(Towards) Translating Temporal Logic to Controller Specifications

Georgios E. Fainekos, Savvas G. Loizou and George J. Pappas

GRASP Lab
Departments of CIS, MEAM and ESE
University of Pennsylvania
Motivation – Motion Planning
Task: “Stay always in $\pi_0$ and visit area $\pi_2$, then area $\pi_3$, then area $\pi_4$ and, finally, return to and stay in region $\pi_1$ while avoiding areas $\pi_2$ and $\pi_3$”

RTL: $\square \pi_0 \land (\pi_2 \land (\pi_3 \land (\pi_4 \land (\neg \pi_2 \land \neg \pi_3) \cup \pi_1)))$
Problem Definition

Temporal Logic Controller Synthesis
Given a dynamical system $\Sigma$, a set of initial conditions $X_0$ and a flat RTL formula $\varphi$ over $\Pi$, construct a closed-loop system in the form of a hybrid automaton $H_\varphi$ such that the resulting system trajectories $x(t)$ starting at some point $x(0) \in X_0$ satisfy the formula $\varphi$.

$$\Sigma: \dot{x}(t) = f(x(t), u(t)) \quad x(t) \in X \subseteq \mathbb{R}^n \quad u(t) \in U \subseteq \mathbb{R}^n$$
In the past ...

Approaches to synthesis of hybrid systems:

- **Given controllers + switching rules** $\implies$ verify that HS satisfies spec
- **Given controllers + spec** $\implies$ keep necessary controllers
- **Given closed-loop dynamics + spec** $\implies$ design switching rules

**Researchers:** Alur, Henzinger, Tabuada, Davoren, Moor, Koutsoukos, Antsaklis, Stiver, Lemmon, Tomlin, Lygeros, Sastry, Habets, van Schuppen, Asarin, Bournez, Dang, Maler, Pnueli, Bemporad, Morari, Giorgetti, Lafferriere, Kloetzer, Belta, Kyriakopoulos, Kress-Gazit, Loizou, Pappas, Fainekos and many others ...

- **Given spec** $\implies$ design controllers
In this presentation ...

A top-down approach:

Input: Software Specification

Algorithm

Hybrid Automaton

(List of Controllers)

1

2

Tomlin, Lygeros, Sastry, Habets, van Schuppen, Bemporad, Morari, Conner, Rizzi, Choset, Burridge, Koditschek, Koutsoukos, Antsaklis, Stiver, Lemmon, many other

Your favorite control engineer

List of Controllers

{Init, Inv, Goal}
A hybrid automaton is a tuple

\[ H = (X, V, E, \text{Inv}, \text{Flow}, \text{Init}, \text{Guard}, F) \]

where

- \( X \) is the state space of the system \( \Sigma \)
- \( V \) is the set of control locations,
- \( E \subseteq V \times V \) is the set of control switches
- \( \text{Inv} : V \rightarrow P(X) \) assigns an invariant set to each location
- \( \text{Flow} : V \times X \rightarrow P(\mathbb{R}^n) \) constraints the time derivative of the continuous part of the state
- \( \text{Init} : V \rightarrow P(X) \) assigns to each control location a set of initial conditions,
- \( \text{Guard} : E \rightarrow P(X) \) is the guard condition that enables a control switch \( e \) in \( E \)
- \( F \subseteq V \) is the set of final locations
Trajectories of the Hybrid Automaton

Formally, the semantics of a hybrid automaton are given in terms of timed transition systems $T_H = (H, H_0, \rightarrow)$.

Note: the reset map is the identity, transitions are forced.
Trajectories of the Hybrid Automaton

Formally, the semantics of a hybrid automaton are given in terms of timed transition systems $T_H = (H, H_0, \rightarrow)$.
Language of the timed transition system

The set of all trajectories $\eta$ of $T_H$ starting from a state in $H_0$ is the language $L(T_H)$ of the timed transition system $T_H$. 

$H_0$ $\eta : \mathbb{R}^+ \rightarrow X$
Linear Temporal Logic: Continuous Time

Like propositional logic, but also reasoning wrt time ...

**Basic operators:**
- Propositional: $\land$, $\lor$, $\neg$
- Temporal: $\Diamond$, $\Box$, $U$, $R$

$\Diamond$(red)
Eventually red

$\Box$(gray)
Always gray

(gray) $U$ (red)
gray Until red
Flat LTL with continuous time semantics

Let $\Pi$ be a set of atomic propositions and $\Pi^*$ be the set of Boolean combinations of atomic propositions, i.e. $(\pi_1 \land \pi_2) \lor \neg \pi_3$.

Define the denotation $[[.]] : \Pi \rightarrow P(X)$ which extends naturally over $\Pi^*$ as:

$[[\pi_1 \land \pi_2]] = [[\pi_1]] \cap [[\pi_2]]$ and $[[\neg \pi]] = [[\pi]]^c$

Syntax in NNF:

$\phi ::= \pi^* \mid \phi_1 \land \phi_2 \mid \phi_1 \lor \phi_2 \mid \pi^* U \phi_2 \mid \pi^* R \phi_2$

Semantics:

$\eta \models \pi^*$ iff $\eta(0) \in [[\pi^*]]$

$\eta \models \phi_1 \land \phi_2$ iff $\eta \models \phi_1$ and $\eta \models \phi_2$

$\eta \models \phi_1 \lor \phi_2$ iff $\eta \models \phi_1$ or $\eta \models \phi_2$

$\eta \models \pi^* U \phi$ iff $\exists t \geq 0$ s.t. $\eta|_t \models \phi_2$ and $\forall s \in [0, t] \eta(s) \in [[\pi^*]]$

$\eta \models \pi^* R \phi$ iff $\forall t \geq 0 \eta|_t \models \phi_2$ or $\exists s \in [0, t]$ s.t. $\eta(s) \in [[\pi^*]]$

Some notation:

$\eta|_t(s) = \eta(t+s)$

$\Diamond \phi = TU \phi$

$LTL(op_1, ..., op_n)$

$\Box \phi = FR \phi$
Modular construction using the structure of $\varphi$.
(≡ proof by induction)

Syntax in NNF: $\varphi ::= \pi^* \mid \varphi_1 \land \varphi_2 \mid \varphi_1 \lor \varphi_2 \mid \pi^* U \varphi$

Ex. $\Diamond (\pi_2 \land \boxtimes (\pi_3 \land (\pi_4 \land (\neg \pi_2 \land \neg \pi_3) U \pi_1))))$
Proposition 2 (base case): Let $\varphi = \pi$, then

$$H_{\pi^*} = (X, \{v\}, \emptyset, X, \mathbb{R}^n, [[\pi^*]], \emptyset, \{v\})$$

proof immediate from definition: $\eta \models \pi^*$ iff $\eta(0) \in [[\pi^*]]$
Basic Building Blocks

$\varphi = \varphi_1 \land \varphi_2$ or $\varphi = \varphi_1 \lor \varphi_2$

**Proposition 1 [Henzinger 95]:**
Let $H_1$ and $H_2$ be hybrid automata.
If $H = H_1 \cup H_2$, then $L(T_H) = L(T_{H_1}) \cup L(T_{H_2})$.
If $H = H_1 \cap H_2$, then $L(T_H) = L(T_{H_1}) \cap L(T_{H_2})$. 
LTL(U, V, ∧) to HA

**Proposition 3:** Let \( \varphi = \pi^* U \psi \), then

\[
H_{\varphi} = (X, V \cup \{v\}, E', \text{Inv}', \text{Flow}', \text{Init}', \text{Guard}', F)
\]

Let \( H_{\psi} = (X, V, E, \text{Inv}, \text{Flow}, \text{Init}, \text{Guard}, F) \). Then:

- \( \text{Inv}'(v') = \text{Init}'(v') = [\langle \pi^* \rangle] \) and \( \text{Flow}(v') = \mathbb{R}^n \)
- \( \text{Inv}'(v) = \text{Inv}(v) \) for all \( v \in V \)
- \( \text{Flow}'(v) = \text{Flow}(v) \) for all \( v \in V \)
- \( \text{Init}'(v) = \text{Init}(v) \cap [\langle \pi^* \rangle] \) for all \( v \in V \)
- \( E' = E \cup \{(v', v) \mid v \in V_{in}\} \)
- \( \text{Guard}'(v', v) = \text{Init}'(v) \) for all \( v \in V_{in} \)
- \( \text{Guard}'(e) = \text{Guard}(e) \) for all \( e \in E \)

where \( V_{in} = \{v \in V \mid \text{Init}(v) \cap [\langle \pi^* \rangle] \neq \emptyset\} \)

Proof immediate from definition:

\[
\eta \models \pi^* \psi \iff \exists t \geq 0 \text{ s.t. } \eta|_t \models \psi \text{ and } \forall s \in [0, t] \eta(s) \models [\langle \pi^* \rangle]
\]
LTL(\(U, \land, \lor\)) to HA

**Algorithm 1** The LTL(\(U, \land, \lor\)) Fragment

**Input:** A formula \(\phi \in \text{LTL}(\mathcal{U}, \land, \lor)\)

**Output:** The hybrid automaton \(\mathcal{H}_\phi\)

1: **procedure** LTL\(_{\mathcal{U}}\)TOHA(\(\phi\))
2: \[\text{if } \phi = \overline{\pi} \text{ then} \]
3: \[\text{return } \mathcal{H}_{\overline{\pi}} \quad \triangleright \text{ Proposition 2} \]
4: \[\text{else if } \phi = \phi_1 \land \phi_2 \text{ then} \]
5: \[\text{return } \text{LTL}_{\mathcal{U}}\text{TOHA}(\phi_1) \cap \text{LTL}_{\mathcal{U}}\text{TOHA}(\phi_2) \]
6: \[\text{else if } \phi = \phi_1 \lor \phi_2 \text{ then} \]
7: \[\text{return } \text{LTL}_{\mathcal{U}}\text{TOHA}(\phi_1) \cup \text{LTL}_{\mathcal{U}}\text{TOHA}(\phi_2) \]
8: \[\text{else if } \phi = \overline{\pi}U\psi \text{ then} \]
9: \[\mathcal{H}_{\overline{\pi}} \leftarrow \text{LTL}_{\mathcal{U}}\text{TOHA}(\psi) \]
10: \[\text{return } \mathcal{H}_\phi \quad \triangleright \text{ Proposition 3} \]
11: **end if**
12: **end procedure**
LTL(U, ∨, ∧) to HA

Remarks:

- "Fairness conditions"

- Computational complexity
  - Size of H1 ∩ H2: $O(n_1 n_2)$
  - Size of final automaton $O(\exp(|\phi|))$
  - Set intersections and complementation
    - set theoretic operations, not reachability etc.

- Note: graph is acyclic

- Final hybrid automaton might have empty guards and invariant sets
  - Then formula might be unsatisfiable
LTL_{U,□} fragment

- Similar to LTL(U,∨,∧) we can get an algorithm for LTL(□,∨,∧)

- General solution for $\varphi_1 \land \varphi_2$ or $\varphi_1 \lor \varphi_2$
  - with $\varphi_1 \in LTL(U,∨,∧)$ and $\varphi_2 \in LTL(□,∨,∧)$

- More general solution for any formula:
  - $\varphi ::= \pi^* | \varphi | \varphi_1 \land \varphi_2 | \varphi_1 \lor \varphi_2 | \pi^* U \varphi$
  - $\varphi \in LTL(□,∨,∧)$

  - $\textbf{Ex. } □ \pi_0 □ (\pi_2 □ (\pi_3 □ (\pi_4 □ (\neg \pi_2 \land \neg \pi_3) U \pi_1)))$
“Abstract” Hybrid Automaton

\[ \mathcal{G} \]

\[ \pi_0 \land \pi_2 \land \pi_3 \]

\[ s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow s_3 \]

Algorithm
"Abstract" Hybrid Automaton

**Example:** $\Box \pi_0 \land \Box \pi_2 \land \Box \pi_3$

**Algorithm**

- $s_0$
- $s_1$
- $s_2$
- $s_3$
A practical solution

Backward reachability using Breadth First Search:

- fix initial conditions
- one outgoing edge from each control location
Derive controller constraints

Feedback control law $g_c : X \rightarrow U$

Controller constraints $c = \{A, B, \Gamma\}$

- $A$ initial conditions, i.e. $x(0) \in A$
- $\Gamma$ final conditions, i.e. $x(t_g) \in \Gamma$
- $B$ invariant, $\forall t \in [0, t_g]. x(t) \in B$

\[
\begin{align*}
c_3 &= \{\bigcup_{e \in I_3} \text{Guard}(e), \text{Inv}(s_3), \emptyset\} \quad \Rightarrow \quad g_{c3} \\
c_1 &= \{\text{Guard}(e_{I_1}) \cup \text{Init}(s_1), \text{Inv}(s_1), \text{Guard}(e_{O_1})\} \quad \Rightarrow \quad g_{c1} \\
c_2 &= \{\text{Init}(s_2), \text{Inv}(s_2), \text{Guard}(e_{O_2})\} \quad \Rightarrow \quad g_{c2} \\
c_0 &= \{\text{Init}(s_0), \text{Inv}(s_0), \text{Guard}(e_{O_1})\} \quad \Rightarrow \quad g_{c0}
\end{align*}
\]
Main result

Let \( \varphi \in \text{LTL}_{U,\Box} \) be satisfiable wrt to our model of computation. Then:

(i) \( \varphi \) can be converted into list of controller specifications.

(ii) The hybrid automaton \( H \) resulting from the closed-loop dynamics satisfies: \( \forall \eta \in \text{L}(H) . \eta \models \varphi \)
Back to the toy example ...

RTL: $\square \pi_0 \land (\pi_2 \land (\pi_3 \land (\pi_4 \land (\neg \pi_2 \land \neg \pi_3) \cup \pi_1)))$
Back to the toy example ...

Conner et. al. IROS 2003
Conclusions

☑ Top-down approach for the synthesis of HA from temporal logic
  ☑ identified a fragment of LTL that provides a modular construction
  ☑ algorithm is based on the closure properties of HA

☑ Advantages of the proposed framework:
  ☑ Clean separation between software specification and controller design
  ☑ Potentially smaller number of required controllers
  ☑ The closed-loop dynamics can be designed using different methodology in each control location
  ☑ “Real” continuous time semantics
    ☑ Ability to distinguish between events that must hold at a point in time or for an interval
Future work

- Introduce robustness into the system

- Complete the framework for the full LTL
  - i.e. add “liveness”

- Design for distributed specifications

- Built a complete toolbox with libraries of controllers
Thank You!

Any Question(s)?