Towards a Language for Multi-Model CPS Integration Properties

Ivan Ruchkin, Bradley Schmerl, David Garlan
Institute for Software Research
Carnegie Mellon University

CPS V&V I&F Workshop
May 12, 2017
Models of Cyber-Physical Systems (CPS)

- Diverse engineering disciplines
- Heterogeneous formalisms
- Disparate levels of abstraction
- Partial overlap of referents
Semantic Gaps in CPS Models

Models of Cyber-Physical Systems (CPS)
- Diverse engineering disciplines
- Heterogeneous formalisms
- Disparate levels of abstraction
- Partial overlap of referents

Challenge: semantic gaps
- Differences in meaning
- Implicit overlaps
- How to express properties of both models?
System Example

Mission: navigate to the goal
Multiple ways to reach the goal
Limited battery capacity
Can recharge at power stations
Planning impl
System Example

map
curLoc
curPlan
vel

M_{pl}

Refines

Planning impl
System Example

$M_{pl}$

Refines

Planning impl

Power data
System Example

\[ M_{pl} \]

\[ M_{po} \]

Refines

Planning impl

Underlies

Power data
System Example

Map: curLoc, curPlan, vel

M_{pl} \rightarrow ? \rightarrow M_{po}

Refines
Planning impl

Underlies
Power data

Battery: curBat, expEnrCost
System Example

Property: “Robot does not run out of power between charging stations”
Relating with architectural views

Problem

How do we specify and check properties of several CPS models?

Requirements for solution

Expressiveness: mixed-model properties, “natural” syntax
Decidability: procedure for automated verification of properties
Semantic modularity: models not dependent on each other
Our Approach In a Nutshell

\[ V_{pl} \xrightarrow{\text{map}} \text{vels} \xrightarrow{\text{curLoc}} \text{curPlan} \xrightarrow{\text{time}} \text{vel} \]

\[ V_{po} \xrightarrow{\text{PowRate}} \text{battery} \xrightarrow{\text{curBat}} \text{expEnrCost} \]
Our Approach In a Nutshell

\[
\begin{align*}
V_{pl} & \xrightarrow{\text{map}} \text{vels} \\
M_{pl} & \xrightarrow{\text{curLoc}} \text{curPlan} \\
& \xrightarrow{\text{time}} \text{vel} \\
V_{po} & \xrightarrow{\text{PowRate}} \text{battery} \\
M_{po} & \xrightarrow{\text{curBat}} \text{expEnrCost}
\end{align*}
\]
Our Approach In a Nutshell

First-order quantification

Linear temporal modalities

\[ V_{pl} \quad \text{map} \quad \text{vels} \]

\[ V_{po} \quad \frac{\text{PowRate}}{\text{battery}} \]

\[ M_{pl} \quad \text{curLoc} \quad \text{curPlan} \quad \text{time} \quad \text{vel} \]

\[ M_{po} \quad \text{curBat} \quad \text{expEnrCost} \]
"For each charging station, there is a speed such that"

$$\forall ch : \text{map.chargeSts} \cdot \forall \nu : \text{vels} \cdot$$

"whenever it is chosen by the planner at that station,"

$$G \ (\text{loc} = ch \land \text{Vel(plan)} = \nu \rightarrow$$

"its power rate, multiplied by the time of the plan, does not exceed the maximum battery capacity."

$$\text{PowRate}(\nu) \times \text{Time(plan)} \leq \text{battery.maxcap}).$$
Integration Property

"For each charging station, there is a speed such that"

$$\forall ch : \text{map.chargeSts} \cdot \forall v : \text{vels} \cdot$$

"whenever it is chosen by the planner at that station,"

$$G (\text{loc} = ch \land \text{Vel(plan)} = v \rightarrow$$

"its power rate, multiplied by the time of the plan, does not exceed the maximum battery capacity."

$$\text{PowRate}(v) \ast \text{Time(plan)} \leq \text{battery.maxcap}.$$
Integration Property Language (IPL)

Syntax

Declarative, one formula = one property
Scope: one (dynamic) model and \( n \) (static) views
Combines symbols from views and the model
Integration Property Language (IPL)

Syntax

- Declarative, one formula = one property
- Scope: one (dynamic) model and $n$ (static) views
- Combines symbols from views and the model

Semantics

- Gives meaning to formulas allowed by the syntax
- Reduces subformulas to either view or model semantics
- Enables a verification algorithm
IPL verification: views, model ⊨ formula
Verification of IPL Formulas

**IPL verification:** views, model \models formula

**Model checking:** for all traces \( \omega \) in a model, determine if \( \omega \models model\_formula \)
IPL verification: views, model $\models$ formula

SMT solving: determine all values of variables $\mu$ s.t. views, $\mu \models$ view_formula($\mu$)

Model checking: for all traces $\omega$ in a model, determine if $\omega \models$ model_formula
Verification of IPL Formulas

**IPL verification:** views, model $\models$ formula

**SMT solving:** determine all values of variables $\mu$ s.t.
views, $\mu \models \text{view\_formula}(\mu)$

**Model checking:** for all traces $\omega$ in a model,
determine if $\omega \models \text{model\_formula}$
Flexible terms/formulas - can change with time

E.g., current position of robot $\text{curPos}$

Only interpretable at model level
Technique: Rigid & Flexible

Flexible terms/formulas - can change with time
  E.g., current position of robot \textit{curPos}
  Only interpretable at model level

Rigid terms/formulas - cannot change with time
  E.g., map of the location \textit{map}
  Interpretable at view level (and sometimes at model level)
Technique: Rigid & Flexible

Flexible terms/formulas - can change with time
E.g., current position of robot \textit{curPos}
Only interpretable at model level

Rigid terms/formulas - cannot change with time
E.g., map of the location \textit{map}
Interpretable at view level (and sometimes at model level)

Syntactic constraint on interleaving models/views
Contributes to semantic modularity
IPL Syntax

Rigid terms
IPL Syntax

Terms

Rigid terms
IPL Syntax

Atomic formulas

Terms

Rigid terms

Atomic rigid formulas
IPL Syntax

IPL formulas

Atomic formulas

Terms

Rigid terms

Atomic rigid formulas

IPL formulas

FORMULA
FORMULA \land FORMULA

\forall VAR : RTERM \cdot QTATOM

\exists TATOM

TATOM \land TATOM

\neg TATOM

X TATOM

\neg ATOM

ATOM \land ATOM

ATOM

STATEVAR

MFUNC(TERM, ... TERM)

BFUNC(TERM, ... TERM)

FORMULA

QTATOM

\forall VAR : RTERM \cdot QTATOM

TATOM

RATOM

RATOM \land RATOM

RTERM

VAR

CONST

ELEM

RTERM,PROP

VRFUNC(RTERM, ... RTERM)

BFUNC(RTERM, ... RTERM)
Application of IPL

Verification algorithm

SMT solver

Model checker

Encoding of property

\[ \forall ch : \text{map.chargeSts} \cdot \forall v : \text{vels} \cdot G (loc = ch \land \text{Vel(plan)} = v \rightarrow \text{PowRate}(v) \cdot \text{Time(plan)} \leq \text{battery.maxcap}) \]
Requirements Revisited

**Expressiveness - provided:**
- IPL expresses behavioral properties of one model
- Multiple views give “shallow” semantics of other models

**Decidability - guaranteed:**
- Decision procedure reduces to other decidable problems

**Semantic modularity - preserved:**
- Models are completely oblivious of views & each other
- Views have no behavioral semantics
When IPL Works Well

When models are implicitly related
When IPL Works Well

When models are implicitly related

When models & views are available
When IPL Works Well

When models are implicitly related

When models & views are available

With behavioral models equivalent to automata

Anything that can be verified against a modal formula
Limitations

**Fundamental: reliant on existing formal methods**

- Satisfiability solvers
- Model checkers
- Theorem provers
Limitations

**Fundamental: reliant on existing formal methods**
- Satisfiability solvers
- Model checkers
- Theorem provers

**Fundamental: reliant on the model-view fwk**
- Soundness and completeness of model-view abstraction
- Soundness of view-view mappings
- Quality of integration with IPL $\leq$ quality of models
Limitations

**Fundamental: reliant on existing formal methods**

Satisfiability solvers
Model checkers
Theorem provers

**Fundamental: reliant on the model-view fwk**

Soundness and completeness of model-view abstraction
Soundness of view-view mappings
Quality of integration with IPL \(\leq\) quality of models

**Incidental: reliant on linear temporal logic**

Other temporal logics possible (e.g., metric, signal, comp. tree)
Other models possible
What is Next?

More expressive behavioral properties
- Integrals, differential/difference equations
- Hybrid/probabilistic models

Implementation for IPL
- Parser
- Verification engine

Application to other systems
- Expressiveness
- Compare performance of verification to other approaches
- Pointers to properties & models welcome!