Synthesis Techniques from Discrete Event Systems
in 10 minutes or less...

Stéphane Lafortune

EECS Department
University of Michigan, USA

ExCAPE Kick-off Meeting

4 June 2012
Modeling: Automata

- Event Set $E$
- State space $Q$; set of marked states $Q_m$
- Dynamics:
  - Automaton: $G$, with partial transition function
- Set of trajectories of $G$:
  - Language $\mathcal{L}(G) \subseteq E^*$
  - Marked Language $\mathcal{L}_m(G) \subseteq \mathcal{L}(G)$

Modeling: Petri nets

- Petri net structure: places, transitions, bipartite graph
- Event Set $E$ for transition labels
- State space $X \subseteq \mathbb{N}^n$; set of marked states $X_m$
- Dynamics:
  - Tokens: transition firing rules
- Set of trajectories of $N$:
  - Reachable state space $R(N)$
  - Language $\mathcal{L}(N) \subseteq E^*$
  - Marked Language $\mathcal{L}_m(N) \subseteq \mathcal{L}(N)$

Safety properties: usually expressed as a regular sublanguage of $\mathcal{L}(G)$, $\mathcal{L}(N)$ or as a subset of $R(N)$

Nonblocking properties: avoidance of deadlocks and livelocks

Optimality criterion is set inclusion: Maximal Permissiveness (if it exists)
The Basic Control Problem

Let: \( E = E_c \cup E_{uc} \) and \( E = E_o \cup E_{uo} \)

Given: System: \( G, E_c, E_o \) + Spec: \( \mathcal{L}(H) \subseteq \mathcal{L}(G) \)

Synthesize: Supervisor \( S \) such that \( S/G \) (the “closed-loop” system is:

safe and nonblocking and maximally permissive
Controller must separate *safe* states from *unsafe* states

*Unsafe*: violate safety or nonblocking
→ usually, set of unsafe states must be determined by iterative process

If all events are controllable and observable, $\text{Trim}(H \parallel G)$ suffices ($\parallel$ is parallel composition)

*Supervisory Control Theory* was developed to automate synthesis of $S$ when $E_c \subset E$ and/or $E_o \subset E$ and to identify necessary and sufficient conditions for the existence of $S$ in various control architectures
The Essence of the Problem
The Basic Control Problem: Solution

- **Full Observation:** \( E_o = E \)

  \[
  \mathcal{L}_m(S/G) = [\mathcal{L}(H) \cap \mathcal{L}_m(G)]^{\uparrow C}
  \]

  where \( \uparrow C = \text{supremal controllable operation} \)

- **Partial Observation:** \( E_o \subset E \)

  \[
  \mathcal{L}_m(S/G) = [\mathcal{L}(H) \cap \mathcal{L}_m(G)]^{\uparrow CN}
  \]

  where \( \uparrow CN = \text{supremal controllable normal operation} \)

  or compute a maximal observable and controllable sublanguage

- safe, nonblocking, maximally permissive (if \( E_c \subseteq E_o \))

- \( \uparrow C \): quadratic complexity in \( H \parallel G \)

- \( \uparrow CN \): polynomial complexity in \( \text{deterministic}(H \parallel G) \)
The Basic Control Problem: Solution

- Key features:
  - DE-system-theoretic properties: controllability, observability, normality, coobservability, nonconflictingness, diagnosability
  - Control architectures: monolithic, decentralized-information, horizontal and vertical modularity

- Ramadge & Wonham, SIJCOPT, 1987
  Ramadge & Wonham, Proc. IEEE, 1989;
  Cassandras & Lafortune, 2008 (Chapter 3)
  Wonham’s Notes at http://www.control.utoronto.ca/DES/

- Software tools: TCT (Toronto), DESUMA (Michigan), SUPREMICA (Chalmers), IDES (Queen’s), libFAUDES (Erlangen), DESPOT (McMaster), DESLAB (Rio de Janeiro)
Scalability with Automata

- Modular approaches: $G_1 \ldots G_n$, $H_1 \ldots H_m$, $S_1 \ldots S_r$
  See recent WODES, CDC, J-DEDS, TAC, Automatica, etc.

- Structural approaches:
  - Efficient generation and analysis of reachability graphs of a special class of Petri nets (resource allocation systems)
    Wang et al., WODES 2012 (submitted)
  - Efficient $\uparrow C$ for automata built by abstraction of first-order continuous dynamics
    Dallal et al., CDC 2012 (submitted)
Scalability with Petri Nets

- Deadlock and liveness in *Gadara* Petri nets can be mapped to presence of certain types of *siphons*
  
  → siphons: set of places where input transitions $\subseteq$ output transitions


- Can write *linear inequalities* on state vector that express avoidance of bad siphons

- Supervision Based on Place Invariants (SBPI): control technique for specification expressed as set of linear inequalities
  
  - maximally permissive (if $E_c = E_o = E$)
  - “low” control overhead: one Petri net control place per linear inequality

  *Iordache and Antsaklis, Supervisory Control of Concurrent Systems: A Petri Net Structural Approach, Birkhäuser, 2006*
Gadara nets: model multithreaded code with mutex locks

Control Synthesis Algorithm based on avoidance of bad siphons: ICOG
- maximally permissive even when \( E_c \subset E \)

Bottleneck of ICOG: finding the bad siphons
- MIP approach: scalable to tens of millions of states
- gadara.eecs.umich.edu, POPL 2009, J-DEDS 2012, CDC 2010, CDC 2011
- SAT-based approach: ICOG-SAT scalable to a few billion states
- Standard Dining Philosopher problem: ICOG-SAT can synthesize controller for 2500 philosophers in less than 2 hours
- *Stanley et al., WIP*

### Table: Dining Philosopher Deadlock Prevention Scalability

<table>
<thead>
<tr>
<th>Philosophers</th>
<th>State Size*</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5000</td>
<td>$10^{4515}$</td>
<td></td>
</tr>
<tr>
<td>2500</td>
<td>$10^{2257}$</td>
<td>6932.75</td>
</tr>
<tr>
<td>1000</td>
<td>$10^{903}$</td>
<td>207.16</td>
</tr>
<tr>
<td>500</td>
<td>$10^{451}$</td>
<td>310.41</td>
</tr>
<tr>
<td>250</td>
<td>$10^{225}$</td>
<td>19.16</td>
</tr>
<tr>
<td>100</td>
<td>$10^{90}$</td>
<td>0.61</td>
</tr>
<tr>
<td>50</td>
<td>$10^{45}$</td>
<td>0.07</td>
</tr>
<tr>
<td>10</td>
<td>$10^9$</td>
<td>0.03</td>
</tr>
</tbody>
</table>
Recall: The Essence of the Problem

- SAFE
- UNSAFE
- uncontrollable
- indistinguishable
Minimize the number of linear inequalities to separate safe and unsafe states

- Novel approach based on classification theory and state space pruning
  - maximal safe states and minimal unsafe states
  - exploits linear separability of state spaces of binary vectors

- MIP problem to solve
- At present, scalable to a few million states

Employ SBPI to enforce the linear inequalities
→ results in minimum number of control places

Nazeem, Reveliotis, et al., TAC 2012
Discussion

- Discrete-event model building: how to automate; use of suitable abstractions
- Exploitation of DES Supervisory Control Theory to solve for maximally-permissive controls in the presence of uncontrollable and unobservable
- Exploitation of structure of model for more efficient computation of solution (supremal controllable sublanguage)
  - Special classes of Petri nets
  - Customized algorithms for automata models
- Efficient representation of the synthesized control logic
- Scalability, scalability, scalability