Compositional Heterogeneous Abstraction

Akshay Rajhans Bruce H. Krogh

Department of Electrical and Computer Engineering
Carnegie Mellon University, Pittsburgh, PA 15213
{ arajhans | krogh }@ece.cmu.edu

ABSTRACT

In model-based development, abstraction provides insight and tractability. Different formalisms are often used at different levels of abstraction to represent the variety of concerns that need to be addressed when designing complex cyber-physical systems. In this paper, we consider the problem of establishing abstraction across heterogeneous formalisms in a compositional manner. We use the framework of behavioral semantics to elucidate the general conditions that must be satisfied to assure that the composition of abstractions for individual components is an abstraction for the composition of the components. The theoretical concepts are illustrated using an example of a cooperative intersection collision avoidance system (CICAS).

Categories and Subject Descriptors

G.4 [**Mathematical Software**]: Verification; I.6.4 [**Simulation and Modeling**]: Model Validation and Analysis

Keywords

Heterogeneous Verification; Compositional Reasoning

1. INTRODUCTION

Model-based development (MBD) refers to the creation of mathematical models of systems under design and checking those models against design specifications using suitable analysis tools. The MBD approach aims to catch errors early in the design process, thereby avoiding costly re-design/re-development cycles. For all but the most trivial cyber-physical systems (CPS), abstraction is essential for making analysis and verification tractable. Different modeling formalisms are often used in various abstractions to facilitate the design of particular aspects of the system. In our previous work, we proposed the use of behavior relations to support abstraction across heterogeneous modeling formalisms [19]. In this paper, we address the problem of establishing heterogeneous abstraction in a compositional manner.

Several tools have been developed to support simulation using heterogeneous models. Ptolemy II, for example, supports hierarchical integration of multiple "models of computation" into a single simulation model based on an actor-oriented formalism [8]. MILAN [17] is an integrated simulation framework that allows different components of a system to be built using different tools. The Metropolis toolchain [5] supports multiple analysis tools for design and simulation. None of these tools deals with abstraction, however, other than in the form of encapsulation for components and subsystems.

Heterogeneous abstraction across particular pairs of modeling formalisms appears often in the literature. Examples include hybrid abstractions of nonlinear systems [13, 10], linear hybrid automata abstractions of hybrid systems with linear continuous dynamics [12], discrete abstractions of hybrid systems [4, 9, 3], and continuous abstractions of hybrid systems [2]. In addition to being specific in the formalisms that are used, many of these methods are not compositional. Compositional methods, such as assume-guarantee reasoning, with abstraction defined by language inclusion [16] and simulation relations [11, 14], are usually defined in the context of a single formalism. Behavior-interaction-priority framework for embedded software uses structured interaction invariants to support compositional analysis but only for transition system models [6]. Our objective is to develop a general framework that elucidates the basic conditions for compositional abstraction between any pair of heterogeneous formalisms.

The notion of tagged signal semantics has been proposed to compare and compose heterogeneous reactive systems [18, 7]. Julius creates a behavioral framework for modeling control as a behavior interconnection problem [15] . These approaches use system trajectories or behaviors as a mathematical framework for creating relations between the semantics of different modeling formalisms. We have used a similar approach to establish abstractions across different formalisms, which can then be used then be used for heterogeneous verification [19]. We use the behavioral framework in this paper to develop compositional heterogeneous abstraction.

The paper is organized as follows. We introduce the notation and the problem description in Sect. 2. Sect. 3 develops a framework for relating the local semantics of a component model with its global semantics for the purposes of composing it with other component models. Sect. 4 develops

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
HSCC'13, April 8–11, 2013, Philadelphia, Pennsylvania, USA.
Copyright 2013 ACM 978-1-4503-1567-8/13/04 ...$15.00.

compositional heterogeneous abstraction for the case when components at each level share the same local behavioral domain. Sect. 5 extends the development to the more common case where components are developed within their own distinct local behavior domains. Sect. 6 illustrates the theoretical concepts using an example of a cooperative intersection collision avoidance system (CICAS). The concluding section summarizes the results in this paper and discusses directions for future work.

2. MATHEMATICAL PRELIMINARIES

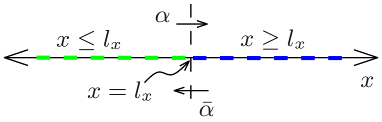
A *model* M is a mathematical description of a system using a *modeling formalism* $\mathcal{M}$, which is a collection of modeling primitives and syntactic rules for building models. Modeling formalisms typically used for CPS include transition systems, hybrid automata, signal-flow models, acausal equation-based models, and queuing networks. The semantics of a model M is defined by a set of *legal behaviors* from a given *behavior domain* B, where the behavior domain is a member of a given *class of behavior domains* $\mathcal{B}$. Behavior classes used to define semantics for CPS models include discrete traces, continuous trajectories and hybrid trajectories. For each behavior class, we assume there exists a syntax, called the *behavior formalism*, which can be used to precisely define behavior domains and individual behaviors. $[\![M]\!]^B$ denotes the set of legal behaviors for a given model M with semantics defined in a given behavior domain B.

Given two models M_0, M_1 with semantic interpretations in the same behavior domain B, model M_1 is called an *abstraction* of model M_0 if $[\![M_0]\!]^B \subseteq [\![M_1]\!]^B$. This is the standard definition of abstraction in the literature, using, for example, language or trace inclusion. We denote abstraction in this case by the notation $M_0 \sqsubseteq^B M_1$. We introduce the following notion of *behavior abstraction functions* as semantic mappings between different behavior domains, perhaps from two different behavior classes.

Definition 1 (Behavior Abstraction Functions) *Given two behavior classes $\mathcal{B}_0$ and $\mathcal{B}_1$ and behavior domains from each behavior class $B_0 \in \mathcal{B}_0$ and $B_1 \in \mathcal{B}_1$, a* behavior abstraction function *$\mathcal{A} : B_0 \to B_1$ maps each behavior in B_0 to a corresponding abstract behavior in B_1.*

Behavior abstraction functions are special cases of behavior relations from [19]. In particular, they are relations that are also functions, i.e., $R \subseteq B_0 \times B_1$ s.t. $(b_0, b_1) \in R$ and $(b_0, b_1') \in R$ only if $b_1 = b_1'$.

Example 1 Consider a behavior domain $B_0 = \mathbb{R}^{\mathbb{R}_+}$ as the set of all 1-d continuous trajectories starting at time 0. Let the variable name for the single dimension be x. Consider another behavior domain $B_1 = \Sigma^* \cup \Sigma^\omega$ defined as the set of all finite or infinite traces with event labels in $\Sigma = \{\alpha, \bar{\alpha}\}$. Consider a usual behavior abstraction technique frequently used in the literature — state-space partitioning, illustrated below. The continuous state-space $\mathbb{R}$ is partitioned in two halves $x \le l_x$ and $x \ge l_x$ at a boundary $x = l_x$ as follows.



The event corresponding to a continuous trajectory crossing the partition going from $x \le l_x$ to $x \ge l_x$ is associated with the label α and that from $x \ge l_x$ to $x \le l_x$ is associated with the label $\bar{\alpha}$.

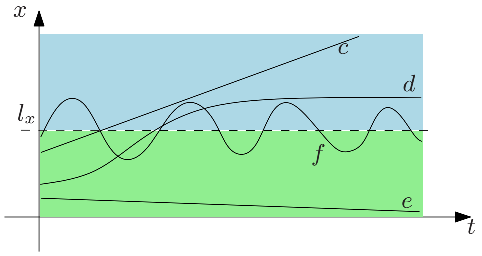
Consider $b_0 \in B_0$ and $b_1 \in B_1$ where $b_0 = x(t)$ for $t \in \mathbb{R}_+$ and $b_1 = \sigma_0\sigma_1 \cdots \sigma_N$, for $N \in \mathbb{N} \cup \{\infty\}$. In words, the abstraction function states that $\mathcal{A}(b_0) = b_1$ if (i) $\exists$ event times $t_i \in \mathbb{R}_+$, $i = 0, 1, \ldots, N$ that correspond to the continuous trajectory crossing the boundary in the right direction associated with the label σ_i (i.e., from "$\textsc{from}(\sigma_i)$" to "$\textsc{to}(\sigma_i)$" according to the following table) and (ii) there are no crossings between any consecutive event times t_i and t_{i+1}. Mathematically, these conditions can be written as

$$\begin{aligned}
&\forall t' \in [0, t_0), && x(t') \in \textsc{from}(\sigma_0),\\
&\forall t' \in [t_{i-1}, t_i), && x(t') \in \textsc{to}(\sigma_{i-1}) \cap \textsc{from}(\sigma_i),\\
&\forall t' \ge t_N, && x(t') \in \textsc{to}(\sigma_N),
\end{aligned}$$

σ	$\textsc{from}(\sigma)$	$\textsc{to}(\sigma)$
α	$x \le l_x$	$x \ge l_x$
$\bar{\alpha}$	$x \ge l_x$	$x \le l_x$

Otherwise, $\mathcal{A}(b_0)$ is an empty behavior ε.

With respect to the following picture, $\mathcal{A}(c) = \mathcal{A}(d) = \alpha$ and $\mathcal{A}(f)$ is the infinite string $\alpha\bar{\alpha}\alpha\bar{\alpha}\alpha\bar{\alpha}\cdots$. In contrast, $\mathcal{A}(e) = \varepsilon$ since e never crosses the boundary.



□

Behavior abstraction functions are typically used in the context of models and serve as mappings between the semantics of the two models defined in the two behavior domains under consideration. Therefore, they are usually inferred from relationships between models from given modeling formalisms and associated definitions of the relationships between model primitives and their semantic interpretations.

The set-valued extensions of behavior abstraction functions are defined in the usual way. For a given behavior abstraction function $\mathcal{A} : B_0 \to B_1$ and a set of behaviors $B_1' \subseteq B_1$, the set-valued inverse $\mathcal{A}^{-1}$ is defined as $\mathcal{A}^{-1}(B_1') = \{b_0 \mid \mathcal{A}(b_0) \in B_1'\}$.

Given behavior abstraction functions as semantic mappings, heterogeneous abstraction between two models is defined as follows.

Definition 2 (Heterogeneous Abstraction) *Given heterogeneous behavior classes $\mathcal{B}_0$, $\mathcal{B}_1$, suppose behavior domains $B_0 \in \mathcal{B}_0$ and $B_1 \in \mathcal{B}_1$ are used to define the semantics of models M_0 and M_1, respectively, and that there is an abstraction function $\mathcal{A} : B_0 \to B_1$. Model M_1 is a* heterogeneous abstraction *of M_0 through $\mathcal{A}$, written $M_0 \sqsubseteq^{\mathcal{A}} M_1$, if*

$$[\![M_0]\!]^{B_0} \subseteq \mathcal{A}^{-1}([\![M_1]\!]^{B_1}).$$

This definition asserts that for every behavior of model M_0 in B_0, the abstraction function $\mathcal{A}$ associates a corresponding abstract behavior of model M_1 in B_1.

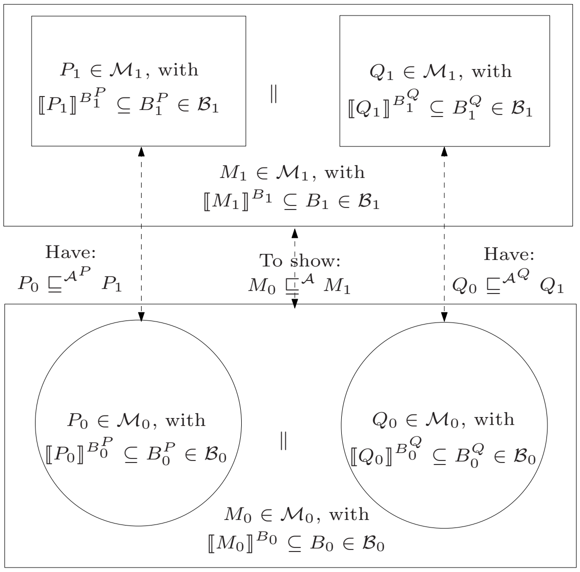

$P_1 \in \mathcal{M}_1$, with
$[\![P_1]\!]^{B_1^P} \subseteq B_1^P \in \mathcal{B}_1$
$\|$
$Q_1 \in \mathcal{M}_1$, with
$[\![Q_1]\!]^{B_1^Q} \subseteq B_1^Q \in \mathcal{B}_1$
$M_1 \in \mathcal{M}_1$, with
$[\![M_1]\!]^{B_1} \subseteq B_1 \in \mathcal{B}_1$
Have:
$P_0 \sqsubseteq^{\mathcal{A}^P} P_1$
To show:
$M_0 \sqsubseteq^{\mathcal{A}} M_1$
Have:
$Q_0 \sqsubseteq^{\mathcal{A}^Q} Q_1$
$P_0 \in \mathcal{M}_0$, with
$[\![P_0]\!]^{B_0^P} \subseteq B_0^P \in \mathcal{B}_0$
$\|$
$Q_0 \in \mathcal{M}_0$, with
$[\![Q_0]\!]^{B_0^Q} \subseteq B_0^Q \in \mathcal{B}_0$
$M_0 \in \mathcal{M}_0$, with
$[\![M_0]\!]^{B_0} \subseteq B_0 \in \mathcal{B}_0$

Figure 1: A schematic of compositional heterogeneous abstraction analysis.

Fig. 1 illustrates the compositional heterogeneous abstraction problem considered in this paper. For each of the two levels of abstraction, $i = 0, 1$, we assume there is a modeling formalism $\mathcal{M}_i$ and a behavior class $\mathcal{B}_i$. Component models $P_i, Q_i \in \mathcal{M}_i$ have their semantics defined in terms of *local* behavior domains $B_i^P, B_i^Q \in \mathcal{B}_i$. These local domains include only the variables relevant to the given component. Heterogeneous abstraction between the two models of each component is established via behavior abstraction functions $\mathcal{A}^P$ and $\mathcal{A}^Q$ that are mappings between the respective local behavior domains. To compose the two models to form the system models $M_i \in \mathcal{M}_i$, the local semantics are lifted to global behavior domains $B_i \in \mathcal{B}_i$ to include variables from both components. We seek conditions under which heterogeneous abstraction between component models in their local behavior domains implies heterogeneous abstraction between the composite system models in the global behavior domains.

3. LOCAL VS. GLOBAL SEMANTICS

We begin by defining the relationship between behaviors in local domains for component models and a global domain for the composition.

Definition 3 (Behavior Localization) *Given a behavior class $\mathcal{B}$ and two behavior domains $B, B' \in \mathcal{B}$, an onto function $\downarrow : B \to B'$ (i.e., every element of B' has at least one pre-image in B) is called a* (behavior) localization *of behavior domain B to behavior domain B'.*

Given a localization $\downarrow$ of B to B', for $b \in B$, we will let $b\downarrow$ denote $\downarrow(b)$. The set-valued extension of localization can be defined in the usual way. Next, we consider two common types of variable elimination as examples of behavior localization projections.

Example 2 (Event label removal) Let $L \subseteq \Sigma^*$ be a language over a global alphabet Σ. Let $A \subseteq \Sigma$ be a local alphabet relevant for a component. The localization due to natural projection of L onto the set of strings over A^*, written $L\downarrow_A := \{s\downarrow | s \in L\}$, where $s\downarrow$ is recursively defined as follows.

1. The empty string is projected onto itself, i.e. $\varepsilon\downarrow = \varepsilon$.
2. For any string $s \in \sigma^*$ and $a \in \Sigma$
   - $(s \circ a)\downarrow = (s)\downarrow \circ\ a \ldots$ if $a \in A$
   - $(s \circ a)\downarrow = (s)\downarrow \ldots$ if $a \notin A$ □

Example 3 (Continuous Variable Elimination) Consider a global behavior domain $B = (\mathbb{R}^2)^{\mathbb{R}_+}$ of 2-d continuous trajectories, with the variables along the two dimensions named x and y. Let a local behavior domain be $B' = (\mathbb{R})^{\mathbb{R}_+}$ with the variable name x. Let $\tilde{B} \subseteq B = \{[x(t)\ y(t)]^T |\ \forall t \in \mathbb{R}_+, x(t) \geq 0, y(t) \in [0, 1]\}$. Then the localization due to elimination of variable y can be written in terms of its existential quantification. $\tilde{B}\downarrow$ can be defined as the set $\{x(t) | \exists y(t) \text{ s.t. } \forall t \in \mathbb{R}_+, x(t) \geq 0, y(t) \in [0, 1]\}$. □

For $b' \in B'$ we will let $b'\uparrow$ denote the set-valued function $\uparrow : B' \to 2^B - \{\emptyset\}$ defined by $\uparrow(b') = \downarrow^{-1}(b') = \{b \in B | b\downarrow = b'\}$. We will call the function $\uparrow$ a *(behavior) globalization* of B' to B. Note that $b'\uparrow$ is always non-empty since the localization function $\downarrow$ is onto.

Behavior localization and globalization are generally inferred from relationships between models from given modeling formalisms and associated definitions of the relationships between model primitives and their semantic interpretations.

Note that in case of compositional heterogeneous analysis as depicted in Fig. 1, there are four different behavior localizations (or globalizations) – one for each component and one at each level of abstraction. We index these with subscripts $i = 0, 1$ for the two levels of abstraction and superscripts $j = 1, 2$ or $j = P, Q$ to distinguish between these wherever necessary.

Given behavior globalizations at the abstract and concrete levels of abstraction, we next define the globalization of a behavior abstraction function between the abstract and concrete local domains.

Definition 4 (Abstraction Globalization) *Given two behavior classes $\mathcal{B}_0$ and $\mathcal{B}_1$, behavior domains from each behavior class: $B_0, B_0' \in \mathcal{B}_0$ and $B_1, B_1' \in \mathcal{B}_1$, localizations $\downarrow_i$ of B_i to B_i' for $i = 1, 2$, and a behavior abstraction function $\mathcal{A}'$ of B_0' to B_1', a behavior abstraction function $\mathcal{A}$ of B_0 to B_1 is said to be a* globalization *of $\mathcal{A}'$ if*

$$\forall b_0 \in B_0 : \mathcal{A}'(b_0\downarrow_0) = \mathcal{A}(b_0)\downarrow_1. \tag{1}$$

In words, the definition of abstraction globalization states that given any global concrete behavior b_0, the abstraction of its localization $b_0\downarrow_0$ at the concrete level 0 through the local abstraction function $\mathcal{A}'$ should be the same as the localization at the abstract level 1 of its corresponding abstract behavior $\mathcal{A}(b_0)$. This concept is illustrated by the following diagram: $\mathcal{A}$ is a globalization of $\mathcal{A}'$ if the diagram commutes.

$$\begin{tikzcd}
B_1' & B_1 \arrow[l, "\downarrow_1"'] \\
B_0' \arrow[u, "\mathcal{A}'"] & B_0 \arrow[l, "\downarrow_0"] \arrow[u, "\mathcal{A}"']
\end{tikzcd}$$

We write $\mathcal{A} = \mathcal{A}'\Uparrow$ if $\mathcal{A}$ is a globalization of $\mathcal{A}'$. We call $\mathcal{A}'$ a *localization* of $\mathcal{A}$, written $\mathcal{A}' = \mathcal{A}\Downarrow$, iff $\mathcal{A} = \mathcal{A}'\Uparrow$.

Note that in case of compositional heterogeneous analysis as depicted in Fig. 1, there are two different abstraction localizations/globalizations – one for each component.

We note the following existence and uniqueness properties of localization/globalization of behavior abstraction functions.

- **Existence of globalization.** For a given local abstraction function $\mathcal{A}'$, it is always possible to construct a globalization $\mathcal{A}'\Uparrow$ s.t. the diagram commutes. This is due to the fact that both localizations $\downarrow_i$, $i = 0, 1$ are onto functions. Therefore, for any local behaviors b'_0 and $b'_1 = \mathcal{A}'(b'_0)$, $b'_0\uparrow_0$ and $b'_1\uparrow_1$ are non-empty. One can then associate every behavior $b_0 \in b'_0\uparrow_0$ with some behavior $b_1 \in b'_1\uparrow_1$, which results in a valid globalization of $\mathcal{A}'$.

- **Non-uniqueness of globalization.** For a given local abstraction function $\mathcal{A}'$, its globalization $\mathcal{A}'\Uparrow$ is not unique. For a b'_0 with $\mathcal{A}'(b'_0) = b'_1$ and $b'_1\uparrow_1 = \{b_1^0, b_1^1\}$, consider a global behavior $b_0 \in b'_0\uparrow_0$. Then $\mathcal{A}^0$ with $\mathcal{A}^0(b_0) = b_1^0$ and $\mathcal{A}^1$ with $\mathcal{A}^1(b_0) = b_1^1$ can both be globalizations of $\mathcal{A}'$. Since localization causes loss of information, its set-valued inverse provides some freedom for creating mappings at the global level; appropriate ones need to be chosen.

- **Non-existence of localization.** For a given global abstraction function $\mathcal{A}$, its localization $\mathcal{A}\Downarrow$ may not exist, i.e., the diagram may not commute for any $\mathcal{A}'$. Consider b_0^0, b_0^1 with $\mathcal{A}(b_0^0) = b_1^0$ and $\mathcal{A}(b_0^1) = b_1^1$, and $b_0^0\downarrow_0 = b_0^1\downarrow_0$, but $b_1^0\downarrow_0 \neq b_1^1\downarrow_0$. For such a case, there can be no $\mathcal{A}'$ s.t. $\mathcal{A}'\Uparrow = \mathcal{A}$.

- **Uniqueness of localization.** For a given global abstraction function $\mathcal{A}$, if $\mathcal{A}\Downarrow$ exists, it is unique. This is simply due to the diagram commuting. $\forall$ b_0, behaviors $b_0\downarrow_0$, $\mathcal{A}(b_0) =: b_1$, and $b_1\downarrow_1$ are unique. Therefore, for every given mapping $\mathcal{A}(b_0) = b_1$, there is a unique mapping $\mathcal{A}'(b_0\downarrow_0) = b_1\downarrow_1$.

- **Globalization and localization are not necessarily inverse operations.** From the uniqueness of localization and non-uniqueness of globalization, it is straightforward to show that
$$(\mathcal{A}'\Uparrow)\Downarrow = \mathcal{A}'; \tag{2}$$
but $(\mathcal{A}\Downarrow)\Uparrow$ may not be equal to $\mathcal{A}$.

Given the theoretical machinery developed in this section, in the next two sections we find conditions under which compositional heterogeneous abstraction w.r.t. Fig. 1 can be used.

4. HETEROGENEOUS ABSTRACTION IN GLOBAL BEHAVIOR DOMAINS

We start with a simple special-case scenario w.r.t. Fig. 1 in which the semantics of component models P_i and Q_i are defined in the same local behavior domain at each level of abstraction. In this case, the global behavior domains are the same as the local behavior domains , i.e., $B_i^P = B_i^Q = B_i \in \mathcal{B}_i$, $i = 0, 1$. For this special case, only one behavior abstraction function $\mathcal{A}$ is sufficient, as we can set $\mathcal{A}^P = \mathcal{A}^Q = \mathcal{A}$. In this case, we define the *semantic composition* of two component models as follows.

Definition 5 (Semantic Composition) *Given component models P and Q from the same modeling formalism $\mathcal{M}$ with semantics defined in behavior domain B, the composition $P||Q$ is a model in $\mathcal{M}$ s.t.*
$$\llbracket P||Q\rrbracket^B = \llbracket P\rrbracket^B \cap \llbracket Q\rrbracket^B. \tag{3}$$

This definition of composition as the intersection of behavior sets is consistent with the literature for composition using specific behavior domains [15, 18, 7]. For a given modeling formalism $\mathcal{M}$, syntactic techniques may exist for creating a composition, e.g., construction of product automata. We support all such procedures so long as (3) holds.

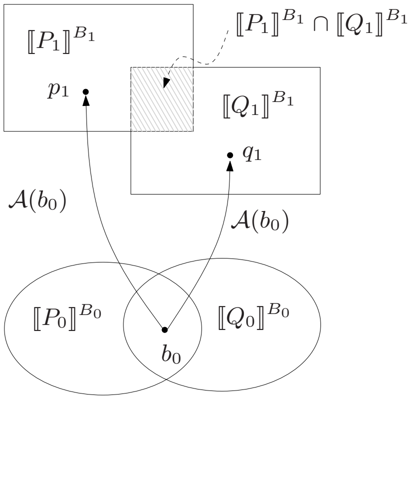
The following proposition gives conditions for compositional heterogeneous abstraction.

Proposition 1 *For each abstraction level $i = 0, 1$, given component models P_i, Q_i with the semantics of each model interpreted over a behavior domain B_i, and a behavior abstraction function $\mathcal{A} : B_0 \to B_1$, if $P_0 \sqsubseteq^{\mathcal{A}} P_1$ and $Q_0 \sqsubseteq^{\mathcal{A}} Q_1$, then*
$$P_0||Q_0 \sqsubseteq^{\mathcal{A}} P_1||Q_1.$$

PROOF. From $P_0 \sqsubseteq^{\mathcal{A}} P_1$ and $Q_0 \sqsubseteq^{\mathcal{A}} Q_1$, we have $\llbracket P_0\rrbracket^{B_0} \subseteq \mathcal{A}^{-1}(\llbracket P_1\rrbracket^{B_1})$ and $\llbracket Q_0\rrbracket^{B_0} \subseteq \mathcal{A}^{-1}(\llbracket Q_1\rrbracket^{B_1})$. Therefore,
$$\begin{aligned}
\llbracket P_0||Q_0\rrbracket^{B_0} &= \llbracket P_0\rrbracket^{B_0} \cap \llbracket Q_0\rrbracket^{B_0} \\
&\subseteq \mathcal{A}^{-1}(\llbracket P_1\rrbracket^{B_1}) \cap \mathcal{A}^{-1}(\llbracket Q_1\rrbracket^{B_1}) \\
&= \mathcal{A}^{-1}(\llbracket P_1\rrbracket^{B_1} \cap \llbracket Q_1\rrbracket^{B_1}) \\
&= \mathcal{A}^{-1}(\llbracket P_1||Q_1\rrbracket^{B_1}).
\end{aligned}$$
■

This proposition states that with global semantics, composition of abstractions is the abstraction of the composition.

Remark 6 (Insufficiency of Behavior Relations) We note that arbitrary behavior relations from [19] that are not functions are not sufficient in even this simple case of compositional heterogeneous abstraction. If a behavior relation $\mathcal{A}$ is not a function, it is possible to have a behavior $b_0 \in \llbracket P_0\rrbracket^{B_0} \cap \llbracket Q_0\rrbracket^{B_0}$ with $(b_0, p_1) \in \mathcal{A}$, $(b_0, q_1) \in \mathcal{A}$, s.t. $p_1 \in \llbracket P_1\rrbracket^{B_1}\backslash\llbracket Q_1\rrbracket^{B_1}$ and $q_1 \in \llbracket Q_1\rrbracket^{B_1}\backslash\llbracket P_1\rrbracket^{B_1}$ but $\nexists$ $b_1 \in \llbracket P_1\rrbracket^{B_1} \cap \llbracket Q_1\rrbracket^{B_1}$ with $(b_0, b_1) \in \mathcal{A}$, as shown in the following Venn diagram.


$\llbracket P_1\rrbracket^{B_1}$
$\llbracket P_1\rrbracket^{B_1} \cap \llbracket Q_1\rrbracket^{B_1}$
p_1
$\llbracket Q_1\rrbracket^{B_1}$
q_1
$\mathcal{A}(b_0)$
$\mathcal{A}(b_0)$
$\llbracket P_0\rrbracket^{B_0}$
$\llbracket Q_0\rrbracket^{B_0}$
b_0

For this b_0, we have $b_0 \in \mathcal{A}^{-1}(\llbracket P_1 \rrbracket^{B_1}) \cap \mathcal{A}^{-1}(\llbracket Q_1 \rrbracket^{B_1})$ but $b_0 \notin \mathcal{A}^{-1}(\llbracket P_1 \rrbracket^{B_1} \cap \llbracket Q_1 \rrbracket^{B_1})$ and therefore the above proof does not hold. The arbitrary mappings that are the source of these counterexamples – those that allow one concrete behavior to be associated with more than one abstract behaviors – are perhaps not necessary in practice. The restriction from behavior relations to functions disallows the possibility of having several abstract behaviors correspond to a single concrete behavior, while still allowing several concrete behaviors to be mapped to a single abstract behavior. □

In the next section, we consider the general case where the local semantics of the two components are defined in terms of distinct behavior domains.

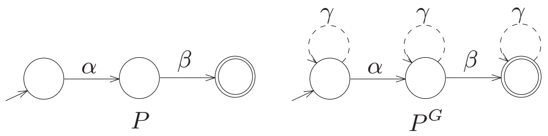
5. HETEROGENEOUS ABSTRACTION IN LOCAL BEHAVIOR DOMAINS

We now consider a more general scenario w.r.t. Fig. 1 in which the component models P_i and Q_i have different local behavior domains B_i^P and B_i^Q in behavior class $\mathcal{B}_i$, for levels of abstraction $i = 0, 1$. In this case, we need to lift the local semantics of the components to common global behavior domains before we can compose them.

Definition 7 (Model Globalization) *Given a global behavior domain B, a model P with its local behavior domain B', and a behavior localization function $\downarrow : B \to B'$, the* (model) globalization *of P is a model P^G s.t. $\llbracket P^G \rrbracket^B = \llbracket P \rrbracket^{B'}\uparrow$.*

For a given modeling formalism $\mathcal{M}$, syntactic approaches for globalization may exist, e.g., addition of self loops for newly added event labels for discrete transition systems, or addition of state variables with unconstrained dynamics for continuous dynamic systems. We support all such procedures that lead to models with the set of behaviors $\llbracket P \rrbracket^{B'}\uparrow$.

Example 4 Consider transition system model P as shown below. The local alphabet is $\Sigma^P = \{\alpha, \beta\}$ and the local behavior domain $B^P = \Sigma^*$. Let the global alphabet be $\Sigma = \{\alpha, \beta, \gamma\}$ and the global behavior domain $B = \Sigma^*$. Let the projection function $\downarrow : B \to B^P$ be defined as per Ex. 2.



The semantic interpretation of P in the local behavior domain B^P is the set $\{\alpha\beta\}$. The globalization of the set $\{\alpha\beta\}$ in the global behavior domain B is the set $\{\gamma^*\alpha\gamma^*\beta\gamma^*\}$. Note that the syntactic globalization procedure by introducing self loops for the new label γ results in a model P^G shown above, and $\llbracket P^G \rrbracket^B = \llbracket P \rrbracket^{B^P}\uparrow$. □

The following lemma states that heterogeneous abstraction between model globalizations via a global abstraction function is equivalent to heterogeneous abstraction between original models via the localization of the global abstraction function.

Lemma 1 *For abstraction levels $i = 0, 1$, given component models P_i with local behavior domains B_i', behavior localization functions $\downarrow_i : B_i \to B_i'$, let their corresponding globalized models be P_i^G with global behavior domains B_i. If $\mathcal{A} : B_0 \to B_1$ is a global behavior abstraction function and $\mathcal{A}' : B_0' \to B_1'$ is a localization of $\mathcal{A}$, then*

$$P_0^G \sqsubseteq^{\mathcal{A}} P_1^G \Leftrightarrow P_0 \sqsubseteq^{\mathcal{A}'} P_1.$$

Proof. From the definition of model globalization, we have

$$b_i \in \llbracket P_i^G \rrbracket^{B_i} \Leftrightarrow b_i{\downarrow_i} \in \llbracket P_i \rrbracket^{B_i'} \tag{4}$$

and

$$b_i' \in \llbracket P_i \rrbracket^{B_i'} \Leftrightarrow b_i'{\uparrow_i} \subseteq \llbracket P_i^G \rrbracket^{B_i}. \tag{5}$$

Case I: $P_0^G \sqsubseteq^{\mathcal{A}} P_1^G \Rightarrow P_0 \sqsubseteq^{\mathcal{A}'} P_1$
For any given $b_0 \in \llbracket P_0^G \rrbracket^{B_0}$, let $b_1 := \mathcal{A}(b_0)$. From $P_0^G \sqsubseteq^{\mathcal{A}} P_1^G$, we have $b_1 \in \llbracket P_1^G \rrbracket^{B_1}$. From (1), $\mathcal{A}'(b_0' := b_0{\downarrow_0}) = b_1{\downarrow_1}$. Hence, from (4), we have that $\forall b_0' \in \llbracket P_0 \rrbracket^{B_0'}, \mathcal{A}'(b_0') \in \llbracket P_1 \rrbracket^{B_1'}$, which implies $\llbracket P_0 \rrbracket^{B_0'} \subseteq \mathcal{A}'^{-1}(\llbracket P_1 \rrbracket^{B_1'})$, i.e., $P_0 \sqsubseteq^{\mathcal{A}'} P_1$.

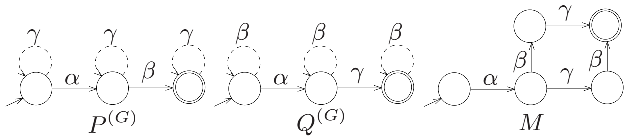
Case II: $P_0^G \sqsubseteq^{\mathcal{A}} P_1^G \Leftarrow P_0 \sqsubseteq^{\mathcal{A}'} P_1$
From $P_0 \sqsubseteq^{\mathcal{A}'} P_1$, we have $b_0' \in \llbracket P_0 \rrbracket^{B_0'} \Rightarrow \mathcal{A}'(b_0') =: b_1' \in \llbracket P_1 \rrbracket^{B_1'}$. From Def. 4 and (5), for any $b_0' \in \llbracket P_0 \rrbracket^{B_0'}$, $b_0 \in b_0'{\uparrow_0} \subseteq \llbracket P_0^G \rrbracket^{B_0} \Rightarrow \mathcal{A}(b_0) =: b_1 \in b_1'{\uparrow_1} \subseteq \llbracket P_1^G \rrbracket^{B_1}$. Therefore, $\llbracket P_0^G \rrbracket^{B_0} \subseteq \mathcal{A}^{-1}(\llbracket P_1^G \rrbracket^{B_1})$, i.e., $P_0^G \sqsubseteq^{\mathcal{A}} P_1^G$. ■

In terms of Fig. 1, the implication of Lemma 1 is the following. When the abstract and concrete models of a component are considered in isolation, it does not matter whether one does the heterogeneous abstraction analysis in the global domains or the local domains.

We now use the result from Lemma 1 in a compositional setting when the component models are composed to form a system model. The following definition generalizes the notion of semantic composition from Def. 5.

Definition 8 (Globalized Semantic Composition) *Given a global behavior domain B, component models P and Q with their corresponding local behavior domains B^P and B^Q, and behavior localizations $\downarrow^P : B \to B^P$ and $\downarrow^Q : B \to B^Q$, the* globalized semantic composition *of P and Q in the global behavior domain B, denoted by $P||^G Q$ is the semantic composition of models P^G and Q^G, which are the globalizations of P and Q respectively, i.e., $P||^G Q = P^G||Q^G$.*

Example 5 Consider two transition system models P and Q as shown below (without the dashed self loops).



The local alphabets of P and Q are $\Sigma^P = \{\alpha, \beta\}$ and $\Sigma^P = \{\alpha, \gamma\}$, and corresponding behavior domains $B^P = \Sigma^{P^*}$ and $B^Q = \Sigma^{Q^*}$ respectively. Let the global alphabet be $\Sigma = \{\alpha, \beta, \gamma\}$, and the global behavior domain $B = \Sigma^*$. Let $\downarrow^P$ and $\downarrow^Q$ be the natural projections as defined in Ex. 2.

The local sets of behaviors for the two components are $\llbracket P \rrbracket^{B^P} = \{\alpha\beta\}$ and $\llbracket P \rrbracket^{B^P} = \{\alpha\gamma\}$. The semantic globalizations of the two component models yield $\llbracket P \rrbracket^{B^P}\uparrow^P = \{\gamma^*\alpha\gamma^*\beta\gamma^*\}$ and $\llbracket Q \rrbracket^{B^Q}\uparrow^Q = \{\beta^*\alpha\beta^*\gamma\beta^*\}$. The composition $M := P||^G Q$ has corresponding sets of behaviors given by $\llbracket M \rrbracket^B = \llbracket P \rrbracket^{B^P}\uparrow^P \cap \llbracket Q \rrbracket^{B^Q}\uparrow^Q = \{\alpha\beta\gamma, \alpha\gamma\beta\}$.

Note that the syntactic globalization procedure of introducing self loops yields models P^G and Q^G, whose syntactic composition results in a model M as shown above, s.t. $M = P||^G Q$. □

Example 6 Let $B^j := \mathbb{R}^{\mathbb{R}_+}$ be the sets of 1-d continuous trajectories with variable names x_j, $j = p, q$ respectively. Let two components P given by $\dot{x_p} \in [1,2]$ and Q given by $\dot{x_q} \in [3,5]$ respectively have their semantics defined in domains B^j, $j = p, q$. Let $B := (\mathbb{R}^2)^{\mathbb{R}_+}$ be the system behavior domain of 2-d continuous trajectories with variable names along the two dimensions x_p and x_q. The globalizations of P and Q add the missing dimension and leave it unconstrained. Therefore, P^G and Q^G can be obtained as

$$P^G \equiv \begin{bmatrix} \dot{x_p} \\ \dot{x_q} \end{bmatrix} \in \begin{bmatrix} [1,2] \\ (-\infty,\infty) \end{bmatrix}, Q^G \equiv \begin{bmatrix} \dot{x_p} \\ \dot{x_q} \end{bmatrix} \in \begin{bmatrix} (-\infty,\infty) \\ [3,5] \end{bmatrix}.$$

Their composition is $P||^G Q \equiv \begin{bmatrix} \dot{x_p} \\ \dot{x_q} \end{bmatrix} \in \begin{bmatrix} [1,2] \\ [3,5] \end{bmatrix}$. □

For the following discussion, we let models M_i, with the global behavior domains B_i, be the globalized compositions $P_i||^G Q_i$ of component models P_i and Q_i with their local behavior domains B_i^P and B_i^Q, for levels of abstraction $i = 0, 1$ as depicted in Fig. 1. We consider two scenarios in which the source of the abstraction is at the system and component levels respectively.

5.1 Centralized development

First, we consider the case where an abstraction function $\mathcal{A} : B_0 \to B_1$ between the global behavior domains B_0 and B_1 is given. For this case, the following proposition shows that the problem of establishing $M_0 \sqsubseteq^{\mathcal{A}} M_1$ can be reduced to solving two smaller problems $P_0 \sqsubseteq^{\mathcal{A}\Downarrow^P} P_1$ and $Q_0 \sqsubseteq^{\mathcal{A}\Downarrow^Q} Q_1$.

Proposition 2 *For abstraction levels $i = 0, 1$, given component models P_i and Q_i with corresponding local behavior domains B_i^P and B_i^Q, let their globalized semantic compositions be $M_i := P_i||^G Q_i$ in global behavior domains B_i with behavior localizations $\downarrow_i^j : B_i \to B_i^j$, where $j = P, Q$, and a global behavior abstraction function $\mathcal{A} : B_0 \to B_1$. If localizations $\mathcal{A}\Downarrow^P$ and $\mathcal{A}\Downarrow^Q$ of $\mathcal{A}$ exist and $P_0 \sqsubseteq^{\mathcal{A}\Downarrow^P} P_1$ and $Q_0 \sqsubseteq^{\mathcal{A}\Downarrow^Q} Q_1$, then $M_0 \sqsubseteq^{\mathcal{A}} M_1$.*

Proof. From $P_0 \sqsubseteq^{\mathcal{A}\Downarrow^P} P_1$ and $Q_0 \sqsubseteq^{\mathcal{A}\Downarrow^Q} Q_1$, we know from Lemma 1 that $P_0^G \sqsubseteq^{\mathcal{A}} P_1^G$ and $Q_0^G \sqsubseteq^{\mathcal{A}} Q_1^G$, i.e., that $[\![P_0^G]\!]^{B_0} \subseteq \mathcal{A}^{-1}([\![P_1^G]\!]^{B_1})$ and $[\![Q_0^G]\!]^{B_0} \subseteq \mathcal{A}^{-1}([\![Q_1^G]\!]^{B_1})$. We have,

$$\begin{aligned} [\![P_0||^G Q_0]\!]^{B_0} &= [\![P_0^G]\!]^{B_0} \cap [\![Q_0^G]\!]^{B_0} \\ &\subseteq \mathcal{A}^{-1}([\![P_1^G]\!]^{B_1}) \cap \mathcal{A}^{-1}([\![Q_1^G]\!]^{B_1}) \\ &= \mathcal{A}^{-1}([\![P_1^G]\!]^{B_1} \cap [\![Q_1^G]\!]^{B_1}) \\ &= \mathcal{A}^{-1}([\![P_1||^G Q_1]\!]^{B_1}). \end{aligned}$$

■

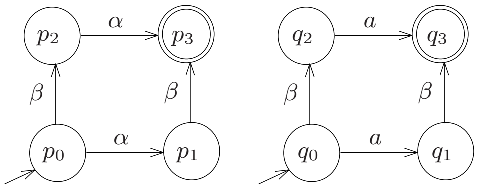
Prop. 2 states that we can establish $M_0 \sqsubseteq^{\mathcal{A}} M_1$ in the global behavior domains by establishing $P_0 \sqsubseteq^{\mathcal{A}\Downarrow^P} P_1$ and $Q_0 \sqsubseteq^{\mathcal{A}\Downarrow^Q} Q_1$ in the local behavior domains of the two components.

Example 7 Consider component models

$$P_0 \equiv \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} \in \begin{bmatrix} [2,4] \\ [1,2] \end{bmatrix}, \begin{bmatrix} x \\ y \end{bmatrix}(0) \in \begin{bmatrix} [0,l_x) \\ [0,l_y) \end{bmatrix} \text{ and}$$

$$Q_0 \equiv \begin{bmatrix} \dot{x} \\ \dot{z} \end{bmatrix} \in \begin{bmatrix} [3,5] \\ [1,2] \end{bmatrix}, \begin{bmatrix} x \\ z \end{bmatrix}(0) \in \begin{bmatrix} [0,l_x) \\ [0,l_z) \end{bmatrix}.$$

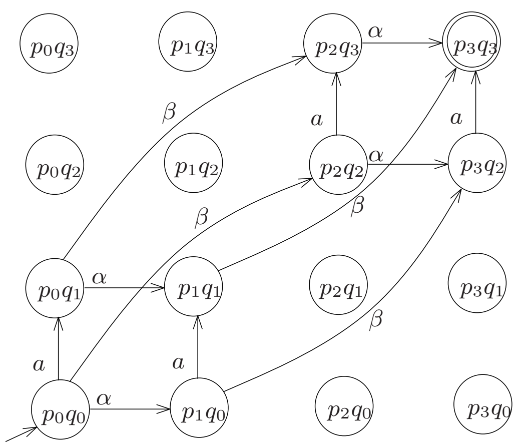
Let P_1 and Q_1 be as follows.



P_1 Q_1

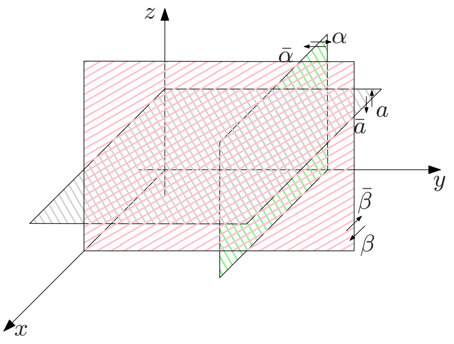
The compositions are $M_0 := P_0||^G Q_0$ given by

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \in \begin{bmatrix} [3,4] \\ [1,2] \\ [1,2] \end{bmatrix}, \begin{bmatrix} x \\ y \\ z \end{bmatrix}(0) \in \begin{bmatrix} [0,l_x) \\ [0,l_y) \\ [0,l_z) \end{bmatrix} \text{ and}$$



$M_1 := P_1||^G Q_1$.

Global behavior domain $B_0 := (\mathbb{R}^3)^{\mathbb{R}_+}$ for M_0 is the set of 3-d trajectories with variable names x, y, z; while global behavior domain $B_1 := \Sigma^*$ for M_1 is the set of all finite traces over the alphabet $\Sigma = \{\alpha, \bar{\alpha}, \beta, \bar{\beta}, a, \bar{a}\}$. Let the behavior abstraction function $\mathcal{A} : B_0 \to B_1$ be defined by partitioning the continuous state space as follows.



Given $b_0 = [x(t)\ y(t)\ z(t)]^T =: \bar{x}(t)$, $t \in \mathbb{R}_+$ and $b_1 = \sigma_0\sigma_1\cdots$, $\mathcal{A}^j(b_0) = b_1$ if $\exists$ times $t_i \in \mathbb{R}_+$ s.t.

$$\begin{array}{ll} \forall t' \in [0, t_0), & \bar{x}(t) \in \text{FROM}(\sigma_0), \\ \forall t' \in [t_{i-1}, t_i), & \bar{x}(t) \in \text{TO}(\sigma_{i-1}) \cap \text{FROM}(\sigma_i), \\ \forall t' \geq t_N & \bar{x}(t) \in \text{TO}(\sigma_N) \end{array}$$

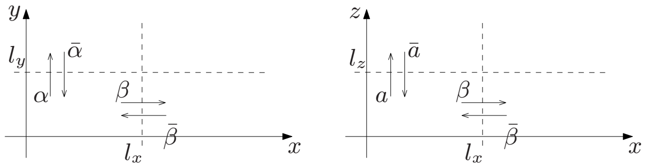
where $i = 1, \ldots, N$ for some $N \in \mathbb{N}$ and $\text{FROM}(\cdot)$ and $\text{TO}(\cdot)$ are given in the following table.

σ	$\text{FROM}(\sigma)$	$\text{TO}(\sigma)$
a	$z \leq l_z,\ x, y \in \mathbb{R}$	$z \geq l_z,\ x, y \in \mathbb{R}$
$\bar{a}$	$z \geq l_z,\ x, y \in \mathbb{R}$	$z \leq l_z,\ x, y \in \mathbb{R}$
α	$y \leq l_y,\ x, z \in \mathbb{R}$	$y \geq l_y,\ x, z \in \mathbb{R}$
$\bar{\alpha}$	$y \geq l_y,\ x, z \in \mathbb{R}$	$y \leq l_y,\ x, z \in \mathbb{R}$
β	$x \leq l_x,\ y, z \in \mathbb{R}$	$x \geq l_x,\ y, z \in \mathbb{R}$
$\bar{\beta}$	$x \geq l_x,\ y, z \in \mathbb{R}$	$x \leq l_x,\ y, z \in \mathbb{R}$

Otherwise, $\mathcal{A}(b_0) = \varepsilon$.

The problem of establishing $M_0 \sqsubseteq^{\mathcal{A}} M_1$ for above $\mathcal{A}$ can be reduced to two smaller problems $P_0 \sqsubseteq^{\mathcal{A}\Downarrow^P} P_1$ and $Q_0 \sqsubseteq^{\mathcal{A}\Downarrow^Q} Q_1$ as follows.

Local behavior domains for the two models of component P are $B_0^P = (\mathbb{R}^2)^{\mathbb{R}_+}$ with variable names for the dimensions x and y; and $B_1^P = \Sigma^{P*}$ with $\Sigma^P = \{\alpha, \bar{\alpha}, \beta, \bar{\beta}\}$. Similarly, in case of the two models of component Q, $B_0^Q = (\mathbb{R}^2)^{\mathbb{R}_+}$ with variable names for the dimensions x and z; and $B_1^Q = \Sigma^{Q*}$ with $\Sigma^Q = \{a, \bar{a}, \beta, \bar{\beta}\}$. Let behavior localization functions for the two components at the two levels of abstractions be variable elimination and natural projection as in Ex. 2 and 3. We get the behavior abstraction function localizations $\mathcal{A}^P : B_0^P \to B_1^P$ and $\mathcal{A}^Q : B_0^Q \to B_1^Q$, where $\mathcal{A}^P = \mathcal{A}\Downarrow^P$ and $\mathcal{A}^Q = \mathcal{A}\Downarrow^Q$ are as follows.



Let $\bar{x}^P = [xy]^T$ and $\bar{x}^Q = [xz]^T$. Given $b_0^P = \bar{x}^P(t)$, $b_0^Q = \bar{x}^Q(t)$ for $t \in \mathbb{R}_+$ and $b_1^j = \sigma_0^j\sigma_1^j\cdots$, $\mathcal{A}^j$, $j = P, Q$, are defined as $\mathcal{A}^j(b_0^j) = b_1^j$ if $\exists$ times $t_i^j \in \mathbb{R}_+$ s.t.

$$\begin{array}{ll} \forall t' \in [0, t_0^j), & \bar{x}^j(t') \in \text{FROM}^j(\sigma_0^j), \\ \forall t' \in [t_{i-1}^j, t_i^j), & \bar{x}^j(t') \in \text{TO}^j(\sigma_{i-1}^j) \cap \text{FROM}^j(\sigma_i^j), \\ \forall t' \geq t_N^j, & \bar{x}^j(t') \in \text{TO}^j(\sigma_N^j), \end{array}$$

where $i = 1, \ldots, N$ for some $N \in \mathbb{N}$ and $\text{FROM}^j(\cdot)$ and $\text{TO}^j(\cdot)$ are given in the following tables.

σ	$\text{FROM}^P(\sigma)$	$\text{TO}^P(\sigma)$
α	$y \leq l_y, x \in \mathbb{R}$	$y \geq l_y, x \in \mathbb{R}$
$\bar{\alpha}$	$y \geq l_y, x \in \mathbb{R}$	$y \leq l_y, x \in \mathbb{R}$
β	$x \leq l_x, y \in \mathbb{R}$	$x \geq l_x, y \in \mathbb{R}$
$\bar{\beta}$	$x \geq l_x, y \in \mathbb{R}$	$x \leq l_x, y \in \mathbb{R}$

σ	$\text{FROM}^Q(\sigma)$	$\text{TO}^Q(\sigma)$
a	$z \leq l_z, x \in \mathbb{R}$	$z \geq l_z, x \in \mathbb{R}$
$\bar{a}$	$z \geq l_z, x \in \mathbb{R}$	$z \leq l_z, x \in \mathbb{R}$
β	$x \leq l_x, z \in \mathbb{R}$	$x \geq l_x, z \in \mathbb{R}$
$\bar{\beta}$	$x \geq l_x, z \in \mathbb{R}$	$x \leq l_x, z \in \mathbb{R}$

Otherwise, $\mathcal{A}^j(b_0^j) = \varepsilon$.

From the initial conditions and the monotonicity of the dynamics of P_0 (resp. Q_0), we can see that every behavior of the concrete model crosses the $x = l_x$ and $y = l_y$ (resp. $z = l_z$) boundaries in either order and have corresponding behaviors $\alpha\beta$ (resp. $a\beta$) or $\beta\alpha$ (resp. βa) at the discrete level that they map to. Therefore $P_0 \sqsubseteq^{\mathcal{A}^P} P_1$ and $Q_0 \sqsubseteq^{\mathcal{A}^Q} Q_1$. Using Prop. 2, $M_0 \sqsubseteq^{\mathcal{A}} M_1$.

Here, analyzing P_is and Q_is is much easier than analyzing M_is directly. In general, the extent of savings achieved by doing the heterogeneous abstraction analysis compositionally is depends on how much smaller the local behavior domains are compared to the global ones. □

5.2 Decentralized development

Now, we consider the case where the abstraction functions $\mathcal{A}^P : B_0^P \to B_1^P$ and $\mathcal{A}^Q : B_0^Q \to B_0^Q$ between the local behavior domains B_i^P and B_i^Q are given and heterogeneous abstractions of component models $P_0 \sqsubseteq^{\mathcal{A}^P} P_1$ and $Q_0 \sqsubseteq^{\mathcal{A}^Q} Q_1$ are established independently. This is the more common situation in practice, particularly for distributed development. In this case, the following proposition states that if the globalizations of abstraction functions $\mathcal{A}^P\Uparrow^P$ and $\mathcal{A}^Q\Uparrow^Q$ are defined consistently, the heterogeneous abstraction results for the components carry over to their compositions.

Proposition 3 *For abstraction levels $i = 0, 1$, given component models P_i and Q_i with local behavior domains B_i^P and B_i^Q, let their compositions be $P_i||^G Q_i$ in global behavior domains B_i and local behavior abstraction functions be $\mathcal{A}^P : B_0^P \to B_1^P$ and $\mathcal{A}^Q : B_0^Q \to B_1^Q$ s.t. $P_0 \sqsubseteq^{\mathcal{A}^P} P_1$ and $Q_0 \sqsubseteq^{\mathcal{A}^Q} Q_1$. If $\mathcal{A}^P\Uparrow^P = \mathcal{A}^Q\Uparrow^Q =: \mathcal{A}$, then $P_0||^G Q_0 \sqsubseteq^{\mathcal{A}} P_1||^G Q_1$.*

PROOF. The result follows due to $(\mathcal{A}^P\Uparrow^P)\Downarrow^P = \mathcal{A}^P$ and $(\mathcal{A}^Q\Uparrow^Q)\Downarrow^Q = \mathcal{A}^Q$ from (2) and Prop. 2. ■

Prop. 3 states that the heterogeneous abstraction results for component models P_i and Q_i via possibly very different abstraction functions $\mathcal{A}^P$ and $\mathcal{A}^Q$ follow over to the system heterogeneous abstraction so long as $\mathcal{A}^P$ and $\mathcal{A}^Q$ are consistent, i.e., that it is possible to find globalizations$\mathcal{A}^P\Uparrow^P$ and $\mathcal{A}^Q\Uparrow^Q$ that are in agreement with each other. Note from the non-uniqueness of globalization of abstraction functions that there is some design freedom while constructing the semantic mappings at the global behavior domains for the two components such that they agree.

We note the following conditions for agreement of the globalizations of the local abstraction functions from the two components.

- **Disjoint behavior domains.** If the local behavior domains are disjoint (no common variables), the abstraction functions are disjoint. Therefore, when globalized, they are not mutually restrictive and it is always possible to construct globalizations that agree.

- **Agreement in intersection.** For non-disjoint local behavior domains, it is necessary for globalization agreement that the local abstraction functions agree on the "intersection" of the two behavior domains, say $B_i^{\cap}$, i.e., along the variables common to the two components. If localizations $\mathcal{A}^P\Downarrow^{\cap} : B_0^{\cap} \to B_1^{\cap}$ and $\mathcal{A}^Q\Downarrow^{\cap} : B_0^{\cap} \to B_1^{\cap}$ of $\mathcal{A}^P$ and $\mathcal{A}^Q$ agree, it is always possible to construct globalizations of $\mathcal{A}^P$ and $\mathcal{A}^Q$ that agree due to the fact that variables not common to the two components are not mutually constraining.

We illustrate distributed compositional heterogeneous abstraction analysis in the following section.

6. EXAMPLE

Consider a cooperative intersection collision avoidance system for stop-sign assist (CICAS-SSA) [1] from Fig. 2, which depicts a *subject vehicle* (SV) waiting at a stop-sign-controlled intersection to cross through traffic on a major road. The objective is to augment human judgment of the SV driver about whether a given gap in oncoming traffic is safe by sensing the positions and/or velocities of the oncoming vehicles and doing some computations based on vehicle dynamics, intersection geometry and speed limits. The oncoming vehicle is called the *principal other vehicle* (POV). The SV modeled is allowed to (but doesn't have to) enter the intersection only if the POV is far enough away from the intersection to allow the SV to pass completely through the intersection before the POV has arrived at the intersection. Otherwise the SV has to remain stopped.

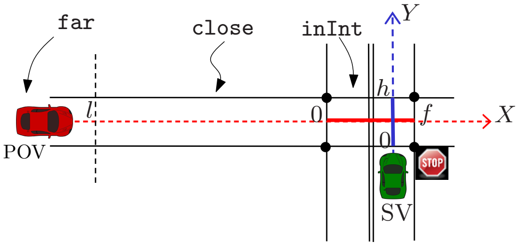


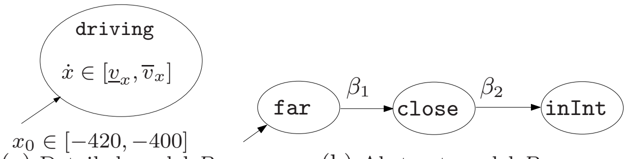
Figure 2: A simple variant of CICAS-SSA.

We model this system at two levels of abstraction. At the detailed level we model the two vehicles using their hybrid dynamics, while at the abstract level, we model simple discrete dynamics. The discrete dynamics can be used to verify protocols such as "if SV hasn't entered in the intersection already, then it doesn't do so once it sees POV get close to the intersection." This protocol verification can then be used in conjunction with other information to construct a hierarchical heterogeneous verification of the system to guarantee safety specifications such as "the two cars are never in the intersection at the same time" as demonstrated in [19]. In this paper, we consider the problem of establishing in a distributed compositional manner that the discrete model of the system used in the protocol verification is a heterogeneous abstraction of the underlying hybrid model of the system. In this distributed compositional heterogeneous abstraction, we use two different kinds of abstraction functions for two components – one using state-space partitioning and another by retaining the discrete transition graph by projecting away all the continuous dynamics.

The POV drives along the major road with its position x increasing over time, at a velocity between a minimum and a maximum, both limits being positive (i.e., it cannot drive in reverse on the highway). We assume that the SV is able to sense the position of the POV to make its decision. Once in the intersection, SV keeps driving with a velocity in the range $[\underline{v}_y, \overline{v}_y]$, both limits assumed positive, eventually clearing the intersection at $y = h$.

6.1 Heterogeneous abstraction for POV

The hybrid model P_0 and the discrete model P_1 for the POV are shown in Fig. 3.



(a) Detailed model P_0 (b) Abstract model P_1

Figure 3: Hybrid and discrete POV models.

The local behavior domains are B_0^{POV}: 1-d hybrid traces, i.e., evolution of the hybrid state $h^{POV} := (l^{POV}, x)$ over time, with $l^{POV} \in \mathcal{L}^{POV} := \{\texttt{driving}\}$ and $x \in \mathbb{R}$; and $B_1^{POV} := \Sigma^{POV*}$ for set of event labels $\Sigma^{POV} = \{\beta_1, \beta_2\}$. The model semantics are $[\![P_0]\!]^{B_0}$: the set of all hybrid traces with the discrete location **driving** and x that starts in the initial condition set $[-420, -400]$ and evolves along any arbitrary derivative in the range $[\underline{v}_x, \overline{v}_x]$, and $[\![P_1]\!]^{B_1}$: the singleton set $\{\beta_1\beta_2\}$.

A behavior abstraction function $\mathcal{A}^{POV} : B_0^{POV} \to B_1^{POV}$ constructed by partitioning the continuous dimension x at boundaries $x = l$ and $x = 0$ is written mathematically as follows. Given $b_0^{POV} = h^{POV}(t) \in B_0^{POV}$ and $b_1^{POV} = \sigma_0\sigma_1 \cdots \in B_1^{POV}$, $\mathcal{A}^{POV}(b_0^{POV}) = b_1^{POV}$ iff $\exists$ times $t_i \in \mathbb{R}_+$ s.t. $\forall t' \in [0, t_0)$, $x(t') \in \text{FROM}(\sigma_0)$; $\forall t' \in [t_{i-1}, t_i)$, $x(t') \in \text{TO}(\sigma_{i-1}) \cap \text{FROM}(\sigma_i)$ for $i = 1, \ldots, N$ for some $N \in \mathbb{N}$; and $\forall t' \geq t_N$, $x(t') \in \text{TO}(\sigma_N)$, where $\text{FROM}(\cdot)$ and $\text{TO}(\cdot)$ are given in the following table.

σ	$\text{FROM}(\sigma)$	$\text{TO}(\sigma)$
β_1	$x \leq l$	$x \in [l, 0]$
β_2	$x \in [l, 0]$	$x \geq 0$

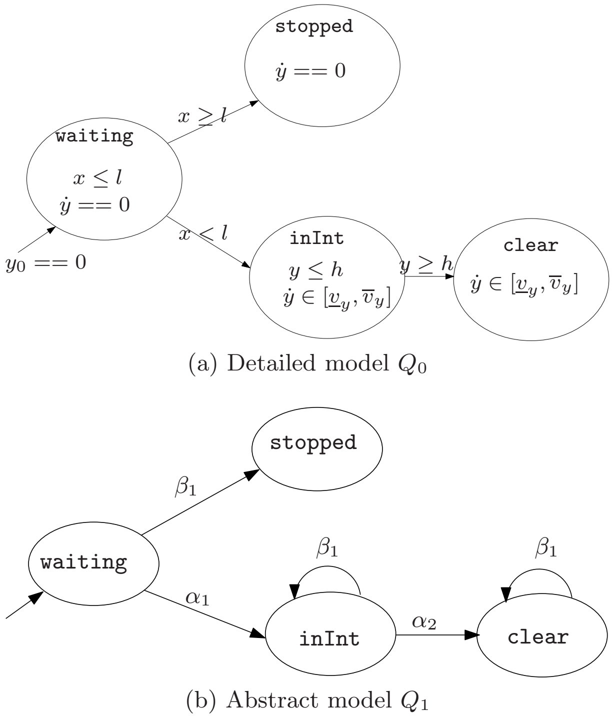
Otherwise, $\mathcal{A}^{POV}(b_0^{POV}) = \varepsilon$.

Suppose the boundary l is at -300. Since the range of velocities is positive, and initial condition is in the range $[-420, -400]$, it is straightforward to show that $\forall b_0^{POV} \in B_0^{POV}$, $\mathcal{A}^{POV}(b^{POV}) = \beta_1\beta_2$. Therefore, $P_0 \sqsubseteq^{\mathcal{A}^{POV}} P_1$. Note that if l is say -410, $\mathcal{A}^{POV}(b^{POV}) = \beta_2$ for some b^{POV} and $P_0 \not\sqsubseteq^{\mathcal{A}^{POV}} P_1$.

6.2 Heterogeneous abstraction for SV

The hybrid model Q_0 and the discrete model Q_1 for the SV are shown in Fig. 4. At the hybrid (respectively, discrete) level, the SV is able to sense the POV position x as a continuous input variable (respectively, the event β_1).

The local behavior domains are B_0^{SV}: the set of 2-d hybrid trajectories $h^{SV}(t)$, where $h^{SV} := (l^{SV}, x, y)$ are the hybrid



(a) Detailed model Q_0

(b) Abstract model Q_1

Figure 4: Hybrid and discrete SV models.

states that take values in $\mathcal{L}^{SV} \times \mathcal{X}^{SV}$, for the discrete set of locations $\mathcal{L}^{SV} := \{\texttt{waiting}, \texttt{stopped}, \texttt{inInt}, \texttt{clear}\}$ and the continuous state space $\mathcal{X}^{SV} := \mathbb{R}^2$; and $B_1^{SV} := \Sigma^{SV*}$ with $\Sigma^{SV} := \{\alpha_1, \alpha_2, \beta_1\}$, where α's signify SV entering and exiting the intersection.

A behavior abstraction function $\mathcal{A}^{SV} : B_0^{SV} \to B_1^{SV}$, constructed by only keeping the discrete part of the hybrid model and adding transition labels, is written formally as follows. Given $b_0^{SV} = h^{SV}(t)$, where $t \in \mathbb{R}_+$ and $h^{SV} = (l^{SV}, x, y)$, and $b_1^{POV} = \sigma_0\sigma_1\cdots$ with states $q_i^{SV} \in \mathcal{L}_i^{SV}$ s.t. $q_i^{SV} \xrightarrow{\sigma_i} q_{i+1}^{SV}$, $\mathcal{A}^{SV}(b_0^{SV}) = b_1^{SV}$ iff ∃ times $t_i \in \mathbb{R}_+$ s.t. $\forall\ t' \in [t_i, t_{i+1})$ with $t_0 = 0$, $l^{SV}(t') == q_i^{SV}$. Otherwise, $\mathcal{A}^{POV}(b_0^{POV}) = \varepsilon$.

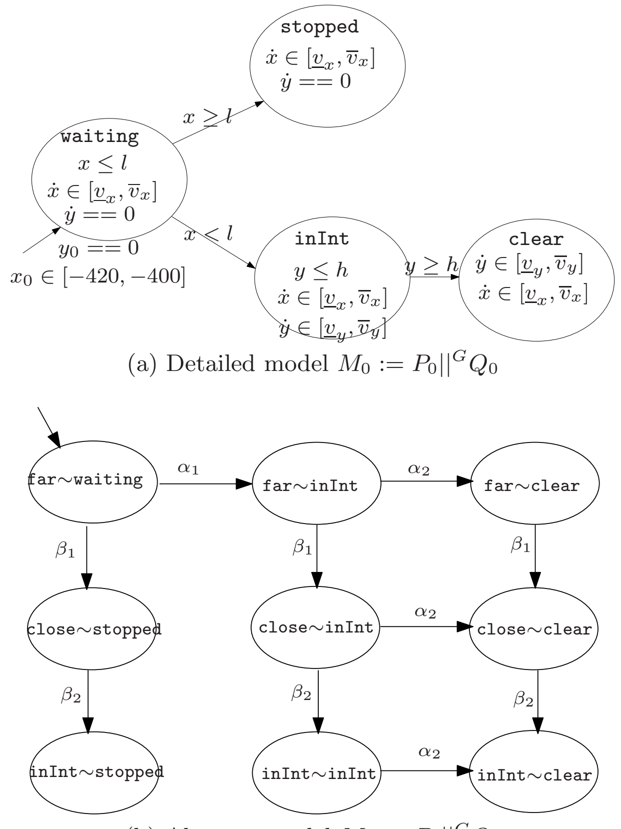
Because Q_1 has the exact same discrete transition graph as that of Q_0, for every hybrid behavior $b_0^{SV} \in [\![Q_0]\!]^{B_0^{SV}}$, $\mathcal{A}^{SV}(b_0^{SV}) \in [\![Q_1]\!]^{B_1^{SV}}$, i.e., $Q_0 \sqsubseteq^{\mathcal{A}^{SV}} Q_1$.

6.3 Abstraction between compositions

At the discrete level of abstraction, the global unified behavior domain B_1 is Σ^*, where $\Sigma = \Sigma^{POV} \cup \Sigma^{SV} = \{\alpha_1, \alpha_2, \beta_1, \beta_2\}$. Behavior localizations $\downarrow_1^j$, $j = P, Q$ are discrete event projection functions that replace a string not in the local label set by the empty string ε. In this case, the syntactic procedures of adding self loops on the missing labels α_1, α_2 in P_1 and β_2 in Q_1 take care of the globalizations and their composition is simply their product. At the hybrid level, we add an unrestricted continuous variable y in P_0 leaving Q_0 unchanged, and take the parallel composition of the resulting hybrid automata. The resultant system models $M_i := P_i||^G Q_i$, $i = 0, 1$ are as shown in Fig. 5.

The variables common to local behavior domains B_i^{POV} and B_i^{SV} are x and β_1. We have to make sure that the localizations $\mathcal{A}^{POV}\Downarrow^\cap$ and $\mathcal{A}^{SV}\Downarrow^\cap$ of abstraction functions $\mathcal{A}^{POV}$ and $\mathcal{A}^{SV}$ onto these common variables , i.e., the mappings from behaviors in x to behaviors in $\{\beta_1\}^*$ agree. $\mathcal{A}^{POV}\Downarrow^\cap$ is essentially the same as $\mathcal{A}^{POV}$, with the row for β_2 discarded. $\mathcal{A}^{SV}$ puts indirect restrictions on x due to the guard and invariant conditions of the hybrid transitions $(\texttt{waiting}, x) \to (\texttt{stopped}, x)$ that are mapped with the discrete transition $\texttt{waiting} \xrightarrow{\beta_1} \texttt{stopped}$. Such a hybrid transition occurs iff $x \leq l$ and $x \geq l$ hold before and after the transition, i.e., while crossing the boundary $x = l$ in the increasing direction, which agrees with $\mathcal{A}^{POV}\Downarrow^\cap$. In the self-loop β_1 transitions, x does not appear and is therefore unrestricted, and in agreement with $\mathcal{A}^{POV}\Downarrow^\cap$.

Therefore, using Prop. 3, we can conclude (without having to analyze models M_0 and M_1 directly) that $M_0 \sqsubseteq^{\mathcal{A}} M_1$.



(a) Detailed model $M_0 := P_0||^G Q_0$

(b) Abstract model $M_1 := P_1||^G Q_1$

Figure 5: Hybrid and discrete system models.

6.4 Need for consistency between abstraction functions

Note that if for some reason, the parameter l is different in models P_0 and Q_0, the consistency condition in Prop. 3 cannot be satisfied and the heterogeneous approach cannot be used. Suppose the reference marker in the POV component is l' rather than l, but the SV thinks it is l. Physically this may correspond to, e.g., a measurement error or parallax for a human SV driver. In this case, there is a disagreement between the two models as to what corresponds to the β_1 event of POV going from `far` to `close`. Since the two abstraction functions disagree on the mapping between behaviors in the variable x and the event β_1 that are common to the local behavior domains of the two components, the design freedom in the non-uniqueness of globalizations while adding the remaining variables and events does not help us resolve this mismatch. Therefore, we cannot find any agreeing globalizations of the two abstraction functions. In such a case, although heterogeneous abstraction still holds for the two components individually, it does not carry over to their composition.

7. CONCLUSION

This paper presents a compositional approach to heterogeneous abstraction. Behavior abstraction functions are proposed to establish semantic associations between heterogeneous formalisms across different levels of abstraction, and localizations/globalizations are used to associate local component behavior domains and global system behavior domains at a given level of abstraction. Sufficient conditions are developed under which heterogeneous abstraction between component models implies heterogeneous abstraction between their compositions. The theoretical concepts are illustrated using the example of a cooperative intersection collision avoidance system (CICAS).

As noted in the paper, abstraction relations as well as globalization and localization typically can be inferred directly from the structure and syntactic rules for constructing models. Future work will address the possibility of heterogeneity between the interacting component models within a given level of abstraction. This requires the development of heterogeneous generalizations of abstraction globalization and globalized semantic composition.

Acknowledgments

The authors gratefully acknowledge support by NSF grants CNS 1035800-NSF and CCF-0926181.

8. REFERENCES

[1] Cooperative intersection collision avoidance systems (CICAS). `http://www.its.dot.gov/cicas/`.

[2] M. Althoff, A. Rajhans, B. H. Krogh, S. Yaldiz, X. Li, and L. Pileggi. Formal Verification of Phase-Locked Loops Using Reachability Analysis and Continuization. In *Proceedings of the IEEE/ACM 2011 International Conference on Computer-Aided Design (ICCAD)*, San Jose, Nov 2011.

[3] R. Alur, T. Dang, and F. Ivancic. Predicate abstraction for reachability analysis of hybrid systems. *ACM Transactions on Embedded Computing Systems*, 5(1):152–199, 2006.

[4] R. Alur, T. A. Henzinger, G. Laffarriere, and G. J. Pappas. Discrete Abstractions of Hybrid Systems. *Proceedings of the IEEE*, 88:971–984, 2000.

[5] F. Balarin, Y. Watanabe, H. Hsieh, L. Lavagno, C. Passerone, and A. Sangiovanni-Vincentelli. Metropolis: an integrated electronic system design environment. *Computer*, 36(4):45–52, april 2003.

[6] S. Bensalem, M. Bozga, T.-H. Nguyen, and J. Sifakis. Compositional verification for component-based systems and application. *IET Software*, 4(3):181–193, 2010.

[7] A. Benveniste, L. P. Carloni, P. Caspi, and A. L. Sangiovanni-Vincentelli. Composing Heterogeneous Reactive Systems. *ACM Transactions on Embedded Computing Systems*, 7(4), July 2008.

[8] S. S. Bhattacharyya, E. Cheong, and I. Davis. PTOLEMY II Heterogeneous Concurrent Modeling and Design in Java. Technical report, University of California, Berkeley, 2003.

[9] A. Chutinan and B. H. Krogh. Verification of Infinite-State Dynamic Systms Using Approximate Quotient Transition Systems. *IEEE Transactions on Automatic Control*, 46:1401–1410, 2001.

[10] T. Dang, O. Maler, and R. Testylier. Accurate Hybridization of Nonlinear Systems. In *Proceedings of the International Conference on Hybrid Systems: Computation and Control (HSCC)*, 2010.

[11] G. Frehse. *Compositional Verification of Hybrid Systems using Simulation Relations*. PhD thesis, Radboud Universiteit Nijmegen, 2005.

[12] G. Frehse. PHAVer: Algorithmic Verification of Hybrid Systems past HyTech. *International Journal on Software Tools for Technology Transfer (STTT)*, 10(3), 2008.

[13] T. A. Henzinger, P.-H. Ho, and H. Wong-Toi. Algorithmic Analysis of Nonlinear Hybrid Systems. *IEEE Transactions on Automatic Control*, 43:225–238, 1998.

[14] T. A. Henzinger, S. Qadeer, S. K. Rajamani, and S. Tasiran. An assume-guarantee rule for checking simulation. *ACM Trans. Program. Lang. Syst.*, 24(1):51–64, Jan. 2002.

[15] A. A. Julius. *On interconnection and equivalence of continuous and discrete systems: a behavioral perspective*. PhD thesis, University of Twente, 2005.

[16] D. Kaynar and N. Lynch. Decomposing Verification of Timed I/O Automata. In *In the Proceedings of the Joint Conference on Formal Modelling and Analysis of Timed Systems (FORMATS) Formal Techniques in Real-Time and Fault Tolerant System (FTRTFT)*, 2004.

[17] A. Ledeczi, J. Davis, S. Neema, and A. Agrawal. Modeling methodology for integrated simulation of embedded systems. *ACM Trans. Model. Comput. Simul.*, 13:82–103, January 2003.

[18] E. A. Lee and A. Sangiovanni-Vincentelli. A Framework for Comparing Models of Computations. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 17(12):1217–1229, Dec 1998.

[19] A. Rajhans and B. H. Krogh. Heterogeneous verification of cyber-physical systems using behavior relations. In *Proceedings of the 15th ACM International Conference on Hybrid Systems: Computation and Control (HSCC)*, 2012.