A Symbolic Approach to Control via Approximate Bisimulations

Antoine Girard

Laboratoire Jean Kuntzmann, Université Joseph Fourier
antoine.girard@imag.fr

GT-SDH
Paris, France, June 10 2010
Controller synthesis from high level (temporal logic) specifications:
Controller synthesis from high level (temporal logic) specifications:
### Linear temporal logic (LTL): wide variety of properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>LTL Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Safety</td>
<td>□S (Always S)</td>
</tr>
<tr>
<td>Reachability</td>
<td>♦T (Eventually T)</td>
</tr>
<tr>
<td>Stability</td>
<td>♦(□T)</td>
</tr>
<tr>
<td>Recurrence</td>
<td>□♦T</td>
</tr>
<tr>
<td>Sequencing</td>
<td>♦(T₁ ∧ ♦T₂)</td>
</tr>
<tr>
<td>Coverage</td>
<td>♦T₁ ∧ ♦T₂</td>
</tr>
<tr>
<td>Fault recovery</td>
<td>□(F ⇒ ♦R)</td>
</tr>
</tbody>
</table>

- LTL formula admits an equivalent (Büchi) automaton.
Controller synthesis from high level (temporal logic) specifications:

\[ \dot{x}(t) = f(x(t), u(t)) \]

Physical System:

Temporal Logic Specif.:
Motivation

Controller synthesis from high level (temporal logic) specifications:

\[ \dot{x}(t) = f(x(t), u(t)) \]

The problem is hard because the model and the specification are heterogeneous.
Symbolic (discrete) model that is approximately equivalent to the (continuous) dynamics of the physical system:
Symbolic (discrete) model that is approximately equivalent to the (continuous) dynamics of the physical system:

\[
\dot{x}(t) = f(x(t), u(t))
\]
Symbolic (discrete) model that is approximately equivalent to the (continuous) dynamics of the physical system:

\[ \dot{x}(t) = f(x(t), u(t)) \]

Hybrid Controller:
\[ q(t^+) = g(q(t), x(t)) \]
\[ u(t) = k(q(t), x(t)) \]

Refinement

\[ \approx \]

Physical System:

Symbolic Model:

Discrete Controller:
Symbolic Approach to Control Synthesis

A three step approach to controller synthesis:

1. Computation of a symbolic abstraction of the physical system.
2. Discrete controller synthesis for the symbolic abstraction.
3. Hybrid controller synthesis via discrete controller refinement.

This allows us to leverage discrete controller synthesis techniques:

- Use supervisory control, algorithmic game theory...
- Modular approaches for rich specifications.
- Possibility of optimizing some performance criteria to choose among admissible controllers: dynamic programming, shortest path algorithms, branch and bound, $\alpha - \beta$ pruning...
A three step approach to controller synthesis:
1. Computation of a symbolic abstraction of the physical system.
2. Discrete controller synthesis for the symbolic abstraction.
3. Hybrid controller synthesis via discrete controller refinement.

This allows us to leverage discrete controller synthesis techniques:
- Use supervisory control, algorithmic game theory...
- Modular approaches for rich specifications.
- Possibility of optimizing some performance criteria to choose among admissible controllers: dynamic programming, shortest path algorithms, branch and bound, $\alpha$-$\beta$ pruning...
Correct-by-design embedded control software synthesis

- Full-day workshop on December 14, 2010, Atlanta, USA

- Organizers:
  - Antoine Girard, Université Joseph Fourier
  - Giordano Pola, University of L’Aquila
  - Paulo Tabuada, UCLA

- Program:
  - Discrete controller synthesis.
  - Discrete abstractions of continuous control systems.
  - Refinement of discrete controllers to hybrid controllers and software implementations.
  - Hands-on Pessoa tutorial.
Outline of the Talk

1. Approximation relationships for discrete and continuous systems
   - Approximate bisimulation.
   - Symbolic abstractions of switched systems.

2. Controller synthesis using approximately bisimilar abstractions
   - Synthesis for safety specifications.
   - Synthesis for reachability specifications under time optimization.
Definition

A transition system is a tuple $T = (X, U, \delta, Y, H)$ where

- $X$ is a (discrete or continuous) set of states;
- $U$ is a (discrete or continuous) set of inputs;
- $\delta : X \times U \rightarrow 2^X$ is a transition relation;
- $Y$ is a (discrete or continuous) set of outputs;
- $H : X \rightarrow Y$ is an output map.

$X = \{\text{red, blue, green, yellow}\}$

$U = \{a, b\}$

$Y = \{0, 1, 2\}$
A trajectory of the transition system $T$ is a finite sequence:

$$s = (x_0, u_0), (x_1, u_1), \ldots, (x_{N-1}, u_{N-1}), x_N$$

where $x_{k+1} \in \delta(x_k, u_k), \forall k \in \{0, \ldots, N - 1\}$.

The associated observed trajectory is

$$o = y_0, y_1, \ldots, y_{N-1}, y_N \text{ where } y_k = H(x_k), \forall k \in \{0, \ldots, N\}.$$  

The transition system is said to be deterministic if for all $x \in X$, $u \in U$, $\delta(x, u)$ has zero or one element.

The transition system is said to be discrete or symbolic if $X$ and $U$ are countable or finite. Otherwise, it is said to be uncountable.
Approximate Bisimulation

Let $T_i = (X_i, U, \delta_i, Y, H_i)$, $i \in \{1, 2\}$, be transition systems with a common set of inputs $U$ and outputs $O$ equipped with a metric $d$.

**Definition**

Let $\varepsilon \in \mathbb{R}^+$, a relation $R \subseteq X_1 \times X_2$ is an $\varepsilon$-approximate bisimulation relation if for all $(x_1, x_2) \in R$:

1. $d(H_1(x_1), H_2(x_2)) \leq \varepsilon$;
2. $\forall u \in U, \forall x'_1 \in \delta_1(x_1, u), \exists x'_2 \in \delta_2(x_2, u)$, such that $(x'_1, x'_2) \in R$;
3. $\forall u \in U, \forall x'_2 \in \delta_2(x_2, u), \exists x'_1 \in \delta_1(x_1, u)$, such that $(x'_1, x'_2) \in R$.

**Definition**

$T_1$ and $T_2$ are $\varepsilon$-approximately bisimilar ($T_1 \sim_\varepsilon T_2$) if:

1. For all $x_1 \in X_1$, there exists $x_2 \in X_2$, such that $(x_1, x_2) \in R$;
2. For all $x_2 \in X_2$, there exists $x_1 \in X_1$, such that $(x_1, x_2) \in R$. 
Approximate Bisimulation

\[ d(H_1(x_1), H_2(x_2)) \leq \varepsilon \]

\[ X_2 \]

\[ X_1 \]

\[ R \]

\[ x_1 \]
Approximate Bisimulation

\[ d(H_1(x_1), H_2(x_2)) \leq \varepsilon \]

\( X_2 \)

\( R \)

\( X_1 \)

\( x_1 \)

\( x_2 \)
Approximate Bisimulation

\[ d(H_1(x_1), H_2(x_2)) \leq \varepsilon \]

\[ x_1 \in X_1, x_2 \in X_2, x'_1 \in \delta_1(x_1, u) \]

A. Girard (LJK-UJF)

A Symbolic Approach to Control
Approximate Bisimulation

\[ d(H_1(x_1), H_2(x_2)) \leq \varepsilon \]

\[ x_2' \in \delta_2(x_2, u) \]

\[ x_1' \in \delta_1(x_1, u) \]
Approximate Bisimulation

Proposition

If \( T_1 \sim_\varepsilon T_2 \), then for all trajectories of \( T_1 \), \((x_{0}^{1}, u_{0}), \ldots, (x_{N-1}^{1}, u_{N-1}), x_{N}^{1}\), there exists a trajectory of \( T_2 \), \((x_{0}^{2}, u_{0}), \ldots, (x_{N-1}^{2}, u_{N-1}), x_{N}^{2}\) with the same sequence of inputs, such that

\[
\forall k \in \{0, \ldots, N\}, \quad (x_{k}^{1}, x_{k}^{2}) \in R.
\]

The associated observed trajectories \( y_{0}^{1}, \ldots, y_{N}^{1} \) and \( y_{0}^{2}, \ldots, y_{N}^{2} \) satisfy

\[
\forall k \in \{0, \ldots, N\}, \quad d(y_{k}^{1}, y_{k}^{2}) \leq \varepsilon.
\]

For \( \varepsilon = 0 \), we recover the usual notion of bisimulation relation used in computer science for studying equivalence of discrete systems.
Outline of the Talk

1. Approximation relationships for discrete and continuous systems
   - Approximate bisimulation.
   - Symbolic abstractions of switched systems.

2. Controller synthesis using approximately bisimilar abstractions
   - Synthesis for safety specifications.
   - Synthesis for reachability specifications under time optimization.
A switched system is a tuple $\Sigma = (\mathbb{R}^n, P, \mathcal{F})$ where:

- $\mathbb{R}^n$ is the state space;
- $P = \{1, \ldots, m\}$ is the finite set of modes;
- $\mathcal{F} = \{f_p: \mathbb{R}^n \to \mathbb{R}^n | p \in P\}$ is the collection of vector fields.

For a switching signal $p: \mathbb{R}^+ \to P$, initial state $x \in \mathbb{R}^n$, $x(t, x, p)$ denotes the trajectory of $\Sigma$ given by:

$$\dot{x}(t) = f_{p(t)}(x(t)), \ x(0) = x.$$ 

For $p \in P$, $x(t, x, p)$ denotes the trajectory of $\Sigma$ associated to the constant switching signal $p(t) = p$. 
Consider a switched system $\Sigma = (\mathbb{R}^n, P, \mathcal{F})$ and a time sampling parameter $\tau > 0$.

Let $T_\tau(\Sigma)$ be the transition system where:

- the set of states is $X = \mathbb{R}^n$;
- the set of inputs is $U = P$;
- the transition relation is given by
  \[ x' \in \delta(x, p) \iff x' = x(\tau, x, p); \]
- the set of outputs is $Y = \mathbb{R}^n$;
- the output map $H$ is the identity map over $\mathbb{R}^n$. 

The transition system $T_\tau(\Sigma)$ is uncountable, can we compute a symbolic abstraction?
Consider a switched system $\Sigma = (\mathbb{R}^n, P, \mathcal{F})$ and a time sampling parameter $\tau > 0$.

Let $T_{\tau}(\Sigma)$ be the transition system where:

- the set of states is $X = \mathbb{R}^n$;
- the set of inputs is $U = P$;
- the transition relation is given by
  \[ x' \in \delta(x, p) \iff x' = x(\tau, x, p); \]
- the set of outputs is $Y = \mathbb{R}^n$;
- the output map $H$ is the identity map over $\mathbb{R}^n$.

The transition system $T_{\tau}(\Sigma)$ is uncountable, can we compute a symbolic abstraction?
We start by approximating the set of states $\mathbb{R}^n$ by:

$$[\mathbb{R}^n]_\eta = \left\{ z \in \mathbb{R}^n \mid z_i = k_i \frac{2\eta}{\sqrt{n}}, \, k_i \in \mathbb{Z}, \, i = 1, \ldots, n \right\},$$

where $\eta > 0$ is a state sampling parameter:

$$\forall x \in \mathbb{R}^n, \, \exists z \in [\mathbb{R}^n]_\eta, \, \|x - z\| \leq \eta.$$
Computation of the Symbolic Abstraction

- We start by approximating the set of states $\mathbb{R}^n$ by:

$$ [\mathbb{R}^n]_{\eta} = \left\{ z \in \mathbb{R}^n \mid z_i = k_i \frac{2\eta}{\sqrt{n}}, k_i \in \mathbb{Z}, \ i = 1, \ldots, n \right\}, $$

where $\eta > 0$ is a state sampling parameter:

$$ \forall x \in \mathbb{R}^n, \exists z \in [\mathbb{R}^n]_{\eta}, \|x - z\| \leq \eta. $$

- Approximation of the transition relation = “rounding”:

![Diagram showing the approximation of the transition relation](attachment:image.png)
We define the transition system $T_{\tau,\eta}(\Sigma)$ where:

- the set of states is $X = [\mathbb{R}^n]_\eta$;
- the set of inputs is $U = P$;
- the transition relation is given by
  $$z' \in \delta(z, p) \iff z' = \arg \min_{q \in [\mathbb{R}^n]_\eta} (\|x(\tau, z, p) - q\|).$$
- the set of outputs is $Y = \mathbb{R}^n$;
- the output map is given by $H(z) = z \in \mathbb{R}^n$.

The transition system $T_{\tau,\eta}(\Sigma)$ is discrete and deterministic.
Computation of the Symbolic Abstraction

- We define the transition system $T_{\tau,\eta}(\Sigma)$ where:
  - the set of states is $X = [\mathbb{R}^n]_\eta$;
  - the set of inputs is $U = P$;
  - the transition relation is given by
    $$z' \in \delta(z, p) \iff z' = \arg \min_{q \in [\mathbb{R}^n]_\eta} \|x(\tau, z, p) - q\|.$$  
  - the set of outputs is $Y = \mathbb{R}^n$;
  - the output map is given by $H(z) = z \in \mathbb{R}^n$.

- The transition system $T_{\tau,\eta}(\Sigma)$ is discrete and deterministic.

- Are $T_{\tau}(\Sigma)$ and $T_{\tau,\eta}(\Sigma)$ approximately bisimilar?
**Computation of the Symbolic Abstraction**

- We define the transition system $T_{\tau,\eta}(\Sigma)$ where:
  - the set of states is $X = [\mathbb{R}^n]_\eta$;
  - the set of inputs is $U = P$;
  - the transition relation is given by
    $$z' \in \delta(z, p) \iff z' = \arg\min_{q \in [\mathbb{R}^n]_\eta} (\|x(\tau, z, p) - q\|).$$
  - the set of outputs is $Y = \mathbb{R}^n$;
  - the output map is given by $H(z) = z \in \mathbb{R}^n$.

- The transition system $T_{\tau,\eta}(\Sigma)$ is discrete and deterministic.

- Are $T_{\tau}(\Sigma)$ and $T_{\tau,\eta}(\Sigma)$ approximately bisimilar?

- Yes, if switched system $\Sigma$ is *incrementally stable*. 

Incremental Stability

**Definition**

The switched system $\Sigma$ is *incrementally globally uniformly asymptotically stable* ($\delta$-GUAS) if there exists a $\mathcal{KL}$ function $\beta$ such that for all initial conditions $x_1, x_2 \in \mathbb{R}^n$, for all switching signals $p : \mathbb{R}^+ \rightarrow P$, for all $t \in \mathbb{R}^+$:

$$\|x(t, x_1, p) - x(t, x_2, p)\| \leq \beta(\|x_1 - x_2\|, t) \xrightarrow{t \to +\infty} 0.$$
Definition

\( V : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^+ \) is a common \( \delta \)-GUAS Lyapunov function for \( \Sigma \) if there exist \( K_\infty \) functions \( \alpha, \bar{\alpha} \) and \( \kappa \in \mathbb{R}^+ \) such that for all \( x_1, x_2 \in \mathbb{R}^n \):

\[
\alpha(\|x_1 - x_2\|) \leq V(x_1, x_2) \leq \bar{\alpha}(\|x_1 - x_2\|),
\]

\[
\forall p \in P, \quad \frac{\partial V}{\partial x_1}(x_1, x_2)f_p(x_1) + \frac{\partial V}{\partial x_2}(x_1, x_2)f_p(x_2) \leq -\kappa V(x_1, x_2).
\]
**Definition**

$V : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^+$ is a common $\delta$-GUAS Lyapunov function for $\Sigma$ if there exist $\mathcal{K}_\infty$ functions $\alpha$, $\bar{\alpha}$ and $\kappa \in \mathbb{R}^+$ such that for all $x_1, x_2 \in \mathbb{R}^n$:

$$
\alpha(\|x_1 - x_2\|) \leq V(x_1, x_2) \leq \bar{\alpha}(\|x_1 - x_2\|),
$$

$$
\forall p \in \mathcal{P}, \quad \frac{\partial V}{\partial x_1}(x_1, x_2)f_p(x_1) + \frac{\partial V}{\partial x_2}(x_1, x_2)f_p(x_2) \leq -\kappa V(x_1, x_2).
$$

**Theorem**

*If there exists a common $\delta$-GUAS Lyapunov function, then $\Sigma$ is $\delta$-GUAS.*
δ-GAS Lyapunov Functions

Definition

\( V : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^+ \) is a common \emph{δ-GUAS Lyapunov function} for \( \Sigma \) if there exist \( \mathcal{K}_\infty \) functions \( \alpha, \alpha^- \) and \( \kappa \in \mathbb{R}^+ \) such that for all \( x_1, x_2 \in \mathbb{R}^n \):

\[
\alpha(\|x_1 - x_2\|) \leq V(x_1, x_2) \leq \alpha^-(\|x_1 - x_2\|),
\]

\[
\forall p \in P, \quad \frac{\partial V}{\partial x_1}(x_1, x_2)f_p(x_1) + \frac{\partial V}{\partial x_2}(x_1, x_2)f_p(x_2) \leq -\kappa V(x_1, x_2).
\]

Theorem

If there exists a common \emph{δ-GUAS Lyapunov function}, then \( \Sigma \) is \emph{δ-GUAS}.

Supplementary assumption (true if working on a compact subset of \( \mathbb{R}^n \)):

There exists a \( \mathcal{K}_\infty \) function \( \gamma \) such that

\[
\forall x_1, x_2, x_3 \in \mathbb{R}^n, \quad |V(x_1, x_2) - V(x_1, x_3)| \leq \gamma(\|x_2 - x_3\|).
\]
Approximation Theorem

Theorem

Let us assume that there exists \( V : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^+ \) which is a common \( \delta \)-GUAS Lyapunov function for \( \Sigma \). Consider sampling parameters \( \tau, \eta \in \mathbb{R}^+ \) and a desired precision \( \varepsilon \in \mathbb{R}^+ \). If

\[
\eta \leq \min \left\{ \gamma^{-1} \left( (1 - e^{-\kappa \tau}) \alpha(\varepsilon) \right), \overline{\alpha}^{-1} (\alpha(\varepsilon)) \right\}
\]

then, the relation \( R \subseteq \mathbb{R}^n \times [\mathbb{R}^n]_\eta \) given by

\[
R = \{(x, z) \in \mathbb{R}^n \times [\mathbb{R}^n]_\eta | V(x, z) \leq \alpha(\varepsilon)\}
\]

is an \( \varepsilon \)-approximate bisimulation relation and \( T_{\tau}(\Sigma) \sim_\varepsilon T_{\tau,\eta}(\Sigma) \).
Approximation Theorem

Theorem

Let us assume that there exists $V : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^+$ which is a common $\delta$-GUAS Lyapunov function for $\Sigma$. Consider sampling parameters $\tau, \eta \in \mathbb{R}^+$ and a desired precision $\varepsilon \in \mathbb{R}^+$. If

$$\eta \leq \min \left\{ \gamma^{-1} \left( (1 - e^{-\kappa \tau}) \alpha(\varepsilon) \right), \bar{\alpha}^{-1} (\alpha(\varepsilon)) \right\}$$

then, the relation $R \subseteq \mathbb{R}^n \times [\mathbb{R}^n]_\eta$ given by

$$R = \left\{ (x, z) \in \mathbb{R}^n \times [\mathbb{R}^n]_\eta \mid V(x, z) \leq \alpha(\varepsilon) \right\}$$

is an $\varepsilon$-approximate bisimulation relation and $T_\tau(\Sigma) \sim_\varepsilon T_{\tau, \eta}(\Sigma)$.

Main idea of the proof: show that accumulation of successive “rounding errors” is contained by incremental stability.
Comments on the Approximation Theorem

- For a given time sampling parameter $\tau$, any precision $\varepsilon$ can be achieved by choosing appropriately the state sampling parameter $\eta$ (the smaller $\tau$ or $\varepsilon$, the smaller $\eta$).

- If all vector fields are affine, one can search for a quadratic common $\delta$-GUAS Lyapunov functions by solving a set of LMIs.

- For switched systems that do not admit a common $\delta$-GUAS Lyapunov functions, the result can be extended by using multiple $\delta$-GUAS Lyapunov functions and by imposing a minimum dwell time.

- A similar result applies to incrementally stable continuous control systems.
Example: DC-DC Converter

- Power converter with switching control:

\[
\begin{align*}
\dot{x}(t) &= A_p x(t) + b, \quad p \in \{1, 2\}.
\end{align*}
\]

- State variable: \(x(t) = [i_l(t), v_c(t)]^T\).

- Common \(\delta\)-GUAS Lyapunov function of the form:

\[
V(x, y) = \sqrt{(x - y)^T M (x - y)}.\]
(Useless) symbolic abstraction: $\tau = 0.5, \eta = \frac{1}{40\sqrt{2}} \implies \varepsilon = 2.6.$
Outline of the Talk

1. Approximation relationships for discrete and continuous systems
   - Approximate bisimulation.
   - Symbolic abstractions of switched systems.

2. Controller synthesis using approximately bisimilar abstractions
   - Synthesis for safety specifications.
   - Synthesis for reachability specifications under time optimization.
Definition

Let \( T = (X, U, \delta, Y, H) \), a state-feedback controller for \( T \) is a map \( S : X \rightarrow 2^U \). The dynamics of the controlled system is described by the transition system \( T_S = (X, U, \delta_S, Y, H) \) where the transition relation \( \delta_S \) is given for all \( x \in X, u \in U, x' \in X \) by

\[
x' \in \delta_S(x, u) \iff (u \in S(x) \land x' \in \delta(x, u)).
\]
Controllers for Safety Specifications

Definition

Let $T = (X, U, \delta, Y, H)$, a state-feedback controller for $T$ is a map $S : X \rightarrow 2^U$. The dynamics of the controlled system is described by the transition system $T_S = (X, U, \delta_S, Y, H)$ where the transition relation $\delta_S$ is given for all $x \in X$, $u \in U$, $x' \in X$ by

$$x' \in \delta_S(x, u) \iff (u \in S(x) \land x' \in \delta(x, u)).$$

Definition

Let $Y_s \subseteq Y$ be a set of outputs associated with safe states.
A controller $S$ is safe for specification $Y_s$ if, for all $x_0 \in X$ with $S(x_0) \neq \emptyset$, for all trajectories of $T_S$ starting from $x_0$, $(x_0, u_0), \ldots, (x_{N-1}, u_{N-1}), x_N$, the following conditions hold:

- $\forall k \in \{0, \ldots, N\}, \ H(x_k) \in Y_s$;
- $S(x_N) \neq \emptyset$. 
Maximal Safe Controller

If for all $x \in X$, $S(x) = \emptyset$, then $S$ is safe... We need a notion of “best” safe controller.

Definition

Controller $S_1$ is more permissive than controller $S_2$ ($S_2 \preceq S_1$) if, for all $x \in X$, $S_2(x) \subseteq S_1(x)$.

Definition

$S^*$ is the maximal safe controller for specification $Y_s$ if, $S^*$ is safe and for all safe controllers $S$, $S \preceq S^*$.

The maximal safe controller exists and is unique. It can be determined by fixed point computation of the largest controlled-invariant of $T$, included in $H^{-1}(Y_s)$. 
Maximal Safe Controller

If for all \( x \in X \), \( S(x) = \emptyset \), then \( S \) is safe... We need a notion of “best” safe controller.

**Definition**

Controller \( S_1 \) is *more permissive* than controller \( S_2 \) (\( S_2 \preceq S_1 \)) if, for all \( x \in X \), \( S_2(x) \subseteq S_1(x) \).

**Definition**

\( S^* \) is the *maximal safe controller* for specification \( Y_s \) if, \( S^* \) is safe and for all safe controllers \( S \), \( S \preceq S^* \).
Maximal Safe Controller

If for all $x \in X$, $S(x) = \emptyset$, then $S$ is safe... We need a notion of “best” safe controller.

**Definition**

Controller $S_1$ is *more permissive* than controller $S_2$ ($S_2 \preceq S_1$) if, for all $x \in X$, $S_2(x) \subseteq S_1(x)$.

**Definition**

$S^*$ is the *maximal safe controller* for specification $Y_s$ if, $S^*$ is safe and for all safe controllers $S$, $S \preceq S^*$.

- The maximal safe controller exists and is unique.
- It can be determined by fixed point computation of the *largest controlled-invariant* of $T$, included in $H^{-1}(Y_s)$. 

The controlled-predecessor of $F \subseteq X$ is

$$\text{Pred}(F) = \{x \in X | \exists u \in U, (\delta(x, u) \neq \emptyset) \land (\forall x' \in \delta(x, u), x' \in F)\}.$$ 

$F$ is controlled-invariant if $F \subseteq \text{Pred}(F)$.

**Algorithm**

**Computation of $F^*$, largest controlled-invariant of $T$ included in $H^{-1}(Y_s)$:**

$$F^0 := H^{-1}(Y_s)$$

repeat

$$F^{k+1} := F^k \cap \text{Pred}(F^k)$$

until $F^{k+1} = F^k$

$$F^* := F^k$$
Computation of the Largest Controlled-Invariant

- The *controlled-predecessor* of $F \subseteq X$ is
  \[
  \text{Pred}(F) = \left\{ x \in X \mid \exists u \in U, (\delta(x, u) \neq \emptyset) \land (\forall x' \in \delta(x, u), x' \in F) \right\}.
  \]

- $F$ is *controlled-invariant* if $F \subseteq \text{Pred}(F)$.

**Algorithm**

*Computation of $F^*$, largest controlled-invariant of $T$ included in $H^{-1}(Y_s)$:*

1. $F^0 := H^{-1}(Y_s)$
2. repeat
   - $F^{k+1} := F^k \cap \text{Pred}(F^k)$
3. until $F^{k+1} = F^k$
4. $F^* := F^k$

The algorithm terminates in a finite number of steps for discrete transition systems if $H^{-1}(Y_s)$ is finite. No guarantee of termination for uncountable transition systems.
Computation of the Maximal Safe Controller

**Theorem**

Let $S^* : X \rightarrow 2^U$ be the controller for $T$ defined, for all $x \in X \setminus F^*$, by $S^*(x) = \emptyset$, and for all $x \in F^*$ by

$$S^*(x) = \{ u \in U \mid (\delta(x, u) \neq \emptyset) \land (\forall x' \in \delta(x, u), x' \in F^*) \}.$$  

Then, $S^*$ is the maximal safe controller for the specification $Y_s$. 
Computation of the Maximal Safe Controller

Theorem

Let \( S^* : X \rightarrow 2^U \) be the controller for \( T \) defined, for all \( x \in X \setminus F^* \), by \( S^*(x) = \emptyset \), and for all \( x \in F^* \) by

\[
S^*(x) = \{ u \in U \mid (\delta(x, u) \neq \emptyset) \land (\forall x' \in \delta(x, u), x' \in F^*) \}.
\]

Then, \( S^* \) is the maximal safe controller for the specification \( Y_s \).

A simple example:
Computation of the Maximal Safe Controller

Theorem

Let \( S^* : X \rightarrow 2^U \) be the controller for \( T \) defined, for all \( x \in X \setminus F^* \), by \( S^*(x) = \emptyset \), and for all \( x \in F^* \) by

\[
S^*(x) = \{ u \in U \mid (\delta(x, u) \neq \emptyset) \land (\forall x' \in \delta(x, u), x' \in F^*) \}.
\]

Then, \( S^* \) is the maximal safe controller for the specification \( Y_s \).

A simple example:
Computation of the Maximal Safe Controller

Theorem

Let $S^* : X \rightarrow 2^U$ be the controller for $T$ defined, for all $x \in X \setminus F^*$, by $S^*(x) = \emptyset$, and for all $x \in F^*$ by

$$S^*(x) = \{ u \in U | (\delta(x, u) \neq \emptyset) \land (\forall x' \in \delta(x, u), x' \in F^*) \}.$$ 

Then, $S^*$ is the maximal safe controller for the specification $Y_s$.

A simple example:
Let $S^*: X \rightarrow 2^U$ be the controller for $T$ defined, for all $x \in X \setminus F^*$, by $S^*(x) = \emptyset$, and for all $x \in F^*$ by

$$S^*(x) = \{ u \in U \mid (\delta(x, u) \neq \emptyset) \land (\forall x' \in \delta(x, u), x' \in F^*) \}.$$ 

Then, $S^*$ is the maximal safe controller for the specification $Y_s$.

A simple example:
Computation of the Maximal Safe Controller

**Theorem**

Let $S^* : X \to 2^U$ be the controller for $T$ defined, for all $x \in X \setminus F^*$, by $S^*(x) = \emptyset$, and for all $x \in F^*$ by

$$S^*(x) = \{ u \in U \mid (\delta(x, u) \neq \emptyset) \land (\forall x' \in \delta(x, u), x' \in F^*) \}.$$

Then, $S^*$ is the maximal safe controller for the specification $Y_s$.

A simple example:
Maximal safe controllers are easy to compute for symbolic abstractions...

We need a controller refinement procedure!

**Definition**

Let $Y' \subseteq Y$ and $\phi \geq 0$. The $\phi$-contraction of $Y'$ is the subset of $Y$

$$C_\phi(Y') = \{ y \in Y' | \forall y' \in Y, d(y, y') \leq \phi \Rightarrow y' \in Y' \}.$$ 

**Theorem**

Let $T_1 \sim_\epsilon T_2$, let $R \subseteq X_1 \times X_2$ denote the $\epsilon$-approximate bisimulation relation between $T_1$ and $T_2$. Let $S^*_2, \epsilon$ be the maximal safe controller for $T_2$ for the specification $C_\epsilon(Y_s)$. Let $S_1$ be the controller for $T_1$ given by

$$\forall x_1 \in X_1, S_1(x_1) = \bigcup x_2 \in R(x_1) S^*_2, \epsilon(x_2)$$

where $x_2 \in R(x_1)$ means $(x_1, x_2) \in R$. Then, $S_1$ is safe for specification $Y_s$. 

A. Girard (LJK-UJF)
Maximal safe controllers are easy to compute for symbolic abstractions...
We need a controller refinement procedure!

**Definition**

Let \( Y' \subseteq Y \) and \( \varphi \geq 0 \). The \( \varphi \)-contraction of \( Y' \) is the subset of \( Y \) is

\[ C_\varphi(Y') = \{ y \in Y' | \forall y' \in Y, d(y, y') \leq \varphi \implies y' \in Y' \} \]

**Theorem**

Let \( T_1 \sim_\varepsilon T_2 \), let \( R \subseteq X_1 \times X_2 \) denote the \( \varepsilon \)-approximate bisimulation relation between \( T_1 \) and \( T_2 \). Let \( S_{2,\varepsilon}^* \) be the maximal safe controller for \( T_2 \) for the specification \( C_\varepsilon(Y_s) \). Let \( S_1 \) be the controller for \( T_1 \) given by

\[ \forall x_1 \in X_1, \ S_1(x_1) = \bigcup_{x_2 \in R(x_1)} S_{2,\varepsilon}^*(x_2) \]

where \( x_2 \in R(x_1) \) means \((x_1, x_2) \in R\). Then, \( S_1 \) is safe for specification \( Y_s \).
Sketch of the Proof

Induction step:

- Let $x_1 \in X_1$, such that $S_1(x_1) \neq \emptyset$:
Induction step:

- Let $x_1 \in X_1$, such that $S_1(x_1) \neq \emptyset$:
  - There exists $x_2 \in R(x_1)$ such that $S_{2,\varepsilon}(x_2) \neq \emptyset$. 

  \[ S^*_{2,\varepsilon}(x_2) \neq \emptyset \]
  \[ x_2 \in R(x_1) \text{ gives } d(H_1(x_1), H_2(x_2)) \leq \varepsilon. \]

Then, $H_1(x_1) \in Y_{\text{sys}}$. 

Let $u \in S_1(x_1)$, $x'_1 \in \delta_1(x_1, u)$:

- There exists $x_2 \in R(x_1)$ such that $u \in S_{2,\varepsilon}(x_2)$.

  \[ x_2 \in R(x_1) \text{ gives that there exists } x'_2 \in \delta_2(x_2, u) \text{ such that } x'_2 \in R(x'_1). \]

  \[ u \in S^*_{2,\varepsilon}(x_2) \text{ gives } S^*_{2,\varepsilon}(x'_2) \neq \emptyset. \]

Then, $S_1(x'_1) \neq \emptyset$. 

A. Girard (LJK-UJF)
Sketch of the Proof

Induction step:

- Let $x_1 \in X_1$, such that $S_1(x_1) \neq \emptyset$:
  - There exists $x_2 \in R(x_1)$ such that $S^*_2,\varepsilon(x_2) \neq \emptyset$.
  - $S^*_2,\varepsilon(x_2) \neq \emptyset$ gives $H_2(x_2) \in C_\varepsilon(Y_s)$. 
Sketch of the Proof

Induction step:

- Let $x_1 \in X_1$, such that $S_1(x_1) \neq \emptyset$:
  - There exists $x_2 \in R(x_1)$ such that $S^*_{2,\varepsilon}(x_2) \neq \emptyset$.
  - $S^*_{2,\varepsilon}(x_2) \neq \emptyset$ gives $H_2(x_2) \in C_\varepsilon(Y_s)$.
  - $x_2 \in R(x_1)$ gives $d(H_1(x_1), H_2(x_2)) \leq \varepsilon$. 
Sketch of the Proof

Induction step:

- Let $x_1 \in X_1$, such that $S_1(x_1) \neq \emptyset$:
  - There exists $x_2 \in R(x_1)$ such that $S_{2,\varepsilon}^*(x_2) \neq \emptyset$.
  - $S_{2,\varepsilon}^*(x_2) \neq \emptyset$ gives $H_2(x_2) \in C_\varepsilon(Y_s)$.
  - $x_2 \in R(x_1)$ gives $d(H_1(x_1), H_2(x_2)) \leq \varepsilon$.
  - Then, $H_1(x_1) \in Y_s$. 

(Proof continues with more details on next pages)
Sketch of the Proof

Induction step:

- Let \( x_1 \in X_1 \), such that \( S_1(x_1) \neq \emptyset \):
  - There exists \( x_2 \in R(x_1) \) such that \( S^*_{2,\varepsilon}(x_2) \neq \emptyset \).
  - \( S^*_{2,\varepsilon}(x_2) \neq \emptyset \) gives \( H_2(x_2) \in C_\varepsilon(Y_s) \).
  - \( x_2 \in R(x_1) \) gives \( d(H_1(x_1), H_2(x_2)) \leq \varepsilon \).
  - Then, \( H_1(x_1) \in Y_s \).

- Let \( u \in S_1(x_1) \), \( x'_1 \in \delta_1(x_1, u) \):
Sketch of the Proof

Induction step:

- Let $x_1 \in X_1$, such that $S_1(x_1) \neq \emptyset$:
  - There exists $x_2 \in R(x_1)$ such that $S^*_{2,\varepsilon}(x_2) \neq \emptyset$.
  - $S^*_{2,\varepsilon}(x_2) \neq \emptyset$ gives $H_2(x_2) \in C_\varepsilon(Y_s)$.
  - $x_2 \in R(x_1)$ gives $d(H_1(x_1), H_2(x_2)) \leq \varepsilon$.
  - Then, $H_1(x_1) \in Y_s$.

- Let $u \in S_1(x_1)$, $x'_1 \in \delta_1(x_1, u)$:
  - There exists $x_2 \in R(x_1)$ such that $u \in S^*_{2,\varepsilon}(x_2)$. 


Sketch of the Proof

Induction step:

- Let \( x_1 \in X_1 \), such that \( S_1(x_1) \neq \emptyset \):
  - There exists \( x_2 \in R(x_1) \) such that \( S_{2,\varepsilon}^*(x_2) \neq \emptyset \).
  - \( S_{2,\varepsilon}^*(x_2) \neq \emptyset \) gives \( H_2(x_2) \in C_\varepsilon(Y_s) \).
  - \( x_2 \in R(x_1) \) gives \( d(H_1(x_1), H_2(x_2)) \leq \varepsilon \).
  - Then, \( H_1(x_1) \in Y_s \).

- Let \( u \in S_1(x_1) \), \( x'_1 \in \delta_1(x_1, u) \):
  - There exists \( x_2 \in R(x_1) \) such that \( u \in S_{2,\varepsilon}^*(x_2) \).
  - \( x_2 \in R(x_1) \) gives that there exists \( x'_2 \in \delta_2(x_2, u) \) such that \( x'_2 \in R(x'_1) \).
Sketch of the Proof

Induction step:

- Let $x_1 \in X_1$, such that $S_1(x_1) \neq \emptyset$:
  - There exists $x_2 \in R(x_1)$ such that $S_{2,\varepsilon}(x_2) \neq \emptyset$.
  - $S_{2,\varepsilon}(x_2) \neq \emptyset$ gives $H_2(x_2) \in C_\varepsilon(Y_s)$.
  - $x_2 \in R(x_1)$ gives $d(H_1(x_1), H_2(x_2)) \leq \varepsilon$.
  - Then, $H_1(x_1) \in Y_s$.

- Let $u \in S_1(x_1), x'_1 \in \delta_1(x_1, u)$:
  - There exists $x_2 \in R(x_1)$ such that $u \in S_{2,\varepsilon}(x_2)$.
  - $x_2 \in R(x_1)$ gives that there exists $x'_2 \in \delta_2(x_2, u)$ such that $x'_2 \in R(x'_1)$.
  - $u \in S_{2,\varepsilon}(x_2)$ gives $S_{2,\varepsilon}(x'_2) \neq \emptyset$. 

Sketch of the Proof

Induction step:

- Let $x_1 \in X_1$, such that $S_1(x_1) \neq \emptyset$:
  - There exists $x_2 \in R(x_1)$ such that $S_{2,\varepsilon}(x_2) \neq \emptyset$.
  - $S_{2,\varepsilon}(x_2) \neq \emptyset$ gives $H_2(x_2) \in C_\varepsilon(Y_s)$.
  - $x_2 \in R(x_1)$ gives $d(H_1(x_1), H_2(x_2)) \leq \varepsilon$.
  - Then, $H_1(x_1) \in Y_s$.

- Let $u \in S_1(x_1)$, $x_1' \in \delta_1(x_1, u)$:
  - There exists $x_2 \in R(x_1)$ such that $u \in S_{2,\varepsilon}(x_2)$.
  - $x_2 \in R(x_1)$ gives that there exists $x_2' \in \delta_2(x_2, u)$ such that $x_2' \in R(x_1')$.
  - $u \in S_{2,\varepsilon}(x_2)$ gives $S_{2,\varepsilon}(x_2') \neq \emptyset$.
  - Then, $S_1(x_1') \neq \emptyset$. 
Distance to the Maximal Safe Controller

Theorem

Let $S_1$ be the safe controller for $T_1$ for specification $Y_s$ defined in the previous theorem. Let $S^*_1$ and $S^*_{1,2\varepsilon}$ be the maximal safe controllers for $T_1$ for specifications $Y_s$ and $C_{2\varepsilon}(Y_s)$, respectively. Then,

$$S^*_{1,2\varepsilon} \preceq S_1 \preceq S^*_1.$$
Distance to the Maximal Safe Controller

Theorem

Let $S_1$ be the safe controller for $T_1$ for specification $Y_s$ defined in the previous theorem. Let $S^*_1$ and $S^*_{1,2\varepsilon}$ be the maximal safe controllers for $T_1$ for specifications $Y_s$ and $C_{2\varepsilon}(Y_s)$, respectively. Then,

$$S^*_{1,2\varepsilon} \preceq S_1 \preceq S^*_1.$$ 

Sketch of proof:

$$S^*_{1,2\varepsilon}(q_1) \subseteq S_1(q_1) \subseteq S^*_1(q_1)$$
Distance to the Maximal Safe Controller

Theorem

Let $S_1$ be the safe controller for $T_1$ for specification $Y_s$ defined in the previous theorem. Let $S^*_1$ and $S^*_{1,2\varepsilon}$ be the maximal safe controllers for $T_1$ for specifications $Y_s$ and $C_{2\varepsilon}(Y_s)$, respectively. Then,

$$S^*_{1,2\varepsilon} \subseteq S_1 \subseteq S^*_1.$$  

Sketch of proof:

$$S^*_{1,2\varepsilon}(q_1) \subseteq S^*_{2,\varepsilon}(q_2) \subseteq S^*_1(q_1)$$

$$S_1(q_1) = \bigcup_{q_2 \in R(q_1)} S^*_{1,2\varepsilon}(q_1)$$
Distance to the Maximal Safe Controller

Theorem

Let $S_1$ be the safe controller for $T_1$ for specification $Y_s$ defined in the previous theorem. Let $S^*_1$ and $S^*_{1,2\varepsilon}$ be the maximal safe controllers for $T_1$ for specifications $Y_s$ and $C_{2\varepsilon}(Y_s)$, respectively. Then,

$$S^*_{1,2\varepsilon} \preceq S_1 \preceq S^*_1.$$

Sketch of proof:

$$S^*_{1,2\varepsilon}(q_1) \subseteq \tilde{S}_1(q_1) = \bigcup_{q_2 \in R(q_1)} S_{2,\varepsilon}(q_2) \subseteq S_1(q_1) = \bigcup_{q_2 \in R(q_1)} S^*_{2,\varepsilon}(q_2) \subseteq S^*_1(q_1)$$
Example: Safe Controller for the DC-DC Converter

- Abstraction parameters: $\tau = 0.5$, $\eta = 17 \times 10^{-5} \implies \varepsilon = 25 \times 10^{-3}$.
- $Y_s = [1.275, 1.525] \times [5.625, 5.775] \implies C_\varepsilon(Y_s) = [1.3, 1.5] \times [5.65, 5.75]$.
- The symbolic abstraction has 337431 states, the synthesis algorithm terminates in 5 iterations.
The synthesized controller is non-deterministic.
Several implementations of the controller are possible.
Possibility to ensure a posteriori secondary control objective.
Outline of the Talk

1. Approximation relationships for discrete and continuous systems
   - Approximate bisimulation.
   - Symbolic abstractions of switched systems.

2. Controller synthesis using approximately bisimilar abstractions
   - Synthesis for safety specifications.
   - Synthesis for reachability specifications under time optimization.
The goal is to steer the state of the system to a desired target while keeping the system safe along the way.

**Definition**

Let $T = (X, U, \delta, Y, H)$ and $S : X \to 2^U$ be a controller for $T$. Let $Y_s \subseteq Y$ be a set of outputs associated with safe states and $Y_t \subseteq Y_s$ be a set of outputs associated with target states.

The *entry time of $T_S$ from $x_0 \in X$ for specification $(Y_s, Y_t)$* is the smallest $N \in \mathbb{N}$ such that for all trajectories of $T_S$ of length $N$ and starting from $(x_0, u_0), \ldots, (x_{N-1}, u_{N-1}), x_N$, there exists $K \in \{0, \ldots, N\}$ such that:

- $\forall k \in \{0, \ldots, K\}, \ H(x_k) \in Y_s$;
- $H(x_K) \in Y_t$.

The entry time is denoted by $J(T_S, Y_s, Y_t, x_0)$. If such $N$ does not exist, then we define $J(T_S, Y_s, Y_t, x_0) = +\infty$. 

The control objective is to minimize the entry time.

**Definition**

We say that a controller $S^*$ is *time-optimal* for specification $(Y_s, Y_t)$ if, for all controllers $S$:

$$\forall x \in X, \ J(T_{S^*}, Y_s, Y_t, x) \leq J(T_S, Y_s, Y_t, x).$$

We define the *value function* of the time-optimal control problem as

$$J^*(T, Y_s, Y_t, x) = J(T_{S^*}, Y_s, Y_t, x).$$
The control objective is to minimize the entry time.

**Definition**

We say that a controller $S^*$ is *time-optimal* for specification $(Y_s, Y_t)$ if, for all controllers $S$:

$$\forall x \in X, \quad J(T_{S^*}, Y_s, Y_t, x) \leq J(T_S, Y_s, Y_t, x).$$

We define the *value function* of the time-optimal control problem as $J^*(T, Y_s, Y_t, x) = J(T_{S^*}, Y_s, Y_t, x)$.

- There exists a time-optimal controller (may be not unique).
- It can be determined by dynamic programming and fixed point computation of the value function of the time-optimal control problem.
Computation of the Value Function

Algorithm

Computation of the value function $J^*(T, Y_s, Y_t, x)$:

\[ F^0 := H^{-1}(Y_t) \]
\[ \forall x \in F^0, \ J^*(T, Y_s, Y_t, x) := 0 \]

repeat

\[ F^{k+1} := F^k \cup (\text{Pred}(F^k) \cap H^{-1}(Y_s)) \]
\[ \forall x \in F^{k+1} \setminus F^k, \ J^*(T, Y_s, Y_t, x) := k + 1 \]

until $F^{k+1} = F^k$

\[ \forall x \in X \setminus F^k, \ J^*(T, Y_s, Y_t, x) := +\infty \]
Computation of the Value Function

Algorithm

Computation of the value function $J^*(T, Y_s, Y_t, x)$:

$$F^0 := H^{-1}(Y_t)$$

$$\forall x \in F^0, \ J^*(T, Y_s, Y_t, x) := 0$$

repeat

$$F^{k+1} := F^k \cup (\text{Pred}(F^k) \cap H^{-1}(Y_s))$$

$$\forall x \in F^{k+1} \setminus F^k, \ J^*(T, Y_s, Y_t, x) := k + 1$$

until $F^{k+1} = F^k$

$$\forall x \in X \setminus F^k, \ J^*(T, Y_s, Y_t, x) := +\infty$$

- $F^k$ is the set of states from which the system can reach the target in at most $k$ transitions while remaining in the safe set.
Computation of the Value Function

**Algorithm**

**Computation of the value function** $J^*(T, Y_s, Y_t, x)$:

\[
F^0 := H^{-1}(Y_t) \\
\forall x \in F^0, \quad J^*(T, Y_s, Y_t, x) := 0
\]

repeat

\[
F^{k+1} := F^k \cup (\operatorname{Pred}(F^k) \cap H^{-1}(Y_s))
\]

\[
\forall x \in F^{k+1} \setminus F^k, \quad J^*(T, Y_s, Y_t, x) := k + 1
\]

until $F^{k+1} = F^k$

\[
\forall x \in X \setminus F^k, \quad J^*(T, Y_s, Y_t, x) := +\infty
\]

- $F^k$ is the set of states from which the system can reach the target in at most $k$ transitions while remaining in the safe set.

- The algorithm terminates in a finite number of steps for discrete transition systems if $H^{-1}(Y_s)$ is finite. No guarantee of termination for infinite transition systems.
Computation of a Time-Optimal Controller

**Theorem**

Let $S^* : X \to 2^U$ be the controller for $T$ defined, for all $x \in X$ by

$$S^*(x) = \arg\min_{u \in U} \left( \max_{x' \in \delta(x,u)} J^*(T, Y_s, Y_t, x') \right).$$

Then, $S^*$ is a time-optimal controller for the specification $(Y_s, Y_t)$. 

---

A simple example:
Computation of a Time-Optimal Controller

Theorem

Let $S^* : X \rightarrow 2^U$ be the controller for $T$ defined, for all $x \in X$ by

$$S^*(x) = \arg \min_{u \in U} \left( \max_{x' \in \delta(x,u)} J^*(T, Y_s, Y_t, x') \right).$$

Then, $S^*$ is a time-optimal controller for the specification $(Y_s, Y_t)$.

A simple example:
Theorem

Let $S^*: X \rightarrow 2^U$ be the controller for $T$ defined, for all $x \in X$ by

$$S^*(x) = \arg\min_{u \in U} \left( \max_{x' \in \delta(x,u)} J^*(T, Y_s, Y_t, x') \right) .$$

Then, $S^*$ is a time-optimal controller for the specification $(Y_s, Y_t)$.

A simple example:
Computation of a Time-Optimal Controller

**Theorem**

Let $S^* : X \rightarrow 2^U$ be the controller for $T$ defined, for all $x \in X$ by

$$S^*(x) = \arg\min_{u \in U} \left( \max_{x' \in \delta(x,u)} J^*(T, Y_s, Y_t, x') \right).$$

Then, $S^*$ is a time-optimal controller for the specification $(Y_s, Y_t)$.

A simple example:

![Graph representation of a simple example](image-url)
Computation of a Time-Optimal Controller

**Theorem**

Let $S^* : X \rightarrow 2^U$ be the controller for $T$ defined, for all $x \in X$ by

$$S^*(x) = \arg\min_{u \in U} \left( \max_{x' \in \delta(x,u)} J^*(T, Y_s, Y_t, x') \right).$$

Then, $S^*$ is a time-optimal controller for the specification $(Y_s, Y_t)$.

A simple example:
Computation of a Time-Optimal Controller

**Theorem**

Let $S^* : X \rightarrow 2^U$ be the controller for $T$ defined, for all $x \in X$ by

$$S^*(x) = \arg\min_{u \in U} \left( \max_{x' \in \delta(x,u)} J^*(T, Y_s, Y_t, x') \right).$$

Then, $S^*$ is a time-optimal controller for the specification $(Y_s, Y_t)$.

A simple example:
Computation of a Time-Optimal Controller

Theorem

Let $S^*: X \rightarrow 2^U$ be the controller for $T$ defined, for all $x \in X$ by

$$S^*(x) = \arg \min_{u \in U} \left( \max_{x' \in \delta(x, u)} J^*(T, Y_s, Y_t, x') \right).$$

Then, $S^*$ is a time-optimal controller for the specification $(Y_s, Y_t)$.

A simple example:

- A. Girard (LJK-UJF)

A Symbolic Approach to Control
Theorem

Let $S^*: X \rightarrow 2^U$ be the controller for $T$ defined, for all $x \in X$ by

$$S^*(x) = \arg\min_{u \in U} \left( \max_{x' \in \delta(x,u)} J^*(T, Y_s, Y_t, x') \right).$$

Then, $S^*$ is a time-optimal controller for the specification $(Y_s, Y_t)$.

A simple example:
Suboptimal Controller Synthesis via Symbolic Abstractions

Time-optimal controllers are easy to compute for symbolic abstractions… We need a controller refinement procedure!

Theorem

Let $T_1 \sim_\epsilon T_2$, let $R \subseteq X_1 \times X_2$ denote the $\epsilon$-approximate bisimulation relation between $T_1$ and $T_2$. Let $S_{2,\epsilon}^*$ be a time-optimal controller for $T_2$ for the specification $(C_{\epsilon}(Y_s), C_{\epsilon}(Y_t))$. Let $S_1$ be the controller for $T_1$ given by

$$\forall x_1 \in X_1, S_1(x_1) = S_{2,\epsilon}^* \left( \arg \min_{x_2 \in R(x_1)} J^*(T_2, C_{\epsilon}(Y_s), C_{\epsilon}(Y_t), x_2) \right).$$

The entry time of $T_{S_1}$ for specification $(Y_s, Y_t)$ satisfies for all $x_1 \in X_1$:

$$J^*(T_1, Y_s, Y_t, x_1) \leq J(T_1, S_1, Y_s, Y_t, x_1) \leq J^*(T_1, C_{2\epsilon}(Y_s), C_{2\epsilon}(Y_t), x_1).$$

Proof is close to the case of safety controllers.
Example: Suboptimal Controller for the DC-DC Converter

- Abstraction parameters: $\tau = 0.5$, $\eta = 13 \times 10^{-4} \implies \epsilon = 0.2$.
- $Y_s = [0.65, 1.65] \times [4.95, 5.95] \implies C_\epsilon(Y_s) = [0.85, 1.45] \times [5.15, 5.75]$.
- $Y_t = [1.1, 1.6] \times [5.4, 5.9] \implies C_\epsilon(Y_t) = [1.3, 1.4] \times [5.6, 5.7]$.
- The symbolic abstraction has 94249 states, the synthesis algorithm terminates in 237 iterations.
Example: Suboptimal Controller for the DC-DC Converter

- Suboptimal controller computed using the refinement procedure.
- Controller $S_1$ seems to be more “regular” than $S_{2,\varepsilon}$.
- Entry time ranges from 0 to 94 when $J^*(T_2, C_\varepsilon(Y_s), C_\varepsilon(Y_t), x_2)$ range from 0 to 237.
Conclusions

- Approximately bisimilar symbolic abstractions:
Conclusions

- Approximately bisimilar symbolic abstractions:
  - A rigorous tool for controller synthesis: controllers are “correct by design” with bounds on the distance to optimality...

- Allows to leverage efficient algorithmic techniques from discrete systems to continuous and hybrid systems.
- Computable for interesting classes of systems: switched systems, continuous control systems...
- Incremental stability needed for approximate bisimulation.
- Ongoing and future work:
  - Multiscale and adaptive symbolic models.
  - On the fly computation of symbolic models.
  - Controller synthesis for other type of specifications.
  - Complexity reduction of synthesized controllers.
Approximately bisimilar symbolic abstractions:

- A rigorous tool for controller synthesis: controllers are “correct by design” with bounds on the distance to optimality...
- Allows to leverage efficient algorithmic techniques from discrete systems to continuous and hybrid systems.
Conclusions

- Approximately bisimilar symbolic abstractions:
  - A rigorous tool for controller synthesis: controllers are “correct by design” with bounds on the distance to optimality...
  - Allows to leverage efficient algorithmic techniques from discrete systems to continuous and hybrid systems.
  - Computable for interesting classes of systems: switched systems, continuous control systems...

Ongoing and future work:
- Multiscale and adaptive symbolic models.
- On the fly computation of symbolic models.
- Controller synthesis for other type of specifications.
- Complexity reduction of synthesized controllers.
Conclusions

- Approximately bisimilar symbolic abstractions:
  - A rigorous tool for controller synthesis: controllers are “correct by design” with bounds on the distance to optimality...
  - Allows to leverage efficient algorithmic techniques from discrete systems to continuous and hybrid systems.
  - Computable for interesting classes of systems: switched systems, continuous control systems...
  - Incremental stability needed for approximate bisimulation.
Conclusions

- Approximately bisimilar symbolic abstractions:
  - A rigorous tool for controller synthesis: controllers are “correct by design” with bounds on the distance to optimality...
  - Allows to leverage efficient algorithmic techniques from discrete systems to continuous and hybrid systems.
  - Computable for interesting classes of systems: switched systems, continuous control systems...
  - Incremental stability needed for approximate bisimulation.

- Ongoing and future work:
Conclusions

- Approximately bisimilar symbolic abstractions:
  - A rigorous tool for controller synthesis: controllers are “correct by design” with bounds on the distance to optimality...
  - Allows to leverage efficient algorithmic techniques from discrete systems to continuous and hybrid systems.
  - Computable for interesting classes of systems: switched systems, continuous control systems...
  - Incremental stability needed for approximate bisimulation.

- Ongoing and future work:
  - Multiscale and adaptive symbolic models.
Conclusions

- Approximately bisimilar symbolic abstractions:
  - A rigorous tool for controller synthesis: controllers are “correct by design” with bounds on the distance to optimality...
  - Allows to leverage efficient algorithmic techniques from discrete systems to continuous and hybrid systems.
  - Computable for interesting classes of systems: switched systems, continuous control systems...
  - Incremental stability needed for approximate bisimulation.

- Ongoing and future work:
  - Multiscale and adaptive symbolic models.
  - On the fly computation of symbolic models.
Conclusions

- Approximately bisimilar symbolic abstractions:
  - A rigorous tool for controller synthesis: controllers are “correct by design” with bounds on the distance to optimality...
  - Allows to leverage efficient algorithmic techniques from discrete systems to continuous and hybrid systems.
  - Computable for interesting classes of systems: switched systems, continuous control systems...
  - Incremental stability needed for approximate bisimulation.

- Ongoing and future work:
  - Multiscale and adaptive symbolic models.
  - On the fly computation of symbolic models.
  - Controller synthesis for other type of specifications.
Conclusions

- Approximately bisimilar symbolic abstractions:
  - A rigorous tool for controller synthesis: controllers are “correct by design” with bounds on the distance to optimality...
  - Allows to leverage efficient algorithmic techniques from discrete systems to continuous and hybrid systems.
  - Computable for interesting classes of systems: switched systems, continuous control systems...
  - Incremental stability needed for approximate bisimulation.

- Ongoing and future work:
  - Multiscale and adaptive symbolic models.
  - On the fly computation of symbolic models.
  - Controller synthesis for other type of specifications.
  - Complexity reduction of synthesized controllers.
References

- Approximation relationships for systems:

- Computation of approximately bisimilar abstractions:

- Synthesis using approximately bisimilar abstractions: