Relational Abstraction in HybridSAL

Ashish Tiwari

SRI International

Menlo Park

CA 94025

Relational Abstraction: Concept

Consider a dynamical system $(X, \rightarrow)$ where

X :variables defining state space of the system

$\rightarrow$:binary relation over state space defining system dynamics

We do not care if

- the system is discrete- or continuous- or hybrid-time, or
- the system has a discrete, continuous, or hybrid state space

For discrete-time systems, $\rightarrow$ is the one-step transition relation

For continuous-time systems, $\rightarrow = \cup_{t \geq 0} \xrightarrow{t}$ where $\xrightarrow{t}$ is the transition relation corresponding to an elapse of t time units

Relational Abstraction: Concept

Relational abstraction of a dynamical system $(X, \rightarrow)$ is another dynamical system $(X, \rightarrow)$ such that

$$\text{TransitiveClosure}(\rightarrow) \ \subseteq \ \rightarrow$$

Relational Abstraction: An over-approximation of the transitive closure of the transition relation

Benefit:

Eliminates need for iterative fixpoint computation

Useful for proving safety properties, and establishing conservative safety bounds

Relational Abstraction: Example

For the continuous-time continuous-space dynamical system:

$$\frac{dx}{dt} = -x + y$$
$$\frac{dy}{dt} = -x - y$$

we have the following continuous-space discrete-time relational abstraction:

$$(x, y) \to (x', y') \quad := \quad \texttt{max}(|x|, |y|) \geq \texttt{max}(|x'|, |y'|)$$

If initially $x \in [0, 3]$, $y \in [-2, 2]$, then in any future time, x, y will remain in the range $[-3, 3]$

Relational Abstraction: Challenge

Is it possible to compute relational abstractions?

We do not want to abstract discrete-time transition relations, because model checkers (and static analyzers) can handle them (compute fixpoint)

Is it possible to compute relational abstractions of continuous-time dynamics?

Computing Relational Abstractions

We have an algorithm for computing relational abstractions of linear systems

Dynamics	Relational Abstraction
$\dot{x} = 1, \dot{y} = 1$	$x' - x = y' - y$
$\dot{x} = 2, \dot{y} = 3$	$(x' - x)/2 = (y' - y)/3$
$\dot{\vec{x}} = A\vec{x}$	$(0 \leq p' \leq p) \vee (0 \geq p' \geq p)$, where $p = \vec{c}^T\vec{x}$, $\vec{c}$ eigenvector of A^T corr. to negative eigenvalue
$\dot{\vec{x}} = A\vec{x} + \vec{b}$	$\ldots$

Computing Relational Abstractions

For linear systems, we can use plenty of linear algebra to automatically generate relational abstractions

More generally, there is a standard way to generate relational abstractions using constraint solving

The relation $\rightarrow$ is a rel.abs. of $\rightarrow$ if

$$\begin{array}{rcl} \mathtt{Init}(\vec{x}) & \Rightarrow & \vec{x} \rightarrow \vec{x} \\ \vec{x} \rightarrow \vec{x} \ \wedge \ \vec{x'} \rightarrow \vec{y} & \Rightarrow & \vec{x} \rightarrow \vec{y} \end{array}$$

We can search for $\rightarrow$ that satisfies the above two formulas.

We can do so by fixing a form/template for $\rightarrow$

Then, we will get a $\exists\forall$ constraint, whose solution will give us a relational invariant

Note for Linear Systems

The algorithm for creating relational abstractions can be viewed as a special case of the generic method described above, where the $\exists\forall$ problems are being solved using linear algebra tricks.

HybridSAL Implementation of RelAbs

Old HybridSAL:

$$\text{HybridSAL} \overset{\texttt{QualitativeAbstraction}}{\Longrightarrow} \text{SAL}$$

Resulting SAL was finite-state model, could be model checked

Now we can also create relational abstractions from HybridSAL:

$$\text{HybridSAL} \overset{\texttt{RelationalAbstraction}}{\Longrightarrow} \text{SAL}$$

Resulting SAL is infinite-state model, can be infinite bounded model checked