Refinement for open automata

Quentin Corradi

March 18, 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 $\{i \mapsto x_i \,|\, i \in I\}$, $\{i \leftarrow x_i \,|\, i \in I\}$ 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 $R \in X \times Y \to Z$ might be noted using the relational notation $[x \; R \; y \,|\, z]$ 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 $E \coloneqq \mathcal{T} \uplus \mathcal{F}$ where $\mathcal{T}$ is a term algebra and $\mathcal{F}$ the formulas over $\mathcal{T}$.

We also define convenience notations $\mathcal{T}(E) ::= \mathcal{T}$ and $\mathcal{F}(E) ::= \mathcal{F}$.

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

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

More generally a closed expression is an expression without variable. In a standard context it would be without free variables but here the "no quantifier" constraint means that there is no such thing as "bound variables".

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

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

The set V is not required to be a subset of $\mathit{vars}(e)$. In the case $U := V \setminus \mathit{vars}(e), U \neq \emptyset$, 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 and satisfaction are supposed to be decidable. We are not interested in how $\mathcal{P}(E)$ is defined. Whether it is predefined or caracterised by $\mathcal{P}(E) ::= [\![\mathcal{T}_0(E)]\!]$ 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 $\mathcal{P}(E) \subseteq \mathcal{T}_0(E)$, in the other cases this will be used as a shorthand for substitution with any $t \in [\![v]\!]^{-1}$.

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

- Let $f \in \mathcal{F}_0(E), \vdash f_0 ::= \vdash(f_0) = 1$ and $\nvdash f_0 ::= \vdash(f_0) = 0$;
- Let $f \in \mathcal{F}(E), \sigma \in \mathcal{P}(E)^V$,
  $\sigma \vdash f ::= \exists \varphi \in \mathcal{P}(E)^{\mathit{vars}(f) \setminus V}, f\{\!\{\sigma \uplus \varphi\}\!\} \in \mathcal{F}_0(E) \wedge \vdash f\{\!\{\sigma \uplus \varphi\}\!\}$.

The first notation is quite classical when satisfaction is defined as a relation rather than a function. The second notation means that the disjunction of the right formulas is satisfiable, not necessarily with the same value for a variable name between formulas, for all valid valuation of the left formula.

Definition 5 (Formulas combination, Expression algebra combination)**.** The propositional logic built without existential quantifiers on top of two propositional logics $\mathcal{F}(E_1)$ and $\mathcal{F}(E_2)$ is $\mathcal{F}(E_1) + \mathcal{F}(E_2)$. By extension $E_1 + E_2 ::= (\mathcal{F}(E_1) + \mathcal{F}(E_2)) \uplus (\mathcal{T}(E_1) \cup \mathcal{T}(E_2))$.

With these common definitions and notations settled, the objects of interest can now be defined.

Definition 6 (Open automaton)**.** An open automaton is a tuple $\langle S, i, E, V, \varphi, J, T \rangle$ with S the set of states, $i \in S$ the initial state, E an expression algebra, V the set of variable names unique to this automaton, $\varphi \in \mathcal{P}(E)^V$ 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.

The variable names may clash when considering two automaton, 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 7 (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, $g \in \mathcal{F}(E)$ the guard and $\psi \in \mathcal{T}(E)^V$ the new value of the variables.

The set of action labels A is arbitrary and implicit. To simplify formulas, the notations $\alpha ::= \alpha(v_\alpha)$ 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: $\bigcup_{v\in V} vars(\psi(v)) \cup vars(g) \subseteq V \cup \{v_\alpha\} \cup \{v_{\beta j} \,|\, j \in J'\}$.

Now we can define some utilitary functions:

Definition 8 (Guard, Variables, Out-transition, In-transition)**.** $\mathrm{OT}(s)$ are called the out-transitions of s and $\mathrm{IT}(s)$ and its in-transitions.

$$guard\left(\frac{\beta_j^{j\in J'}, g, \psi}{s \xrightarrow{\alpha} s'}\right) ::= g \qquad vars\left(\frac{(\beta_j(v_{\beta j}))^{j\in J'}, g, \psi}{s \xrightarrow{\alpha(v_\alpha)} s'}\right) ::= \{v_\alpha\} \cup \{v_{\beta j} \,|\, j \in J'\}$$

$$\mathrm{OT}(s) ::= \left\{\frac{\beta_j^{j\in J'}, g, \psi}{s \xrightarrow{\alpha} s'} \in T\right\} \qquad \mathrm{IT}(s') ::= \left\{\frac{\beta_j^{j\in J'}, g, \psi}{s \xrightarrow{\alpha} s'} \in T\right\}$$

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 auomaton is in a state $s \in S$ with a valuation of its variables $\sigma \in \mathcal{P}(E)^V$, it cannot perform transitions $t := \frac{\beta_j^{j\in J'}, g, \psi}{s \xrightarrow{\alpha} s'} \in T$ such that $\nvdash g\{\!\{\sigma\}\!\}$. After performing the transition and emitting action α, the automaton is in the state s' with valuation $\psi\{\!\{\sigma \uplus \varphi\}\!\}$ for some $\varphi \in \mathcal{P}(E)^{vars(t)}$. The holes are synchronised using a handshake like mecanism 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 9 (Composition of open automata)**.** The composition of $\mathrm{OA}_c := \langle S_c, i_c, E_c, V_c, \varphi_c, J_c, T_c\rangle$

in the hole $k \in J_p$ of $\mathrm{OA}_p \coloneqq \langle S_p, i_p, E_p, V_p, \varphi_p, J_p, T_p \rangle$ is:

$$\mathrm{OA}_p[\mathrm{OA}_c/k] ::= \langle S_p \times S_c, (i_p, i_c), E_p + E_c, V_p \uplus V_c, \varphi_p \uplus \varphi_c, J_c \uplus J_p \setminus \{k\}, T \rangle$$

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

$$\cup \left\{ \frac{\beta_j^{j \in J'_p}, g_p, \psi_p}{(s_p, s_c) \xrightarrow{\alpha_p} (s'_p, s_c)} \;\middle|\; \frac{\beta_j^{j \in J'_p}, g_p, \psi_p}{s_p \xrightarrow{\alpha_p} s'_p}, k \notin J', s_c \in S_c \right\}$$

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

The actions emitted when OA_c makes a transition is sychronised 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:

Deadlock preservation: If $a \leq b$ then a does not introduces new deadlocks.

Deadlock preservation conflicts with "composition as 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 deadlock preserving 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 $a \leq b \wedge b \leq a$, i.e. it is not a strict refinement.

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.

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 $\mathrm{OA}_1 := \langle S_1, i_1, E_1, J_1, V_1, \varphi_1, T_1 \rangle$ and $\mathrm{OA}_2 := \langle S_2, i_2, E_2, J_2, V_2, \varphi_2, T_2 \rangle$, the following refinement relations are defined.

This relation is the basis for all the following ones. It applies on automaton with same holes. The goal is to caracterise automata that have a more determined behaviour (more deterministic) 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:

$$\begin{aligned}\mathrm{OA}_1 \leq_{\equiv} \mathrm{OA}_2 ::= \exists R \in (S_1 \times S_2) \to \mathcal{F}(E)_1 + \mathcal{F}(E)_2,& \\ \big(\exists p \in \mathcal{F}(E)_1 + \mathcal{F}(E)_2, \quad [i_1\ R\ i_2 \,|\, p] \quad \wedge \quad \vdash p\{\!\{\varphi_1 \uplus \varphi_2\}\!\}\big)& \\ \wedge\ R \text{ hole-identical simulation of } \mathrm{OA}_1 \text{ in } \mathrm{OA}_2&\end{aligned}$$

As in other simulation-like relations, R can be seen as a witness of $\mathrm{OA}_1 \leq_{\equiv} \mathrm{OA}_2$. 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.

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 curent 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:

$$\forall [s_1 \ R \ s_2 \,|\, p],$$
$$\left(\begin{array}{l} \forall t_1 := \frac{\beta_{1j}^{j\in J'}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s_1'} \in T_1, \exists \left(\frac{\beta_{2xj}^{j\in J'}, g_{2x}, \psi_{2x}}{s_2 \xrightarrow{\alpha_{2x}} s_{2x}'} \in T_2, p_x' \in \mathcal{F}(E)_1 + \mathcal{F}(E)_2 \right)^{x\in X}, \\ (\forall x \in X, [s_1' \ R \ s_{2x}' \,|\, p_x']) \wedge \forall \sigma \in \mathcal{P}(E_1)^{V_1 \uplus vars(t_1)} \times \mathcal{P}(E_2)^{V_2}, \\ \quad \sigma \vdash \left(p \wedge g_1 \implies \bigvee_{x\in X} \left(\alpha_1 = \alpha_{2x} \wedge \bigwedge_{j\in J'} \beta_{1j} = \beta_{2xj} \wedge g_{2x} \wedge p_x' \{\!\{ \psi_1 \uplus \psi_{2x} \}\!\} \right) \right) \end{array} \right)$$
$$\wedge \vdash \left(p \wedge \bigvee_{t_2 \in \mathrm{OT}(s_2)} guard(t_2) \implies \bigvee_{t_1 \in \mathrm{OT}(s_1)} guard(t_1) \right)$$

Being stable wrt. progress is the same as stating that for all related states, transitions can be matched and target states are also related (two first lines). For a bisimulation transitions are matched one-to-many from each automaton to the other, but for a simulation only one way is used. There are some conditions to ensure that transitions are not matched to incompatible ones (3^{rd} line): For all variable assignement 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 ($p_x' \{\!\{ \psi_1 \uplus \psi_{2x} \}\!\}$).

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 indicies are the same and identical holes indicies have to perform the same actions. On top of that the last line ensures deadlock preservation. 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 trasition can be simulated in a deadlock, hence deadlock "preservation" and not "reduction".

Example 3.

Theorem 1 (Hole-identical refinement correction)**.**

Proof. □

Definition 12 (Second order formulas)**.** Let E be an expression algebra, $SO(\mathcal{F}(E))$ is an extension with pairs of terms, fixed sets and the set membership relation. By extension $SO(E) ::= \mathcal{T}(E) \uplus SO(\mathcal{F}(E))$.

Theorem 2. *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 $R = \left\{ (s,s) \mapsto \bigwedge_{v \in V} v = v' \,\middle|\, s \in S_1 \right\}$ is a witness of $\mathrm{OA} \leq \mathrm{OA}$. Checking it is a simple exercise left to the reader in order to understand how the definition works.

Transitivity: Let $\mathrm{OA}_1, \mathrm{OA}_2, \mathrm{OA}_3$ be open automata with respectively variables in V_1, V_2, V_3. And let R_{12} be a witness of $\mathrm{OA}_1 \leq_{\equiv} \mathrm{OA}_2$ and R_{23} be a witness of $\mathrm{OA}_2 \leq_{\equiv} \mathrm{OA}_3$.

$$R_{13} \coloneqq \left\{ \left(s_1, s_3, merge_{V_1,V_3}(p_{12}, p_{23}) \right) \,\middle|\, [s_1 \; R_{12} \; s_2 \,|\, p_{12}] \wedge [s_2 \; R_{23} \; s_3 \,|\, p_{23}] \right\}$$

$$merge_{V_1,V_3}(p_{12}, p_{23}) \coloneqq p_{12} \wedge p_{23}$$

□

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 13 (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:

$$\begin{aligned} \mathrm{OA}_1 \leq_{\subseteq} \mathrm{OA}_2 ::= \exists R \in (S_1 \times S_2) \to \mathcal{F}(E)_1 + \mathcal{F}(E)_2, \\ \left(\exists p \in \mathcal{F}(E)_1 + \mathcal{F}(E)_2, [i_1 \; R \; i_2 \,|\, p] \wedge \vdash p \{\!\{ \varphi_1 \uplus \varphi_2 \}\!\} \right) \\ \wedge R \text{ hole-subset simulation of } \mathrm{OA}_1 \text{ in } \mathrm{OA}_2 \end{aligned}$$

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

Definition 14 (Hole-subset simulation)**.** R is a hole-subset simulation of OA_1 in OA_2 is defined as:

$$\begin{aligned} &\forall [s_1 \; R \; s_2 \,|\, p], \\ &\left(\begin{array}{l} \forall \frac{\beta_{1j}^{j \in J_1'}, g_1, \psi_1}{s_1 \xrightarrow{\alpha_1} s_1'} \in T_1, \exists \left(\frac{\beta_{2xj}^{j \in J_{2x}'}, g_{2x}, \psi_{2x}}{s_2 \xrightarrow{\alpha_{2x}} s_{2x}'} \in T_2, p_x' \in \mathcal{F}(E)_1 + \mathcal{F}(E)_2 \right)^{x \in X}, \\ (\forall x \in X, [s_1' \; R \; s_{2x}' \,|\, p_x'] \quad \wedge \quad J_1' = J_{2x}' \cap J_1) \\ \quad \wedge (p \wedge g_1) \vdash \left(\alpha_1 = \alpha_{2x} \wedge \bigwedge_{j \in J_1'} \beta_{1j} = \beta_{2xj} \wedge g_{2x} \wedge p_x' \{\!\{ \psi_1 \uplus \psi_{2x} \}\!\} \right)^{x \in X} \end{array} \right) \\ &\qquad \wedge \vdash \left(p \wedge \bigvee_{t_2 \in \mathrm{OT}(s_2)} guard(t_2) \implies \bigvee_{t_1 \in \mathrm{OT}(s_1)} guard(t_2) \right) \end{aligned}$$

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 differenciated 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 4.

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

Proof. □

Proposition 4 (Hole-subset refinement is a pre-order)**.**

Proof. □

Theorem 3 (Composition is a refinement)**.**

Proof. □

Theorem 4 (Context refinement)**.**

Proof. □

Theorem 5 (Congruence with composition)**.**

Proof. □

A relation for which filling holes with one hole open automata is a refinement.

Definition 15 (Hole-matching refinement)**.** If $|J_2| = |J_1|$ then "OA_2 is a hole-matching refinement of OA_1" is defined, and its definition is as follows:

$$\begin{aligned}\mathrm{OA}_2 \leq_{\#} \mathrm{OA}_1 ::= \exists R \in (S_1 \times S_2) \to \mathcal{F}(E)_1 + \mathcal{F}(E)_2, \quad f \in J_2 \setminus J_1 \to J_1 \setminus J_2,\\ \big(\exists p \in \mathcal{F}(E)_1 + \mathcal{F}(E)_2, [i_1 \; R \; i_2 \,|\, p] \wedge \vdash (\llbracket \varphi_1 \rrbracket \wedge \llbracket \varphi_2 \rrbracket \implies p)\big) \quad \wedge \quad f(J_2 \setminus J_1) = J_1 \setminus J_2\\ \wedge R \text{ hole-f-matching simulation between } \mathrm{OA}_1 \text{ and } \mathrm{OA}_2\end{aligned}$$

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 16 (Hole-f-matching simulation)**.** R is a hole-f-matching simulation between OA_1 and OA_2 is defined as:

$$\forall [s_1 \ R \ s_2 \,|\, p],$$
$$\left(\begin{array}{l} \forall \frac{\beta_{2j}^{j\in J_2'}, g_2, \psi_2}{s_2 \xrightarrow{\alpha_2} s_2'} \in T_2, \exists \left(\frac{\beta_{1xj}^{j\in J_{1x}'}, g_{1x}, \psi_{1x}}{s_1 \xrightarrow{\alpha_{1x}} s_{1x}'} \in T_1, p_x' \in \mathcal{F}(E)_1 + \mathcal{F}(E)_2 \right)^{x\in X}, \\ (\forall x \in X, [s_{1x}' \ R \ s_2' \,|\, p_x'] \quad \wedge \quad J_2' \cap J_1 = J_{1x}' \cap J_2 \quad \wedge \quad f(J_2' \setminus J_1) \subseteq J_{1x}') \\ \qquad \wedge (p \wedge g_2) \vdash \left(\alpha_2 = \alpha_{1x} \wedge \bigwedge_{j \in J_2' \cap J_1} \beta_{2j} = \beta_{1xj} \wedge g_{1x} \wedge p_x' \{\!\{ \psi_2 \uplus \psi_{1x} \}\!\} \right)^{x\in X} \end{array} \right)$$
$$\wedge \vdash \left(p \wedge \bigvee_{t_1 \in \mathrm{OT}(s_1)} \mathit{guard}(t_1) \implies \bigvee_{t_2 \in \mathrm{OT}(s_2)} \mathit{guard}(t_2) \right)$$

$$\mathrm{OA}_1 \leq \mathrm{OA}_2 ::= \exists f \in J_2 \setminus J_1 \to J_1 \setminus J_2$$
$$\wedge \exists R \in (S_1 \times S_2) \to \mathcal{F}(E)_1 + \mathcal{F}(E)_2, \exists p, [i_1 \ R \ i_2 \,|\, p] \wedge (v_1 \wedge v_2 \Rightarrow p)$$
$$\wedge \ R \text{ f-refinement relation}$$

$$R \text{ f-refinement relation} ::= \forall [s_1 \ R \ s_2 \,|\, p],$$
$$\left(\begin{array}{l} \forall \frac{(\beta_{2j}(x_{2j}))^{j\in J_2'}, g_2, \psi_2}{s_2 \xrightarrow{\alpha_2(x_2)} s_2'} \in T_2, \exists \left(\frac{(\beta_{1yj}(x_{1yj}))^{j\in J_{1y}'}, g_{1y}, \psi_{1y}}{s_1 \xrightarrow{\alpha_{1y}(x_{1y})} s_{1y}'} \in T_1 \right)^{y\in Y}, \\ \big(\forall y \in Y, \big[s_{1y}' \ R \ s_2' \,\big|\, p_y'\big]\big) \wedge \big(\forall y \in Y, J_{1y}' \cap J_2 = J_1 \cap J_2'\big) \wedge \big(\forall y, f(J_2' \setminus J_1) \subseteq J_{1y}' \setminus J_2\big) \\ \quad \wedge \left(p \wedge g_2 \Rightarrow \bigvee_{y\in Y} \left(\begin{array}{l} \left(\bigwedge_{j \in J_1' \cap J_2'} \beta_{2j} = \beta_{1yj} \wedge x_{2j} = x_{1yj} \right) \\ \wedge \alpha_2 = \alpha_{1y} \wedge x_2 = x_{1y} \\ \wedge g_{1y} \wedge p_y' \{\!\{ \psi_2 + \psi_{1y} \}\!\} \end{array} \right) \right) \end{array} \right)$$
$$\wedge \left(p \wedge \bigvee_{t_1 \in \mathrm{OT}(s_1)} \mathit{guard}(t_1) \Rightarrow \bigvee_{t_2 \in \mathrm{OT}(s_2)} \mathit{guard}(t_2) \right)$$

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 automaton are deadlock equivalent.

7 New equivalence relation induced by pre-order

8 Weak refinement relation

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

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

$$\begin{aligned}R \text{ weak refinement relation} ::= {} & \forall [s_1 \; R \; s_2 \,|\, p], \\ & \left(\begin{array}{l} \forall \frac{-, g_2, \psi_2}{s_2 \xrightarrow{\alpha_2} s_2'} \in T_2, \exists \frac{-, g_{1x}, \psi_{1x}}{s_1 \xrightarrow{\alpha_{1x}} s_{1x}'} \in T_1, (\forall x \in X, [s_{1x}' \; R \; s_2' \,|\, p_x']) \\ \wedge \Big(p \wedge g_2 \Rightarrow \Big(\bigwedge_{j \in J_1 \cap J_2} \beta_{2j} = \beta_{1jx} \Big) \wedge \alpha_2 = \alpha_{1x} \wedge g_{1x} \wedge p_x' \{\!\{ \psi_2 + \psi_{1x} \}\!\} \Big) \end{array} \right) \\ & \wedge \left(p \wedge \bigvee_{t_1 \in \mathrm{OT}(s_1)} guard(t_1) \Rightarrow \bigvee_{t_2 \in \mathrm{OT}(s_2)} guard(t_2) \right)\end{aligned}$$

9 Hole equivalence wrt equivalence

10 Conclusion