{T}^V$, $e\{\psi\}$.

The set V is not required to be a subset of $\text{vars}(e)$. In the case $U := V \setminus \text{vars}(e)$, $U \neq \emptyset$, the variables in U are not substituted. The substitutions might give a nonsensical expression; for instance terms can be integers and pairs with addition, $(a + b)\{a \mapsto 7, b \mapsto (4, 5)\}$ is not a valid term. This can be guarded with $e[t/x] \in E$ and $e\{\psi\} \in E$ when there is quantification on t and ψ ; in the other cases it is supposed to give a valid expression. The interpretation of values is supposed to be decidable. The satisfaction of formulae is not supposed to be decidable, however it is better if it is ;). We are not interested in how \mathcal{P} is defined. Whether it is predefined or characterised by $\mathcal{P} ::= \llbracket \mathcal{T}_0 \rrbracket$ is not the point of this definition. A value v may be used for keeping a variable state, and then injected in terms for substitution. While this is correct when $\mathcal{P} \subseteq \mathcal{T}_0$, in the other cases this will be used as a shorthand for substitution with any $t \in \llbracket v \rrbracket^{-1}$.

Definition 4. The following notations will be used for convenience:

- Let $f \in \mathcal{F}_0$, $\vdash f_0 ::= \vdash(f_0) = 1$ and $\not\vdash f_0 ::= \vdash(f_0) = 0$;
- Let $f \in \mathcal{F}$, $\sigma \in \mathcal{P}^V$,
 $\sigma \vdash f ::= \exists \varphi \in \mathcal{P}^{\text{vars}(f) \setminus V}, f\{\sigma \uplus \varphi\} \in \mathcal{F}_0 \wedge \vdash f\{\sigma \uplus \varphi\}$.

The first notation is quite classical when satisfaction is defined as a relation rather than a function. The second notation is the satisfiability of formulae under a (partial) valuation.

From there the expression algebra is fixed. With these common definitions and notations settled, the objects of interest can now be defined.

Definition 5 (Open automaton). An open automaton is a tuple $\langle S, i, V, \varphi, J, T \rangle$ with S the set of states, $i \in S$ the initial state, V the set of variable names unique to this automaton, $\varphi \in \mathcal{P}^V$ the initial valuation of variables, J the set of hole names and T a set of open-transitions.

S, V, J are arbitrary sets, **only J is required to be finite.**

Is S also required to be finite?

The variable names may clash when considering two automata, in that case we suppose that we can still distinguish the variables. All the possible open transitions, from which transitions in T are selected, are defined below.

Definition 6 (Open transition). An open transition is a tuple $\frac{(\beta_j(v_{\beta_j}))^{j \in J'}, g, \psi}{s \xrightarrow{\alpha(v_\alpha)} s'}$ with $s, s' \in S$ the source and target states, $\alpha \in A$ an action label, $J' \subseteq J$ the holes involved, $\beta_j \in A$ the action labels of the holes, $v_\alpha, v_{\beta_j}^{j \in J'}$ fresh distinct variables, $g \in \mathcal{F}$ the guard and $\psi \in \mathcal{T}^V$ the new value of the variables.

The set of action labels A is arbitrary and implicit. To simplify formulae, the notations $\alpha ::= \alpha(v_{\alpha})$ for the mono- and $\alpha = \beta ::= \alpha = \beta \wedge v_{\alpha} = v_{\beta}$ will be used. The guard and new values of the variables can use any defined variable: $\bigcup_{v \in V} \text{vars}(\psi(v)) \cup \text{vars}(g) \subseteq V \cup \{v_{\alpha}\} \cup \{v_{\beta_j} \mid j \in J'\}$.

Now we can define some utility functions:

Definition 7 (Guard, Variables, Out-transition, In-transition). $\text{OT}(s)$ are called the out-transitions of s and $\text{IT}(s)$ and its in-transitions.

$$\begin{aligned} \text{guard}\left(\frac{\beta_i^{j \in J'}, g, \psi}{s \xrightarrow{\alpha} s'}\right) &::= g & \text{vars}\left(\frac{(\beta_i(v_{\beta_j}))^{j \in J'}, g, \psi}{s \xrightarrow{\alpha(v_{\alpha})} s'}\right) &::= \{v_{\alpha}\} \cup \{v_{\beta_j} \mid j \in J'\} \\ \text{OT}(s) &::= \left\{ \frac{\beta_i^{j \in J'}, g, \psi}{s \xrightarrow{\alpha} s'} \in T \right\} & \text{IT}(s') &::= \left\{ \frac{\beta_i^{j \in J'}, g, \psi}{s \xrightarrow{\alpha} s'} \in T \right\} \end{aligned}$$

The intuition of an open automaton is a partially defined LTS with variables, guards on transitions and parametrised actions.

4 Semantic and composition of Open Automata

When the automaton is in a state $s \in S$ with a valuation of its variables $\sigma \in \mathcal{P}^V$, it cannot perform transitions $t := \frac{\beta_i^{j \in J'}, g, \psi}{s \xrightarrow{\alpha} s'} \in T$ such that $\not\models g\{\sigma\}$. After performing the transition and emitting action α , the automaton is in the state s' with valuation $\psi\{\sigma \uplus \varphi\}$ for some $\varphi \in \mathcal{P}^{\text{vars}(t)}$. The holes are synchronised using a handshake like mechanism when the transition happens.

Example 1.

An open automaton can be partially specified, the partial specification comes from the holes. A hole can be filled with an open automaton, this operation is called composition.

Definition 8 (Composition of open automata). The composition of $\text{OA}_c := \langle S_c, i_c, V_c, \varphi_c, J_c, T_c \rangle$ in the hole $k \in J_p$ of $\text{OA}_p := \langle S_p, i_p, V_p, \varphi_p, J_p, T_p \rangle$ is:

$$\begin{aligned} \text{OA}_p[\text{OA}_c/k] &::= \langle S_p \times S_c, (i_p, i_c), V_p \uplus V_c, \varphi_p \uplus \varphi_c, J_c \uplus J_p \setminus \{k\}, T \rangle \\ \text{With } T &:= \left\{ \frac{\beta_i^{j \in J'_c \uplus J'_p \setminus \{k\}}, g_p \wedge g_c \wedge \alpha_c = \beta_k, \psi_p \uplus \psi_c}{(s_p, s_c) \xrightarrow{\alpha_p} (s'_p, s'_c)} \mid \frac{\beta_i^{j \in J'_p}, g_p, \psi_p}{s_p \xrightarrow{\alpha_p} s'_p} \in T_p, \frac{\beta_i^{j \in J'_c}, g_c, \psi_c}{s_c \xrightarrow{\alpha_c} s'_c} \in T_c \right\} \\ &\cup \left\{ \frac{\beta_i^{j \in J'_p}, g_p, \psi_p}{(s_p, s_c) \xrightarrow{\alpha_p} (s'_p, s_c)} \mid \frac{\beta_i^{j \in J'_p}, g_p, \psi_p}{s_p \xrightarrow{\alpha_p} s'_p}, k \notin J', s_c \in S_c \right\} \end{aligned}$$

Similarly to expression substitution, the parallel composition is noted $\text{OA}\{\{\text{OA}_j^{j \in J}\}\}$.

The actions emitted when OA_c makes a transition is synchronised with the action of the hole k in transitions of OA_p which have it as a hole actions (first transition set, $\alpha_c = \beta_k$). No transition of OA_c is performed when a transition that do not refer to the hole k is performed in OA_p (second transition set, $k \notin J'$). The composition may look like a handshake, with both automata running in parallel (product of states and joint variables) but it is actually not symmetric.

Example 2.

5 What is a refinement for Open Automata

There are several properties that we may want from a refinement relation. Depending on these properties, several kind of refinement may be used. For example if we are interested in being able to produce the same sequence of action we may want to use trace set inclusion as a refinement.

The important properties we will consider here are:

Reflexivity: $\forall a, a \leq a$;

Transitivity: $\forall a b c, a \leq b \wedge b \leq c \implies a \leq c$;

Preorder: Reflexivity and transitivity;

Composition is a refinement: $\forall a b, a[b] \leq a$;

Composition context equivalence: $\forall a b c, a \leq b \wedge a[c] \leq a \implies a[c] \leq b[c]$;

Composition congruence: $\forall a b c, a \leq b \wedge c[a] \leq c \implies c[a] \leq c[b]$;

Composition compatibility: $\forall a b c d, a \leq b \wedge c \leq d \wedge c[a] \leq c \implies c[a] \leq d[b]$;

FH-bisimulation compatibility: $\forall a b c d, a = b \wedge c = d \wedge a \leq c \implies b \leq d$.

The interesting but not necessary properties are:

No deadlock introduction: If $a \leq b$ then a does not introduces new deadlocks.

“No deadlock introduction” conflicts with “composition is a refinement” by disallowing some unwanted cases where an automaton in a hole cannot produce any action that out-transitions expect. In particular filling a hole with the deadlock automaton (one state without out-transitions) is not considered a refinement for a no deadlock introductive refinement. This property has another impact which lead to write prerequisites like $a \leq b \wedge a[c] \leq a \implies a[c] \leq b[c]$ for context equivalence because $a[c] \leq a$ might not be true.

On the other hand the interesting but actually unwanted properties are:

Daisy equivalence: Filling a hole of a with a **daisy** gives b such that $a \leq b$ and $b \leq a$, i.e. it is not a strict refinement. How is it called in english?

While daisy equivalence might seem to be a natural property because the daisy being the meaning of a hole, there is actually a difference between the two. Daisy equivalence allows to refine by filling a daisy but also by going the other way, that is putting new action restrictions to any transition (and duplicating transitions). The issue especially arise when two independant holes are filled with a daisy, then the other way is simulated by putting back the actions but as a unique hole, effectively merging independant holes.

Example 3.

Definition 9 (Refinement simulation correction).

6 Refinement relation

The main objective of the refinement relation in this section is to be able to say that a composition is a refinement of the base automaton ($a[b] \leq a$). However composing an open automaton can give an automaton where not much can be said in terms of hole indicies. So looking at restricted cases where holes are related in a specific manner can help understanding the general case. For two open automata $OA_1 := \langle S_1, i_1, J_1, V_1, \varphi_1, T_1 \rangle$ and $OA_2 := \langle S_2, i_2, J_2, V_2, \varphi_2, T_2 \rangle$, the following refinement relations are defined.

This relation is the basis for all the other ones. It applies on automaton with same holes. The goal is to characterise open automata that have a more determined (more deterministic) behaviour without adding deadlocks.

Definition 10 (Hole-identical refinement). If $J_1 = J_2$ then “ OA_1 is a hole-identical refinement of OA_2 ” is defined, and its definition is as follows:

$$OA_1 \leq OA_2 ::= \exists R \in (S_1 \times S_2) \rightarrow \mathcal{F},$$

$$(\exists p \in \mathcal{F}, \quad [i_1 R i_2 | p] \quad \wedge \quad \vdash p\{\!\!\{\varphi_1 \uplus \varphi_2\}\!\!\})$$

$$\wedge R \text{ hole-identical simulation of } OA_1 \text{ in } OA_2$$

As in other simulation-like relations, R can be seen as a witness of $OA_1 \leq OA_2$. Hence when two open automata are in relation, such a simulation will be called a witness. To be a witness the standard requirement is that R relates initial states, R is closed under progress and related states are simultaneously final. R' progresses to R if R' relates target states of transitions with the same label and with source related states in R . Being closed under progress is the same as stating that for all related states, transitions can be matched and target states are also related. The latter will be used. For a bisimulation transitions are matched one-to-many from each automaton to the other, but for a simulation only one way is used.

In this model there is no final states so there is no need for that last part. The progress closure is stated after as R being a hole-identical refinement relation. The specificities of open automata comes in when relating the (initial) states. The relation $[i_1 R i_2 | p]$ means that i_1 is related to i_2 under the condition that the current value of variables satisfies p . The predicate p is used to take into account the fact that a state is also constituted of the value of the variables. This is already used in FH-bisimulation introduced in previous articles about open automata. So it is normal to find right after $\vdash p\{\!\!\{\varphi_1 \uplus \varphi_2\}\!\!\}$ which means that the initial valuation satisfies p .

Definition 11 (Hole-identical simulation). “ R is a hole-identical simulation of OA_1 in OA_2 ” is

defined as:

$$\begin{aligned} & \forall [s_1 \ R \ s_2 \mid p], \\ & \left(\begin{aligned} & \forall t_1 := \frac{\beta_{1j}^{j \in J'}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s'_1} \in T_1, \exists \left(\frac{\beta_{2xj}^{j \in J'}, g_{2x}, \psi_{2x}}{s_2 \xrightarrow{\alpha_{2x}} s'_{2x}} \in T_2, p'_x \in \mathcal{F}_1 + \mathcal{F}_2 \right)^{x \in X}, \\ & (\forall x \in X, [s'_1 \ R \ s'_{2x} \mid p'_x]) \wedge \forall \sigma \in \mathcal{P}^{V_1 \uplus \text{vars}(t_1) \uplus V_2}, \\ & \sigma \vdash \left(p \wedge g_1 \implies \bigvee_{x \in X} \left(\alpha_1 = \alpha_{2x} \wedge \bigwedge_{j \in J'} \beta_{1j} = \beta_{2xj} \wedge g_{2x} \wedge p'_x \llbracket \psi_1 \uplus \psi_{2x} \rrbracket \right) \right) \end{aligned} \right) \\ & \wedge \vdash \left(p \wedge \bigvee_{t_2 \in \text{OT}(s_2)} \text{guard}(t_2) \implies \bigvee_{t_1 \in \text{OT}(s_1)} \text{guard}(t_1) \right) \end{aligned}$$

The first two lines describe the fact that transitions can be matched and source and target states are related. The 3rd line is the condition to ensure that transitions are not matched to incompatible ones: For all variable assignment of the two automata and all hole actions ($\forall \sigma$ part), assuming that the states were related and the transition in OA_1 is possible ($p \wedge g_1$ part) then there is always a transition, not necessarily only one, which can be performed (g_{2x}), produce the same action ($\alpha_1 = \alpha_{2x}$), accept the same action from the holes and satisfy the predicate for relating the target states after variable update ($p'_x \llbracket \psi_1 \uplus \psi_{2x} \rrbracket$).

The notion of refinement here is stating that OA_1 is a refinement of OA_2 if OA_1 can be simulated in OA_2 . The hole-identical part is referring to the fact that holes indices are the same and identical holes indices have to perform the same actions. On top of that the last line ensures no deadlock introduction. It can be interpreted “assuming the predicate holds and there is any possible transition in OA_2 (= no deadlock), then there must be a possible transition in OA_1 (= no deadlock)”. It does not have to ensure that this transition isn’t garbage because the first part of the simulation already does it. Also if there was a deadlock then there is no transition because no transition can be simulated in a deadlock, that’s why it is more a deadlock equivalence than a deadlock reduction.

In order to prove that a relation R is a witness of $\text{OA}_1 \leq \text{OA}_2$ for some automata OA_1 and OA_2 , there are usually N steps:

1. Proving that $\exists p \in \mathcal{F}, [i_1 \ R \ i_2 \mid p] \wedge \vdash p \llbracket \varphi_1 \uplus \varphi_2 \rrbracket$ which is often skipped in concrete examples because it is obvious.
- 2.
- 3.
- 4.

Example 4.

Theorem 1. *The hole-identical refinement relation is a preorder.*

Proof. Reflexivity: Let OA be an open automaton with variables in V . The automaton variables on the right hand side of the relation will be noted v' for the equivalent variable $v \in V$ in the automaton on the left hand side. The simulation $R = \left\{ (s, s) \mapsto \bigwedge_{v \in V} v = v' \mid s \in S_1 \right\}$ is a

witness of $OA \leq OA$. Checking it is a simple exercise left to the reader in order to understand how the definition works.

Transitivity: Let OA_1, OA_2, OA_3 be open automata with respectively variables in V_1, V_2, V_3 . And let R_{12} be a witness of $OA_1 \leq OA_2$ and R_{23} be a witness of $OA_2 \leq OA_3$.

$$R_{13} := \left\{ \left(s_1, s_3, merge_{V_2}(p_{12}, p_{23}) \right) \mid [s_1 \ R_{12} \ s_2 \mid p_{12}] \wedge [s_2 \ R_{23} \ s_3 \mid p_{23}] \right\}$$

$$merge_{V_2}(p_{12}, p_{23}) := p_{12} \wedge p_{23}$$

Now let's prove that R_{13} is a witness of $OA_1 \leq OA_3$:

•

□

Theorem 2 (Hole-identical refinement correction).

Proof.

□

The goal of the following relation is to capture the case where holes are filled with automata that do not have any hole. It is supposed to be a relation for which filling holes with fully specified automata is a considered refinement.

Definition 12 (Hole-subset refinement). If $J_1 \subseteq J_2$ then “ OA_1 is a hole-subset refinement of OA_2 ” is defined, and its definition is as follows:

$$OA_1 \leq_{\subseteq} OA_2 ::= \exists R \in (S_1 \times S_2) \rightarrow \mathcal{F},$$

$$(\exists p \in \mathcal{F}, [i_1 \ R \ i_2 \mid p] \wedge \vdash p \llbracket \varphi_1 \uplus \varphi_2 \rrbracket)$$

$$\wedge R \text{ hole-subset simulation of } OA_1 \text{ in } OA_2$$

This definitions is essentially the same as the hole-identical one excepted the constraint on holes which has been softened.

Definition 13 (Hole-subset simulation). R is a hole-subset simulation of OA_1 in OA_2 is defined as:

$$\forall [s_1 \ R \ s_2 \mid p],$$

$$\left(\begin{array}{l} \forall \frac{\beta_{1j}^{j \in J'_1}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s'_1} \in T_1, \exists \left(\frac{\beta_{2xj}^{j \in J'_{2x}}, g_{2x}, \psi_{2x}}{s_2 \xrightarrow{\alpha_{2x}} s'_{2x}} \in T_2, p'_x \in \mathcal{F}_1 + \mathcal{F}_2 \right)^{x \in X}, \\ (\forall x \in X, [s'_1 \ R \ s'_{2x} \mid p'_x] \wedge J'_1 = J'_{2x} \cap J_1) \wedge \forall \sigma \in \mathcal{P}^{V_1 \uplus vars(t_1) \uplus V_2}, \\ \sigma \vdash \left(p \wedge g_1 \implies \bigvee_{x \in X} \left(\alpha_1 = \alpha_{2x} \wedge \bigwedge_{j \in J'_1} \beta_{1j} = \beta_{2xj} \wedge g_{2x} \wedge p'_x \llbracket \psi_1 \uplus \psi_{2x} \rrbracket \right) \right) \end{array} \right)$$

$$\wedge \vdash \left(p \wedge \bigvee_{t_2 \in OT(s_2)} guard(t_2) \implies \bigvee_{t_1 \in OT(s_1)} guard(t_1) \right)$$

The difference with the hole-identical simulation is that the holes involved in the matched transitions don't have to be exactly the same anymore. The new constraint is that the holes in common (that is J'_1) should be involved at the same time and their action have to match. What happens on the other holes is unspecified except for the fact that, by being free variables they still need to be valued such that the transition is possible.

An alternative version of this definition could enforce $\exists J'_2 \subseteq J_2, \forall x \in X, J'_{2x} = J'_2$. Let's call it the alternative hole-subset simulation. This version would mean that the holes involved in every matched transitions must be the same. While this may seem more natural this leads to the unwanted behaviour that an automaton FH-bisimilar to a refinement of some specification might not be a refinement of that specification:

Proposition 1 (Alternative hole-subset simulation is not compatible with FH-bisimulation). *TODO: collapse a transition that was differentiated by having one filled hole involved and it should be simple*

This is a sufficient reason to discard this version although it seems more natural.

Proposition 2 (Hole-subset refinement is compatible with FH-bisimulation).

Example 5.

Proposition 3 (Hole-subset refinement is an extension of hole-identical refinement). *Hole-subset refinement and hole-identical match when holes are identical.*

Proof. When $J_1 = J_2$, $J'_1 = J'_{2x} \cap J_1 \iff J'_1 = J'_{2x} \cap J_2 \iff J'_1 = J'_{2x}$, which is the implicit constraint on hole actions in the hole-identical simulation. By that rewriting their definition match. \square

Let's assume that this relation is also correct, the proof of correctness is given later for a more general refinement relation.

Theorem 3 (Composition is a refinement).

Proof. \square

Theorem 4 (Context refinement).

Proof. \square

Theorem 5 (Congruence with composition).

Proof. \square

The goal of the following relation is to capture the case where holes are filled with one hole automata. It is supposed to be a relation for which filling holes with one hole automata is a considered refinement.

Definition 14 (Hole-matching refinement). If $|J_1| = |J_2|$ then “ OA_1 is a hole-matching refinement of OA_2 ” is defined, and its definition is as follows:

$$\begin{aligned} \text{OA}_1 \leq_{\#} \text{OA}_2 ::= & \exists R \in (S_1 \times S_2) \rightarrow \mathcal{F}, \quad f \in J_1 \setminus J_2 \rightarrow J_2 \setminus J_1, \\ & (\exists p \in \mathcal{F}, [i_1 \ R \ i_2 \mid p] \wedge \vdash p \{\!\{ \varphi_1 \uplus \varphi_2 \}\!\}) \quad \wedge \quad f(J_1 \setminus J_2) = J_2 \setminus J_1 \\ & \wedge \ R \text{ hole-f-matching simulation of } \text{OA}_1 \text{ in } \text{OA}_2 \end{aligned}$$

This definitions is essentially the same as the hole-identical except that the constraint on holes has been softened and it is compensated with a invertible map between non-shared holes.

Definition 15 (Hole-f-matching simulation). R is a hole-f-matching simulation of OA_1 in OA_2 is defined as:

$$\begin{aligned} & \forall [s_1 R s_2 | p], \\ & \left(\begin{aligned} & \forall \frac{\beta_{1j}^{j \in J'_1}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s'_1} \in T_1, \exists \left(\frac{\beta_{2xj}^{j \in J'_{2x}}, g_{2x}, \psi_{2x}}{s_2 \xrightarrow{\alpha_{2x}} s'_{2x}} \in T_2, p'_x \in \mathcal{F} \right)^{x \in X}, \\ & (\forall x \in X, [s'_1 R s'_{2x} | p'_x] \wedge J'_1 \cap J_2 = J'_{2x} \cap J_1 \wedge f(J'_1 \setminus J_2) \subseteq J'_{2x}) \\ & \wedge (p \wedge g_1) \vdash \left(\alpha_1 = \alpha_{2x} \wedge \bigwedge_{j \in J'_1 \cap J_2} \beta_{1j} = \beta_{2xj} \wedge g_{2x} \wedge p'_x \llbracket \psi_1 \uplus \psi_{2x} \rrbracket \right)^{x \in X} \end{aligned} \right) \\ & \wedge \vdash \left(p \wedge \bigvee_{t_2 \in OT(s_2)} guard(t_2) \implies \bigvee_{t_1 \in OT(s_1)} guard(t_1) \right) \end{aligned}$$

Definition 16 (Open automata refinement).

$$\begin{aligned} OA_1 \leq OA_2 &::= \exists f \in J_2 \setminus J_1 \rightarrow J_1 \setminus J_2 \\ & \wedge \exists R \in (S_1 \times S_2) \rightarrow \mathcal{F}_1 + \mathcal{F}_2, \exists p, [i_1 R i_2 | p] \wedge (v_1 \wedge v_2 \Rightarrow p) \\ & \wedge R \text{ f-refinement relation} \end{aligned}$$

$$\begin{aligned} R \text{ f-refinement relation} &::= \forall [s_1 R s_2 | p], \\ & \left(\begin{aligned} & \forall \frac{(\beta_{2j}(x_{2j}))^{j \in J'_2}, g_2, \psi_2}{s_2 \xrightarrow{\alpha_2(x_2)} s'_2} \in T_2, \exists \left(\frac{(\beta_{1yj}(x_{1yj}))^{j \in J'_{1y}}, g_{1y}, \psi_{1y}}{s_1 \xrightarrow{\alpha_{1y}(x_{1y})} s'_{1y}} \in T_1 \right)^{y \in Y}, \\ & (\forall y \in Y, [s'_{1y} R s'_2 | p'_y]) \wedge (\forall y \in Y, J'_{1y} \cap J_2 = J_1 \cap J'_2) \wedge (\forall y, f(J'_2 \setminus J_1) \subseteq J'_{1y} \setminus J_2) \\ & \wedge \left(p \wedge g_2 \Rightarrow \bigvee_{y \in Y} \left(\begin{aligned} & \bigwedge_{j \in J'_1 \cap J'_2} \beta_{2j} = \beta_{1yj} \wedge x_{2j} = x_{1yj} \\ & \wedge \alpha_2 = \alpha_{1y} \wedge x_2 = x_{1y} \\ & \wedge g_{1y} \wedge p'_y \llbracket \psi_2 + \psi_{1y} \rrbracket \end{aligned} \right) \right) \end{aligned} \right) \\ & \wedge \left(p \wedge \bigvee_{t_1 \in OT(s_1)} guard(t_1) \Rightarrow \bigvee_{t_2 \in OT(s_2)} guard(t_2) \right) \end{aligned}$$

The objective of the first part of this definition is to be able to simulate OA_2 in OA_1 when holes of the same name receive the same closed automaton while other holes receive “compatible” closed automata. The objective of the other part of this definition is to prevent the appearance of new deadlocks, which should mean with the first property that the compared automata are deadlock equivalent.

7 New equivalence relation induced by pre-order

8 Weak refinement relation

$$\begin{aligned} \text{OA}_1 \leq_{fH} \text{OA}_2 &::= f \in J_2 \setminus J_1 \rightarrow J_1 \setminus J_2 \wedge H \subseteq A \\ &\quad \wedge \exists R \in (S_1 \times S_2) \rightarrow \mathcal{F}_1 + \mathcal{F}_2, [i_1 \ R \ i_2 \mid p] \wedge (v_1 \wedge v_2 \Rightarrow p) \\ &\quad \wedge R \text{ weak f-refinement relation} \end{aligned}$$

Often $H = \{\tau\}$.

$$\begin{aligned} R \text{ weak refinement relation} &::= \forall [s_1 \ R \ s_2 \mid p], \\ &\left(\begin{array}{l} \forall \frac{_, g_2, \psi_2}{s_2 \xrightarrow{\alpha_2} s'_2} \in T_2, \exists \frac{_, g_{1x}, \psi_{1x}}{s_1 \xrightarrow{\alpha_{1x}} s'_{1x}} \in T_1, (\forall x \in X, [s'_{1x} \ R \ s'_2 \mid p'_x]) \\ \wedge \left(p \wedge g_2 \Rightarrow \left(\bigwedge_{j \in J_1 \cap J_2} \beta_{2j} = \beta_{1jx} \right) \wedge \alpha_2 = \alpha_{1x} \wedge g_{1x} \wedge p'_x \llbracket \psi_2 + \psi_{1x} \rrbracket \right) \end{array} \right) \\ &\quad \wedge \left(p \wedge \bigvee_{t_1 \in \text{OT}(s_1)} \text{guard}(t_1) \Rightarrow \bigvee_{t_2 \in \text{OT}(s_2)} \text{guard}(t_2) \right) \end{aligned}$$

9 Hole equivalence wrt equivalence

10 Conclusion