UNIVERSITY
OF TRENTO

DIPARTIMENTO DI INGEGNERIA E SCIENZA DELL'INFORMAZIONE

38123 Povo – Trento (Italy), Via Sommarive 5
http://www.disi.unitn.it

An Operational Contract Framework for Heterogeneous Systems

Thi Thieu Hoa Le, Uli Fahrenberg, Axel Legay and Roberto Passerone
August 2012

Technical Report Number: DISI-12-029

An Operational Contract Framework for Heterogeneous Systems

Thi Thieu Hoa Le[1], Uli Fahrenberg[2], Axel Legay[2], and Roberto Passerone[1]

[1] DISI, University of Trento, Italy
[2] INRIA/IRISA, Rennes Cedex, France

Abstract. Modern computing systems are increasingly being built by assembling components that are pre-designed or developed concurrently in a distributed manner. In this context, contracts play a vital role for ensuring interoperability of components and adherence to specifications. For the design of e.g. embedded systems, additional complexity is found in heterogeneity of components: such systems are composed of components of very different nature, e.g. mechanical or electronic.
Heterogeneity adds extra complexity to systems design, as composition of heterogeneous components is generally not well-defined, hence making design and verification difficult. So far, few approaches have attempted to address heterogeneity for embedded systems, and yet none of them has demonstrated to be really effective. Meanwhile, denotational mathematical frameworks for reasoning effectively on heterogeneous composition have recently been made available in the literature, but their operational application to a contract-based design flow is still missing.
In this work, we propose a heterogeneous contract theory for embedded systems built on the tag machine formalism. We introduce heterogeneous composition, refinement, dominance, and compatibility of contracts, altogether enabling a formalized and rigorous design process for heterogeneous embedded systems.

Keywords: contract theory, heterogeneity, tag machine

1 Introduction

Modern computing systems are increasingly being built by composing components which are developed concurrently and independently by different design teams. In such a paradigm, the distinction between what is constrained on environments, and what must be guaranteed by a system given the constraint satisfaction, reflects the different roles and responsibilities in the system design procedure. Following [7], a *contract* is a component model that can truly capture such distinction. Formally, a contract is a pair of *assumptions* and *guarantees* which are properties to be satisfied by all inputs and all outputs of a design. This separation between assumptions and guarantees in contracts supports the distributed, concurrent development of complex systems, as different teams can develop different parts of the overall system and synchronize by relying on the notions of component models and their associated contracts.

In the particular context of embedded systems, *heterogeneity* is a typical characteristic. This stems from the fact that these systems usually are composed of various components of different natures (e.g. mechanical, electronic) whose interactions would not be allowed without a heterogeneous composition mechanism. Such heterogeneity usually appears across different layers of abstraction in the design flow, making the evaluation of whether certain properties passed from the higher level of abstraction are maintained at the lower level become extremely difficult. To deal with such heterogeneity, Benveniste et.al. [2] have proposed a generic behavioural framework based on *tagged events*. The framework considers a system as a set of behaviours, each of which are finite sets of tagged events. Different notions of time, including physical time, logical time, etc., can be captured by the proposed tagging mechanism. Given that each component can be modelled as a tag system over some tag structure, by mapping different tag structures to one same tag structure, heterogeneous systems can then be constructed formally.

Due to the significant inherent complexity of heterogeneity, there have been few attempts at addressing heterogeneity in contract-based models. One such attempt was made by researchers from the SPEEDS project[3] to deal with different viewpoints (such as functional, time, safety, etc.) of a single component [5]. However, the notion of heterogeneity in general is much broader than that between multiple viewpoints. On the other hand, [2] proposes a foundational framework for handling heterogeneity behaviourally, but this has not been related to contract-based design flows. This has motivated us to study a methodology which allows contract-based systems to be heterogeneous, or alternatively, which enables heterogeneous systems to be interconnected in a contract-based fashion.

Being inspired by Benveniste et.al.'s behavioural theory, our methodology also addresses heterogeneity by analysing the behaviours of heterogeneous systems which are modelled as tag systems. To build an operational contract framework on top of our methodology, we advocate using tag machines [3] to represent tag systems. Tag machines were first introduced as finitary generators of homogeneous traces. Since our methodology is to deal with heterogeneity, we have also made a heterogeneous extension to tag machines and developed a specification theory with basic operators (e.g. composition, refinement, quotient, conjunction) for them. We then propose a tag contract framework for combining heterogeneous systems based on tag machines. Our framework enables automatic checks on various contract operations such as satisfaction, refinement, dominance and compatibility.

The rest of the paper is organized as follows. Section 2 describes how tag machines are extended to represent heterogeneous systems. Section 3 presents a specification theory for tag machines and Section 4 details our tag machine based contract framework for heterogeneous systems. Finally the paper concludes in Section 5.

Related Work. The notion of contract was first introduced by Bertrand Meyer in his design-by-contract method [17], based on ideas by Dijkstra [10], Lam-

[3] www.speeds.eu.com

port [14], Jones [13] and others, where systems are viewed as abstract boxes achieving their common goal by verifying specified contracts. Such technique guarantees that methods of a class provide some post-conditions at their termination, as long as the pre-conditions under which they operate are satisfied. De Alfaro and Henzinger subsequently introduced interface automata [8] for documenting components, thereby enabling them to be reused effectively. This formalism establishes a more general notion of contract, as pre-conditions and post-conditions, which originally appeared in the form of predicates, are generalized to behavioural interfaces. The central issues when introducing the formalism of interface automata are compatibility, composition and refinement. While the differentiation between assumptions and guarantees is somewhat implicit in [8], it has become explicit in [4,5,6], where component behaviours are considered as sets of traces (or runs). The relationship between specifications of component behaviours and contracts is further studied in [1]. Here the authors show that a contract framework can be built on top of any *specification theory* equipped with a composition operator and a refinement relation which satisfy certain properties. Traced-based contract theories such as [4,5,6] are also demonstrated to be instances of such framework.

The heterogeneous theory has been evolving in parallel with the contract theory, to assist embedded systems designers in dealing with heterogeneous composition of components with various Models of Computation and Communication (MoCC). Handling heterogeneous MoCCs can be done strictly hierarchically in the pioneering framework of Ptolemy II [16], meaning that each level of the hierarchy is homogeneous while different interaction mechanisms are allowed to be specified at different levels in the hierarchy. While such framework advocates the heterogeneous semantics geared towards the representation and simulation of heterogeneous systems, another framework based on tags [2] is instead oriented towards the formal verification and analysis of those systems. The latter was inspired by the Lee and Sangiovanni-Vincentelli framework of tag signal models [15] which has been long advocated as a unified modelling framework capable of capturing heterogeneous MoCCs. In both models, tags play an important role in capturing various notions of time, and each tag system has its own tag structure that can be used to express a MoCC. By applying mappings between different tag structures, the authors define how to compose heterogeneous systems. Tag machines [3] are subsequently introduced as finite representations of tag systems, yet only the homogeneous composition has been defined. Tag machines have been applied to model a job-shop specification [9] where any trace of the composite tag machine from the start to the final state results in a valid job-shop schedule.

2 The Tag Machine Formalism

A component, a unit of design, can be modelled as a set of behaviours called *tag system* [2]. Each behaviour is a set of events which are characterised by pairs of a data value and a tag. Different notions of time, including logical time, physical

time, causal order, precedence relation, etc., can be represented by means of tags. The ordering among tags makes up a tag structure and is used to resolve the ordering among events at the system interface. Formally, a *tag structure* is a tuple $\langle \mathcal{T}, \leqslant \rangle$ where $\mathcal{T}$ is a set of tags and $\leqslant$ is a partial order which relates tags seen as time stamps. To distinguish the time-stamp order of tag structure $\mathcal{T}$ from that of others, we refer to it as $\underset{\mathcal{T}}{\leqslant}$ if necessary.

Example 1. $\langle \mathbb{N}, \leqslant \rangle$ and $\langle \mathbb{R}+, \leqslant \rangle$ are two tag structures whose time-stamp orders are total and suitable for capturing discrete and dense time behaviours, respectively.

Let V be an underlying set of variables with domain D. A V-behaviour σ is an element of the map $\sigma \in V \mapsto (\mathbb{N} \mapsto (\mathcal{T} \times D))$, meaning that for each variable $v \in V$, the n-th occurrence of v in behaviour σ has tag $\tau \in \mathcal{T}$ and value $d \in D$. An event of σ is a tuple $\langle v, n, \tau, d \rangle \in V \times \mathbb{N} \times \mathcal{T} \times D$ such that $\sigma(v, n) = (\tau, d)$. A *tag system* (component) is a triple $P = \langle V, \mathcal{T}, \Sigma \rangle$ where V is a finite set of variables, $\mathcal{T}$ is a tag structure and Σ is a set of V-behaviours. For tag systems having identical tag structures, their composition is defined by intersection.

Definition 1 (Homogeneous Composition [2]). *Consider two tag systems $P_1 = \langle V_1, \mathcal{T}, \Sigma_1 \rangle$ and $P_2 = \langle V_2, \mathcal{T}, \Sigma_2 \rangle$ with identical tag structures. For $\sigma_i \in \Sigma_i (1 \leq i \leq 2)$, define:*

$$\sigma_1 \underset{\mathcal{T}}{\bowtie} \sigma_2 \Leftrightarrow \sigma_{1|V_1 \cap V_2} = \sigma_{2|V_1 \cap V_2}$$
$$\sigma_1 \underset{\mathcal{T}}{\sqcup} \sigma_2 \stackrel{\text{def}}{=} \sigma_{1|V_1 \cap V_2} \cup \sigma_{1|V_1 \setminus V_2} \cup \sigma_{2|V_2 \setminus V_1}$$
$$\Sigma_1 \wedge \Sigma_2 \stackrel{\text{def}}{=} \{\sigma_1 \underset{\mathcal{T}}{\sqcup} \sigma_2 | \sigma_1 \in \Sigma_1, \sigma_2 \in \Sigma_2 \text{ and } \sigma_1 \underset{\mathcal{T}}{\bowtie} \sigma_2\}$$

where $\sigma_{|W}$ denotes the restriction of behaviour σ to the variables in W. Then the homogeneous composition of P_1 and P_2 is: $P_1 \parallel P_2 \stackrel{\text{def}}{=} \langle V_1 \cup V_2, \mathcal{T}, \Sigma_1 \wedge \Sigma_2 \rangle$.

For tag systems having different tag structures, their heterogeneous composition is defined by intersection w.r.t. a pair of tag morphisms.

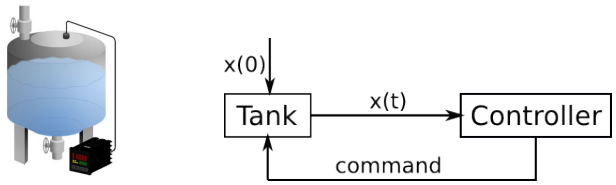
Definition 2 (Tag Morphisms [2]). *For $\mathcal{T}$ and $\mathcal{T}'$ two tag structures, a tag morphism is a total map $\rho : \mathcal{T} \mapsto \mathcal{T}'$ which is increasing w.r.t. the time-stamp orders $\underset{\mathcal{T}}{\leqslant}$ and $\underset{\mathcal{T}'}{\leqslant}$, i.e. $\forall \tau_1, \tau_2 \in \mathcal{T} : \tau_1 \underset{\mathcal{T}}{\leqslant} \tau_2 \Rightarrow \rho(\tau_1) \underset{\mathcal{T}'}{\leqslant} \rho(\tau_2)$.*

For a behaviour $\sigma \in V \mapsto (\mathbb{N} \mapsto (\mathcal{T} \times D))$, replacing $\tau \in \mathcal{T}$ by $\rho(\tau)$ in σ defines a new behaviour denoted by σ_ρ or $\sigma \circ \rho$ having $\mathcal{T}'$ as its tag structure.

Definition 3 (Heterogeneous Composition [2]). *Consider two tag systems $P_1 = \langle V_1, \mathcal{T}_1, \Sigma_1 \rangle$ and $P_2 = \langle V_2, \mathcal{T}_2, \Sigma_2 \rangle$ and two morphisms $\rho_1 : \mathcal{T}_1 \mapsto \mathcal{T}$ and $\rho_2 : \mathcal{T}_2 \mapsto \mathcal{T}$. Let $\mathcal{T}_1 {}_{\rho_1}\!\times_{\rho_2} \mathcal{T}_2 \stackrel{\text{def}}{=} \{\langle \tau_1, \tau_2 \rangle \in \mathcal{T}_1 \times \mathcal{T}_2 \,|\, \rho_1(\tau_1) = \rho_2(\tau_2)\}$ (shortened to $\mathcal{T}_\times$), called the fibered product of $\mathcal{T}_1$ and $\mathcal{T}_2$. For $\sigma_i \in \Sigma_i (1 \leq i \leq 2)$, define:*

$$\sigma_1 \ {}_{\rho_1}\!\bowtie_{\rho_2} \ \sigma_2 \Leftrightarrow (\sigma_1 \circ \rho_1) \underset{\mathcal{T}}{\bowtie} (\sigma_2 \circ \rho_2)$$
$$\sigma_1 \ {}_{\rho_1}\!\sqcup_{\rho_2} \ \sigma_2 \stackrel{\text{def}}{=} \{\langle v, n, \langle \tau_1, \tau_2 \rangle, d \rangle | \langle \tau_1, \tau_2 \rangle \in \mathcal{T}_\times \wedge \langle v, n, \tau, d \rangle \in (\sigma_1 \circ \rho_1) \underset{\mathcal{T}}{\sqcup} (\sigma_2 \circ \rho_2)\}$$
$$\Sigma_1 \ {}_{\rho_1}\!\wedge_{\rho_2} \ \Sigma_2 \stackrel{\text{def}}{=} \{\sigma_1 \ {}_{\rho_1}\!\sqcup_{\rho_2} \ \sigma_2 \mid \sigma_1 \ {}_{\rho_1}\!\bowtie_{\rho_2} \ \sigma_2\}$$

where $\tau = \rho_1(\tau_1) = \rho_2(\tau_2)$. Then the heterogeneous composition of P_1 and P_2 is:



(a) A water-controller (b) System diagram

Fig. 1. Water controlling system

$$P_1 \ {}_{\rho_1}\|_{\rho_2} \ P_2 \stackrel{\text{def}}{=} (V_1 \cup V_2, \mathcal{T}_\times, \Sigma_1 \ {}_{\rho_1 \wedge \rho_2} \ \Sigma_2)$$

Example 2. We consider a simplified version of the water controlling system proposed by Benvenuti et.al. [6]. It consists of two components: a water tank and a water level controller (Fig. 1(a)), connected in a closed-loop fashion (Fig. 1(b)). We assume that the water level $x(t)$ is changed linearly as follows:

$$x(t) \stackrel{\text{def}}{=} \begin{cases} \Delta_t * (f_i - f_o), \text{ when an Open command is received} \\ \mathbf{h} - \Delta_t * f_o, \text{ when a Close command is received} \end{cases}$$

where f_i and f_o denote the constant inlet and outlet flow, respectively, $\mathbf{h}$ denotes the height when the tank is full of water and Δ_t denotes the time elapsed since t_0 at which the tank reaches the maximum/minimum water level $\mathbf{h}$, i.e. $\Delta_t = t - t_0$. Let $P_1 = \langle V_1, \mathcal{T}_1, \Sigma_1 \rangle$ and $P_2 = \langle V_2, \mathcal{T}_2, \Sigma_2 \rangle$ be two tag systems representing the tank and the water controller, respectively, where $\mathcal{T}_1 = \langle \mathbb{R}, \leq \rangle, \mathcal{T}_2 = \langle \mathbb{N}, \leq \rangle$ and $V_1 = V_2 = \{cmd, x\}$. Variable cmd denotes the command values, which can be Open (op) or Close (cl), and variable x denotes the water level, which is of positive real type, i.e. $D_{cmd} = \{\mathbf{op}, \mathbf{cl}\}$ and $D_x = \mathbb{R}+$. Assume that each system has a single behaviour: $\Sigma_1 = \{\sigma_1\}, \Sigma_2 = \{\sigma_2\}$, where $\sigma(v, n)$ is described as follows when the parameter setting is $f_i = 2, f_o = 1, \mathbf{h} = 1$:

σ_1 :

$cmd:$		0.5;op		1.5;cl		2.5;op		3.5;cl		4.5;op	...
$x:$	0;0	0.5;0	1;0.5	1.5;1	2;0.5	2.5;0	3;0.5	3.5;1	4;0.5	4.5;0	...

σ_2 :

$cmd:$		1;op		3;cl		5;op		7;cl		9;op	...
$x:$	0;0	1;0	2;0.5	3;1	4;0.5	5;0	6;0.5	7;1	8;0.5	9;0	...

These are two different behaviours whose heterogeneous composition is possible w.r.t. morphisms such as $\rho_1 : \mathcal{T}_1 \mapsto \mathcal{T}_1$ and $\rho_2 : \mathcal{T}_2 \mapsto \mathcal{T}_1$:

$$\forall \tau_1 \in \mathcal{T}_1 : \rho_1(\tau_1) = \tau_1 \tag{1}$$

$$\forall \tau_2 \in \mathcal{T}_2 : \rho_2(\tau_2) = 0.5 * \tau_2 \tag{2}$$

Tag machines (TM) [3] have been introduced as finite representations of tag systems and a homogeneous machinery for composing homogeneous systems. Since our aim is to develop an operational theory for heterogeneous systems, it is necessary to extend the TM formalism to encompass the heterogeneous composition. In the sequel, we revisit the TM-relevant definitions, thereby proposing a heterogeneous extension to TMs.

Definition 4 (Algebraic Tag Structures [2]). *A tag structure* $\langle \mathcal{T}, \leqslant \rangle$ *is called* algebraic*, written* $\widehat{\mathcal{T}}$*, if* $\mathcal{T}$ *is equipped with an internal binary operator* $\bullet$*, called concatenation, such that:*

i) $\langle \mathcal{T}, \bullet \rangle$ *is a monoid whose identity element is denoted by* $\hat{\imath}_{\widehat{\mathcal{T}}}$
ii) Operator $\bullet$ *is compatible with* $\leqslant$*:* $\tau_1 \leqslant \tau_1'$ *and* $\tau_2 \leqslant \tau_2' \Rightarrow (\tau_1 \bullet \tau_2) \leqslant (\tau_1' \bullet \tau_2')$
iii) There is a special tag $\epsilon_{\widehat{\mathcal{T}}} \in \widehat{\mathcal{T}}$*, called the minimum tag, for which* $\epsilon_{\widehat{\mathcal{T}}} \leqslant \tau$ *and* $\epsilon_{\widehat{\mathcal{T}}} \bullet \tau = \tau \bullet \epsilon_{\widehat{\mathcal{T}}} = \epsilon_{\widehat{\mathcal{T}}}$ *for all* $\tau \in \mathcal{T}$

Example 3. The tag structure $\langle \mathbb{N} \cup \{-\infty\}, + \rangle$ is algebraic, because $(\mathbb{N}, +)$ is a monoid with 0 being the identity element and $\forall \tau_1, \tau_1', \tau_2, \tau_2' \in \mathbb{N} : \tau_1 \leq \tau_1' \wedge \tau_2 \leq \tau_2' \Rightarrow \tau_1 + \tau_2 \leq \tau_1' + \tau_2'$. The minimum element is $-\infty$, given that one defines $-\infty + \tau = \tau + (-\infty) = \tau$ for all $\tau \in \mathbb{N}$.

Note that an algebraic tag structure as above is almost the same as a *naturally ordered semiring* [11], with operations $\oplus = \mathbf{max}^{\leqslant}$ the maximum w.r.t. the time-stamp order and $\oplus = \bullet$. We do not, however, require $\mathbf{max}^{\leqslant}$ to be always defined; it may well be only a *partial* operation.

Definition 5 (Tag Pieces [2]). *Let* V *be an underlying set of variables with domain* D*. A* V-tag piece *is a tuple* $(\widehat{\mathcal{T}}, V, \mu)$ *where* $\widehat{\mathcal{T}}$ *is an algebraic tag structure and* μ *is a matrix:* $V \times V \mapsto \widehat{\mathcal{T}}$*. By abusing notation, a tag piece will also be denoted by* μ*. Tag pieces are applied to vectors of tags as follows. Let* $\overrightarrow{\tau_0} = (\tau_0^{v_1}, \tau_0^{v_2}, \ldots) \stackrel{\text{def}}{=} (\tau_0^v)^{v \in V}$ *be a vector of tags indexed by the set* V *of underlying variables. Applying* μ *to* $\overrightarrow{\tau_0}$ *results in another vector of tags* $\overrightarrow{\tau_0}'$*, written* $\overrightarrow{\tau_0}' \stackrel{\text{def}}{=} \overrightarrow{\tau_0} \bullet \mu$ *, whose* v*-th coordinate is computed by:*

$$(\overrightarrow{\tau_0} \bullet \mu)^v \stackrel{\text{def}}{=} \mathbf{max}^{\leqslant}(\tau_0^w \bullet \mu^{wv})^{w \in V}$$

where μ^{wv} *denotes the entry* $\mu[w, v]$ *in the matrix* μ*. The map* $(\overrightarrow{\tau_0}, \mu) \mapsto \overrightarrow{\tau_0} \bullet \mu$ *is partial if* $\mathbf{max}^{\leqslant}$ *is not always defined and total otherwise.*

Following up on the semiring analogy from above, application of a tag piece to a tag vector is the same as vector-matrix multiplication in the free semimodule over the (partial) semiring $\langle \mathcal{T}, \mathbf{max}^{\leqslant}, \bullet \rangle$. As an example in the tag structure $\langle \mathbb{N} \cup \{-\infty\}, + \rangle$,

$$\begin{bmatrix} 1 & 3 & 5 \end{bmatrix} \bullet \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & \epsilon \\ \epsilon & 0 & \epsilon \end{bmatrix} = \begin{bmatrix} 3 & 5 & 2 \end{bmatrix}$$

A tag piece μ can be labelled with a variable assignment $\nu : V \mapsto D$, which may be a partial function, forming a labelled tag piece $\widehat{\mu} \stackrel{\text{def}}{=} \langle \mu, \nu \rangle$. Let $Dom(\nu) = \{v |\ \nu(v) \text{ is defined}\}$ be the domain of ν. For $v \in Dom(\nu)$, we say that $\widehat{\mu}$ has an event for v. The set of all such $\widehat{\mu}$ is denoted by $L(V, \widehat{\mathcal{T}})$. We present below our extended definitions of TMs and unifiable labelled tag pieces on which the heterogeneous TM composition will be defined.

Definition 6 (Tag Machines). *A* tag machine M *is a finite automaton where transitions are annotated with labelled tag pieces. Formally, tag machine* M *is a tuple* $\langle S, s_0, \overrightarrow{\tau_0}, V, \widehat{\mathcal{T}}, E\rangle$*, where:*

- S *is a finite set of states and* s_0 *is the initial state,*
- $\overrightarrow{\tau_0} = (\tau^v)^{v\in V}$ *is the initial tag vector where* $\tau^v = \hat{\imath}_{\widehat{\mathcal{T}}}$*,*
- V *is a set of underlying variables,*
- $\widehat{\mathcal{T}}$ *is an algebraic tag structure,*
- $E \subseteq S \times L(V, \widehat{\mathcal{T}}) \times S$ *is a set of edges* $\langle s, \widehat{\mu}, s'\rangle$ *defining the transition relation.*

For $1 \le i \le 2$, labelled tag pieces $\widehat{\mu_i} \in L(V_i, \widehat{\mathcal{T}_i})$ are *unifiable* w.r.t. morphisms $\rho_i : \mathcal{T}_i \mapsto \mathcal{T}$, written $\widehat{\mu_1}\ {}_{\rho_1}\!\bowtie_{\rho_2} \widehat{\mu_2}$, if the following conditions hold (recall that μ^{wv} denotes the $[w, v]$-entry of the matrix μ):

$$\forall w, v \in V_1 \cap V_2 : \rho_1(\mu_1^{wv}) = \rho_2(\mu_2^{wv}) \tag{3}$$

$$Dom(\nu_1) \cap V_1 \cap V_2 = Dom(\nu_2) \cap V_1 \cap V_2 \tag{4}$$

$$\forall w, v \in Dom(\nu_1) \cap Dom(\nu_2) \cap V_1 \cap V_2 : \nu_1(v) = \nu_2(v) \tag{5}$$

Unifying $\widehat{\mu_1}$ and $\widehat{\mu_2}$ results in a set of labelled tag pieces $\widehat{\mu} \in L(V_1 \cup V_2, \widehat{\mathcal{T}_\times})$, written $\widehat{\mu} \stackrel{\text{def}}{=} \widehat{\mu_1}\ {}_{\rho_1}\!\sqcup_{\rho_2} \widehat{\mu_2}$ where:

$$\mu^{wv} = \begin{cases} \langle \mu_1^{wv}, \mu_2^{wv}\rangle, \text{ for } w, v \in V_1 \cap V_2 \\ \langle \mu_1^{wv}, \tau\rangle, \text{ for } w \in V_1 \setminus V_2, v \in V_1 \text{ and } \rho_1(\mu_1^{wv}) = \rho_2(\tau) \\ \langle \mu_1^{wv}, \tau\rangle, \text{ for } w \in V_1, v \in V_1 \setminus V_2 \text{ and } \rho_1(\mu_1^{wv}) = \rho_2(\tau) \\ \langle \tau, \mu_2^{wv}\rangle, \text{ for } w \in V_2, v \in V_2 \setminus V_1 \text{ and } \rho_1(\tau) = \rho_2(\mu_2^{wv}) \\ \langle \tau, \mu_2^{wv}\rangle, \text{ for } w \in V_2 \setminus V_1, v \in V_2 \text{ and } \rho_1(\tau) = \rho_2(\mu_2^{wv}) \\ \langle \epsilon_{\widehat{\mathcal{T}_1}}, \epsilon_{\widehat{\mathcal{T}_2}}\rangle, \text{otherwise} \end{cases}$$

$$\nu(v) = \begin{cases} \nu_1(v) = \nu_2(v), \text{ for } v \in Dom(\nu_1) \cap Dom(\nu_2) \cap V_1 \cap V_2 \\ \nu_1(v), \text{ for } v \in Dom(\nu_1) \setminus V_2 \\ \nu_2(v), \text{ for } v \in Dom(\nu_2) \setminus V_1 \\ \text{undefined, otherwise} \end{cases}$$

When $\widehat{\mathcal{T}_1} = \widehat{\mathcal{T}_2} = \widehat{\mathcal{T}}$ and ρ_1, ρ_2 are identity morphisms, the tag pieces are said to be homogeneous and μ^{wv} is considered to be a singleton instead of a pair of identical tags, ρ_i is thus omitted in the notations. This convention is also applied to the other notations. We next define the language of a tag machine.

Definition 7 (Tag Machine Language). *Let* $M \stackrel{\text{def}}{=} \langle S, s_0, \overrightarrow{\tau_0}, V, \widehat{\mathcal{T}}, E\rangle$ *and* r^M *be a run of* M *over a sequence of labelled tag pieces* $w = \widehat{\mu_1}\widehat{\mu_2}\ldots\widehat{\mu_n}(n \ge 1)$*, i.e.* $r^M : s_0 \xrightarrow{\widehat{\mu_1}} s_1 \xrightarrow{\widehat{\mu_2}} s_2 \ldots \xrightarrow{\widehat{\mu_n}} s_n$ *where* $\langle s_{i-1}, \widehat{\mu_i}, s_i\rangle \in E$*,* $\forall 1 \le i \le n$*. For such a run, let* τ_i *denote the tag vector at state* s_i *and* $ind_i(v_j)$ *denote the number of events that have happened to variable* $v_j \in V$ *up to state* s_i*:*

$$(\tau_i, \nu_i) \stackrel{\text{def}}{=} (\overrightarrow{\tau_{i-1}} \underset{\mathcal{T}}{\bullet} \mu_i, \nu_i) \stackrel{\text{def}}{=} (\overrightarrow{\tau_{i-1}} \underset{\mathcal{T}}{\bullet} \widehat{\mu_i})$$

$$ind_i(v_j) \stackrel{\text{def}}{=} \begin{cases} ind_{i-1}(v_j) + 1, \text{ if } \widehat{\mu_i} \text{ has an event for } v_j \\ ind_{i-1}(v_j), \text{ otherwise} \end{cases}$$

$$ind_0(v_j) \stackrel{\text{def}}{=} 0$$

Let w^i denote $\widehat{\mu_1}\widehat{\mu_2}\dots\widehat{\mu_i}$ (i.e. the length-i prefix of w) and r^i denote the sub-run of r^M over w^i. Let σ^{w^i} denote the set of events happened to all variables in V along the sub-run r^i. The semantics of the run r^M over w is the behaviour σ^{w^n} defined recursively through σ^{w^i} as follows:

$$\sigma^{w^i}(v_j,k) \stackrel{\text{def}}{=} \begin{cases} \sigma^{w^{i-1}}(v_j,k), \text{ if } k < ind_i(v_j) \\ \sigma^{w^{i-1}}(v_j,k), \text{ if } k = ind_i(v_j) \text{ and } \widehat{\mu_i} \text{ has no event for } v_j \\ (\overrightarrow{\tau_{i-1}} \underset{\mathcal{T}}{\bullet} \widehat{\mu_i})(v_j) \text{ if } k = ind_i(v_j) \text{ and } \widehat{\mu_i} \text{ has an event for } v_j \end{cases}$$

$$\sigma^{w^0} \stackrel{\text{def}}{=} \emptyset$$

The set of behaviours accepted by M consists of all behaviours σ^{w^n} such that σ^{w^n} is the semantics of some run over w in M and n is the length of w. The $\widehat{\mathcal{T}}$-language of M contains all such w and is denoted by $L(M)$.

As an example, the $\widehat{\mathcal{T}}$-language of the tag machine in Fig. 2(c), written as a regular expression, is $(\widehat{\mu_4^2} + \widehat{\mu_9^2})^+$.

Because each TM trace can represent a behaviour of tag systems, TMs can be used as an operational representation for a set of behaviours of a tag system. Therefore, the TM composition should be defined in such a way that it could also represent the tag system composition. Two TMs M_i defined over two algebraic tag structures $\widehat{\mathcal{T}_i}$ $(1 \le i \le 2)$ can be composed if there exists a pair of morphisms preserving the concatenation operations while mapping the source tag structures (i.e. $\widehat{\mathcal{T}_i}$) into another tag structure $\widehat{\mathcal{T}}$. We refer to such morphisms as *algebraic* morphisms and present a definition for them below.

Definition 8 (Algebraic Morphisms). *For $\widehat{\mathcal{T}}$ and $\widehat{\mathcal{T}'}$ two algebraic tag structures, a tag morphism $\rho : \mathcal{T} \mapsto \mathcal{T}'$ is said to be algebraic, written $\rho : \widehat{\mathcal{T}} \mapsto \widehat{\mathcal{T}'}$ if $\rho(\hat{\imath}_{\widehat{\mathcal{T}}}) = \hat{\imath}_{\widehat{\mathcal{T}'}}$ and $\rho(\epsilon_{\widehat{\mathcal{T}}}) = \epsilon_{\widehat{\mathcal{T}'}}$ and $\rho \circ \underset{\mathcal{T}}{\bullet} = \underset{\mathcal{T}'}{\bullet} \circ (\rho, \rho)$.*

Whenever two TMs can be composed, their composition is another machine defined on $\widehat{\mathcal{T}_\times}$ (or $\widehat{\mathcal{T}_1}\,{}_{\rho_1}\!\times_{\rho_2}\widehat{\mathcal{T}_2}$) $\stackrel{\text{def}}{=} \langle \mathcal{T}_\times, \underset{\mathcal{T}_\times}{\leqslant}, \underset{\mathcal{T}_\times}{\bullet} \rangle$ where $\underset{\mathcal{T}_\times}{\leqslant} \stackrel{\text{def}}{=} \langle \underset{\mathcal{T}_1}{\leqslant}, \underset{\mathcal{T}_2}{\leqslant} \rangle$ and $\underset{\mathcal{T}_\times}{\bullet} \stackrel{\text{def}}{=} \langle \underset{\mathcal{T}_1}{\bullet}, \underset{\mathcal{T}_2}{\bullet} \rangle$. In order for the TM composition to represent the tag system composition, the component machines must be *interoperable*. The intuition of TM interoperability relies on the local independence of shared variables, which requires that non-shared variables should not influence how shared ones are tagged in each system. A set $\overline{V} \subseteq V$ is *locally independent* in machine $M = \langle S, s_0, \overrightarrow{\tau_0}, V, \widehat{\mathcal{T}}, E \rangle$, written $\mathfrak{lind}_M^{\overline{V}}$, if the tag value of any $v \in \overline{V}$ evolves algebraically depending only on those of variables in $\overline{V}$. Let $\mathfrak{lind}_\mu^{\overline{V}} \stackrel{\text{def}}{=} (\forall w \in V \setminus \overline{V}, \forall v \in \overline{V} : \mu^{wv} = \epsilon_{\widehat{\mathcal{T}}})$, then $\mathfrak{lind}_M^{\overline{V}} \stackrel{\text{def}}{=} (\forall \langle s, \widehat{\mu}, s' \rangle \in E : \mathfrak{lind}_\mu^{\overline{V}})$. The interoperability between two tag machines M_1 and M_2 w.r.t. two algebraic morphisms ρ_1 and ρ_2 is denoted by $M_1 \,{}_{\rho_1}\!\bowtie_{\rho_2} M_2$ and is defined as follows.

Definition 9 (Interoperable Tag Machines). *For two tag machines M_i defined on $\widehat{\mathcal{T}_i}$ and V_i $(1 \le i \le 2)$:*

$$M_1 \,{}_{\rho_1}\!\bowtie_{\rho_2}\, M_2 \stackrel{\text{def}}{=} \mathfrak{lind}_{M_1}^{V_1 \cap V_2} \wedge \mathfrak{lind}_{M_2}^{V_1 \cap V_2} \wedge \exists \rho_1, \rho_2 : \rho_i : \widehat{\mathcal{T}_i} \mapsto \widehat{\mathcal{T}}$$

Example 4. Consider two TMs in Fig. 2(a) and 2(c) on page 11. They are interoperable w.r.t. ρ_1 and ρ_2 as defined in (1) and (2). This is because they are defined on the same set of variables and so the requirement of the shared variable tags' algebraic evolution depending on each other is trivially satisfied.

3 Tag Machine Specification Theory

Our goal is to provide an operational framework for verification and analysis of contract-based design of heterogeneous systems. To this end, we limit TMs to their *deterministic* form where labelled tag pieces annotated on transitions going out of a state are all different. Being inspired by the generic contract framework proposed by Bauer et.al. [1], we also base our framework on a TM specification theory which is a theory developed for a class of TMs acting as specifications of tag systems. Like other specification theories, the TM specification theory is also equipped with essential binary operators such as composition to combine two TMs and refinement to relate two TMs. Given two TMs $M_i = \langle S_i, s_{i0}, \overrightarrow{\tau_{i0}}, V_i, \widehat{\mathcal{T}_i}, E_i\rangle$ $(1 \leq i \leq 2)$ s.t. $M_1 \ {}_{\rho_1}\!\bowtie_{\rho_2} M_2$, the definitions below provide operations for them.

Definition 10 (TM Composition). *The composition of M_1 and M_2 is a tag machine denoted by $M_1 \ {}_{\rho_1}\!\|_{\rho_2} \ M_2 = \langle S, s_0, \overrightarrow{\tau_0}^{\times}, V_\times, \widehat{\mathcal{T}_\times}, E\rangle$, where:*

- $S = \{\langle s_1, s_2\rangle | s_i \in S_i\}$, $s_0 = \langle s_{10}, s_{20}\rangle$, $V_\times = V_1 \cup V_2$, $\widehat{\mathcal{T}_\times} = \widehat{\mathcal{T}_1} \ {}_{\rho_1}\!\times_{\rho_2} \widehat{\mathcal{T}_2}$
- $E = \left\{\left\langle \langle s_1, s_2\rangle, \widehat{\mu_1} \ {}_{\rho_1}\!\sqcup_{\rho_2} \widehat{\mu_2}, \langle s_1', s_2'\rangle\right\rangle | \langle s_i, \widehat{\mu_i}, s_i'\rangle \in E_i \wedge \widehat{\mu_1} \ {}_{\rho_1}\!\bowtie_{\rho_2} \widehat{\mu_2}\right\}$

Definition 11 (TM Refinement). *M_1 refines M_2, written $M_1 \ {}_{\rho_1}\!\preceq_{\rho_2} M_2$, if there exists a binary relation $\mathcal{R} \subseteq S_1 \times S_2$ such that:*

i) $(s_{10}, s_{20}) \in \mathcal{R}$,
ii) $\forall (s_{1i}, \widehat{\mu_1}, s_{1i}') \in E_1 :$
$$\left(\exists s_{2j} \in S_2 : (s_{1i}, s_{2j}) \in \mathcal{R}\right) \Rightarrow \left(\exists (s_{2j}, \widehat{\mu_2}, s_{2j}') \in E_2 : \begin{array}{l}(\widehat{\mu_1} \ {}_{\rho_1}\!\bowtie_{\rho_2} \widehat{\mu_2}) \ \wedge \\ (s_{1i}', s_{2j}') \in \mathcal{R}\end{array}\right)$$

The intuition of the TM composition matches the tag system composition, and the TM refinement means that the refined TM can produce the same (possibly even more) events to the variables as the refining TM can. When the morphisms ρ_1, ρ_2 are identities, we say that the refinement is *homogeneous* and write $M_1 \preceq M_2$. The following theorem is one of *independent implementability*: refinement is preserved when composing components.

Theorem 1. *For $1 \leq i \leq 2$, let M_i and M_i' be TMs defined on $\widehat{\mathcal{T}_i}$ and V_i. If $M_i \preceq M_i'$ and $M_1 \ {}_{\rho_1}\!\bowtie_{\rho_2} M_2$ and $M_1' \ {}_{\rho_1}\!\bowtie_{\rho_2} M_2'$ for some algebraic morphisms $\rho_i : \widehat{\mathcal{T}_i} \mapsto \widehat{\mathcal{T}}$, then $(M_1 \ {}_{\rho_1}\!\|_{\rho_2} M_2) \preceq (M_1' \ {}_{\rho_1}\!\|_{\rho_2} M_2')$.*

Definition 12 (TM quotient). *Let $V = V_1 \cap V_2$. The quotient of M_1 and M_2 is a tag machine denoted by $M_1 \ {}_{\rho_1}\!/_{\rho_2} \ M_2 = \langle S, s_0, \overrightarrow{\tau_0}, V_\times, \widehat{\mathcal{T}_\times}, E\rangle$, where:*

- $S = \{\langle s_1, s_2\rangle | s_i \in S_i\} \cup \{(\!|\mathfrak{u}|\!)\}$ *where $(\!|\mathfrak{u}|\!)$ is a new state, and* $s_0 = \langle s_{10}, s_{20}\rangle$

- $V_\times = V_1 \cup V_2$, $\widehat{\mathcal{T}_\times} = \widehat{\mathcal{T}_1}\,{}_{\rho_1}\!\times_{\rho_2}\widehat{\mathcal{T}_2}$
- $E = \big\{\big\langle\langle s_1, s_2\rangle, \widehat{\mu_1}\ {}_{\rho_1}\!\sqcup_{\rho_2}\ \widehat{\mu_2}, \langle s_1', s_2'\rangle\big\rangle \big| \langle s_i, \widehat{\mu_i}, s_i'\rangle \in E_i \wedge \widehat{\mu_1}\ {}_{\rho_1}\!\bowtie_{\rho_2}\ \widehat{\mu_2}\big\} \cup$
$\Big\{\big\langle\langle s_1, s_2\rangle, \widehat{\mu_1}\ {}_{\rho_1}\!\sqcup_{\rho_2}\ \widehat{\mu_2}, \textcircled{\mathfrak{u}}\big\rangle \Big| \begin{array}{l}\langle s_1, \widehat{\mu_1}, s_1'\rangle \in E_1 \wedge \langle s_2, \widehat{\mu_2}, s_2'\rangle \notin E_2 \wedge \\ \widehat{\mu_1}\ {}_{\rho_1}\!\bowtie_{\rho_2}\ \widehat{\mu_2} \ \wedge\ \mathfrak{lind}^V_{\mu_2}\end{array}\Big\} \cup$
$\Big\{\big\langle\langle s_1, s_2\rangle, \widehat{\mu_1}\ {}_{\rho_1}\!\sqcup_{\rho_2}\ \widehat{\mu_2}, \textcircled{\mathfrak{u}}\big\rangle \Big| \begin{array}{l}\langle s_1, \widehat{\mu_1}, s_1'\rangle \notin E_1 \wedge \langle s_2, \widehat{\mu_2}, s_2'\rangle \notin E_2 \wedge \\ \widehat{\mu_1}\ {}_{\rho_1}\!\bowtie_{\rho_2}\ \widehat{\mu_2} \ \wedge\ \mathfrak{lind}^V_{\mu_1} \wedge \mathfrak{lind}^V_{\mu_2}\end{array}\Big\} \cup$
$\big\{\langle \textcircled{\mathfrak{u}}, \mu, \textcircled{\mathfrak{u}}\rangle \big| \mu \in L(V_\times, \widehat{\mathcal{T}_\times}) \wedge \mathfrak{lind}^{V_1}_\mu \wedge \mathfrak{lind}^{V_2}_\mu\big\}$

An example of a quotient is displayed in Figure 3(b) on page 13, which shows the quotient $M_{\mathcal{G}_{tk}}/M_{\mathcal{A}_{tk}}$ of the TMs of Figures 2(a) and 2(b).

Theorem 2. *The quotient $M_1\ {}_{\rho_1}\!/_{\rho_2}\ M_2$ in Definition 12 satisfies the refinement $\big(M_2\ {}_{\mathsf{id}}\|_{\mathsf{proj}}\ (M_1\ {}_{\rho_1}\!/_{\rho_2}\ M_2)\big)\ {}_{\mathsf{proj}'}\!\preceq_{\mathsf{id}'} M_1$, where:*

$$\begin{aligned}\forall \tau_2 \in \widehat{\mathcal{T}_2} : \mathsf{id}(\tau_2) &= \tau_2\\ \forall \tau_1 \in \widehat{\mathcal{T}_1} : \mathsf{id}'(\tau_1) &= \tau_1\\ \forall \langle \tau_1, \tau_2\rangle \in \widehat{\mathcal{T}_\times} : \mathsf{proj}(\tau_1, \tau_2) &= \tau_2\\ \forall \big\langle \tau_2, \langle \tau_1, \tau_2\rangle\big\rangle \in \widehat{\mathcal{T}_2}\,{}_{\mathsf{id}}\!\times_{\mathsf{proj}}\widehat{\mathcal{T}_\times} : \mathsf{proj}'(\tau_2, (\tau_1, \tau_2)) &= \tau_1\end{aligned}$$

Moreover, for any tag machine M defined on $\widehat{\mathcal{T}_\times}$ and $V_\times$, it holds that:

$$\mathfrak{lind}^{V_1}_M \wedge \mathfrak{lind}^{V_2}_M \wedge \big((M_2\ {}_{\mathsf{id}}\|_{\mathsf{proj}}\ M)\ {}_{\mathsf{proj}'}\!\preceq_{\mathsf{id}'} M_1\big) \Rightarrow M \preceq M_1\ {}_{\rho_1}\!/_{\rho_2}\ M_2 \qquad (6)$$

Hence the quotient $M_1\ {}_{\rho_1}\!/_{\rho_2}\ M_2$ is the *greatest*, in the (homogeneous) refinement preorder, of all tag machines M which satisfy (6). This universal property is generally expected of quotients, *cf.* [1], and it alone implies that the quotient is uniquely defined up to two-sided homogeneous refinement [12].

Finally, the operator of *heterogeneous conjunction*, denoted ${}_{\rho_1}\!\curlywedge_{\rho_2}$, is defined as the greatest lower bound of the refinement order. Conjunction, therefore, amounts to computing the intersection of the behaviour sets, in order to find the largest common refinement. Thus, for tag systems, conjunction can be computed similarly to composition. The two operators, however, serve very different purposes, and must not therefore be confused.

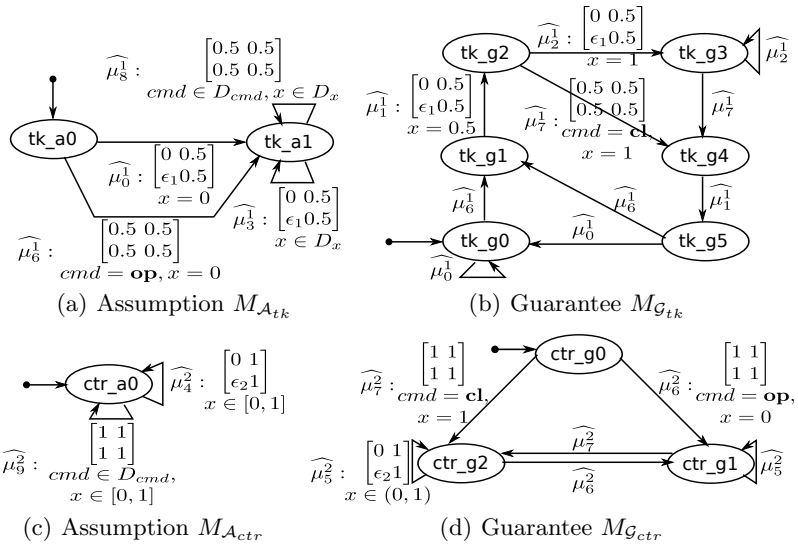
4 A Contract Framework for Tag Systems

We present in this section our contract framework for tag systems which is built on top of the TM specification theory in the previous section. We use the term *tag contract* to mean that in our framework each contract is coupled with an algebraic tag structure and a variable set which allow the contract assumption and guarantee to be represented by tag machines.

Definition 13 (Tag Contract). *A tag contract $\mathcal{C}$ is a homogeneous pair of TMs $(M_\mathcal{A}, M_\mathcal{G})$ defined over an algebraic tag structure $\widehat{\mathcal{T}}$ and variable set V:*

$$\begin{aligned}M_\mathcal{A} &\stackrel{\text{def}}{=} (S_\mathcal{A}, s_0^\mathcal{A}, \overrightarrow{\tau_0}, V, \widehat{\mathcal{T}}, E_\mathcal{A})\\ M_\mathcal{G} &\stackrel{\text{def}}{=} (S_\mathcal{G}, s_0^\mathcal{G}, \overrightarrow{\tau_0}, V, \widehat{\mathcal{T}}, E_\mathcal{G})\end{aligned}$$

Example 5. Consider the simplified water controlling system in Example 2. Assume that the algebraic tag structures of the tank and the controller are $\widehat{\mathcal{T}_1} = (\mathbb{R}+ \cup \{\epsilon_{\widehat{\mathcal{T}_1}}\}, +)$ and $\widehat{\mathcal{T}_2} = (\mathbb{N} \cup \{\epsilon_{\widehat{\mathcal{T}_2}}\}, +)$ respectively where $\epsilon_{\widehat{\mathcal{T}_1}} = \epsilon_{\widehat{\mathcal{T}_2}} = -\infty$. We present in this example a contract $\mathcal{C}_{tk} = (M_{\mathcal{A}_{tk}}, M_{\mathcal{G}_{tk}})$ for the tank component guaranteeing a linear evolution of the water level x given the assumption satisfaction. Fig. 2(a) depicts $M_{\mathcal{A}_{tk}}$ which simply requires the water tank to be empty initially and accepts Open/Close commands from the controller and Fig. 2(b) describes the guarantee $M_{\mathcal{G}_{tk}}$ on the water level evolution. That is, as long as the controlling command is received at the right time (i.e. Open when the tank is empty and Close when it is full), the water level will evolve linearly as specified in Example 2. For the sake of simplicity, the events described by the tank contract are timestamped periodically every 0.5 time unit.



(a) Assumption $M_{\mathcal{A}_{tk}}$ (b) Guarantee $M_{\mathcal{G}_{tk}}$

(c) Assumption $M_{\mathcal{A}_{ctr}}$ (d) Guarantee $M_{\mathcal{G}_{ctr}}$

Fig. 2. The tank and controller contract

The controller contract is shown in the same figure, where the controller assumption (Fig. 2(c)) states an admissible requirement on the water level input, i.e. 0 <= x <= **h** and places no requirement on its output which is the command signal. As long as such assumption is satisfied, the controller guarantees (Fig. 2(d)) to send an Open/Close command upon knowing of the tank emptiness/fullness. While the tank system uses physical time to stamp its behaviours, the controller system instead timestamps its events logically which can be described by the integer tag set $\mathbb{N}$. For the sake of expressiveness, some of the labelled tag pieces can be represented symbolically. For example, to capture any event of variable x happening at a specific time point, we label with the tag piece

capturing that time point expressions such as $x \in D_x$, meaning that in such an event x can take any value in its domain.

Definition 14 (Tag Contract Semantics). *For a tag contract* $\mathcal{C} = (M_\mathcal{A}, M_\mathcal{G})$*, define:*

- $M_\mathcal{I} \underset{M_\mathcal{A}}{\preceq} M_\mathcal{G}$ *iff* $\forall M_\mathcal{E} : (M_\mathcal{E} \preceq M_\mathcal{A} \Rightarrow (M_\mathcal{I} \parallel M_\mathcal{E}) \preceq (M_\mathcal{G} \parallel M_\mathcal{E}))$
- $[\![\mathcal{C}]\!]_{env} = \{M_\mathcal{E} \mid M_\mathcal{E} \preceq M_\mathcal{A}\}$ – *the set of* environments *of* $\mathcal{C}$
- $[\![\mathcal{C}]\!]_{impl} = \{M_\mathcal{I} \mid M_\mathcal{I} \underset{M_\mathcal{A}}{\preceq} M_\mathcal{G}\}$ – *the set of* implementations *of* $\mathcal{C}$

When contract refinement can be checked independently of the contract assumption, we say that $\mathcal{C}$ is in normalised form.

Definition 15 (Normalized Tag Contract). *A tag contract* $\mathcal{C} = (M_\mathcal{A}, M_\mathcal{G})$ *is in normalised form if the following holds:* $\forall M_\mathcal{I} : M_\mathcal{I} \in [\![C]\!]_{impl} \Leftrightarrow M_\mathcal{I} \preceq M_\mathcal{G}$

Theorem 3. *Any tag contract* $\mathcal{C} = (M_\mathcal{A}, M_\mathcal{G})$ *can be normalised by replacing* $M_\mathcal{G}$ *with* $M_\mathcal{G}/M_\mathcal{A}$*.*

Whenever a tag contract is in normalised form, checking contract satisfaction is reduced to finding a refinement relation between two TMs. As we will see later, working with normalised tag contracts can simplify the definition of contract operators and relations as well as provide a unique representation for equivalent contracts, thus we assume contracts to be in normalised form hereafter.

Example 6. Consider the tag contracts in Example 5. Let $!\widehat{\mu}$ be the class of all labelled tag pieces different from $\widehat{\mu}$, we perform the quotient of the guarantees and assumptions to normalise the contracts. The quotients are shown in Fig. 3 where

$$\widehat{\mu^1_{00}} \overset{\text{def}}{=} !\widehat{\mu^1_0} \cup !\widehat{\mu^1_6}$$
$$\widehat{\mu^1_{01}} = \widehat{\mu^1_{51}} \overset{\text{def}}{=} [(!\widehat{\mu^1_0} \cup !\widehat{\mu^1_6}) \cap (!\widehat{\mu^1_3} \cup !\widehat{\mu^1_8})]$$
$$\widehat{\mu^1_{11}} = \widehat{\mu^1_{41}} \overset{\text{def}}{=} [!\widehat{\mu^1_1} \cap (!\widehat{\mu^1_3} \cup !\widehat{\mu^1_8})]$$
$$\widehat{\mu^1_{21}} = \widehat{\mu^1_{31}} \overset{\text{def}}{=} [(!\widehat{\mu^1_2} \cup !\widehat{\mu^1_7}) \cap (!\widehat{\mu^1_3} \cup !\widehat{\mu^1_8})]$$

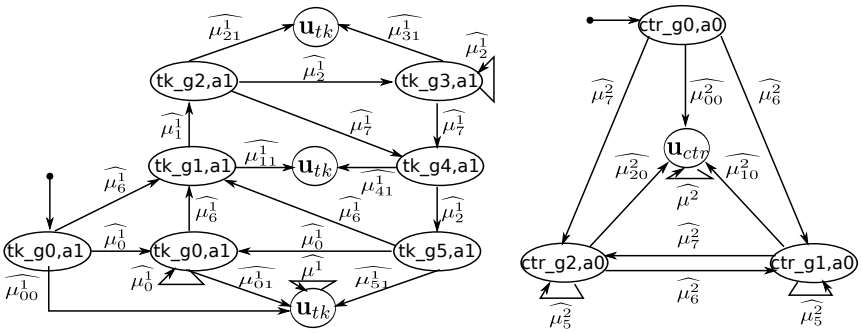
and

$$\widehat{\mu^2_{00}} \overset{\text{def}}{=} [(!\widehat{\mu^2_6} \cup !\widehat{\mu^2_7}) \cap (!\widehat{\mu^2_4} \cup !\widehat{\mu^2_9})]$$
$$\widehat{\mu^2_{10}} \overset{\text{def}}{=} [(!\widehat{\mu^2_5} \cup !\widehat{\mu^2_7}) \cap (!\widehat{\mu^2_4} \cup !\widehat{\mu^2_9})]$$
$$\widehat{\mu^2_{20}} \overset{\text{def}}{=} [(!\widehat{\mu^2_5} \cup !\widehat{\mu^2_6}) \cap (!\widehat{\mu^2_4} \cup !\widehat{\mu^2_9})]$$
$$\widehat{\mu^1} \overset{\text{def}}{=} L(V_1, \widehat{\mathcal{T}_1}), \widehat{\mu^2} \overset{\text{def}}{=} L(V_2, \widehat{\mathcal{T}_2})$$

It is easy to see that the tag systems P_1 and P_2 in Example 2 are an implementation of $\mathcal{C}_{tk}$ and $\mathcal{C}_{ctr}$ respectively, because their behaviours are included in the behaviour set of $M_{\mathcal{G}_{tk}}/M_{\mathcal{A}_{tk}}$ and $M_{\mathcal{G}_{ctr}}/M_{\mathcal{A}_{ctr}}$.

Like other contract frameworks, our tag contract framework is also equipped with contract refinement and dominance operators which are defined only when the contract operands are interoperable.

Definition 16 (Interoperable Tag Contract). *For two tag contracts* $\mathcal{C}_i = (M_{\mathcal{A}_i}, M_{\mathcal{G}_i})$ *defined over* $\widehat{\mathcal{T}_i}$ *and* V_i *(*$1 \le i \le 2$*), if* $V_1 \cap V_2$ *is locally independent in* $M_{\mathcal{A}_i}$ *and* $M_{\mathcal{G}_i}$ *and there exist two algebraic morphisms* $\rho_i : \widehat{\mathcal{T}_i} \mapsto \widehat{\mathcal{T}}$*, then* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *are said to be interoperable w.r.t.* ρ_1 *and* ρ_2*, written* $\mathcal{C}_1 \ {}_{\rho_1}\!\bowtie_{\rho_2} \mathcal{C}_2$*.*



(a) $M_{\mathcal{G}_{tk}}/M_{\mathcal{A}_{tk}}$ (b) $M_{\mathcal{G}_{ctr}}/M_{\mathcal{A}_{ctr}}$

Fig. 3. Guarantees of normalised $\mathcal{C}_{tk}$ and $\mathcal{C}_{ctr}$

4.1 Tag Contract Refinement

The refinement relation between two interoperable contracts is determined by that between their sets of implementations and environments.

Definition 17 (Tag Contract Refinement). *Given two tag contracts $\mathcal{C}_i = (M_{\mathcal{A}_i}, M_{\mathcal{G}_i})$ interoperable w.r.t. ρ_i for $1 \leq i \leq 2$. Contract $\mathcal{C}_1$ refines contract $\mathcal{C}_2$ w.r.t. ρ_i, written $\mathcal{C}_1 \ {}_{\rho_1}\preceq_{\rho_2} \mathcal{C}_2$, if the following two conditions are met:*

i) $\forall M_{\mathcal{I}_1} \in [\![\mathcal{C}_1]\!]_{impl} : \exists M_{\mathcal{I}_2} \in [\![\mathcal{C}_2]\!]_{impl} : (M_{\mathcal{I}_1} \ {}_{\rho_1}\bowtie_{\rho_2} \ M_{\mathcal{I}_2}) \wedge (M_{\mathcal{I}_1} \ {}_{\rho_1}\preceq_{\rho_2} M_{\mathcal{I}_2})$
ii) $\forall M_{\mathcal{E}_2} \in [\![\mathcal{C}_2]\!]_{env} : \exists M_{\mathcal{E}_1} \in [\![\mathcal{C}_1]\!]_{env} : (M_{\mathcal{E}_1} \ {}_{\rho_1}\bowtie_{\rho_2} \ M_{\mathcal{E}_2}) \wedge (M_{\mathcal{E}_2} \ {}_{\rho_2}\preceq_{\rho_1} M_{\mathcal{E}_1})$

The following theorem shows that when two tag contracts are in normalised form, checking refinement can be done at the *syntactic* level, i.e. by checking TM refinement between their assumptions and guarantees.

Theorem 4. *Given two normalised tag contracts $\mathcal{C}_i = (M_{\mathcal{A}_i}, M_{\mathcal{G}_i})$ interoperable w.r.t. ρ_i $(1 \leq i \leq 2)$: $\mathcal{C}_1 \ {}_{\rho_1}\preceq_{\rho_2} \mathcal{C}_2 \Leftrightarrow (M_{\mathcal{A}_2} \ {}_{\rho_2}\preceq_{\rho_1} M_{\mathcal{A}_1}) \wedge (M_{\mathcal{G}_1} \ {}_{\rho_1}\preceq_{\rho_2} M_{\mathcal{G}_2})$*

4.2 Tag Contract Dominance and Composition

Our tag contract framework also includes the tag contract composition operator, allowing two contracts defined over different algebraic tag structures and variable sets to be composed when possible.

Definition 18 (Tag Contract Dominance). *Given two tag contracts $\mathcal{C}_i = (M_{\mathcal{A}_i}, M_{\mathcal{G}_i})$ defined on $\widehat{\mathcal{T}_i}$ and V_i and interoperable w.r.t. $\rho_i : \widehat{\mathcal{T}_i} \mapsto \mathcal{T}$ for $1 \leq i \leq 2$. A contract $\mathcal{C} = (M_{\mathcal{A}}, M_{\mathcal{G}})$ dominates over the contract pair $(\mathcal{C}_1, \mathcal{C}_2)$ w.r.t. ρ_i if the following conditions are met:*

i) *$\mathcal{C}$ is defined over $\widehat{\mathcal{T}_\times}$ (or $\widehat{\mathcal{T}_1}\,{}_{\rho_1}\!\times_{\rho_2}\widehat{\mathcal{T}_2}$) and $V_\times \stackrel{\text{def}}{=} V_1 \cup V_2$*
ii) *$\mathfrak{Iind}^{V_1}_{M_{\mathcal{A}}}$ and $\mathfrak{Iind}^{V_2}_{M_{\mathcal{A}}}$ hold.*
iii) $\forall M_{\mathcal{I}_i} \in [\![\mathcal{C}_i]\!]_{impl} : M_{\mathcal{I}_1} \ {}_{\rho_1}\bowtie_{\rho_2} \ M_{\mathcal{I}_2} \Rightarrow (M_{\mathcal{I}_1} \ {}_{\rho_1}\|_{\rho_2} \ M_{\mathcal{I}_2}) \in [\![\mathcal{C}]\!]_{impl}$

$$\textit{iv)}\ \forall M_{\mathcal{E}} \in [\![\mathcal{C}]\!]_{env} : \begin{cases} \left(\forall M_{\mathcal{I}_1} \in [\![\mathcal{C}_1]\!]_{impl} : (M_{\mathcal{I}_1}\ \mathsf{id}_{\mathcal{T}_1} \|_{\mathsf{proj}_{\mathcal{T}_1}}\ M_{\mathcal{E}})\ \mathsf{proj}'_{\mathcal{T}_2} \preceq_{\mathsf{id}_{\mathcal{T}_2}} M_{\mathcal{A}_2}\right) \wedge \\ \left(\forall M_{\mathcal{I}_2} \in [\![\mathcal{C}_2]\!]_{impl} : (M_{\mathcal{I}_2}\ \mathsf{id}_{\mathcal{T}_2} \|_{\mathsf{proj}_{\mathcal{T}_2}}\ M_{\mathcal{E}})\ \mathsf{proj}'_{\mathcal{T}_1} \preceq_{\mathsf{id}_{\mathcal{T}_1}} M_{\mathcal{A}_1}\right) \end{cases}$$

where $\mathsf{id}_{\mathcal{T}_i}$, $\mathsf{proj}_{\mathcal{T}_i}$ *and* $\mathsf{proj}'_{\mathcal{T}_i}$ *are defined as follows:*

$$\forall \tau_i \in \widehat{\mathcal{T}}_i : \mathsf{id}_{\mathcal{T}_i}(\tau_i) = \tau_i \tag{7}$$

$$\forall (\tau_1, \tau_2) \in \widehat{\mathcal{T}}_1\ {}_{\rho_1}\!\times_{\rho_2} \widehat{\mathcal{T}}_2 : \mathsf{proj}_{\mathcal{T}_i}(\tau_1, \tau_2) = \tau_i \tag{8}$$

$$\forall \langle \tau_2, \langle \tau_1, \tau_2 \rangle \rangle \in \widehat{\mathcal{T}}_2\ {}_{\mathsf{id}_{\mathcal{T}_2}}\!\times_{\mathsf{proj}_{\mathcal{T}_2}} \widehat{\mathcal{T}_\times} : \mathsf{proj}'_{\mathcal{T}_1}(\tau_2, (\tau_1, \tau_2)) = \tau_1 \tag{9}$$

$$\forall \langle \tau_1, \langle \tau_1, \tau_2 \rangle \rangle \in \widehat{\mathcal{T}}_1\ {}_{\mathsf{id}_{\mathcal{T}_1}}\!\times_{\mathsf{proj}_{\mathcal{T}_1}} \widehat{\mathcal{T}_\times} : \mathsf{proj}'_{\mathcal{T}_2}(\tau_1, (\tau_1, \tau_2)) = \tau_2 \tag{10}$$

The pair of contracts $(\mathcal{C}_1, \mathcal{C}_2)$ is *dominatible* w.r.t. ρ_1 and ρ_2 if there exists a contract $\mathcal{C}$ dominating over $(\mathcal{C}_1, \mathcal{C}_2)$ w.r.t. the same morphisms. The contract composition is then defined as follows.

Definition 19 (Tag Contract Composition). *For* $1 \leq i \leq 2$, *let* $\mathcal{C}_i = (M_{\mathcal{A}_i}, M_{\mathcal{G}_i})$ *be tag contracts defined over* $\widehat{\mathcal{T}}_i$ *and* V_i. *If* $\mathcal{C}_1\ {}_{\rho_1}\!\bowtie_{\rho_2} \mathcal{C}_2$, *then their composition, denoted by* $\mathcal{C}_1\ {}_{\rho_1}\|_{\rho_2}\ \mathcal{C}_2$, *is defined as follows:*
$\mathcal{C}_1\ {}_{\rho_1}\|_{\rho_2}\ \mathcal{C}_2 \stackrel{\text{def}}{=} \left((M_{\mathcal{A}_1}\ {}_{\rho_1}/_{\rho_2}\ M_{\mathcal{G}'_2}) \curlywedge (M_{\mathcal{A}_2}\ {}_{\rho_2}/_{\rho_1}\ M_{\mathcal{G}'_1})^{-1}, (M_{\mathcal{G}'_1}\ {}_{\rho_1}\|_{\rho_2}\ M_{\mathcal{G}'_2})\right)$, *where*

- $M_{\mathcal{G}'_i}$ *is obtained from* $M_{\overline{\mathcal{G}}_i}$ *by removing all* $\widehat{\mu}$*-transitions for which* $\mathfrak{Iind}_{\mu}^{V_1 \cap V_2}$ *does not hold, and*
- M^{-1} *is a machine created by swapping the tag components in machine* $M = \langle S, s_0, \overrightarrow{\tau_0}', V_\times, \widehat{\mathcal{T}}_2\ {}_{\rho_2}\!\times_{\rho_1} \widehat{\mathcal{T}}_1, E' \rangle$. *Let* swap *be defined s.t.* $\forall \langle \tau_2, \tau_1 \rangle \in \widehat{\mathcal{T}}_2\ {}_{\rho_2}\!\times_{\rho_1} \widehat{\mathcal{T}}_1 : \mathsf{swap}(\tau_2, \tau_1) = (\tau_1, \tau_2)$. *Then* $M^{-1} = \langle S, s_0, \overrightarrow{\tau_0}, V_\times, \widehat{\mathcal{T}}_1\ {}_{\rho_1}\!\times_{\rho_2} \widehat{\mathcal{T}}_2, E \rangle$ *where* $E = \{\langle s, \mu, s' \rangle | (\langle s, \mu', s' \rangle \in E') \wedge (\mu = \mu' \circ \mathsf{swap})\}$.

The following theorem show that the tag contract composition in Def. 19 indeed dominates the contract components.

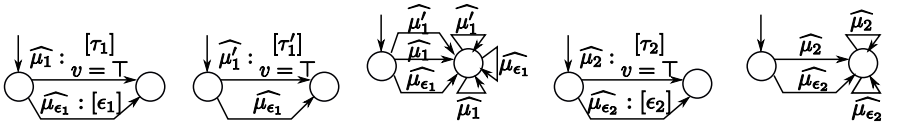
Theorem 5. *For* $1 \leq i \leq 2$, *let* $\mathcal{C}_i = (M_{\mathcal{A}_i}, M_{\mathcal{G}_i})$ *be tag contracts such that* $\mathcal{C}_1\ {}_{\rho_1}\!\bowtie_{\rho_2} \mathcal{C}_2$. *Let* $\mathcal{C} = \mathcal{C}_1\ {}_{\rho_1}\|_{\rho_2}\ \mathcal{C}_2$, *then:*

i) $\mathcal{C}$ *dominates the contract pair* $(\mathcal{C}_1, \mathcal{C}_2)$ *w.r.t.* ρ_i.
ii) For all contracts $\mathcal{C}'$ *which dominate* $(\mathcal{C}_1, \mathcal{C}_2)$ *w.r.t.* ρ_i: $\mathcal{C} \preceq \mathcal{C}'$.

The next theorem show that homogeneous refinement is preserved by heterogeneous composition.

Theorem 6. *For* $1 \leq i \leq 2$, *let* $\mathcal{C}_i = (M_{\mathcal{A}_i}, M_{\mathcal{G}_i})$ *and* $\mathcal{C}'_i = (M_{\mathcal{A}'_i}, M_{\mathcal{G}'_i})$ *be tag contracts such that* $\mathcal{C}_i$ *and* $\mathcal{C}'_i$ *are defined on* $\widehat{\mathcal{T}}_i$ *and* V_i. *If* $\mathcal{C}'_i \preceq \mathcal{C}_i$ *and* $\mathcal{C}_1\ {}_{\rho_1}\!\bowtie_{\rho_2} \mathcal{C}_2$ *and* $\mathcal{C}'_1\ {}_{\rho_1}\!\bowtie_{\rho_2} \mathcal{C}'_2$ *for some algebraic morphisms* $\rho_i : \widehat{\mathcal{T}}_1 \mapsto \widehat{\mathcal{T}}$. *Then:*

i) If $\mathcal{C}$ *dominates the contract pair* $(\mathcal{C}_1, \mathcal{C}_2)$ *w.r.t.* ρ_1 *and* ρ_2 *then it also dominates the contract pair* $(\mathcal{C}'_1, \mathcal{C}'_2)$ *w.r.t. the same morphisms.*
ii) $(\mathcal{C}'_1\ {}_{\rho_1}\|_{\rho_2}\ \mathcal{C}'_2) \preceq (\mathcal{C}_1\ {}_{\rho_1}\|_{\rho_2}\ \mathcal{C}_2)$.



(a) $M_{\mathcal{A}_1}$ (b) $M_{\mathcal{A}'_1}$ (c) $M_{\mathcal{G}_1} \equiv M_{\mathcal{G}'_1}$ (d) $M_{\mathcal{A}_2} \equiv M_{\mathcal{A}'_2}$ (e) $M_{\mathcal{G}_2} \equiv M_{\mathcal{G}'_2}$

Fig. 4. Contract components of Example 7

Contrasting Theorem 6, we note that heterogeneous refinement in general is *not* preserved even by homogeneous composition. The reason is that the former involves two morphisms which are many-to-one functions and can map two different tags into the same tag; meanwhile the latter tends to not expect such mappings. That is, if $\mathcal{C}_1 \ {}_{\rho_1}\preceq_{\rho_2}\mathcal{C}_2$ and $\mathcal{C}'_1 \ {}_{\rho_1}\preceq_{\rho_2}\mathcal{C}'_2$, then it is possible that $(\mathcal{C}_1 \parallel \mathcal{C}'_1) \ {}_{\rho_1}\not\preceq_{\rho_2}(\mathcal{C}_2 \parallel \mathcal{C}'_2)$.

Example 7. Consider as an example where:

- $\widehat{\mathcal{T}_1} = \{\tau_1, \tau'_1, \epsilon_1\}, \mathcal{T}_2 = \{\tau_2, \epsilon_2\}, \mathcal{T} = \{\tau, \epsilon\}$,
- $V_1 = V_2 = \{v\}, D_v = \{\top\}$,
- $\rho_1(\tau_1) = \rho_1(\tau'_1) = \rho_2(t_2) = \tau, \rho_1(\epsilon_1) = \rho_2(\epsilon_2) = \epsilon$

Fig. 4 shows contract components of $\mathcal{C}_i$ and $\mathcal{C}'_i$. Since $V_1 = V_2$, the contracts are interoperable with each other. It is then easy to see that $\mathcal{C}_1 \ {}_{\rho_1}\preceq_{\rho_2}\mathcal{C}_2$ and $\mathcal{C}'_1 \ {}_{\rho_1}\preceq_{\rho_2}\mathcal{C}'_2$ but $(\mathcal{C}_1 \parallel \mathcal{C}'_1) \ {}_{\rho_1}\not\preceq_{\rho_2}(\mathcal{C}_2 \parallel \mathcal{C}'_2)$ because the assumption of $\mathcal{C}_2 \parallel \mathcal{C}'_2$, which is $\big((M_{\mathcal{A}_2}/M_{\mathcal{G}'_2}) \curlywedge (M_{\mathcal{A}'_2}/M_{\mathcal{G}_2})\big)$ and equivalent to $M_{\mathcal{A}_2}$ or $M_{\mathcal{A}'_2}$, cannot heterogeneously refine w.r.t. ρ_1 and ρ_2 that of $\mathcal{C}_1 \parallel \mathcal{C}'_1$, which is $\big((M_{\mathcal{A}_1}/M_{\mathcal{G}'_1}) \curlywedge (M_{\mathcal{A}'_1}/M_{\mathcal{G}_1})\big)$ and equivalent to the empty machine with no transition.

4.3 Tag Contract Compatibility

Of particular interest is the notion of *compatibility* between contracts which depends critically on the particular partition of the variables into inputs and outputs. Intuitively, compatibility denotes the existence of a pair of interoperable implementations whose composition can be driven by some environment to avoid all incompatible states.

Definition 20 (Profile). *A profile is a tuple* $\pi = \langle \mathfrak{in}, \mathfrak{out}, \mathfrak{loc} \rangle$ *where* $\mathfrak{in}/\mathfrak{out}/\mathfrak{loc}$ *denotes a set of input/output/internal ports, respectively. Moreover these sets have to be disjoint, i.e.* $(\mathfrak{in} \cap \mathfrak{out}) = (\mathfrak{out} \cap \mathfrak{loc}) = (\mathfrak{loc} \cap \mathfrak{in}) = \emptyset$.

A tag contract defined on some variable set V has profile $\pi = \langle V^{\mathfrak{in}}, V^{\mathfrak{out}}, V^{\mathfrak{loc}} \rangle$ if and only if $V = V^{\mathfrak{in}} \cup V^{\mathfrak{out}} \cup V^{\mathfrak{loc}}$. When composing two contracts with different profiles $\mathcal{C}_i = (\pi_i, M_{\mathcal{A}_i}, M_{\mathcal{G}_i})$, we have to enforce the property that each output port should be controlled by at most one contract, i.e. $V_1^{\mathfrak{out}} \cap V_2^{\mathfrak{out}} = \emptyset$. In addition, because local ports are only visible to the underlying components, the set of local ports of one contract must be disjoint from that of the other, i.e.

$V_1^{\mathtt{loc}} \cap V_2 = V_1 \cap V_2^{\mathtt{loc}} = \emptyset$. The composite contract is defined as in Def. 19 and with profile $\pi = \langle V^{\mathtt{in}}, V^{\mathtt{out}}, V^{\mathtt{loc}} \rangle$ where: $\begin{cases} V^{\mathtt{in}} = V_1^{\mathtt{in}} \cup V_2^{\mathtt{in}} \setminus (V_1^{\mathtt{out}} \cup V_2^{\mathtt{out}}) \\ V^{\mathtt{out}} = V_1^{\mathtt{out}} \cup V_2^{\mathtt{out}} \\ V^{\mathtt{loc}} = V_1^{\mathtt{loc}} \cup V_2^{\mathtt{loc}} \end{cases}$.

Before establishing the definition of contract compatibility, we need to define the set of incompatible states in composing two tag machines with profile. Intuitively, the composite machine is at an incompatible state if a component machine does not accept some output of the other at the shared ports. This condition is similar to that proposed by de Alfaro and Henzinger for Interface Automata [8]. In the context of tag machines, compatibility also requires that the outputs happen at the same "moment" as the inputs do. In other words, their tags should agree with each other.

Definition 21 (Incompatible States). *Let $M_i = (S_i, s_0^i, \overrightarrow{\tau_0}^i, V_i, \widehat{\mathcal{T}}_i, E_i)$ be a TM with profile $\pi_i = \langle V_i^{\mathtt{in}}, V_i^{\mathtt{out}}, V_i^{\mathtt{loc}} \rangle$ $(1 \leq i \leq 2)$ s.t. $M_1 \ {}_{\rho_1}\!\bowtie_{\rho_2} M_2$. Let $\widehat{\mu_i} = \langle \mu_i, \nu_i \rangle, V^{\mathtt{out}_1} = V_1^{\mathtt{out}} \cap V_2^{\mathtt{in}}, V^{\mathtt{out}_2} = V_2^{\mathtt{out}} \cap V_1^{\mathtt{in}}, V^{\mathtt{shared}} = V_1 \cap V_2, V = Dom(\nu_1) \cap Dom(\nu_2) \cap (V^{\mathtt{out}_1} \cup V^{\mathtt{out}_2})$. A state $\langle s_1, s_2 \rangle$ in $M_1 \ {}_{\rho_1}\|_{\rho_2} M_2$ is incompatible if it is reachable from $\langle s_0^1, s_0^2 \rangle$ and does not satisfy both following conditions:*

i) $\forall \langle s_1, \widehat{\mu_1}, s_1' \rangle \in E_1$:

$$\big(Dom(\nu_1) \cap V^{\mathtt{out}_1} \neq \emptyset\big) \Rightarrow \exists \langle s_2, \widehat{\mu_2}, s_2' \rangle \in E_2 : \begin{cases} \big(Dom(\nu_1) \cap V^{\mathtt{out}_1} \subseteq Dom(\nu_2)\big) \wedge \\ \big(Dom(\nu_2) \cap V^{\mathtt{out}_2} \subseteq Dom(\nu_1)\big) \wedge \\ \big(\forall w \in V : \nu_1(w) = \nu_2(w)\big) \\ \big(\forall w \in V^{\mathtt{shared}}, \forall v \in V : \rho_1(\mu_1^{wv}) = \rho_2(\mu_2^{wv})\big) \end{cases}$$

ii) $\forall \langle s_2, \widehat{\mu_2}, s_2' \rangle \in E_2$:

$$\big(Dom(\nu_2) \cap V^{\mathtt{out}_2} \neq \emptyset\big) \Rightarrow \exists \langle s_1, \widehat{\mu_1}, s_1' \rangle \in E_1 : \begin{cases} \big(Dom(\nu_2) \cap V^{\mathtt{out}_2} \subseteq Dom(\nu_1)\big) \wedge \\ \big(Dom(\nu_1) \cap V^{\mathtt{out}_1} \subseteq Dom(\nu_2)\big) \wedge \\ \big(\forall w \in V : \nu_1(w) = \nu_2(w)\big) \\ \big(\forall w \in V^{\mathtt{shared}}, \forall v \in V : \rho_1(\mu_1^{wv}) = \rho_2(\mu_2^{wv})\big) \end{cases}$$

The set of incompatible states is denoted by $Icmp(M_1 \ {}_{\rho_1}\|_{\rho_2} M_2)$.

Given two interoperable tag machines $M_{\mathcal{I}_i}$ and their incompatible states, we are interested in knowing whether there exists some machine that can drive the composition $M_1 \ {}_{\rho_1}\|_{\rho_2} M_2$ away from their incompatible states. Such a machine should not enforce any constraint on the outputs by not accepting some of them and is referred to as a legal environment for the composition. Let $M_1 \ {}_{\rho_1}\|_{\rho_2} M_2 = \langle S, s_0, \overrightarrow{\tau_0}^{\times}, V_{\times}, \widehat{\mathcal{T}_{\times}}, E \rangle$ as defined in Def. 10 and the profile of $M_1 \ {}_{\rho_1}\|_{\rho_2} M_2$ is $\pi_{\times} = \langle V_{\times}^{\mathtt{in}}, V_{\times}^{\mathtt{out}}, V_{\times}^{\mathtt{loc}} \rangle$ where: $\begin{cases} V_{\times}^{\mathtt{in}} = V_1^{\mathtt{in}} \cup V_2^{\mathtt{in}} \setminus (V_1^{\mathtt{out}} \cup V_2^{\mathtt{out}}) \\ V_{\times}^{\mathtt{out}} = V_1^{\mathtt{out}} \cup V_2^{\mathtt{out}} \\ V_{\times}^{\mathtt{loc}} = V_1^{\mathtt{loc}} \cup V_2^{\mathtt{loc}} \end{cases}$.

Definition 22 (Legal Environment). *A TM $M_{\mathcal{E}} = (S_{\mathcal{E}}, s_0^{\mathcal{E}}, \overrightarrow{\tau_0}^{\times}, V_{\mathcal{E}}, \widehat{\mathcal{T}_{\times}}, E_{\mathcal{E}})$ with profile π_e is a legal environment for $M_c = M_1 \ {}_{\rho_1}\|_{\rho_2} M_2$ if the following conditions are satisfied:*

i) $V_{\mathcal{E}}^{\mathtt{in}} = V_{\times}^{\mathtt{out}}$

ii) $\mathfrak{lind}_{M_\mathcal{E}}^{V_\mathcal{E} \cap V_\times}$ *and* $\mathfrak{lind}_{M_c}^{V_\mathcal{E} \cap V_\times}$
iii) $Icmp(M_\mathcal{E} \parallel M_c) = \emptyset$
iv) The states in $Icmp(M_c) \times S_\mathcal{E}$ *are not reachable from* $\langle \langle s_0^1, s_0^2 \rangle, s_0^\mathcal{E} \rangle$.

We can now define the contract compatibility as follows.

Definition 23 (Contract Compatibility). *Two tag contracts* $\mathcal{C}_i$ *with profile defined over* $\widehat{\mathcal{T}}_i$ *and* V_i *are compatible if the following conditions are satisfied:*

i) $\mathcal{C}_1 \ {}_{\rho_1}\!\bowtie_{\rho_2} \mathcal{C}_2$ *for some algebraic morphisms* $\rho_i : \widehat{\mathcal{T}}_i \mapsto \widehat{\mathcal{T}}$ $(1 \leq i \leq 2)$.
ii) For $1 \leq i \leq 2$, *there exist implementations* $M_i \in [\![\mathcal{C}_i]\!]_{impl}$ *s.t.* $M_1 \ {}_{\rho_1}\!\bowtie_{\rho_2} M_2$ *and* $M_1 \ {}_{\rho_1}\!\|_{\rho_2} M_2$ *is driven by some legal environment to avoid its incompatible states.*

The simplest environment for M_c is the one that accepts all outputs and does not produce any input. For two interoperable implementations with profiles where tagging input and output ports is independent from tagging local ports, if there exists some legal environment for their heterogeneous composition, the simplest environment is also legal for such a composition. Thus, the contract compatibility check can be reduced to checking the existence of the simplest environment.

Theorem 7. *If* $M_\mathcal{E} = (S_\mathcal{E}, s_0^\mathcal{E}, \overrightarrow{\tau_0}^\times, V_\mathcal{E}, \widehat{\mathcal{T}_\times}, E_\mathcal{E})$ *with profile* π_e *is a legal environment for* $M_1 \ {}_{\rho_1}\!\|_{\rho_2} M_2$, *then* $M_{\mathcal{E}_s} = (S_{\mathcal{E}_s}, s_0^{\mathcal{E}_s}, \overrightarrow{\tau_0}^\times, V_{\mathcal{E}_s}, \widehat{\mathcal{T}_\times}, E_{\mathcal{E}_s})$ *with profile* π_{e_s} *such that:*

i) $S_{\mathcal{E}_s} = \{s_0^{\mathcal{E}_s}\}$
ii) $V_{\mathcal{E}_s}^{\mathtt{in}} = V_\mathcal{E}^{\mathtt{in}} = V_\times^{\mathtt{out}}, V_{\mathcal{E}_s}^{\mathtt{out}} = V_\times^{\mathtt{in}}, V_{\mathcal{E}_s}^{\mathtt{loc}} = \emptyset$
iii) The transition relation $E_{\mathcal{E}_s}$ *contains of all transitions* $\langle s_0^{\mathcal{E}_s}, \widehat{\mu}', s_0^{\mathcal{E}_s} \rangle$ *s.t.:*
- *There exists* $\langle s, \widehat{\mu}, s' \rangle$ *in the transition relation of* $M_1 \ {}_{\rho_1}\!\|_{\rho_2} M_2$.
- $\mu'^{wv} = \mu^{wv}$ *for* $w \in V_{\mathcal{E}_s}^{\mathtt{in}} \cup V_{\mathcal{E}_s}^{\mathtt{out}}, v \in V_\mathcal{E}^{\mathtt{in}}$ *and* $\mu'^{wv} = \epsilon_{\widehat{\mathcal{T}_\times}}$ *otherwise*
- $\nu'^w = \nu^w$ *for* $w \in Dom(\nu) \cap V_\mathcal{E}^{\mathtt{in}}$

is also a legal environment for $M_1 \ {}_{\rho_1}\|_{\rho_2} \ M_2$ *provided that* $\mathfrak{lind}_{M_1}^{V_1^{\mathtt{in}} \cup V_1^{\mathtt{out}}}$ *and* $\mathfrak{lind}_{M_2}^{V_2^{\mathtt{in}} \cup V_2^{\mathtt{out}}}$ *hold.*

5 Conclusion

We have proposed a contract-based methodology for combining heterogeneous systems built on tag machines. We aim to develop an operational framework for tag contracts which supports verification and analysis on contract-based design of heterogeneous systems. To this end, we have introduced heterogeneous composition, refinement, dominance, and compatibility of contracts, altogether enabling a formalized and rigorous design process for heterogeneous systems. Our next step is to demonstrate our methodology through a prototype tool and validate it through case studies. The development of such a tool is therefore included in our future work.

References

1. S. S. Bauer, A. David, R. Hennicker, K. G. Larsen, A. Legay, U. Nyman, and A. Wasowski. Moving from specifications to contracts in component-based design. In *FASE*, volume 7212 of *LNCS*, pages 43–58. Springer, 2012.
2. A. Benveniste, B. Caillaud, L. P. Carloni, P. Caspi, and A. L. Sangiovanni-Vincentelli. Composing heterogeneous reactive systems. *ACM Trans. Embed. Comput. Syst.*, 7(4):43:1–43:36, 2008.
3. A. Benveniste, B. Caillaud, L. P. Carloni, and A. Sangiovanni-Vincentelli. Tag machines. In *EMSOFT*, pages 255–263. ACM, 2005.
4. A. Benveniste, B. Caillaud, A. Ferrari, L. Mangeruca, R. Passerone, and C. Sofronis. Multiple viewpoint contract-based specification and design. In *FMCO*, volume 5382 of *LNCS*, pages 200–225. Springer, 2007.
5. A. Benveniste, B. Caillaud, and R. Passerone. Multi-viewpoint state machines for rich component models. In P. Mosterman and G. Nicolescu, editors, *Model-Based Design for Embedded Systems*. CRC Press, 2009.
6. L. Benvenuti, A. Ferrari, L. Mangeruca, E. Mazzi, R. Passerone, and C. Sofronis. A contract-based formalism for the specification of heterogeneous systems. In *FDL*, pages 142–147. IEEE, 2008.
7. W. Damm. Controlling speculative design processes using rich component models. In *ACSD*, pages 118–119. IEEE, 2005.
8. L. de Alfaro and T. A. Henzinger. Interface automata. *SIGSOFT Softw. Eng. Notes*, 26(5):109–120, Sept. 2001.
9. S. Dey, D. Sarkar, and A. Basu. A tag machine based performance evaluation method for job-shop schedules. *IEEE Trans. CAD of Integ. Circ. and Systems*, 29(7):1028–1041, 2010.
10. E. W. Dijkstra. Guarded commands, non-determinancy and a calculus for the derivation of programs. In *Language Hierarchies and Interfaces*, volume 46 of *LNCS*, pages 111–124. Springer, 1975.
11. M. Droste, W. Kuich, and H. Vogler. *Handbook of Weighted Automata*. EATCS Monographs in Theoretical Computer Science. Springer, 2009.
12. U. Fahrenberg, A. Legay, and A. Wasowski. Make a difference! (semantically). In *MoDELS*, volume 6981 of *LNCS*, pages 490–500. Springer, 2011.
13. C. B. Jones. *Development methods for computer programs including a notion of interference*. PhD thesis, Oxford University Computing Laboratory, 1981.
14. L. Lamport. win and sin: Predicate transformers for concurrency. *ACM Trans. Program. Lang. Syst.*, 12(3):396–428, 1990.
15. E. Lee and A. Sangiovanni-Vincentelli. A framework for comparing models of computation. *IEEE Trans. CAD of Integ. Circ. and Systems*, 17(12):1217–1229, 1998.
16. X. Liu, Y. Xiong, and E. A. Lee. The Ptolemy II framework for visual languages. In *HCC*, pages 50–. IEEE, 2001.
17. B. Meyer. Applying "Design by contract". *Computer*, 25(10):40–51, 1992.

Appendix

Lemma 1 (Total Order Preservation). *Given two algebraic morphisms* $\rho_i : \widehat{\mathcal{T}}_i \mapsto \widehat{\mathcal{T}}$. *If* $\mathcal{T}_1$ *and* $\mathcal{T}_2$ *are total orders and one of the two morphisms is strictly increasing, then* $\mathcal{T}_\times$ *is also a total order.*

Proof of Lemma 1. For any two tags $\langle \tau_1, \tau_2 \rangle$ and $\langle \tau_1', \tau_2' \rangle$ in $\mathcal{T}_\times$. Since $\mathcal{T}_1$ is a total order, we can assume that $\tau_1 \underset{\mathcal{T}_1}{\leqslant} \tau_1'$ without loss of generality. Then $\rho_1(\tau_1) \underset{\mathcal{T}}{\leqslant} \rho_1(\tau_1')$ because ρ_1 is an increasing morphism. Since $\mathcal{T}_2$ is another total order, either $\tau_2 \underset{\mathcal{T}_2}{\leqslant} \tau_2'$ or $\tau_2' \underset{\mathcal{T}_2}{\leqslant} \tau_2$ holds. Assume that the latter holds, then $\rho_2(\tau_2') \underset{\mathcal{T}}{\leqslant} \rho_2(\tau_2)$ because ρ_2 is also increasing. Moreover, $\rho_1(\tau_1) = \rho_2(\tau_2) = \tau$ and $\rho_1(\tau_1') = \rho_2(\tau_2') = \tau'$ because $\langle \tau_1, \tau_2 \rangle \in \mathcal{T}_\times$ and $\langle \tau_1', \tau_2' \rangle \in \mathcal{T}_\times$. All of these facts imply $\tau \underset{\mathcal{T}}{\leqslant} \tau'$ and $\tau' \underset{\mathcal{T}}{\leqslant} \tau$ which together mean $\tau = \tau'$ which is not possible because either the morphisms is strictly increasing. Therefore it must be the case that $\tau_2 \underset{\mathcal{T}_2}{\leqslant} \tau_2'$ holds which implies $\langle \tau_1, \tau_2 \rangle \underset{\mathcal{T}_\times}{\leqslant} \langle \tau_1', \tau_2' \rangle$. □

Proof of Theorem 1. For every run $r^{M_c} : s_{c0} \xrightarrow{\widehat{\mu_{c0}}} s_{c1} \xrightarrow{\widehat{\mu_{c1}}} \dots s_{cn} \xrightarrow{\widehat{\mu_{cn}}}$ in the composition $M_c = M_1 \ {}_{\rho_1}\|_{\rho_2} \ M_2$, there exists a run $r^{M_i} : s_{i0} \xrightarrow{\widehat{\mu_{i0}}} s_{i1} \xrightarrow{\widehat{\mu_{i1}}} \dots s_{in} \xrightarrow{\widehat{\mu_{in}}}$ in M_i such that $\widehat{\mu_{ck}} = \widehat{\mu_{1k}} \ {}_{\rho_1}\sqcup_{\rho_2} \widehat{\mu_{2k}}$ for $1 \leq k \leq n$. Because $M_i \preceq M_i'$ and M_i, M_i' are defined on the same variable set V_i, there must exist another run $r^{M_i'}$ in M_i' matching r^{M_i} on all the labels, i.e.

$$r^{M_i'} : s_{i0}' \xrightarrow{\widehat{\mu_{i0}}} s_{i1}' \xrightarrow{\widehat{\mu_{i1}}} \dots s_{in}' \xrightarrow{\widehat{\mu_{in}}}$$

Composing the runs $r^{M_1'}$ and $r^{M_2'}$ results in a run

$$r^{M_c'} : s_{c0}' \xrightarrow{\widehat{\mu_{c0}}} s_{c1}' \xrightarrow{\widehat{\mu_{c1}}} \dots s_{cn}' \xrightarrow{\widehat{\mu_{cn}}}$$

for which r^{M_c} is a refinement. Therefore, $(M_1 \ {}_{\rho_1}\|_{\rho_2} \ M_2) \preceq (M_1' \ {}_{\rho_1}\|_{\rho_2} \ M_2')$ holds. □

Proof of Theorem 2. For $1 \leq k \leq 2$, let $M_k = \langle S_k, s_{k0}, \overrightarrow{\tau_{k0}}, V_k, \widehat{\mathcal{T}_k}, E_k \rangle$ and $M_q = M_1 \ {}_{\rho_1}/_{\rho_2} \ M_2$. We first prove that refinement $\left(M_2 \ {}_{\mathsf{id}}\|_{\mathsf{proj}} \ M_q\right) \ {}_{\mathsf{proj}'} \preceq {}_{\mathsf{id}'} M_1$ is defined and then construct a relation $\mathcal{R}$ containing states of the form $\left\langle \langle s_{2j}, \langle s_{1i}, s_{2j} \rangle \rangle, s_{1i} \right\rangle \in \mathcal{R}$ to show the refinement satisfaction. Finally we prove that the quotient is the most general tag machine satisfying the refinement, that is for any deterministic tag machine M satisfying the same refinement will refine M_q.

(i) By the quotient definition, $V_1 \cap V_2$ is locally independent in M_q. For all $\langle s_q, \widehat{\mu_q}, s_q' \rangle$ in the transition relation E_q of the quotient machine, there exists $\widehat{\mu_1} \in L(V_1, \widehat{\mathcal{T}_1})$ and $\widehat{\mu_2} \in L(V_2, \widehat{\mathcal{T}_2})$ s.t. $\mathfrak{lind}_{\mu_1}^{V_1 \cap V_2}$ and $\mathfrak{lind}_{\mu_2}^{V_1 \cap V_2}$ and $\widehat{\mu_q} = \widehat{\mu_1} \ {}_{\rho_1}\sqcup_{\rho_2} \widehat{\mu_2}$. By the unification rule for two unifiable tag pieces, for $w \in V_1 \setminus V_2$ and $v \in V_1 \cap V_2$: $\mu_q^{wv} = \langle \mu_1^{wv}, \tau_2 \rangle = \langle \epsilon_{\widehat{\mathcal{T}_1}}, \epsilon_{\widehat{\mathcal{T}_2}} \rangle$, since $\rho_1(\mu_1^{wv}) =$

$\rho_2(\tau_2)$ and $\mathfrak{lind}_{M_1}^{V_1 \cap V_2}$ holds. Likewise, for $w \in V_1 \setminus V_2$ and $v \in V_2 \setminus V_1$: $\mu_q^{wv} = \langle \epsilon_{\widehat{\mathcal{T}_1}}, \epsilon_{\widehat{\mathcal{T}_2}} \rangle$. Hence V_2 is locally independent in M_q and $M_2 \ {}_{\mathsf{id}}\|_{\mathsf{proj}} \ M_q$ is defined.

For all $\langle s_c, \widehat{\mu_c}, s_c' \rangle$ in the transition relation of the composition $M_2 \ {}_{\mathsf{id}}\|_{\mathsf{proj}} \ M_q$, there exist $\langle s_2, \widehat{\mu_2}, s_2' \rangle \in E_2$ and $\langle s_q, \widehat{\mu_q}, s_q' \rangle \in E_q$ s.t.$s_c = \langle s_2, s_q \rangle$ and $s_c' = \langle s_2', s_q' \rangle$ and $\widehat{\mu_2} \ {}_{\mathsf{id}}\bowtie_{\mathsf{proj}} \ \widehat{\mu_q}$ and $\widehat{\mu_c} = \widehat{\mu_2} \ {}_{\mathsf{id}}\sqcup_{\mathsf{proj}} \ \widehat{\mu_q}$. By the unification rule for two unifiable tag pieces, for $w \in V_2 \setminus V_1$ and $v \in V_1 \cap V_2$: $\mu_c^{wv} = \langle \mu_2^{wv}, \mu_q^{wv} \rangle = \langle \epsilon_{\widehat{\mathcal{T}_2}}, \langle \epsilon_{\widehat{\mathcal{T}_1}}, \epsilon_{\widehat{\mathcal{T}_2}} \rangle \rangle$, since $V_1 \cap V_2$ is locally independent in M_q and $\mathsf{id}(\epsilon_{\widehat{\mathcal{T}_2}}) = \mathsf{proj}(\langle \epsilon_{\widehat{\mathcal{T}_1}}, \epsilon_{\widehat{\mathcal{T}_2}} \rangle) = \epsilon_{\widehat{\mathcal{T}_2}}$. Likewise, for $w \in V_2 \setminus V_1$ and $v \in V_1 \setminus V_2$: $\mu_c^{wv} = \langle \tau_2, \mu_q^{wv} \rangle = \langle \epsilon_{\widehat{\mathcal{T}_2}}, \langle \epsilon_{\widehat{\mathcal{T}_1}}, \epsilon_{\widehat{\mathcal{T}_2}} \rangle \rangle$ since $\mu_q^{wv} = \langle \epsilon_{\widehat{\mathcal{T}_1}}, \epsilon_{\widehat{\mathcal{T}_2}} \rangle$ (by the quotient definition) and $\mathsf{id}(\epsilon_{\widehat{\mathcal{T}_2}}) = \mathsf{proj}(\langle \epsilon_{\widehat{\mathcal{T}_1}}, \epsilon_{\widehat{\mathcal{T}_2}} \rangle) = \epsilon_{\widehat{\mathcal{T}_2}}$. Hence V_1 is also locally independent in $M_2 \ {}_{\mathsf{id}}\|_{\mathsf{proj}} \ M_q$, resulting in its refinement relation with M_1 is defined in $\left(M_2 \ {}_{\mathsf{id}}\|_{\mathsf{proj}} \ M_q\right) \ {}_{\mathsf{proj}'}\preceq_{\mathsf{id}'} M_1$.

(ii) Initially, $\mathcal{R}$ contains the state $\left\langle \left\langle s_{20}, \langle s_{10}, s_{20} \rangle \right\rangle, s_{10} \right\rangle \in \mathcal{R}$. For any state $\left\langle \left\langle s_{2j}, \langle s_{1i}, s_{2j} \rangle \right\rangle, s_{1i} \right\rangle \in \mathcal{R}$, if the following holds:

$$\left(\exists \widehat{\mu_2} \in L(V_2, \widehat{\mathcal{T}_2}) : s_{2j} \xrightarrow{\widehat{\mu_2}} s_{2j}'\right) \wedge \left(\exists \widehat{\mu_q} \in L(V_\times, \widehat{\mathcal{T}_\times}) : \langle s_{1i}, s_{2j} \rangle \xrightarrow{\widehat{\mu_q}} s_q\right) \ \wedge \ \left(\widehat{\mu_2} \ {}_{\mathsf{id}}\bowtie_{\mathsf{proj}} \ \widehat{\mu_q}\right)$$

then $s_q = \langle s_{1i}', s_{2j}' \rangle$ for some $s_{1i}' \in S_1$. The fact of $\widehat{\mu_2} \ {}_{\mathsf{id}}\bowtie_{\mathsf{proj}} \ \widehat{\mu_q}$ means $(\widehat{\mu_q} \circ \mathsf{proj})_{|V_2} = \widehat{\mu_2}$. By construction, the existence of $s_{2j} \xrightarrow{\widehat{\mu_2}} s_{2j}'$ in M_2 and $\langle s_{1i}, s_{2j} \rangle \xrightarrow{\widehat{\mu_q}} s_q$ in the quotient transition relation where $(\widehat{\mu_q} \circ \mathsf{proj})_{|V_2} = \widehat{\mu_2}$ implies there must exist a transition $s_{1i} \xrightarrow{\widehat{\mu_1}} s_{1i}'$ such that $\widehat{\mu_1} \ {}_{\rho_1}\bowtie_{\rho_2} \ \widehat{\mu_2}$. Since M_1, M_2 and their quotient are deterministic machines, it must be that $\widehat{\mu_q} = \widehat{\mu_1} \ {}_{\rho_1}\sqcup_{\rho_2} \widehat{\mu_2}$ and $s_q = \langle s_{1i}', s_{2j}' \rangle$. Since it can be proved easily that $(\widehat{\mu_2} \ {}_{\mathsf{id}}\sqcup_{\mathsf{proj}} \ \widehat{\mu_q}) \ {}_{\mathsf{proj}'}\bowtie_{\mathsf{id}'} \ \widehat{\mu_1}$, the second refinement condition immediately follows and so $\left\langle \left\langle s_{2j}', \langle s_{1i}', s_{2j}' \rangle \right\rangle, s_{1i}' \right\rangle \in \mathcal{R}$

(iii) Since $\mathfrak{lind}_M^{V_2}$ and $\mathfrak{lind}_M^{V_1}$ hold, refinement $\left(M_2 \ {}_{\mathsf{id}}\|_{\mathsf{proj}} \ M\right) \ {}_{\mathsf{proj}'}\preceq_{\mathsf{id}'} M_1$ is also defined. Assume that M does not refine M_q. So there must exist some runs r^M in M and r^{M_q} in M_q where the last transition of r^M cannot be simulated by r^{M_q}:

$$r^M : s_0 \xrightarrow{\widehat{\mu_0}} s_1 \xrightarrow{\widehat{\mu_1}} \dots s_n \xrightarrow{\widehat{\mu_n}}$$
$$r^{M_q} : s_{q0} \xrightarrow{\widehat{\mu_0}} s_{q1} \xrightarrow{\widehat{\mu_1}} \dots s_{qn} \not\xrightarrow{\widehat{\mu_n}}$$

We look for the longest run r^{M_2} in M_2 that can synchronize with r^M. There are three cases that can happen:

a) r^{M_2} has a shorter length than that of r^M by at least 2 transitions, i.e. its length is at most $n-2$:

$$r^{M_2} : s_{20} \xrightarrow{\widehat{\mu_{20}}} s_{21} \xrightarrow{\widehat{\mu_{21}}} \dots s_{2k} \not\xrightarrow{\widehat{\mu_{2k}}}$$

where $\widehat{\mu_{2i}}\ \mathsf{id} \bowtie_{\mathsf{proj}}\ \widehat{\mu_i}$ for $0 \leq i < k$ and $k < n$. Thus, $(\widehat{\mu_i} \circ \mathsf{proj})_{|V_2} = \widehat{\mu_{2i}}$. Using a similar line of reasoning as done in the previous part of the proof, we can conclude that $s_{qi} = (s_{1i}, s_{2i})$ for $0 \leq i \leq k$ and $s_{1i} \in S_1$. By construction, $s_{2k} \overset{\widehat{\mu_{2k}}}{\not\longrightarrow}$ drives the quotient machine to the universal state $(s_{1k}, s_{2k}) \xrightarrow{\widehat{\mu_k}} (\!|\mathfrak{u}|\!)$ from which we can infer $s_{q(k+1)} = s_{q(k+2)} = \ldots = s_n = (\!|\mathfrak{u}|\!)$. Since all transitions $\xrightarrow{\widehat{\mu}}$ s.t. $\mathfrak{lind}_{\mu}^{V_1}$ and $\mathfrak{lind}_{\mu}^{V_2}$ are allowed at this universal state, $s_{qn} \xrightarrow{\widehat{\mu_n}}$ must be also possible which contradicts the hypothesis.

b) r^{M_2}'s length is $n-1$:

$$r^{M_2} : s_{20} \xrightarrow{\widehat{\mu_{20}}} s_{21} \xrightarrow{\widehat{\mu_{21}}} \ldots s_{2n} \overset{\widehat{\mu_{2n}}}{\not\longrightarrow}$$

Similar to the proof of the previous case, we can infer $s_{qn} \xrightarrow{\widehat{\mu_n}} (\!|\mathfrak{u}|\!)$ which contradicts the hypothesis.

c) r^{M_2}'s length is as same as r^M's length:

$$r^{M_2} : s_{20} \xrightarrow{\widehat{\mu_{20}}} s_{21} \xrightarrow{\widehat{\mu_{21}}} \ldots s_{2n} \xrightarrow{\widehat{\mu_{2n}}}$$

where $\widehat{\mu_{2i}}\ \mathsf{id} \bowtie_{\mathsf{proj}}\ \widehat{\mu_i}$ for $0 \leq i \leq n$, which implies $(\widehat{\mu_i} \circ \mathsf{proj})_{|V_2} = \widehat{\mu_{2i}}$. Since r^{M_2} and r^M are composable, there must exist some run r^{M_1} in M_1:

$$r^{M_1} : s_{10} \xrightarrow{\widehat{\mu_{10}}} s_{11} \xrightarrow{\widehat{\mu_{11}}} \ldots s_{1n} \xrightarrow{\widehat{\mu_{1n}}}$$

where $(\widehat{\mu_{2i}}\ \mathsf{id} \sqcup_{\mathsf{proj}}\ \widehat{\mu_i})\ \mathsf{proj}' \bowtie_{\mathsf{id}'}\ \widehat{\mu_{1i}}$ for $0 \leq i \leq n$, which implies $\widehat{\mu_{1i}}\ \rho_1 \bowtie \rho_2\ \widehat{\mu_{2i}}$ and $\widehat{\mu_i} = \widehat{\mu_{1i}}\ \rho_1 \sqcup_{\rho_2}\ \widehat{\mu_{2i}}$. From the runs r^{M_1} and r^{M_2}, by construction, there must exist a run $\overline{r}^{M_q}$ in M_q:

$$\overline{r}^{M_q} : s_{q0} \xrightarrow{\widehat{\mu'_0}} s'_{q1} \xrightarrow{\widehat{\mu'_1}} \ldots s'_{qn} \xrightarrow{\widehat{\mu'_n}}$$

where $\widehat{\mu'_i} = \widehat{\mu_{1i}}\ \rho_1 \sqcup_{\rho_2}\ \widehat{\mu_{2i}}$ for $0 \leq i \leq n$. Since all the machines are deterministic, all transitions going out of some state are annotated with unique labelled tag pieces $\widehat{\mu'_i}$ which leads us to the conclusion that $\widehat{\mu_i} = \widehat{\mu'_i}$ and $s_{qi} = s'_{qi}$. Hence $s_{qn} \xrightarrow{\widehat{\mu_n}}$ is also possible which again contradicts the hypothesis.

Therefore, there is no run r^M in M that cannot be simulated by some run r^{M_q} in M_q where M satisfies s.t. $\mathfrak{lind}_M^{V_1}$ and $\mathfrak{lind}_M^{V_2}$ and $\left(M_2\ \mathsf{id} \|_{\mathsf{proj}}\ M\right)\ \mathsf{proj}' \preceq_{\mathsf{id}'} M_1$. As a result, M refines M_q. □

Proof of Theorem 3. Let $M_{\overline{\mathcal{G}}} = M_{\mathcal{G}}/M_{\mathcal{A}}$, we first prove that $M_{\overline{\mathcal{G}}}/M_{\mathcal{A}} \preceq M_{\overline{\mathcal{G}}}$. Let $M_q = M_{\overline{\mathcal{G}}}/M_{\mathcal{A}}$, by contraposition, assume that the refinement relation does not hold. Hence, there must exist some run r^{M_q} in M_q and $r^{M_{\overline{\mathcal{G}}}}$ in $M_{\overline{\mathcal{G}}}$ where the last transition of r^{M_q} cannot be simulated by $r^{M_{\overline{\mathcal{G}}}}$:

$$r^{M_q} : s_{q0} \xrightarrow{\widehat{\mu_0}} s_{q1} \xrightarrow{\widehat{\mu_1}} \ldots s_{qn} \xrightarrow{\widehat{\mu_n}}$$
$$r^{M_{\overline{\mathcal{G}}}} : s_0^{\overline{\mathcal{G}}} \xrightarrow{\widehat{\mu_0}} s_1^{\overline{\mathcal{G}}} \xrightarrow{\widehat{\mu_1}} \ldots s_n^{\overline{\mathcal{G}}} \overset{\widehat{\mu_n}}{\not\longrightarrow}$$

By the quotient definition for M_q and the existence of $r^{M_{\overline{\mathcal{G}}}}$, there must exist a run $r^{M_{\mathcal{A}}}$ in $M_{\mathcal{A}}$ such that:

$$r^{M_{\mathcal{A}}} : s_0^{\mathcal{A}} \xrightarrow{\widehat{\mu_0}} s_1^{\mathcal{A}} \xrightarrow{\widehat{\mu_1}} \dots s_k^{\mathcal{A}} \xrightarrow{\widehat{\mu_k}} \not\rightarrow \text{ where } 0 \leq k \leq n.$$

By the quotient definition for $M_{\overline{\mathcal{G}}}$ and the existence of $r^{M_{\mathcal{A}}}$, there must exist a run $r^{M_{\mathcal{G}}}$ in $M_{\mathcal{G}}$ such that:

$$r^{M_{\mathcal{G}}} : s_0^{\mathcal{G}} \xrightarrow{\widehat{\mu_0}} s_1^{\mathcal{G}} \xrightarrow{\widehat{\mu_1}} \dots s_k^{\mathcal{G}}$$

Since all machines are deterministic, the existence of $r^{M_{\mathcal{A}}}$ and $r^{M_{\mathcal{G}}}$ together implies that $s_{k+1}^{\overline{\mathcal{G}}} = s_{k+2}^{\overline{\mathcal{G}}} = \dots s_n^{\overline{\mathcal{G}}} = \textcircled{\mathfrak{u}}$. Hence $s_n^{\overline{\mathcal{G}}} \xrightarrow{\widehat{\mu_n}}$ must exist in $M_{\overline{\mathcal{G}}}$ which contradicts the assumption that it does not. As a result, M_q refines $M_{\overline{\mathcal{G}}}$ or $(M_{\overline{\mathcal{G}}}/M_{\mathcal{A}}) \preceq M_{\overline{\mathcal{G}}}$.

We next show that $\overline{\mathcal{C}} = (M_{\mathcal{A}}, M_{\overline{\mathcal{G}}})$ is in a normalized form by showing that $M_{\mathcal{I}} \in [\![\overline{C}]\!]_{impl} \Leftrightarrow M_{\mathcal{I}} \preceq M_{\overline{\mathcal{G}}}$.

i) $\Rightarrow$:
$M_{\mathcal{I}} \in [\![\overline{C}]\!]_{impl} \Rightarrow \forall M_{\mathcal{E}} \in [\![\overline{C}]\!]_{env} : (M_{\mathcal{I}} \parallel M_{\mathcal{E}}) \preceq (M_{\overline{\mathcal{G}}} \parallel M_{\mathcal{E}})$. Because $(M_{\overline{\mathcal{G}}} \parallel M_{\mathcal{E}}) \preceq M_{\overline{\mathcal{G}}}$, we know that $\forall M_{\mathcal{E}} \in [\![\overline{C}]\!]_{env} : (M_{\mathcal{I}} \parallel M_{\mathcal{E}}) \preceq M_{\overline{\mathcal{G}}}$. By the quotient definition, we can then infer $\forall M_{\mathcal{E}} \in [\![\overline{C}]\!]_{env} : M_{\mathcal{I}} \preceq (M_{\overline{\mathcal{G}}}/M_{\mathcal{E}})$ from which it follows that: $M_{\mathcal{I}} \preceq \parallel_{\forall M_{\mathcal{E}} \in [\![\overline{C}]\!]_{env}} (M_{\overline{\mathcal{G}}}/M_{\mathcal{E}})$.
Since $\parallel_{\forall M_{\mathcal{E}} \in [\![\overline{C}]\!]_{env}} (M_{\overline{\mathcal{G}}}/M_{\mathcal{E}}) \preceq (M_{\overline{\mathcal{G}}}/M_{\mathcal{A}})$ and $(M_{\overline{\mathcal{G}}}/M_{\mathcal{A}}) \preceq M_{\overline{\mathcal{G}}}$, the refinement $M_{\mathcal{I}} \preceq M_{\overline{\mathcal{G}}}$ also follows.

ii) $\Leftarrow$:
$M_{\mathcal{I}} \preceq M_{\overline{\mathcal{G}}} \Rightarrow \forall M_{\mathcal{E}} \in [\![\overline{C}]\!]_{env} : (M_{\mathcal{I}} \parallel M_{\mathcal{E}}) \preceq (M_{\overline{\mathcal{G}}} \parallel M_{\mathcal{E}}) \Rightarrow M_{\mathcal{I}} \in [\![\overline{C}]\!]_{impl}$.

We finally show that $\mathcal{C}$ and $\overline{\mathcal{C}}$ have the same set of environments as well as implementations. The former holds since they have the same assumption. The latter follows by the fact that $(M_{\mathcal{G}} \parallel M_{\mathcal{E}}) \preceq (M_{\overline{\mathcal{G}}} \parallel M_{\mathcal{E}})$ because $M_{\mathcal{G}} \preceq (M_{\mathcal{G}}/M_{\mathcal{A}})$ and $(M_{\overline{\mathcal{G}}} \parallel M_{\mathcal{E}}) \preceq (M_{\mathcal{G}} \parallel M_{\mathcal{E}})$ because the transitions that do not appear in $M_{\mathcal{A}}$ but in $M_{\overline{\mathcal{G}}}$ can never be synchronized by any environment $M_{\mathcal{E}}$ which refines $M_{\mathcal{A}}$.

□

Proof of Theorem 4.

(i) $\Rightarrow$: Because $\mathcal{C}_1 \ {}_{\rho_1}\preceq_{\rho_2} \mathcal{C}_2$ and $M_{\mathcal{G}_1} \in [\![\mathcal{C}_1]\!]_{impl}$, there must exist some implementation $M_{\mathcal{I}_2} \in [\![\mathcal{C}_2]\!]_{impl}$ such that $M_{\mathcal{G}_1} \ {}_{\rho_1}\preceq_{\rho_2} M_{\mathcal{I}_2}$ (by the first condition of Def. 17). Since $M_{\mathcal{I}_2} \preceq M_{\mathcal{G}_2}$, we can infer that $M_{\mathcal{G}_1} \ {}_{\rho_1}\preceq_{\rho_2} M_{\mathcal{G}_2}$. Using a similar line of reasoning, we can also infer that $M_{\mathcal{A}_2} \ {}_{\rho_2}\preceq_{\rho_1} M_{\mathcal{A}_1}$.

(ii) $\Leftarrow$: For all implementations $M_{\mathcal{I}_1}$ of contract $\mathcal{C}_1$, we have that $M_{\mathcal{I}_1} \preceq M_{\mathcal{G}_1}$ (since $\mathcal{C}_1$ is a normalized contract) which deduces the refinement $M_{\mathcal{I}_1} \ {}_{\rho_1}\preceq_{\rho_2} M_{\mathcal{G}_2}$ due to the fact that $M_{\mathcal{G}_1} \ {}_{\rho_1}\preceq_{\rho_2} M_{\mathcal{G}_2}$. Using a similar line of reasoning, we can also deduce that $M_{\mathcal{E}_2} \ {}_{\rho_2}\preceq_{\rho_1} M_{\mathcal{A}_1}$ for any environment $M_{\mathcal{E}_2}$ of contract $\mathcal{C}_2$.

□

Proof of Theorem 5. Note first that $M_{\mathcal{G}'_i}$ satisfies the following formula: $(M_{\mathcal{G}'_i} \preceq M_{\overline{\mathcal{G}}_i} \wedge \mathfrak{Iind}^{V_1\cap V_2}_{M_{\mathcal{G}'_i}}) \wedge (\forall M_{\mathcal{G}''_i} : M_{\mathcal{G}''_i} \preceq M_{\overline{\mathcal{G}}_i} \wedge \mathfrak{Iind}^{V_1\cap V_2}_{M_{\mathcal{G}''_i}} \Rightarrow M_{\mathcal{G}''_i} \preceq M_{\mathcal{G}'_i})$ where $M_{\overline{\mathcal{G}}} = M_{\mathcal{G}}/M_{\mathcal{A}}$.

By Def. 19, $\mathcal{C} = (M_{\mathcal{A}}, M_{\mathcal{G}})$, where:

- $M_{\mathcal{A}} = (M_{\mathcal{A}_1}\ {}_{\rho_1}\!/_{\rho_2}\ M_{\mathcal{G}'_2}) \curlywedge (M_{\mathcal{A}_2}\ {}_{\rho_2}\!/_{\rho_1}\ M_{\mathcal{G}'_1})^{-1}$
- $M_{\mathcal{G}} = M_{\mathcal{G}'_1}\ {}_{\rho_1}\|_{\rho_2}\ M_{\mathcal{G}'_2}$

i) $\mathcal{C}$ dominates over $(\mathcal{C}_1, \mathcal{C}_2)$ w.r.t. ρ_1 and ρ_2 because:
   a) $\mathcal{C}$ is defined over $\widehat{\mathcal{T}}_1\,{}_{\rho_1}\!\times_{\rho_2}\widehat{\mathcal{T}}_2$ and $V_\times \stackrel{\text{def}}{=} V_1 \cup V_2$, by Def. 19.
   b) By the proof of Theorem 2, both V_1 and V_2 are locally independent in both $M_{\mathcal{A}_1}\ {}_{\rho_1}\!/_{\rho_2}\ M_{\mathcal{G}'_2}$ and $M_{\mathcal{A}_2}\ {}_{\rho_2}\!/_{\rho_1}\ M_{\mathcal{G}'_1}$ and since M^{-1} does not change this local independence of machine M, $\mathfrak{Iind}^{V_1}_{M_{\mathcal{A}}} \wedge \mathfrak{Iind}^{V_2}_{M_{\mathcal{A}}}$ hold.
   c) $M_{\mathcal{I}_i} \in [\![\mathcal{C}_i]\!]_{impl} \Rightarrow M_{\mathcal{I}_i} \preceq M_{\overline{\mathcal{G}}_i}$ (by Theorem 3). If $\mathfrak{Iind}^{V_1\cap V_2}_{M_{\mathcal{I}_i}}$ holds then $M_{\mathcal{I}_i} \preceq M_{\mathcal{G}'_i}$ (by definition of $M_{\mathcal{G}'_i}$). Hence, $(M_{\mathcal{I}_1}\ {}_{\rho_1}\|_{\rho_2}\ M_{\mathcal{I}_2}) \preceq (M_{\mathcal{G}'_1}\ {}_{\rho_1}\|_{\rho_2}\ M_{\mathcal{G}'_2})$. Therefore, $(M_{\mathcal{I}_1}\ {}_{\rho_1}\|_{\rho_2}\ M_{\mathcal{I}_2}) \underset{M_{\mathcal{A}}}{\preceq} M_{\mathcal{G}}$, or equivalently $(M_{\mathcal{I}_1}\ {}_{\rho_1}\|_{\rho_2}\ M_{\mathcal{I}_2}) \in [\![\mathcal{C}]\!]_{impl}$.
   d) For all environments $M_{\mathcal{E}}$ of contract $\mathcal{C}$, it holds that $M_{\mathcal{E}} \preceq M_{\mathcal{A}}$ which implies:

$$M_{\mathcal{E}} \preceq (M_{\mathcal{A}_1}\ {}_{\rho_1}\!/_{\rho_2}\ M_{\mathcal{G}'_2}) \tag{11}$$

$$M_{\mathcal{E}} \preceq (M_{\mathcal{A}_2}\ {}_{\rho_2}\!/_{\rho_1}\ M_{\mathcal{G}'_1})^{-1} \tag{12}$$

By Def. 12 and the quotients $(M_{\mathcal{A}_1}\ {}_{\rho_1}\!/_{\rho_2}\ M_{\mathcal{G}'_2})$, $(M_{\mathcal{A}_2}\ {}_{\rho_2}\!/_{\rho_1}\ M_{\mathcal{G}'_1})$, we have that:

$$\big(M_{\mathcal{G}'_2}\ {}_{\mathsf{id}}\|_{\mathsf{proj}}\ (M_{\mathcal{A}_1}\ {}_{\rho_1}\!/_{\rho_2}\ M_{\mathcal{G}'_2})\big)\ {}_{\mathsf{proj}'}\!\preceq_{\mathsf{id}'} M_{\mathcal{A}_1}$$
$$\big(M_{\mathcal{G}'_1}\ {}_{\overline{\mathsf{id}}}\|_{\overline{\mathsf{proj}}}\ (M_{\mathcal{A}_2}\ {}_{\rho_2}\!/_{\rho_1}\ M_{\mathcal{G}'_1})\big)\ {}_{\overline{\mathsf{proj}'}}\!\preceq_{\overline{\mathsf{id}'}} M_{\mathcal{A}_2}$$

where the morphisms are defined as follows:

$$\forall \tau_2 \in \widehat{\mathcal{T}}_2 : \mathsf{id}(\tau_2) = \tau_2$$
$$\forall \tau_1 \in \widehat{\mathcal{T}}_1 : \mathsf{id}'(\tau_1) = \tau_1$$
$$\forall \langle \tau_1, \tau_2 \rangle \in \widehat{\mathcal{T}}_1\,{}_{\rho_1}\!\times_{\rho_2}\widehat{\mathcal{T}}_2 : \mathsf{proj}(\tau_1, \tau_2) = \tau_2$$
$$\forall \big\langle \tau_2, \langle \tau_1, \tau_2 \rangle \big\rangle \in \widehat{\mathcal{T}}_2\,{}_{\mathsf{id}}\!\times_{\mathsf{proj}}(\widehat{\mathcal{T}}_1\,{}_{\rho_1}\!\times_{\rho_2}\widehat{\mathcal{T}}_2) : \mathsf{proj}'(\tau_2, (\tau_1, \tau_2)) = \tau_1$$
$$\forall \tau_1 \in \widehat{\mathcal{T}}_1 : \overline{\mathsf{id}}(\tau_1) = \tau_1$$
$$\forall \tau_2 \in \widehat{\mathcal{T}}_2 : \overline{\mathsf{id}'}(\tau_2) = \tau_2$$
$$\forall \langle \tau_2, \tau_1 \rangle \in \widehat{\mathcal{T}}_2\,{}_{\rho_2}\!\times_{\rho_1}\widehat{\mathcal{T}}_1 : \overline{\mathsf{proj}}(\tau_2, \tau_1) = \tau_1$$
$$\forall \big\langle \tau_1, \langle \tau_2, \tau_1 \rangle \big\rangle \in \widehat{\mathcal{T}}_1\,{}_{\overline{\mathsf{id}}}\!\times_{\overline{\mathsf{proj}}}(\widehat{\mathcal{T}}_2\,{}_{\rho_2}\!\times_{\rho_1}\widehat{\mathcal{T}}_1) : \overline{\mathsf{proj}'}(\tau_1, (\tau_2, \tau_1)) = \tau_2$$

Refinement 11 implies $(M_{\mathcal{I}_2}\ {}_{\mathsf{id}}\|_{\mathsf{proj}}\ M_{\mathcal{E}}) \preceq \big(M_{\mathcal{I}_2}\ {}_{\mathsf{id}}\|_{\mathsf{proj}}\ (M_{\mathcal{A}_1}\ {}_{\rho_1}\!/_{\rho_2}\ M_{\mathcal{G}'_2})\big)$ for some $M_{\mathcal{I}_2} \in [\![\mathcal{C}_2]\!]_{impl}$. Because $V_1 \cap V_2$ is locally independent in

$(M_{\mathcal{A}_1} \ {}_{\rho_1}\!/_{\rho_2} \ M_{\mathcal{G}'_2})$, we can infer that $\big(M_{\mathcal{I}_2} \ {}_{\mathsf{id}} \|_{\mathsf{proj}} \ (M_{\mathcal{A}_1} \ {}_{\rho_1}\!/_{\rho_2} \ M_{\mathcal{G}'_2})\big) \preceq \big(M_{\mathcal{G}'_2} \ {}_{\mathsf{id}} \|_{\mathsf{proj}} \ (M_{\mathcal{A}_1} \ {}_{\rho_1}\!/_{\rho_2} \ M_{\mathcal{G}'_2})\big)$. Hence it follows that:

$$\big(M_{\mathcal{I}_2} \ {}_{\mathsf{id}} \|_{\mathsf{proj}} \ M_{\mathcal{E}}\big) \ {}_{\mathsf{proj}'} \preceq_{\mathsf{id}'} M_{\mathcal{A}_1} \tag{13}$$

Likewise, refinement 12 implies $(M_{\mathcal{E}}^{-1} \preceq M_{\mathcal{A}_2} \ {}_{\rho_2}\!/_{\rho_1} \ M_{\mathcal{G}'_1})$ from which we can infer the following:

$$\big(M_{\mathcal{I}_1} \ {}_{\overline{\mathsf{id}}} \|_{\overline{\mathsf{proj}}} \ M_{\mathcal{E}}^{-1}\big) \ {}_{\overline{\mathsf{proj}'}} \preceq_{\overline{\mathsf{id}'}} M_{\mathcal{A}_2} \tag{14}$$

By noticing that $\mathsf{id} \equiv \overline{\mathsf{id}'} \equiv \mathsf{id}_{\mathcal{T}_2}, \mathsf{id}' \equiv \overline{\mathsf{id}} \equiv \mathsf{id}_{\mathcal{T}_1}, \mathsf{proj} \equiv \mathsf{proj}_{\mathcal{T}_2}, \mathsf{proj}' \equiv \mathsf{proj}'_{\mathcal{T}_1}$ where $\mathsf{id}_{\mathcal{T}_i}$, $\mathsf{proj}_{\mathcal{T}_i}$ and $\mathsf{proj}'_{\mathcal{T}_i}$ are defined as in formulae 7, 8, 9 and 10, refinements 13 can be rewritten as follows:

$$(M_{\mathcal{I}_2} \ {}_{\mathsf{id}_{\mathcal{T}_2}} \|_{\mathsf{proj}_{\mathcal{T}_2}} \ M_{\mathcal{E}}) \ {}_{\mathsf{proj}'_{\mathcal{T}_1}} \preceq_{\mathsf{id}_{\mathcal{T}_1}} M_{\mathcal{A}_1}$$

Also by observing that $\overline{\mathsf{proj}} \equiv \mathsf{proj}_{\mathcal{T}_1} \circ \mathsf{swap}$, refinement 14 can be rewritten as:

$$\big(M_{\mathcal{I}_1} \ {}_{\mathsf{id}_{\mathcal{T}_1}} \|_{\mathsf{proj}_{\mathcal{T}_1} \circ \mathsf{swap}} \ M_{\mathcal{E}}^{-1}\big) \ {}_{\overline{\mathsf{proj}'}} \preceq_{\mathsf{id}_{\mathcal{T}_2}} M_{\mathcal{A}_2}$$

from which we can safely remove the composition with swap. Because such removal only results in commutating the second tag component of the composite tag structure on the left hand side of the refinement and this can be compensated by using $\mathsf{proj}'_{\mathcal{T}_2}$ in place of $\overline{\mathsf{proj}}$. Therefore, refinement 14 can be equivalently rewritten as:

$$(M_{\mathcal{I}_1} \ {}_{\mathsf{id}_{\mathcal{T}_1}} \|_{\mathsf{proj}_{\mathcal{T}_1}} \ M_{\mathcal{E}}) \ {}_{\mathsf{proj}'_{\mathcal{T}_2}} \preceq_{\mathsf{id}_{\mathcal{T}_2}} M_{\mathcal{A}_2}$$

ii) Since $\mathcal{C}$ and $\mathcal{C}'$ are defined on the same algebraic tag structure and variable set, to prove $\mathcal{C} \preceq \mathcal{C}'$, we show that $[\![\mathcal{C}']\!]_{env} \subseteq [\![\mathcal{C}]\!]_{env}$ and $[\![\mathcal{C}]\!]_{impl} \subseteq [\![\mathcal{C}']\!]_{impl}$.
a) $[\![\mathcal{C}']\!]_{env} \subseteq [\![\mathcal{C}]\!]_{env}$:
Contract $\mathcal{C}' = (M_{\mathcal{A}'}, M_{\mathcal{G}'})$ dominating over the contract pair $(\mathcal{C}_1, \mathcal{C}_2)$ implies the satisfaction of the second condition in Definition 18, i.e. for all environments $M_{\mathcal{E}}$ of $\mathcal{C}'$:

$$\begin{pmatrix} \forall M_{\mathcal{I}_1} \in [\![\mathcal{C}_1]\!]_{impl} : (M_{\mathcal{I}_1} \ {}_{\mathsf{id}_{\mathcal{T}_1}} \|_{\mathsf{proj}_{\mathcal{T}_1}} \ M_{\mathcal{E}}) \ {}_{\mathsf{proj}'_{\mathcal{T}_2}} \preceq_{\mathsf{id}_{\mathcal{T}_2}} M_{\mathcal{A}_2} \end{pmatrix} \wedge \\ \begin{pmatrix} \forall M_{\mathcal{I}_2} \in [\![\mathcal{C}_2]\!]_{impl} : (M_{\mathcal{I}_2} \ {}_{\mathsf{id}_{\mathcal{T}_2}} \|_{\mathsf{proj}_{\mathcal{T}_2}} \ M_{\mathcal{E}}) \ {}_{\mathsf{proj}'_{\mathcal{T}_1}} \preceq_{\mathsf{id}_{\mathcal{T}_1}} M_{\mathcal{A}_1} \end{pmatrix}$$

Because $M_{\mathcal{A}'}$ is an environment of contract $\mathcal{C}'$, $M_{\mathcal{G}'_i}$ is an implementation of contract $\mathcal{C}_i$, replacing $M_{\mathcal{E}}$ with $M_{\mathcal{A}'}$ and $M_{\mathcal{I}_i}$ with $M_{\mathcal{G}'_i}$ in the above formula preserves its boolean satisfiability:

$$\begin{pmatrix} (M_{\mathcal{G}'_1} \ {}_{\mathsf{id}_{\mathcal{T}_1}} \|_{\mathsf{proj}_{\mathcal{T}_1}} \ M_{\mathcal{A}'}) \ {}_{\mathsf{proj}'_{\mathcal{T}_2}} \preceq_{\mathsf{id}_{\mathcal{T}_2}} M_{\mathcal{A}_2} \end{pmatrix} \wedge \\ \begin{pmatrix} (M_{\mathcal{G}'_2} \ {}_{\mathsf{id}_{\mathcal{T}_2}} \|_{\mathsf{proj}_{\mathcal{T}_2}} \ M_{\mathcal{A}'}) \ {}_{\mathsf{proj}'_{\mathcal{T}_1}} \preceq_{\mathsf{id}_{\mathcal{T}_1}} M_{\mathcal{A}_1} \end{pmatrix}$$

The second conjunct directly implies that $M_{\mathcal{A}'} \preceq (M_{\mathcal{A}_1} \ {}_{\rho_1}\!/_{\rho_2} \ M_{\mathcal{G}'_2})$ and using a line of reasoning similar to the previous proof, we can infer from the first conjunct that $M_{\mathcal{A}'}^{-1} \preceq (M_{\mathcal{A}_2} \ {}_{\rho_2}\!/_{\rho_1} \ M_{\mathcal{G}'_1})$. Therefore:

$M_{\mathcal{A}'} \preceq \big((M_{\mathcal{A}_1} \ {}_{\rho_1}\!/_{\rho_2} \ M_{\mathcal{G}_2}) \curlywedge (M_{\mathcal{A}_2} \ {}_{\rho_2}\!/_{\rho_1} \ M_{\mathcal{G}_1})^{-1}\big)$, or equivalently $M_{\mathcal{A}'} \preceq M_{\mathcal{A}}$

b) $[\![\mathcal{C}]\!]_{impl} \subseteq [\![\mathcal{C}']\!]_{impl}$:
For all implementations $M_{\mathcal{I}}$ of contract $\mathcal{C}$, the following refinement relation holds for all possible environment of $\mathcal{C}$:

$$\forall M_{\mathcal{E}} \in [\![\mathcal{C}]\!]_{env} : (M_{\mathcal{I}} \parallel M_{\mathcal{E}}) \preceq (M_{\mathcal{G}} \parallel M_{\mathcal{E}})$$

Because $[\![\mathcal{C}']\!]_{env} \subseteq [\![\mathcal{C}]\!]_{env}$ due to the previous proof, such refinement obviously holds for all possible environment of contract $\mathcal{C}'$:

$$\forall M_{\mathcal{E}} \in [\![\mathcal{C}']\!]_{env} : (M_{\mathcal{I}} \parallel M_{\mathcal{E}}) \preceq (M_{\mathcal{G}} \parallel M_{\mathcal{E}}) \tag{15}$$

In addition, because $\mathcal{C}'$ dominates over $(\mathcal{C}_1, \mathcal{C}_2)$, the third condition of Def. 18 must be satisfied:

$$\forall M_{\mathcal{I}_i} \in [\![\mathcal{C}_i]\!]_{impl} : M_{\mathcal{I}_1} \ {}_{\rho_1}\!\bowtie_{\rho_2} M_{\mathcal{I}_2} \Rightarrow (M_{\mathcal{I}_1} \ {}_{\rho_1}\|_{\rho_2} M_{\mathcal{I}_2}) \in [\![\mathcal{C}']\!]_{impl}$$

which implies $(M_{\mathcal{G}'_1} \ {}_{\rho_1}\|_{\rho_2} M_{\mathcal{G}'_2}) \in [\![\mathcal{C}']\!]_{impl}$ since $M_{\mathcal{G}'_i} \in [\![\mathcal{C}_i]\!]_{impl}$ and $\mathfrak{bind}_{\mathcal{G}'_i}^{V_1 \cap V_2}$ holds. As a result, $M_{\mathcal{G}}$ is also an implementation of contract $\mathcal{C}'$ which, in turn, implies a similar refinement relation:

$$\forall M_{\mathcal{E}} \in [\![\mathcal{C}']\!]_{env} : (M_{\mathcal{G}} \parallel M_{\mathcal{E}}) \preceq (M_{\mathcal{G}'} \parallel M_{\mathcal{E}}) \tag{16}$$

Combining together two facts 15 and 16, the following refinement holds for all implementation $M_{\mathcal{I}}$ of contract $\mathcal{C}$:

$$\forall M_{\mathcal{E}} \in [\![\mathcal{C}']\!]_{env} : (M_{\mathcal{I}} \parallel M_{\mathcal{E}}) \preceq (M_{\mathcal{G}'} \parallel M_{\mathcal{E}})$$

Hence, $M_{\mathcal{I}}$ is also an implementation of contract $\mathcal{C}'$ or $[\![\mathcal{C}]\!]_{impl} \subseteq [\![\mathcal{C}']\!]_{impl}$. □

Proof of Theorem 6. The first property holds because the satisfaction of two conditions in Def. 18 can be deduced from the fact that $[\![\mathcal{C}'_1]\!]_{impl} \subseteq [\![\mathcal{C}_1]\!]_{impl}$, $[\![\mathcal{C}'_2]\!]_{impl} \subseteq [\![\mathcal{C}_2]\!]_{impl}$, $M_{\mathcal{A}_1} \preceq M_{\mathcal{A}'_1}$, $M_{\mathcal{A}_2} \preceq M_{\mathcal{A}'_2}$ and $\mathcal{C}$ dominates $(\mathcal{C}_1, \mathcal{C}_2)$ w.r.t. ρ_1 and ρ_2. The second property follows directly from the first property of this theorem and the second property of Theorem 5.

□

Proof of Theorem 7. To prove that $M_{\mathcal{E}_s}$ is a legal environment for $M_c = M_1 \ {}_{\rho_1}\|_{\rho_2} M_2$, we show that the four conditions of Def. 22.

a) Trivial.
b) To prove that $M_1 \ {}_{\rho_1}\|_{\rho_2} M_2$ and $M_{\mathcal{E}_s}$ are interoperable, we show that $V_\times \cap V_{\mathcal{E}_s} = V_\times^{\mathtt{in}} \cup V_\times^{\mathtt{out}} = (V_1^{\mathtt{in}} \cup V_1^{\mathtt{out}}) \cup (V_2^{\mathtt{in}} \cup V_2^{\mathtt{out}})$ is locally independent in both machines.
   i) $V_\times \cap V_{\mathcal{E}_s}$ is locally independent in $M_1 \ {}_{\rho_1}\|_{\rho_2} M_2$:
   For $1 \leq i \leq 2$, let $V_i^{\mathtt{inout}} = V_i^{\mathtt{in}} \cup V_i^{\mathtt{out}}$. Then $V_1 \cap V_2 = V_1^{\mathtt{inout}} \cap V_2^{\mathtt{inout}}$. For all $\langle s_q, \widehat{\mu_c}, s'_q \rangle$ in the transition relation E of the composition machine,

there exist $\widehat{\mu_1} \in L(V_1, \widehat{\mathcal{T}_1})$ and $\widehat{\mu_2} \in L(V_2, \widehat{\mathcal{T}_2})$ s.t. $\mathfrak{bind}_{\mu_1}^{V_1 \cap V_2}$ and $\mathfrak{bind}_{\mu_2}^{V_1 \cap V_2}$ and $\widehat{\mu_c} = \widehat{\mu_1}\ {}_{\rho_1}\sqcup_{\rho_2}\ \widehat{\mu_2}$. By the unification rule for two unifiable tag pieces:

$$\begin{array}{l}\forall w \in V_1^{\mathfrak{loc}}, \forall v \in V_1^{\mathfrak{inout}} \cap V_2^{\mathfrak{inout}} : \mu_c^{wv} = \langle \mu_1^{wv}, \tau_2 \rangle = \langle \epsilon_{\widehat{\mathcal{T}_1}}, \epsilon_{\widehat{\mathcal{T}_2}} \rangle \\ \forall w \in V_1^{\mathfrak{loc}}, \forall v \in V_1^{\mathfrak{inout}} : \mu_c^{wv} = \langle \mu_1^{wv}, \tau_2 \rangle = \langle \epsilon_{\widehat{\mathcal{T}_1}}, \epsilon_{\widehat{\mathcal{T}_2}} \rangle \\ \forall w \in V_1^{\mathfrak{loc}}, \forall v \in V_2^{\mathfrak{inout}} \setminus V_1^{\mathfrak{inout}} : \mu_c^{wv} = \langle \epsilon_{\widehat{\mathcal{T}_1}}, \epsilon_{\widehat{\mathcal{T}_2}} \rangle\end{array}$$

since $\rho_1(\mu_1^{wv}) = \rho_2(\tau_2)$ and $\mathfrak{bind}_{M_1}^{V_1 \cap V_2}$ and $\mathfrak{bind}_{M_1}^{V_1^{\mathfrak{inout}}}$ Hence:

$$\forall w \in V_1^{\mathfrak{loc}}, \forall v \in V_1^{\mathfrak{inout}} \cup V_2^{\mathfrak{inout}} : \mu_c^{wv} = \langle \epsilon_{\widehat{\mathcal{T}_1}}, \epsilon_{\widehat{\mathcal{T}_2}} \rangle \tag{17}$$

Using a similar line of reasoning, we also infer:

$$\forall w \in V_2^{\mathfrak{loc}}, \forall v \in V_1^{\mathfrak{inout}} \cup V_2^{\mathfrak{inout}} : \mu_c^{wv} = \langle \epsilon_{\widehat{\mathcal{T}_1}}, \epsilon_{\widehat{\mathcal{T}_2}} \rangle \tag{18}$$

From 17 and 18, we can conclude that $V_\times \cap V_{\mathcal{E}_s}$ is locally independent in $M_1\ {}_{\rho_1}\|_{\rho_2}\ M_2$.

ii) $V_\times \cap V_{\mathcal{E}_s}$ is locally independent in $M_{\mathcal{E}_s}$:
Let $V^{\mathfrak{shared}} = V_\times \cap V_{\mathcal{E}} = (V_\times^{\mathfrak{in}} \cap V_{\mathcal{E}}^{\mathfrak{in}}) \cup (V_{\mathcal{E}}^{\mathfrak{out}} \cap V_\times^{\mathfrak{in}}) \cup (V_\times^{\mathfrak{out}} \cap V_{\mathcal{E}}^{\mathfrak{in}})$. By assumption $V_{\mathcal{E}}^{\mathfrak{in}} = V_\times^{\mathfrak{out}}$, therefore:

$$V^{\mathfrak{shared}} = (V_{\mathcal{E}}^{\mathfrak{out}} \cap V_\times^{\mathfrak{in}}) \cup V_\times^{\mathfrak{out}} \tag{19}$$

In addition, $V_{\mathcal{E}} \cap V_{\mathcal{E}_s} = (V_{\mathcal{E}}^{\mathfrak{in}} \cup V_{\mathcal{E}}^{\mathfrak{out}} \cup V_{\mathcal{E}}^{\mathfrak{loc}}) \cap (V_{\mathcal{E}_s}^{\mathfrak{in}} \cup V_{\mathcal{E}_s}^{\mathfrak{out}})$. Because $V_{\mathcal{E}}^{\mathfrak{loc}}$ only contains internal ports and $V_{\mathcal{E}}^{\mathfrak{in}} = V_{\mathcal{E}_s}^{\mathfrak{in}}$, we can infer

$$V_{\mathcal{E}} \cap V_{\mathcal{E}_s} = V_{\mathcal{E}}^{\mathfrak{in}} \cup (V_{\mathcal{E}}^{\mathfrak{out}} \cap V_{\mathcal{E}_s}^{\mathfrak{out}}) = (V_{\mathcal{E}}^{\mathfrak{out}} \cap V_\times^{\mathfrak{in}}) \cup V_\times^{\mathfrak{out}} \tag{20}$$

Therefore $V_{\mathcal{E}} \cap V_{\mathcal{E}_s} \subseteq V_\times \cap V_{\mathcal{E}_s}$ and $V^{\mathfrak{shared}} = V_\times \cap V_{\mathcal{E}} = V_{\mathcal{E}} \cap V_{\mathcal{E}_s}$ (from 19 and 20). By construction, for all $\langle s, \widehat{\mu}', s' \rangle$ in $E_{\mathcal{E}_s}$, for $w \notin V_\times \cap V_{\mathcal{E}_s}$ and $v \in V_\times \cap V_{\mathcal{E}_s}$: $\mu'^{wv} = \epsilon_{\widehat{\mathcal{T}_\times}}$. Therefore $V_\times \cap V_{\mathcal{E}_s}$ is also locally independent in $M_{\mathcal{E}_s}$.

c) $Icmp(M_c \parallel M_{\mathcal{E}_s}) = Icmp(M_c \parallel M_{\mathcal{E}}) = \emptyset$: trivial since the simplest environment accepts all possible output and produces no input.

d) $Icmp(M_c) \times S_{\mathcal{E}_s}$ is not reachable:

Let $\begin{cases} \widehat{\mu}_i = \langle \mu_i, \nu_i \rangle, \widehat{\mu}_i{}' = \langle \mu_i', \nu_i' \rangle, \\ V^{\mathfrak{out}_\times} = V_\times^{\mathfrak{out}} \cap V_{\mathcal{E}}^{\mathfrak{in}}, \\ V_s^{\mathfrak{out}_\times} = V_\times^{\mathfrak{out}} \cap V_{\mathcal{E}_s}^{\mathfrak{in}}, \\ V^{\mathfrak{out}_\mathcal{E}} = V_{\mathcal{E}}^{\mathfrak{out}} \cap V_\times^{\mathfrak{in}}, \\ V_s^{\mathfrak{out}_\mathcal{E}} = V_{\mathcal{E}_s}^{\mathfrak{out}} \cap V_\times^{\mathfrak{in}}, \\ V^{\mathfrak{shared}} = V_\times \cap V_{\mathcal{E}}, \\ V_s^{\mathfrak{shared}} = V_\times \cap V_{\mathcal{E}_s} \end{cases}$

Since $M_{\mathcal{E}}$ is a legal environment for M_c, there exists no run leading to the incompatible states of M_c that contains only output or local transitions. An output/local transition is s.t. its labelled tag piece $\widehat{\mu}$ has events only for output/local ports. If there exists such a run called r^i:

$$r^i : s_0 \xrightarrow{\widehat{\mu_1}^c} s_1 \ldots \xrightarrow{\widehat{\mu_n}^c} s_n$$

where $s_n \in Icmp(M_c)$ and $\widehat{\mu_i}^c$ has events for output/local ports only, then there also exists a run $\bar{r}^i$ in $M_\mathcal{E}$:

$$\bar{r}^i : s_0^\mathcal{E} \xrightarrow{\widehat{\mu_1}^\mathcal{E}} s_1^\mathcal{E} \ldots \xrightarrow{\widehat{\mu_n}^\mathcal{E}} s_n^\mathcal{E}$$

where $\widehat{\mu_i}^c \bowtie \widehat{\mu_i}^\mathcal{E}$ for $1 \leq i \leq n$. Indeed, since $M_\mathcal{E}$ is a legal environment for M_c, $\langle s_0, s_0^\mathcal{E} \rangle$ must be compatible. That is if $\widehat{\mu_0}^c$ has only output and local events then there must exist a transition $\langle s_0^\mathcal{E}, \widehat{\mu_1}^\mathcal{E}, s_1^\mathcal{E} \rangle \in E_{\mathcal{E}_s}$ such that $\nu_1^\mathcal{E}$ is defined only for input and local variables:

$$\nu_1^\mathcal{E}(w) = \begin{cases} \nu_1^c(w), \forall w \in Dom(\nu_1^c) \cap V^{\mathtt{out}_\times} \\ \text{undefined}, \forall w \in V_\mathcal{E}^{\mathtt{out}} \end{cases}$$

Moreover, by condition 1 of Def. 21, we know that $\mu_1^c[wv] = \mu_1^\mathcal{E}[wv]$ for $w \in V^{\mathtt{shared}}$ and $v \in V$ where $V = Dom(\nu_1^c) \cap Dom(\nu_1^\mathcal{E}) \cap (V^{\mathtt{out}_\times} \cup V^{\mathtt{out}_\mathcal{E}})$. Since $Dom(\nu_1^\mathcal{E}) \cap V^{\mathtt{out}_\mathcal{E}} = \emptyset$, we can infer that $V = Dom(\nu_1^c) \cap V^{\mathtt{out}_\times}$. For variables in $V^{\mathtt{shared}} \setminus V$, there are no events for them in $\widehat{\mu_1}^c$ and therefore $\mu_1^c[wv] = \epsilon_{\widehat{\mathcal{T}_\times}}$ for $w \in V^{\mathtt{shared}}$ and $v \in V^{\mathtt{shared}} \setminus V$. Likewise, since only variables in V have an event in $\widehat{\mu_1}^\mathcal{E}$, it holds that $\mu_1^\mathcal{E}[wv] = \epsilon_{\widehat{\mathcal{T}_\times}}$. As a consequence, $\widehat{\mu_1}^c \bowtie \widehat{\mu_1}^\mathcal{E}$ holds as well and the transition $\left\langle \langle s_0, s_0^\mathcal{E} \rangle, \widehat{\mu_i}^c \sqcup \widehat{\mu_i}^\mathcal{E}, \langle s_1, s_1^\mathcal{E} \rangle \right\rangle$ is included in the transition relation of M_c. Since $M_\mathcal{E}$ is a legal environment for M_c, $\langle s_1, s_1^\mathcal{E} \rangle \rangle$ is also a compatible state. By inductive reasoning we can similarly prove that $\widehat{\mu_i}^c \bowtie \widehat{\mu_i}^\mathcal{E}$ for $1 < i \leq n$.

Therefore, an incompatible state of M_c can be reached which contradicts the assumption of a legal environment for M_c and implies that no such run r^i should exists in M_c. All runs leading to an incompatible state in M_c thus contain at least one transition where input events are expected to happen. By definition, the simplest environment will not output the expected inputs, thereby blocking M_c from going to any incompatible state as soon as possible.