Heterogeneous Verification of Cyber-Physical Systems using Behavior Relations

Akshay Rajhans
arajhans@ece.cmu.edu

Bruce H. Krogh
krogh@ece.cmu.edu

Department of Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213

ABSTRACT

Today's complex cyber-physical systems are being built increasingly using model-based development (MBD), where mathematical models for the system behavior are checked against design specifications using analysis tools. Different types of models and analysis tools are used to address different aspects of the system. While the use of heterogeneous formalisms supports a divide-and-conquer approach to complexity and allows engineers with different types of expertise to work on various aspects of the design, system integration problems can arise due to the lack of an underlying unifying formalism. In this paper, we introduce the notion of behavior relations to address the problem of heterogeneity and propose constraints over parameters as a mechanism to manage inter-model dependencies and ensure consistency. In addition, we present structured constructs of nested conjunctive and disjunctive analyses to enable multi-model heterogeneous verification. 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, Cyber-Physical Systems, Behavior Relations

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 has the ability to catch errors early in the system design before the system or prototypes are built, thereby avoiding costly re-design/re-development cycles. For all but the most trivial systems, many types of models need to be created and analyzed. This introduces the problem of heterogeneity: without a single comprehensive modeling formalism, how can it be guaranteed that the heterogeneous models are consistent with each other, and how can verification results from the different formalisms be combined to infer system-level properties? In this paper, we propose a general framework based on *behavior relations* and *constraints over parameters* as a formal basis for the design and verification of complex systems using multiple heterogeneous models.

Heterogeneity is inherent in cyber-physical systems (CPS) due to the tight coupling between computation elements, physical dynamics and communication networks. Typical heterogeneous aspects of a CPS are its physical dynamics, control logic, software implementation, real-time execution, communication networking and so on. For example, consider the cooperative intersection collision avoidance system for stop-sign assist (CICAS-SSA) [1] illustrated in Fig. 1. The figure depicts a vehicle called the *subject vehicle* (SV) waiting on a minor road to cross through major-road traffic at a stop-sign-controlled intersection. The system aims to augment human judgment about safe gaps in oncoming traffic by *sensing* the speeds and positions of the oncoming vehicles using cameras, magnetic induction loops or other sensors, *communicating* these values to a decision system via wired or wireless networks and *computing* safe gaps based on the *physical dynamics* of the vehicles and speed limits, implemented either on a dedicated road-side computer or onboard a smart vehicle. There is no good unified formalism for modeling all aspects of this complex heterogeneous system. And even if there were, verifying the correctness of the system design using a single model would be an intractable problem.

MBD of CPS involves creating a collection of different models using a variety of formalisms that are best suited for the different aspects of the overall design problem. Common formalisms used for design and analysis of a CPS include: acausal equation-based models in tools such as MapleSim and Modelica, suited for modeling the underlying physics of a system, e.g., the plant dynamics; signal-flow models in tools such as Simulink, suitable for control design and simulation; finite state machines and labeled transition systems in tools such as LTSA, best suited for modeling decision logic and communication protocols; hybrid-dynamic models such as hybrid automata in tools such as SpaceEx, useful for

HSCC'12, April 17–19, 2012, Beijing, China.

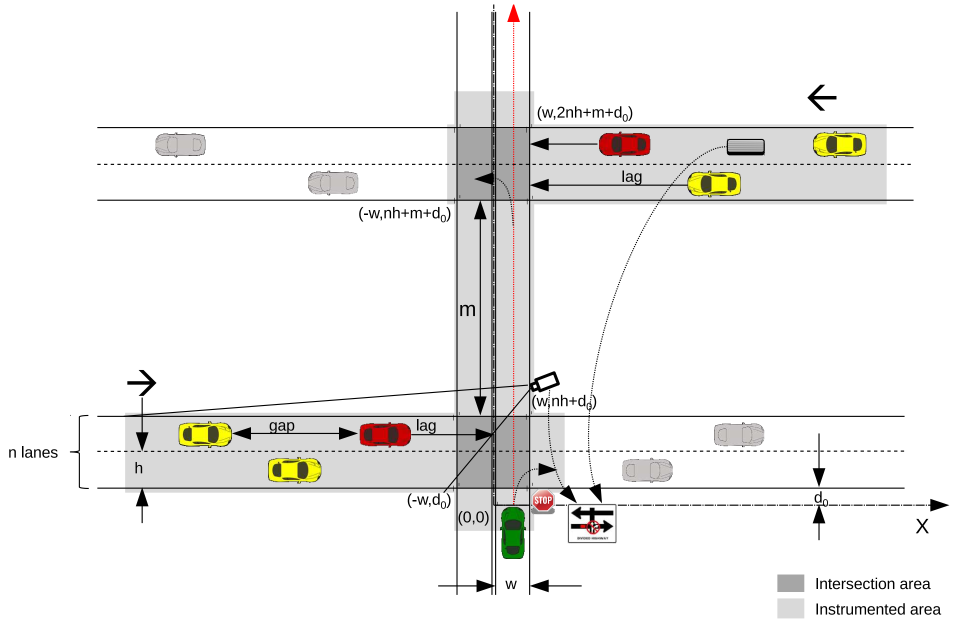

Figure 1: A pictorial sketch of CICAS-SSA. Parameters w, n, h, m, d_0 depend on the intersection geometry.

analyzing abstract unified behaviors of continuous dynamics and discrete mode switches; network simulation models in tools such as OMNET++, useful for analyzing communication network properties such as packet loss, communication delay and so on; and software models in tools such as Spin, useful for analyzing whether the decision logic is correctly implemented.

These heterogeneous models are usually created and analyzed by different engineers due to the wide range of expertise necessary for designing complex systems. In current practice, methods for maintaining consistency between the models and composing verification results from the various models to infer system-level properties are ad hoc at best. This paper presents a formal basis for addressing these problems.

The rest of the paper is organized as follows. We begin with a review of the relevant literature in Sec. 2. In Sec. 3, a general framework is developed for verification using heterogeneous models. Sec. 4 provides conjunctive and disjunctive constructs to enable heterogeneous verification. Sec. 5 illustrates the concepts using the CICAS-SSA example. Sec. 6 introduces the notion of semantic consistency using constraints over parameters and these concepts are illustrated in Sec. 7. The paper concludes with a discussion and future work in Sec. 8.

2. RELATED WORK

The idea of using an abstraction in a simpler modeling formalism in order to verify safety properties of a more complex model in the original formalism has been frequently used in the literature. Hybrid abstractions of nonlinear systems [15, 12], LHA abstractions of linear hybrid systems [13], discrete abstractions of hybrid systems [4, 11, 3] and continuous abstractions of hybrid systems [2] are some of the examples where simpler abstractions are successfully created and used. These approaches use specific pairs of formalisms. Our objective is to create a general framework for abstraction that can support any set of heterogeneous formalisms.

Towards the aim of heterogeneous multi-model development, several research efforts have focused on supporting simulation of heterogeneous elements in a common framework. 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 [18] 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. However, the focus of these efforts has been simulation and not verification.

Inference-based approaches that use ontologies have been proposed for static analysis and type checking [20]. In a similar spirit, the work in [17] focuses on integrating the results of disparate verification efforts and analysis techniques using static and epistemic ontologies. Rather than using an ontology-based approach, we use a behavioral approach to compare and relate behaviors of different types.

The work by Julius [16] uses a behavioral approach in the spirit of Willems' work [22] and creates a framework for comparing and interconnecting behaviors based on the different time axis structures for discrete, continuous and hybrid behaviors. For embedded software applications, the Behavior-Interaction-Priority (BIP) framework [9] leverages the component structure of a system and supports behavioral annotation of the components in the form of state diagrams [6] to support system analysis. In contrast to Julius's approach of incorporating behaviors in the definition of models, we see behaviors as the semantic interpretation of systems, which allows us to observe behaviors in different domains. This idea is similar to the one proposed in [14], where timed and time-abstract traces serve as different semantics for the same hybrid automaton. The notion of tagged signal semantics has been proposed to compare [19] and compose

[7] heterogeneous reactive systems. Unlike [9, 7], the focus of this work is not to compose heterogeneous components into one big system, but rather to use heterogeneous models independently towards a common system-verification goal.

The TLA+ proof system deploys a proof manager that breaks down a complex verification task logically into proof obligations that are proved using theorem provers and SMT solvers [10]. We use a similar approach for logically composing the results of verification activities, but their framework based on temporal logic of actions (TLA) is primarily aimed towards software systems, whereas our framework supports more general (e.g. continuous, hybrid) dynamics and non-deductive analysis methods.

3. HETEROGENEOUS VERIFICATION

Our objective is to use models and their specifications to reason about the underlying system. The first step in analyzing heterogeneous models and specifications together in a common framework is to create a mechanism to compare their associated sets of behaviors. In our previous work, we dealt with heterogeneity based on the assumption that one can create semantic mappings from each model and specification onto one common behavioral domain [21]. Here create we a framework using behavior relations to support true semantic heterogeneity by allowing the use of several different types of behavior formalisms for different models and specifications.

A *behavior formalism* B is the set of all possible behaviors of a particular type. There is no restriction on the type of behaviors: they could be event traces, continuous trajectories, hybrid traces, input-output maps or something else.

Definition 1 (Behavior Relation) *Given behavior formalisms B_1 and B_2, a* behavior relation *is a set $R \subseteq B_1 \times B_2$ that associates pairs of behaviors from the two sets B_1 and B_2.*

For a subset of behaviors $B_1' \subseteq B_1$, let $R(B_1')$ denote the set of behaviors in B_2 associated with behaviors in B_1', i.e., $R(B_1') = \{b_2 \mid \exists b_1 \in B_1' \text{ s.t. } (b_1, b_2) \in R\}$. Similarly, for $B_2' \subseteq B_2$, let $R^{-1}(B_2')$ represent the set of behaviors in B_1 associated with behaviors in B_2', i.e., $R^{-1}(B_2') = \{b_1 \mid \exists b_2 \in B_2' \text{ s.t. } (b_1, b_2) \in R\}$.

A *specification* S is a a logical assertion written in a specification formalism $\mathcal{S}$. There is no restriction on what specification formalism can be used. Specifications could be written in, for example, various temporal logics, Kripke structures, automata, sets of unsafe states to be avoided, or even in English language, so long as their semantic interpretation is clear in terms of the associated behavioral formalism. The *semantic interpretation* of S in a behavior formalism B, denoted by $[\![S]\!]^B$, is defined as the set of all behaviors in B for which, the specification is satisfied.

When semantically interpreted over the same set of behaviors B, a (stronger) specification S_2 is said to imply a (weaker) specification S_1, written $S_2 \Rightarrow^B S_1$ if $[\![S_2]\!]^B \subseteq [\![S_1]\!]^B$. The following definition extends this notion to heterogeneous behavior spaces using behavior relations.

Definition 2 (Heterogeneous Implication) *Given behavior formalisms B_1, B_2 and a behavior relation $R \subseteq B_1 \times B_2$, we say that specification S_2* implies *specification S_1 via R, written $S_2 \Rightarrow^R S_1$, if*

$$R^{-1}([\![S_2]\!]^{B_2}) \subseteq [\![S_1]\!]^{B_1}.$$

This definition requires that if a behavior $b_1 \in B_1$ is associated through R with a behavior in $b_2 \in B_2$ that satisfies S_2, then b_1 satisfies S_1.

A *modeling formalism* $\mathcal{M}$ is a set of models of a particular type. Transition systems, hybrid automata, signal-flow models, acausal equation-based models, and network models are some of the modeling formalisms used in CPS; however the discussion is valid for any modeling formalism. A *model* M is an element of some formalism $\mathcal{M}$. Given a behavior formalism B, the *semantic interpretation* of a model M is the set of behaviors $[\![M]\!]^B \subseteq B$ that it allows.

When interpreted over the same behavioral formalism B, a model M_2 is an *abstraction* of a model M_1, written $M_1 \sqsubseteq^B M_2$, if $[\![M_1]\!]^B \subseteq [\![M_2]\!]^B$. This is the standard definition of abstraction common among the literature, using, for example, language or trace inclusion.

Definition 3 (Heterogeneous Abstraction) *Given behavior formalisms B_1, B_2 and a behavior relation $R \subseteq B_1 \times B_2$, a model M_2 is an* abstraction *of a model M_1 through R, written $M_1 \sqsubseteq^R M_2$, if*

$$[\![M_1]\!]^{B_1} \subseteq R^{-1}([\![M_2]\!]^{B_2}).$$

This definition asserts that for every behavior in B_1 of model M_1, the behavior relation R associates at least one corresponding behavior in B_2 of model M_2.

In a given behavior formalism B, a model M *entails* a specification S, written $M \models^B S$, if $[\![M]\!]^B \subseteq [\![S]\!]^B$. When true, this simply asserts that the set of behaviors of the model M do not violate the set of safe behaviors allowed by the specification S. To establish this type of entailment, formal approaches such as reachability analysis and theorem proving, or semi-formal approaches like systematic state-space exploration, need to be used whenever possible. We do not restrict what method the system designer chooses to use to establish entailment.

Proposition 1 *Given behavior formalisms B_1 and B_2, models M_1 and M_2, specifications S_1 and S_2, and a behavior relation $R \subseteq B_1 \times B_2$, if $M_1 \sqsubseteq^R M_2$, $M_2 \models^{B_2} S_2$ and $S_2 \Rightarrow^R S_1$, then $M_1 \models^{B_1} S_1$.*

Proof. From $M_1 \sqsubseteq^R M_2$, we have

$$\begin{aligned} [\![M_1]\!]^{B_1} &\subseteq R^{-1}([\![M_2]\!]^{B_2}) \\ (\text{From } M_2 \models^{B_2} S_2) &\subseteq R^{-1}([\![S_2]\!]^{B_2}) \\ (\text{From } S_2 \Rightarrow^R S_1) &\subseteq [\![S_1]\!]^{B_1}. \end{aligned}$$

Therefore, $M_1 \models^{B_1} S_1$.

This proposition gives us the conditions under which a heterogeneous abstraction of a complex model can be used to verify a property of the underlying system. In the following section, we further develop this idea to use *several* abstractions to verify properties of a given system.

4. MULTI-MODEL HETEROGENEITY

There are two natural ways of using multiple models and specifications. In one, models individually are abstractions of the underlying system and the conjunction of their associated specifications needs to imply the system specification. Alternatively, each model may represent only a subset of the behaviors of the underlying system, and the collection of models provides an abstraction of the complete system.

In this second case, the specification for each model needs to imply the specification of interest for the underlying system for the set of behaviors covered by the model. The following develops these two notions in the context of heterogeneous verification.

We first consider the case where each model is a heterogeneous abstraction of the underlying system. In this case, we need to ensure that the specifications checked against each model together imply the specification of the underlying system. The following definition makes this notion formal.

Definition 4 (Conjunctive Heterogeneous Implication) *Given a system behavior formalism B_0, behavior formalisms B_i and behavior relations $R_i \subseteq B_0 \times B_i$, $i = 1, \ldots, n$, specifications $S_i, i = 1, \ldots, n$* conjunctively imply *the system specification S_0 if*

$$\bigcap_i R_i^{-1}(\llbracket S_i \rrbracket^{B_i}) \subseteq \llbracket S_0 \rrbracket^{B_0}.$$

This definition allows the individual specifications S_i to not imply S_0, but their conjunction (intersection of the allowed behaviors) is required to be stronger than S_0.

Proposition 2 (Heterogeneous Conjunctive Analysis) *For a system model M_0 with a behavioral formalism B_0 and specification S_0, given models M_i with the corresponding behavior formalisms B_i, specifications S_i and behavior relations $R_i \subseteq B_0 \times B_i$, if $M_0 \sqsubseteq^{R_i} M_i$, specifications S_i conjunctively imply S_0, and $M_i \models^{B_i} S_i$ for each $i = 1, \ldots, n$, then $M_0 \models^{B_0} S_0$.*

Proof. From $M_0 \sqsubseteq^{R_i} M_i$ for each i, we have

$$\begin{aligned} \llbracket M_0 \rrbracket^{B_0} &\subseteq \bigcap_i R_i^{-1}(\llbracket M_i \rrbracket^{B_i}) \\ (\text{since } M_i \models^{B_i} S_i) &\subseteq \bigcap_i R_i^{-1}(\llbracket S_i \rrbracket^{B_i}) \\ (\text{Conj. Het. Implication}) &\subseteq \llbracket S_0 \rrbracket^{B_0}. \end{aligned}$$

Therefore, $M_0 \models^{B_0} S_0$.

Now we consider the case where different models are built to represent different subsets of behaviors of a system. This is typically useful when there are different behaviors in different operating regimes best modeled by different models, where neither one fully represents the whole set of behaviors of the system, but their union does. This notion is made formal by the following definition.

Definition 5 (Model Coverage) *For a system model M_0 with a behavioral formalism B_0, given a set of models M_i with corresponding behavior formalisms B_i and behavior relations $R_i \subseteq B_0 \times B_i$, models M_i, $i = 1, \ldots, n$* cover *M_0 if there exists a partition $\{B_0^1, B_0^2, \ldots, B_0^n\}$ of $\llbracket M_0 \rrbracket^{B_0}$ s.t. $\forall i = 1, 2, \ldots, n$*

$$B_0^i \subseteq R_i^{-1}(\llbracket M_i \rrbracket^{B_i}).$$

This definition requires that every behavior of the underlying system M_0 to be accounted for by at least one model.

Lemma 1 *If models M_i cover M_0 through R_i, $i = 1, \ldots, n$, we have*

$$\llbracket M_0 \rrbracket^{B_0} \subseteq \bigcup_{i=1}^{n} R_i^{-1}(\llbracket M_i \rrbracket^{B_i}).$$

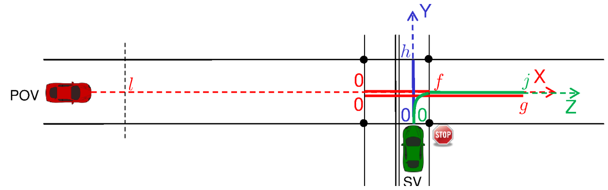

Figure 2: A simple variant of the CICAS-SSA with one near-side oncoming lane with one POV. Road coordinates X, Y and Z are along the POV path, SV path going straight and SV turning right respectively. Conflict areas along the paths shown using bold line segments.

Proof. From the definition of partition, we have

$$\begin{aligned} \llbracket M_0 \rrbracket^{B_0} &= \bigcup_{i=1}^{n} B_0^i \\ (\text{Def. 5}) &\subseteq \bigcup_{i=1}^{n} R_i^{-1}(\llbracket M_i \rrbracket^{B_i}). \end{aligned}$$

In this case, since each model is not an abstraction of the underlying system, to imply a specification for the underlying system it is necessary that we verify specifications that are at least as strong as the system specification, as stated in the following proposition.

Proposition 3 (Heterogeneous Disjunctive Analysis) *For a system model M_0 with a behavioral formalism B_0 and specification S_0, given models M_i with the corresponding behavior formalisms B_i, specifications S_i and behavior relations $R_i \subseteq B_0 \times B_i$, if each specification S_i heterogeneously implies S_0, models M_i cover M_0, and $M_i \models^{B_i} S_i$ for each $i = 1, \ldots, n$, then $M_0 \models^{B_0} S_0$.*

Proof. From the definition of model coverage, we have

$$\begin{aligned} \llbracket M_0 \rrbracket^{B_0} &\subseteq \bigcup_i R_i^{-1}(\llbracket M_i \rrbracket^{B_i}) \\ (\text{since } M_i \models^{B_i} S_i) &\subseteq \bigcup_i R_i^{-1}(\llbracket S_i \rrbracket^{B_i}) \\ (\text{Het. Implication}) &\subseteq \llbracket S_0 \rrbracket^{B_0}. \end{aligned}$$

Therefore, $M_0 \models^{B_0} S_0$.

Finally, we note that the conjunctive and disjunctive analysis constructs can be nested arbitrarily. For example, the j^{th} conjunctive verification subtask $M_j \models^{B_j} S_j$ can be broken down disjunctively into its subtasks $M_{ji} \models^{B_{ji}} S_{ji}$ by creating new models that cover M_j and specifications that imply S_j. Thus, using the nesting of conjunctive and disjunctive constructs, any arbitrary logical breakdown of a system verification task can be achieved. This is illustrated in an example in the following section.

5. EXAMPLE

Consider a simple variant of the CICAS-SSA as shown in Fig. 2, with a single major-road lane and one oncoming principal other vehicle (POV). The subject vehicle (SV) can either go straight or turn right to merge into POV's path. The SV is able to sense the position of the POV, and the decision of whether to start driving or not is made on-board

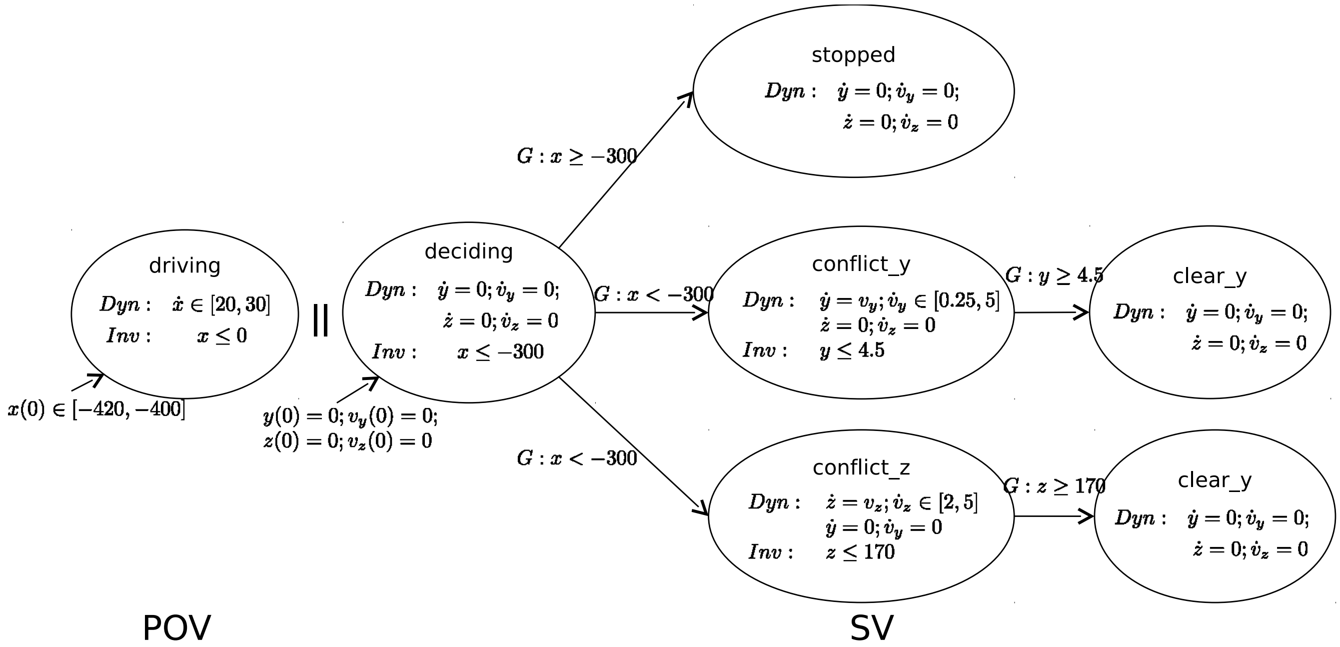

Figure 3: A single universal model M_0 for the simple SSA system.

the SV using this sensed position of the POV. The road coordinates along the path of the POV and along the straight and right-turn paths of the SV are assumed to be along dimensions X, Y and Z respectively. The conflict areas where crashes can occur (depending on the intersection geometry) are either $x \in [0, f = 3]$ and $y \in (0, h = 4.5)$ or $x \in [0, g]$ and $z \in (0, j)$, where f and h depend on the intersection geometry, and g and j are chosen large enough (here, 170m) such that the SV has a chance to accelerate to the highway speed so that after the turn there is no (intersection-related) collision.

Fig. 3 shows a model of the system made up of two hybrid automata components SV and POV. The decision strategy implemented on-board the SV is that if the POV hasn't crossed an imaginary marker at position $l = -300$ along the X axis, the SV is permitted to start driving, but it doesn't have to. When POV crosses l, the SV has to stay stopped, forced by the invariant in `deciding`. Whenever permitted, whether the SV decides to go straight or turn right is represented as a nondeterministic choice; however once it has committed to one, it isn't allowed to change its mind. The evolution stops when the SV clears the conflict regions or when the POV enters the intersection. By the time the POV enters the intersection, if the SV is still in the conflict zone, there is a safety violation (a potential collision). Alternatively, if the SV has cleared the conflict zone or hasn't entered it, there is no safety violation. The objective is to guarantee collision freedom for this particular strategy. The collision-freedom specification S_0 can be defined by the temporal logic formula $S_0 : \Box \neg ((x == 0 \wedge 0 < y < 4.5) \vee (x == 0 \wedge 0 < z < 170))$.

5.1 Disjunctive analysis

We first disjunctively break down the problem into two subproblems. We create two models, one for the case where SV is only allowed to go straight and the other where the SV is only allowed to go right, as shown in Fig. 4 and 5. The behavior domain of M_0 (i.e., B_0) is the set of all five dimensional hybrid traces, while B_1 and B_2 are each sets of all three dimensional hybrid traces. The behavior relations for this breakdown are as follows:

- $R_1 : \{(b_0, b_1) | b_0 \downarrow_{z,v_z} == \bar{0}$ and $b_0 \downarrow_{x,y,v_y} == b_1\}$
- $R_2 : \{(b_0, b_2) | b_0 \downarrow_{y,v_y} == \bar{0}$ and $b_0 \downarrow_{x,z,v_z} == b_2\}$

where $\bar{0}$ represents a 2-d trace of zeros over all time and $\downarrow_{()}$ represents the projection on $()$.

The specifications to be checked for the two models are

- $S_1 : \Box \neg (x == 0 \wedge 0 < y < 4.5)$ and
- $S_2 : \Box \neg (x == 0 \wedge 0 < z < 170)$.

We have heterogeneous implication $S_1 \Rightarrow^{R_1} S_0$ because $R_1^{-1}([\![S_1]\!]^{B_1})$ forces that y be conflict-free and z be 0, which implies that y is conflict-free and z is conflict-free. Similarly, we have $S_2 \Rightarrow^{R_2} S_0$. Further, we note that in every behavior of M_0, either $\{y, v_y\}$ or $\{z, v_z\}$ are zero and both the possibilities are covered by either model. Therefore, from Prop. 3, if $M_1 \models^{B_1} S_1$ and $M_2 \models^{B_2} S_2$, we can conclude $M_0 \models^{B_0} S_0$. Out of these two verification sub-tasks, we show how $M_1 \models^{B_1} S_1$ can be proved using conjunctive analysis in the next subsection. $M_2 \models^{B_2} S_2$ can be shown in a similar manner.

5.2 Conjunctive analysis

Consider the subtask of showing $M_1 \models^{B_1} S_1$. We break down this task conjunctively by creating three models M_{1i} and constructing corresponding specifications $S_{1i}, i = 1, 2, 3$, as shown in Fig. 5. M_{11} models the behaviors of the POV, and is exactly the same as the POV automaton in M_1. M_{12}

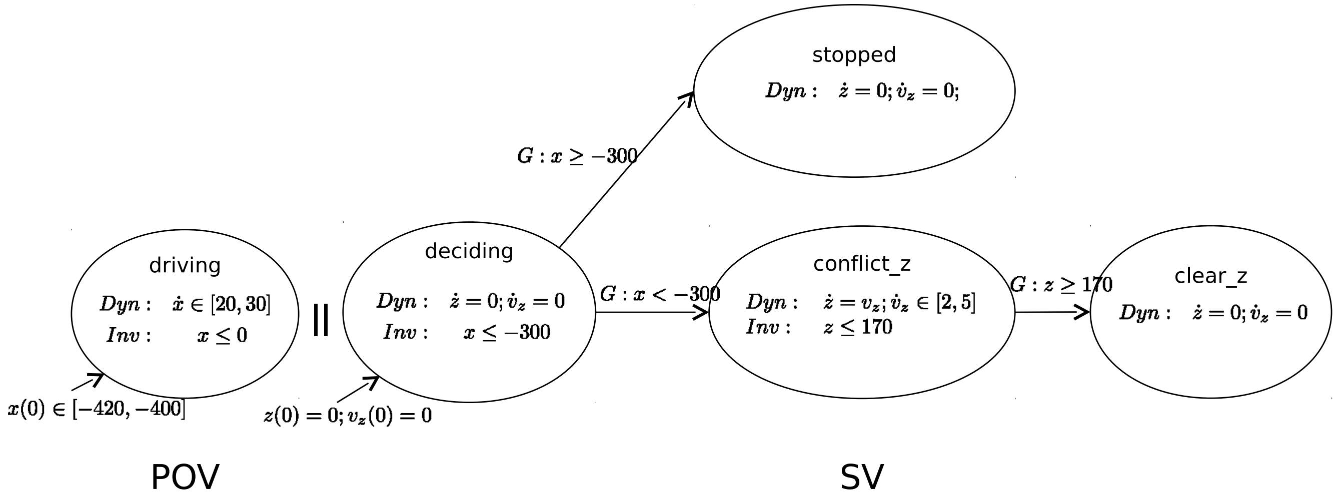

Figure 4: A model M_2 for SV only going right.

models the behavior of the SV only while it is in the conflict zone and has the same dynamics as that of the `conflict_y` location of M_1. M_{13} is a discrete model consisting of two elements. The component POV is a created by partitioning the component POV of M_1 into discrete states `far`,`close`, and `inInt` using predicates $x \leq -300$, $-300 \leq x \leq 0$, and $0 \leq x$. The second component SV is merely a discrete control graph of the hybrid automaton model for SV in M_1. The only synchronized pair of transitions is (`far`$\xrightarrow{\sigma_1}$`close`) and (`deciding`$\xrightarrow{\sigma_1}$`stopped`). Non-blocking self loops have been dropped from the pictorial representation for simplicity.

The behavior relations are

- $R_{11} : \{(b_1, b_{11}) | b_{11} == b_1 \downarrow_x \}$,
- $R_{12} : \{(b_1, b_{12}) | b_{12} == s_1 \downarrow_{y,v_y}$ where s_1 is b_1 restricted to the discrete location `(driving,conflict_y)`$\}$ and
- $R_{13} : \{(b_1, b_{13}) | b_1$ is a hybrid trajectory that visits the discrete locations corresponding to ones in b_{13} in that order $\}$.

For these behavior relations, we first note that $M_1 \sqsubseteq^{R_{1i}} M_{1i}$ because neither of the models M_{1i} is more restrictive than M_1. The specifications for the three models are

- $S_{11} : \Box\ (x == -300 \Rightarrow \Box_9\ x < 0)$,
- $S_{12} : \Box\ (\Diamond_8\ y \geq h)$ and
- $S_{13} : \Box\ ((\phi_1 \wedge \neg\phi_2) \rightarrow \neg(\Diamond\phi_2))$, where ϕ_1 is the predicate "POV is close" satisfied in states (`close`,$\cdot$) and (`inInt`,$\cdot$); and ϕ_2 is the predicate "SV is driving" satisfied in states ($\cdot$,`con_y`).

The behaviors effectively allowed in B_1 by the specifications S_{1i} are as follows:

- $R_{11}^{-1}([\![S_{11}]\!])$: system behaviors where POV takes at least 9 seconds to get from $l = -300$ to the intersection.
- $R_{12}^{-1}([\![S_{12}]\!])$: system behaviors where SV clears the intersection within 8 seconds of starting to drive.
- $R_{13}^{-1}([\![S_{13}]\!])$: system behaviors where SV does not start driving after POV crosses l.

There can only be two cases:

1. The SV has already started driving before the POV crosses l and is in the intersection: in this case, from $R_{11}^{-1}([\![S_{11}]\!])$ and $R_{12}^{-1}([\![S_{12}]\!])$ together, it will clear the intersection in at most 8 seconds and the POV won't get to the intersection in at least 9 seconds, OR
2. The SV hasn't started driving when the POV crosses l: in this case, from $R_{13}^{-1}([\![S_{13}]\!])$, the SV cannot start driving anymore.

Therefore, from all the specifications put together, the two cars can't be in the intersection at the same time, which implies S_1, i.e., we have conjunctive heterogeneous implication.

$M_{11} \models^{B_{11}} S_{11}$ can be shown by algebraic computations: for the fastest velocity (30m/s) it takes 10s to travel 300m. $M_{12} \models^{B_{12}} S_{12}$ can be shown by Newton's laws of motion: the longest time needed to cross 4.5m with initial velocity 0 and minimum acceleration 0.25m/s^2 is $\sqrt{\frac{2*4.5}{0.25}} = 6$ seconds. $M_{13} \models^{B_{13}} S_{13}$ can be shown by using Labeled Transition System Analyzer (LTSA). Under these conditions, using Prop. 2, we can infer that $M_1 \models^{B_1} S_1$.

6. HETEROGENEOUS CONSISTENCY

The framework developed in Sec. 4 treats the model abstraction and model coverage in terms of the entire sets of behaviors. At that level, the interdependencies between individual behaviors of the models are lost. In our earlier work [21], we introduced the use of constraints over parameters as a mechanism to capture interdependencies between models and to ensure consistency. Here, we redevelop a consistency framework based on constraints over parameters for our new approach using behavior relations introduced in Sec. 3 and 4 and extend the idea to also capture interdependencies between specifications.

A *parameter* p of a system is a real-valued static variable that affects the system behavior. The *valuation* of a set of

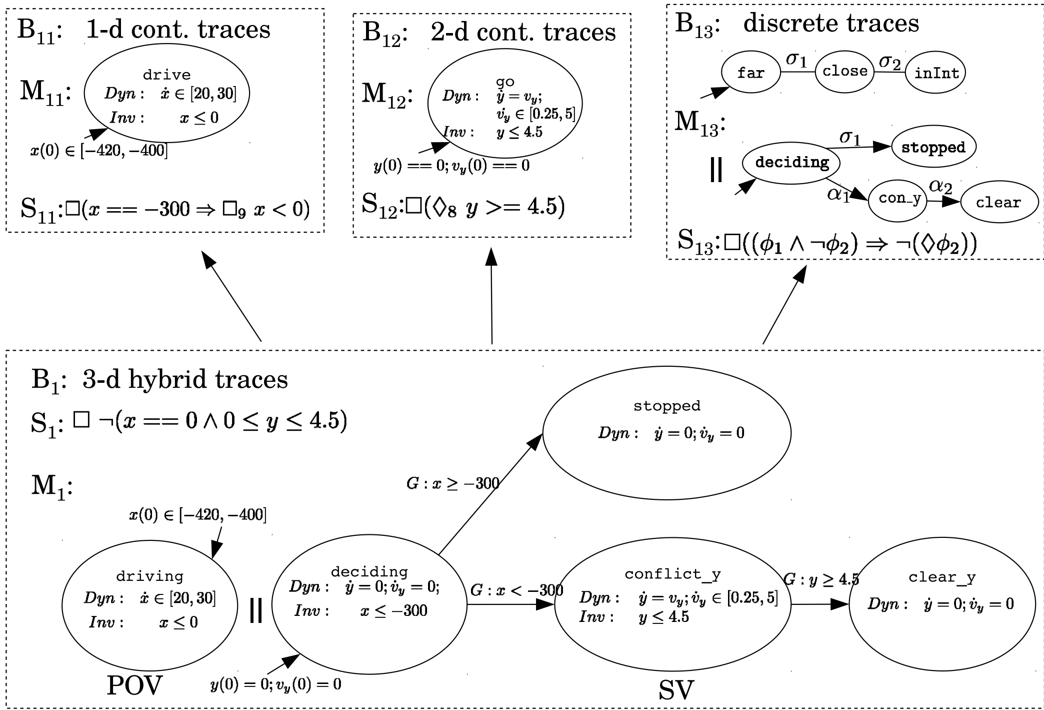

Figure 5: Heterogeneous conjunctive analysis of a model M_1 for SV going only straight.

parameters P is a function $v : P \to \mathbb{R}$ that associates each parameter with a value. $V(P)$ denotes the set of all possible valuations of the parameters in P.

A *constraint* $C(P)$ over a set of parameters P is an expression written in a constraint formalism $\mathcal{C}$, such as first-order logic of real arithmetic. For a given $v \in V(P)$, $[\![C(P)]\!]_v \in \{\top, \bot\}$ denotes the evaluation of the constraint $C(P)$ at v, and $[\![C(P)]\!]$ denotes the set of all valuations v of P for which $[\![C(P)]\!]_v = \top$.

Conjunction of constraints $C_1(P)$ and $C_2(P)$, written $C_1(P) \wedge C_2(P)$, is also a constraint whose corresponding parameter valuations are the intersection of the parameter valuations of the original constraints, i.e., $[\![C_1(P) \wedge C_2(P)]\!] = [\![C_1(P)]\!] \cap [\![C_2(P)]\!]$. Similarly, disjunction of constraints is a constraint whose corresponding parameter valuations are the union of the parameter valuations of the original constraints. We write $C'(P) \Rightarrow C(P)$ when $[\![C'(P)]\!] \subseteq [\![C(P)]\!]$.

Given two sets of parameters P and P', the *projection* of a constraint $C(P)$ onto P', written as $C(P) \downarrow_{P'}$, is the constraint over P' defined by existential quantification of the parameters in $P \setminus P'$. Its valuations $[\![C(P) \downarrow_{P'}]\!]$ are

$$\{v' \in V(P') \mid \exists v \in [\![C(P)]\!] : v'(p') = v(p')\ \forall p' \in P' \cap P\}.$$

We now consider that a set of parameters P_i is introduced for every i^{th} analysis task. Parameters $P_i^M \subseteq P_i$ are associated with the models M_i and parameters $P_i^S \subseteq P_i$ are associated with the specifications S_i. Constraints C_i^M and C_i^S determine the values of the parameters in P_i^M and P_i^S for models M_i and specifications S_i, respectively. The *semantic interpretation* of a parameterized model M_i with a constraint C_i^M, written $[\![C_i^M, M_i]\!]^{B_i}$, is the set of all possible behaviors in B_i associated with the model M_i for all parameter valuations in $[\![C_i^M(P_i^M)]\!]$. Similarly, the semantic interpretation of a parameterized specification $[\![C_i^S, S_i]\!]^{B_i}$ is the set of all behaviors in B_i that are permitted by S_i for the values of parameters P_i^S determined by the constraint C_i^S. The *parametric entailment* $C_i^M, M_i \models^{B_i} C_i^S, S_i$ needs to establish that $[\![C_i^M, M_i]\!]^{B_i} \subseteq [\![C_i^S, S_i]\!]^{B_i}$.

We observe that the set of possible behaviors of a given model grows or shrinks monotonically with increasing or decreasing sets of parameter valuations, i.e., if $C' \Rightarrow C$, then $[\![C', M]\!]^B \subseteq [\![C, M]\!]^B$ for any model M. We assume that the specifications are parameterized such that increasing sets of parameter valuations allow increasing sets of behaviors, i.e., if $C' \Rightarrow C$, then $[\![C', S]\!]^B \subseteq [\![C, S]\!]^B$ for any specification S.

We let the constraint $C_{\mathsf{aux}}(P)$ denote the auxiliary constraints that capture the dependencies across the set of all parameters $P = \bigcup_{j=0}^{n} P_j$, which is the set of all parameters being used, including the original system-level parameters P_0. Without loss of generality we assume the sets P_j, $j = 0, 1, \ldots, n$ are disjoint.

Definition 6 *We say that an auxiliary constraint C_{aux} is* non-conflicting *for a given system-level constraint C_0 if*

$$(C_0 \wedge C_{\mathsf{aux}}) \downarrow_{P_0} = C_0.$$

Definition 7 (Parametric Abstraction) *Given a parameterized model M_0 with a behavioral domain B_0, a parameterized model M_i with a corresponding behavior formalism B_i and a behavior relation $R_i \subseteq B_0 \times B_i$, M_i is said to be a* parametric abstraction *of M_0 under an auxiliary constraint C_{aux} if for any constraint C_0^M such that C_{aux} is non-conflicting for C_0^M, we have*

$$[\![C_0^M, M_0]\!]^{B_0} \subseteq R_i^{-1}([\![(C_{\mathsf{aux}} \wedge C_0^M) \downarrow_{P_i^M}, M_i]\!]^{B_i}).$$

The following definition creates a notion of coverage for parameterized models given their parameter dependencies.

Definition 8 (Parametric Coverage) *For a parameterized system model M_0 with a corresponding behavior formalism B_0, a given set of parameterized models M_i with corresponding behavior formalisms B_i and behavior relations $R_i \subseteq B_0 \times B_i$, $i = 1, \ldots, n$ form a* parametric cover *for M_0 under an auxiliary constraint C_{aux} if for any constraint C_0^M such that C_{aux} is non-conflicting for C_0^M, there exists a partition $\{B_0^1, B_0^2, \ldots, B_0^n\}$ of $[\![C_0^M, M_0]\!]^{B_0}$ s.t. $\forall i = 1, 2, \ldots, n$*

$$B_0^i \subseteq R_i^{-1}([\![(C_{aux} \wedge C_0^M)\downarrow_{P_i^M}, M_i]\!]^{B_i}).$$

Now we develop analogous definitions for parameterized specifications.

Definition 9 (Parametric Implication) *For a parameterized system specification S_0 with a corresponding behavioral formalism B_0, a parameterized specification S_i with a corresponding behavior formalism B_i and a behavior relation $R_i \subseteq B_0 \times B_i$ is said to* parametrically imply *S_0 under an auxiliary constraint C_{aux} if for any constraint C_0^S such that C_{aux} is non-conflicting for C_0^S, we have*

$$R_i^{-1}([\![(C_{aux} \wedge C_0^S)\downarrow_{P_i^S}, S_i]\!]^{B_i}) \subseteq [\![C_0^S, S_0]\!]^{B_0}.$$

Definition 10 (Conjunctive Parametric Implication) *For a parameterized system specification S_0 with a corresponding behavioral formalism B_0, a given set of parameterized specifications S_i with corresponding behavior formalisms B_i and behavior relations $R_i \subseteq B_0 \times B_i$, S_i, $i = 1, \ldots, n$* conjunctively paremetrically imply *S_0 under an auxiliary constraint C_{aux} if for any constraint C_0^S such that C_{aux} is non-conflicting for C_0^S, we have*

$$\bigcap_i R_i^{-1}([\![(C_{aux} \wedge C_0^S)\downarrow_{P_i^S}, S_i]\!]^{B_i}) \subseteq [\![C_0^S, S_0]\!]^{B_0}.$$

Definition 11 *The pair of constraints (C_i^M, C_i^S) for i^{th} analysis task is said to be* original-constraint consistent *if*

$$(C_0^M \wedge C_{aux})\downarrow_{P_i^M} \Rightarrow C_i^M \text{ and } C_i^S \Rightarrow (C_0^S \wedge C_{aux})\downarrow_{P_i^S}.$$

Given these definitions, the following two propositions give sufficient conditions for parametric conjunctive and disjunctive analysis.

Proposition 4 *Given parameterized system model M_0 and specification S_0 with corresponding behavior formalism B_0 and the pair of constraints (C_0^M, C_0^S) over the system-level parameters P_0^M and P_0^S, a set of parameterized models M_i and specifications S_i with corresponding behavior formalisms B_i, behavior relations $R_i \subseteq B_0 \times B_i$ and pairs of constraints (C_i^M, C_i^S) over parameters P_i^M and P_i^S for $i = 1, \ldots, n$, if*

i. constraints (C_i^M, C_i^S) are original-constraint consistent,

ii. each model M_i is a parametric abstraction of M_0,

iii. specifications S_i conjunctively parametrically imply S_0, and

iv. $C_i^M, M_i \models^{B_i} C_i^S, S_i$

then $C_0^M, M_0 \models^{B_0} C_0^S, S_0$.

Proof. From the definition of parametric abstraction, we have

$$\begin{aligned}
[\![C_0^M, M_0]\!]^{B_0} &= \bigcap_i R_i^{-1}([\![(C_{aux} \wedge C_0^M)\downarrow_{P_i^M}, M_i]\!]^{B_i}) \\
\text{(Def. 11)} &\subseteq \bigcap_i R_i^{-1}([\![C_i^M, M_i]\!]^{B_i}) \\
(C_i^M, M_i \models^{B_i} C_i^S, S_i) &\subseteq \bigcap_i R_i^{-1}([\![C_i^S, S_i]\!]^{B_i}) \\
\text{(Def. 11)} &\subseteq \bigcap_i R_i^{-1}([\![(C_{aux} \wedge C_0^S)\downarrow_{P_i^S}, S_i]\!]^{B_i}) \\
\text{(Def. 10)} &\subseteq [\![C_0^S, S_0]\!]^{B_0}
\end{aligned}$$

Therefore, $C_0^M, M_0 \models^{B_0} C_0^S, S_0$.

Proposition 5 *Given parameterized system model M_0 and specification S_0 with a behavior formalism B_0 and the pair of constraints (C_0^M, C_0^S) over the system-level parameters P_0^M and P_0^S, a set of parameterized models M_i and specifications S_i with corresponding behavior formalisms B_i, behavior relations $R_i \subseteq B_0 \times B_i$ and pairs of constraints (C_i^M, C_i^S) over parameters P_i^M and P_i^S for $i = 1, \ldots, n$, if*

i. constraints (C_i^M, C_i^S) are original-constraint consistent,

ii. models M_i form a parametric cover for M_0,

iii. specifications S_i each parametrically imply S_0 and

iv. $C_i, M_i \models^{B_i} C_i^S, S_i$

then $C_0^M, M_0 \models^{B_0} C_0^S, S_0$.

Proof. From the definition of parametric coverage, there exists a partition $\{B_0^1, \ldots, B_0^n\}$ of $[\![C_0^M, M_0]\!]^{B_0}$ s.t.

$$\begin{aligned}
[\![C_0^M, M_0]\!]^{B_0} &\subseteq \bigcup_i R_i^{-1}([\![(C_{aux} \wedge C_0^M)\downarrow_{P_i^M}, M_i]\!]^{B_i}) \\
\text{(Def. 11)} &\subseteq \bigcup_i R_i^{-1}([\![C_i^M, M_i]\!]^{B_i}) \\
(C_i^M, M_i \models^{B_i} C_i^S, S_i) &\subseteq \bigcup_i R_i^{-1}([\![C_i^S, S_i]\!]^{B_i}) \\
\text{(Def. 11)} &\subseteq \bigcup_i R_i^{-1}([\![(C_{aux} \wedge C_0^S)\downarrow_{P_i^S}, S_i]\!]^{B_i}) \\
\text{(Def. 9)} &\subseteq [\![C_0^S, S_0]\!]^{B_0}
\end{aligned}$$

Therefore, $C_0^M, M_0 \models^{B_0} C_0^S, S_0$.

7. EXAMPLE WITH PARAMETERS

To illustrate the use of parametrized models and specifications, we return to the conjunctive analysis example from Sec. 5. The bounds on the POV velocity, the bounds on the SV acceleration, the position of the marker l and the lane width of the major road h are represented as parameters as shown in Fig. 6. These parameters embedded in the unparameterized models are now explicitly identified as follows.

- $P_1^M : \{M_1.\underline{v}_x, M_1.\overline{v}_x, M_1.l, M_1.h, M_1.\underline{a}_y, M_1.\overline{a}_y\}$,
- $P_1^S : \{M_1.h\}$,
- $P_{11}^M : \{M_{11}.\underline{v}_x, M_{11}.\overline{v}_x, M_{11}.l\}$,
- $P_{11}^S : \{M_{11}.l, M_{11}.t_x\}$,

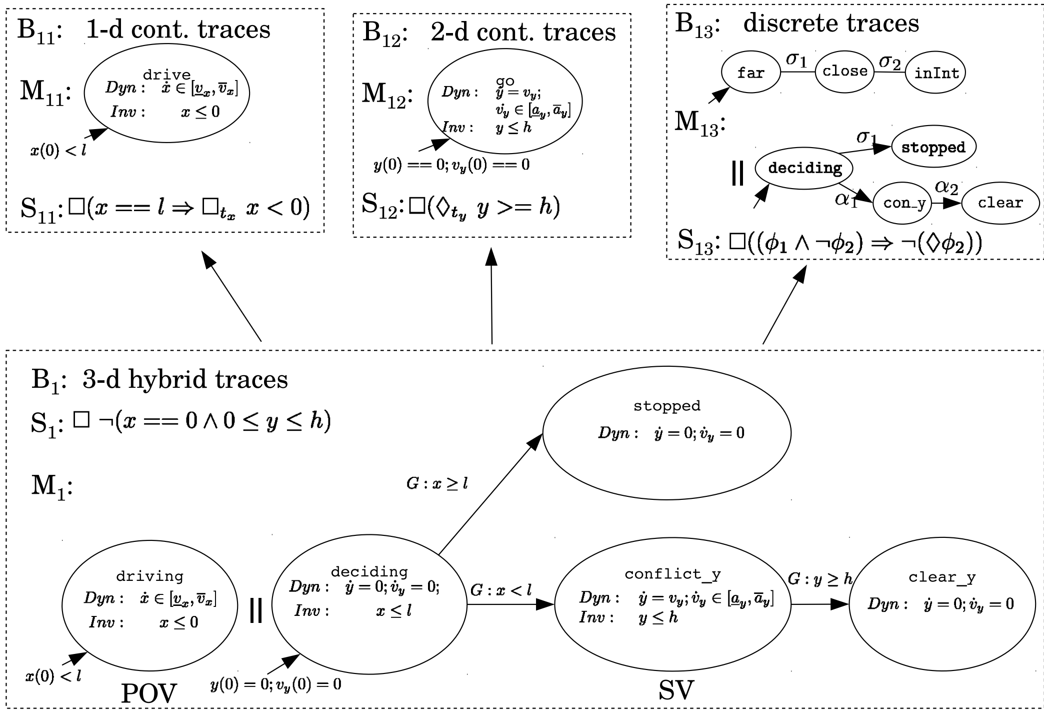

Figure 6: Parametric heterogeneous analysis of CICAS-SSA.

- $P_{12}^M : \{M_{12}.h, M_{12}.\underline{a}_y, M_{12}.\overline{a}_y\}$,
- $P_{12}^S : \{M_{12}.h, M_{12}.t_y\}$,
- $P_{13}^M : \{\}$,
- $P_{13}^S : \{\}$.

The following constraints identify the ranges of these parameters.

- $C_1^M : 20 \le M_1.\underline{v}_x \le M_1.\overline{v}_x \le 30 \wedge M_1.l == -300 \wedge M_1.h == 4.5 \wedge 0.25 \le M_{12}.\underline{a}_y \le M_{12}.\overline{a}_y \le 5$
- $C_1^S : M_1.h == 4.5$
- $C_{11}^M : 18 \le M_{11}.\underline{v}_x \le M_{11}.\overline{v}_x \le 32 \wedge M_{11}.l == -300$
- $C_{11}^S : M_{11}.l == -300 \wedge 9 \le M_{11}.t_x \le 10$
- $C_{12}^M : M_{12}.h == 4.5 \wedge 0.2 \le M_{12}.\underline{a}_y \le M_{12}.\overline{a}_y \le 5.2$
- $C_{12}^S : M_{12}.h == 4.5 \wedge 7 \le M_{12}.t_y \le 8$

Now, we know that the time needed for the POV to get from l to 0 needs to be bigger than the time needed for the SV to start accelerating from a stationary position and clear the intersection (i.e., $t_y < t_x$). From Newton's laws of motion, we note that $\sqrt{\frac{2h}{\underline{a}_y}} \le t_y$ and $t_x \le \frac{-l}{\overline{v}_x}$. We add this to C_{aux} along with the equality constraints between the parameters that are identical between M_{1i}s and M_1:

$$C_{\mathsf{aux}} : \quad (M_1.\underline{v}_x == M_{11}.\underline{v}_x) \wedge \ldots \wedge (M_1.\overline{a}_y == M_{12}.\overline{a}_y) \wedge \\ (\sqrt{\frac{2h}{\underline{a}_y}} \le t_y < t_x \le \frac{-l}{\overline{v}_x})$$

We have a parametric abstraction for each model because due to the equality constraints in C_{aux}, we get equal parameter valuations for the corresponding models, and under the same parameter valuations, M_{1i} are not more restrictive than M_1. Note that we have parametric conjunctive specification implication so long as $t_y < t_x$ holds, and here it does.

$C_{11}^M, M_{11} \models^{B_{11}} C_{11}^S, S_{11}$ and $C_{12}^M, M_{12} \models^{B_{12}} C_{12}^M, S_{12}$ can be shown using Newton's laws so long as $\sqrt{\frac{2h}{\underline{a}_y}} \le t_y$ and $t_x \le \frac{-l}{\overline{v}_x}$ hold, which they do. $C_{13}^M, M_{13} \models^{B_{13}} C_{13}^M, S_{13}$ still holds since it hasn't changed.

Finally, we get the following projections of C_1^M and C_1^S on P_{1i}^M and P_{1i}^S through C_{aux}:

- $(C_1^M \wedge C_{\mathsf{aux}}) \downarrow_{P_{11}^M}$: $20 \le M_{11}.\underline{v}_x \le M_{11}.\overline{v}_x \le 30 \wedge M_{11}.l == -300 \wedge M_{11}.\overline{v}_x < 33.33$
- $(C_1^S \wedge C_{\mathsf{aux}}) \downarrow_{P_{11}^S}$: $\top$
- $(C_1^M \wedge C_{\mathsf{aux}}) \downarrow_{P_{12}^M}$: $0.25 \le M_{12}.\underline{a}_y \le M_{12}.\overline{a}_y \le 5 \wedge M_{12}.h == 4.5 \wedge M_{12}.\underline{a}_y > 0.19$
- $(C_1^S \wedge C_{\mathsf{aux}}) \downarrow_{P_{12}^S}$: $M_{12}.h == 4.5$

We have $(C_1^M \wedge C_{\mathsf{aux}}) \downarrow_{P_{11}^M} \Rightarrow C_{11}^M$, $C_{11}^S \Rightarrow (C_1^S \wedge C_{\mathsf{aux}}) \downarrow_{P_{11}^S}$; and $(C_1^M \wedge C_{\mathsf{aux}}) \downarrow_{P_{12}^M} \Rightarrow C_{12}^M$, $C_{12}^S \Rightarrow (C_1^S \wedge C_{\mathsf{aux}}) \downarrow_{P_{12}^S}$. Now we can use Prop. 4 to turn this into a parametric conjunctive analysis and conclude that $C_1^M, M_1 \models^{B_1} C_1^S, S_1$.

In this parameterized example, because we are able to capture the parameter dependencies, we now know how fast the SV needs to accelerate given ranges of $\overline{v}_x$, h and l. Alternatively, if the system is implemented as a road-side infrastructure-based solution, where $\underline{a}_y$ cannot be chosen but is known empirically from driver behavior data, we know how l should be chosen. While the heterogeneous verification of the unparameterized example succeeds, there is no support for capturing these interdependencies. Therefore, there is value added in exposing parameters and identifying interdependencies.

8. DISCUSSION

This paper addresses the use of heterogeneous models for verifying system-level properties of cyber-physical systems. Behavior relations are introduced to relate the different semantic frameworks used to model different aspects of the system. Structured nesting of verification activities using Boolean combinations of conjunctive and disjunctive constructs is introduced to make it possible to infer system-level properties from the properties of heterogeneous models. The notion of semantic consistency critical for inferring system-level properties from model-level analyses is also introduced based on constraints over parameters.

The application of the proposed approach to real-scale problems will require tool support for managing various behavior relations, parameters, constraints and sufficient conditions for conjunctive and disjunctive analysis constructs. We are currently working on creating this verification management tool support. Future work will focus on integrating structural connectivity information available from architectural modeling of CPS with the semantic information regarding behavior relations and parameter constraints. Another direction is to support dynamic interdependencies between models by using temporal or dynamic logic constraints.

Acknowledgments

The authors gratefully acknowledge helpful discussions of the CICAS application with Prashant Ramachandra and Ken Butts of the Toyota Technical Center, Ann Arbor, MI, and support from NSF Grants CNS 1035800-NSF and CCF-0926181.

9. 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] A. Basu, S. Bensalem, M. Bozga, J. Combaz, M. Jaber, T.-H. Nguyen, and J. Sifakis. Rigorous component-based system design using the BIP framework. *IEEE Software*, 28(3):41–48, 2011.

[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] S. Bliudze and J. Sifakis. The algebra of connectors - structuring interaction in BIP. *IEEE Trans. Computers*, 57(10):1315–1330, 2008.

[10] K. Chaudhari, D. Doligez, L. Lamport, and S. Merz. The TLA+ proof system: building a heterogeneous verification platform. In *International Colloquium on Theoretical Aspects of Computing (ICTAC-7)*, volume LNCS 6256, page 44, Natal, Brazil, 2010.

[11] 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.

[12] 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.

[13] G. Frehse. PHAVer: Algorithmic verification of hybrid systems past HyTech. *International Journal on Software Tools for Technology Transfer (STTT)*, 10(3), 2008.

[14] T. A. Henzinger. The theory of hybrid automata. *Verification of Digital and Hybrid Systems*, 170:296–292, 2000.

[15] 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.

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

[17] R. Kumar, B. H. Krogh, and P. Feiler. An ontology-based approach to heterogeneous verification of embedded control systems. In *Hybrid Systems: Computation and Control*. Springer, 2005.

[18] 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.

[19] 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.

[20] J. Leung, T. Mandl, E. Lee, E. Latronico, C. Shelton, S. Tripakis, and B. Lickly. Scalable semantic annotation using lattice-based ontologies. In *12th International Conference on Model Driven Engineering Languages and Systems*, pages 393–407. ACM/IEEE, October 2009.

[21] A. Rajhans, A. Bhave, S. Loos, B. H. Krogh, A. Platzer, and D. Garlan. Using parameters in architectural views to support heterogeneous design and verification. In *50th IEEE Conference on Decision and Control*, Orlando, Dec 2011.

[22] J. Willems. The behavioral approach to open and interconnected systems. *IEEE Control Systems Magazine*, (6):46–99, 2007.