Rapprochement Between Formal Methods and Control Theory

Richard M. Murray
California Institute of Technology

30 years of the Ramadge-Wonham Theory of Supervisory Control: A Retrospective and Future Perspectives

Conference on Decision and Control
11 December 2017

Outline
• Background: multi-layered architectures, LTL/STL specs for supervisory control
• Review: Synthesis of supervisory controllers for GR(1) specifications
• Example: automated valet parking
• Future: Synthesis of contracts, integration of computer science and controls

Research support by AFOSR, Boeing, DARPA (STARNet), DENSO, IBM and United Technologies Corp.
Layered Approaches to Design

Multi-layer Networked Control System

Model

\[
\dot{x} = f_\alpha(x, u) \\
g_\alpha(x, u, z) \leq 0 \\
\min J = \int_0^T L_\alpha(x, u) \, dt + V(x(T))
\]

Specs

\[
(\phi_{\text{init}} \land \Box \phi_{\text{env}}) \implies \\
(\Box \phi_{\text{safe}} \land \Diamond \phi_{\leq T \phi_{\text{live}}})
\]

\[
y = P_{yu}(s) \, u + P_{yd}(s) \, d \\
\|W(s) \, d(s)\| \leq 1 \\
\|W_1 S + W_2 T\|_\infty < \gamma
\]

\[
\dot{x}^i = f_\alpha(x^i, u^i, d^i) \\
x \in \mathcal{X}, u \in \mathcal{U}, d \in \mathcal{D}
\]

Operating Envelope
Energy Efficiency
Actuator Authority

Specifying Discrete Behavior Using Temporal Logic

Linear temporal logic (LTL)
- $\diamond$ “eventually” - satisfied at some point in the future
- $\square$ “always” - satisfied now and forever into the future
- $\circ$ “next” - true at next step

Signal temporal logic (STL)
- Allow predicates that compare values
- Allow temporal bounds

- $p \rightarrow \diamond q$  p implies eventually q (response)
- $p \rightarrow q U r$  p implies q until r (precedence)
- $\square \diamond p$  always eventually p (progress)
- $\diamond \square p$  eventually always p (stability)
- $\diamond p \rightarrow \diamond q$  eventually p implies eventually q (correlation)
- $V < V_{max}$  $V(t)$ less than threshold ($V_{max}$)
- $\square [t_1, t_2] p$  p true for all time in $[t_1, t_2]$
- $p \rightarrow \diamond [0, t] q$  if p occurs, q will occur w/in time t

Baier and Katoen, Principles of Model Checking, 2007
Synthesis of Reactive Controllers

Reactive Protocol Synthesis

• Find control action that insures that specification is always satisfied
• For LTL, complexity is doubly exponential (!) in the size of system specification

GR(1) synthesis for reactive protocols

• Piterman, Pnueli and Sa’ar, 2006
• Assume environment fixes action before controller (breaks symmetry)
• For certain class of specifications, get complexity cubic in # of states (!)

\[(\phi_{init}^e \land \square \phi_{safe}^e \land \square \diamond \phi_{prog}^e) \rightarrow (\phi_{init}^s \land \square \phi_{safe}^s \land \square \diamond \phi_{prog}^s)\]

Environment assumption System guarantee

• GR(1) = general reactivity formula
• Assume/guarantee style specification

A. Pnueli, 2005
Example: Autonomous Valet Parking - Specification

Layer 3 - supervisory protocol
- Respond to requests to deposit or retrieve a car
- Specification: STL formulas using TLA+ (temporal logic of actions)
- Controller: finite state automata

Layer 2 - trajectory optimization
- Find optimal trajectory minimizing fuel and time + avoid obstacles
- Specification: simplified model + cost function and constraints
- Controller: receding horizon (MPC)

Layer 1 - feedback regulation
- Tracking, disturbance rejection
- Controller: PID w/ gain scheduling
Autonomous Valet Parking - Design and Verification

Layer 3 - supervisory protocol
• Respond to requests to deposit or retrieve a car
• Specification: STL formulas using TLA+ (temporal logic of actions)
• Controller: finite state automata

Layer 2 - trajectory optimization
• Find optimal trajectory minimizing fuel and time + avoid obstacles
• Specification: simplified model + cost function and constraints
• Controller: receding horizon (MPC)

Layer 1 - feedback regulation
• Tracking, disturbance rejection
• Controller: PID w/ gain scheduling
Assume/guarantee contracts

• Assume: properties of other components in the system

• Guarantee: properties that will hold for my component

\[ A_i \Rightarrow G_i \]

\[ G_2 \land G_3 \Rightarrow A_1, G_1 \land G_3 \Rightarrow A_2, \ldots \]

• Contracts can be horizontal (within a layer) or vertical (between two layers)

Synthesis of contracts

• Given a set of (LTL) properties, synthesize GR(1) contracts for components

• Key component is amount of information that must be shared
  - Can minimize subject to constraints

Software (I. Filippidis)

• omega - synthesis of controllers/contracts

• dd - binary decision diagrams in Python
Rapprochement Between Formal Methods and Control Theory

Need to stay rigorous about control of reactive systems
- Systems are too complex to be tested by trial and error
- Systems are too safety-critical to be tested by trial and error
- Move from “design then verify” to
  - specify then synthesize
  - synthesis of contracts

Controlling cyberphysical systems requires solving both problems

\[
\|z\|_2 \leq \gamma \|d\|_2 \quad \text{for all} \quad \|\Delta\| \leq 1
\]

\[
\square \phi^e_{\text{safe}} \land \square \diamond \phi^e_{\text{prog}} \rightarrow \square \phi^s_{\text{safe}} \land \square \diamond \leq T \phi^s_{\text{prog}}
\]