Receding Horizon Temporal Logic Planning
Automatic Synthesis of Planners and Controllers for Dynamical Systems

Nok Wongpiromsarn, Ufuk Topcu, and Richard M. Murray

California Institute of Technology

September 17, 2009
Motivation

- In many applications, dynamical systems need to perform complex tasks and interact with (potentially adversarial) environments.
- These systems are generally designed by hand based on their desired properties (specs). Changes in the specs potentially result in redesigning and reimplementing a large portion of the systems.
- Ideal: Automatically synthesize dynamical systems which are guaranteed, by construction, to satisfy their desired properties.
1 Introduction
   - Methods from Computer Science and Control
   - Describing Desired Properties
   - Synthesis of Digital Designs and Continuous Controllers
   - Problem Formulation

2 Hierarchical Approach
   - Dealing with Adversarial Environments and System Dynamics
   - Abstraction of System Dynamics
   - Synthesizing the Discrete Planner
   - Correctness of the System

3 Receding Horizon Temporal Logic Planning

4 Example
   - System Model and Specification
   - State Space Discretization
   - Receding Horizon Formulation
   - Results
## Methods from Computer Science and Control

### Computer Science

- discrete system, finite domain
- a polynomial-time algorithm to construct finite state automata from their temporal logic specifications.
- automatic synthesis of digital designs.
- a large class of properties are ensured even in the presence of adversary.

### Control

- continuous system, infinite domain
- optimal control, model predictive control, robust analysis, etc
- automatic synthesis of continuous controllers.
- safety and stability properties are ensured even in the presence of disturbances and modeling errors.
Deal with systems with both discrete and continuous components.

Properties of interest are typically limited to stability and safety.

A wider class of properties needs to be considered
  - guarantee: eventually accomplish task 1, 2 and 3 in any order.
  - response: if the system fails, then eventually perform task 1, or perform tasks 1, 2 and 3 infinitely often in any order.
Describing Desired Properties: Linear Temporal Logic

- Built up from
  - atomic propositions: statements on system variables which have unique truth values
  - logical connectives: \( \neg, \lor, \land, \implies \)
  - temporal operators: next (\( \bigcirc \)), always (\( \square \)), eventually (\( \Diamond \)), until (\( \mathcal{U} \))

- Syntax
  \[
  \varphi ::= \pi \mid \neg \varphi \mid \varphi \lor \varphi \mid \bigcirc \varphi \mid \varphi \mathcal{U} \varphi
  \]
  \[
  \varphi \land \psi = \neg (\neg \varphi \lor \neg \psi)
  \]
  \[
  \varphi \implies \psi = \neg \varphi \lor \psi
  \]
  \[
  \Diamond \varphi = \text{True} \mathcal{U} \varphi
  \]
  \[
  \square \varphi = \neg \Diamond \neg \varphi
  \]

- An LTL formula is interpreted over an infinite sequence of states.
  - Given an execution \( \sigma = v_0 v_1 v_2 \ldots \) and an LTL formula \( \varphi \), we say that \( \varphi \) holds at position \( i \geq 0 \) of \( \sigma \) iff \( \varphi \) holds for the remainder of the execution \( \sigma \) starting at position \( i \).
  - \( \square \varphi \) holds at position \( i \) iff \( \varphi \) holds at every position in \( \sigma \) starting at position \( i \)
  - \( \Diamond \varphi \) holds at position \( i \) iff \( \varphi \) holds at some position \( j \geq i \) in \( \sigma \).
  - \( \bigcirc \varphi \) holds at position \( i \) iff \( \varphi \) holds at position \( i + 1 \) in \( \sigma \).

- An execution \( \sigma = v_0 v_1 v_2 \ldots \) satisfies \( \varphi \) iff \( v_0 \models \varphi \).

- The system is said to be correct with respect to its specification \( \varphi \) if all its executions satisfy \( \varphi \).
Linear Temporal Logic

- **Examples of LTL specifications**
  - Safety: $\square \text{dist}(x, obs) > \delta$
  - Guarantee: $\Diamond (ckt_1 \land \Diamond (ckt_2 \land \Diamond ckt_3))$
  - Response: $\square ((\text{system fails}) \implies \Diamond (\text{perform task 1})), \square \Diamond (\text{perform tasks 1}) \land \square \Diamond (\text{perform task 2})$
For a system to be correct, its specification $\varphi$ must be satisfied regardless of what the environment does.

A two-player game between the system and the environment.
- The environment attempts to falsify $\varphi$
- The system attempts to satisfy $\varphi$

$\varphi$ is *realizable* if the system can satisfy $\varphi$ no matter what the environment does.

Interested in a specification of the form

$$ \varphi = (\varphi_e \Rightarrow \varphi_s) $$

- $\varphi_e$ is an LTL formula which characterizes the initial states of the system and the assumptions of the environment
- $\varphi_s$ is an LTL formula which describes the correct behavior of the system

If $\varphi$ is realizable, the synthesis algorithm generates a finite state automaton which satisfies $\varphi$ no matter what the environment does. Otherwise, it provides an initial state starting from which $\varphi$ cannot be satisfied.

Limitation: state explosion
Designing Continuous Controllers

- Optimization-based design and robust analysis
- Receding Horizon Control
  - Solve finite time optimization over $T$ seconds and implement first $\Delta T$ seconds.
  - Software: NTG, OTG, MPT, ...
Problem Formulation

Consider a system with a set of variables \( V = S \cup E \) where \( S \) represents the set of controlled variables and \( E \) represents the set of environment variables.

The controlled state evolves according to the discrete-time piecewise-affine dynamics:

\[
    s[t + 1] = A_k s[t] + B_k u[t] + C_k \text{ if } (s[t], u[t]) \in \Omega_k
\]

\[
    u[t] \in U
\]

Given \( \varphi \), an LTL specification built from a set of atomic propositions \( \Pi \). Assume that \( \varphi \) can be expressed without the next (\( \bigcirc \)) operator.

Interested in designing a controller for the system which ensures that any execution satisfies \( \varphi \).
Hierarchical Approach

- The system must satisfy its specification $\varphi$ regardless of what the environment does.
- In designing a controller, both the adversarial nature of the environment and the dynamics of the system need to be taken into account.
- Want to separate the concern of the environment from the concern of the dynamics.
- Hierarchical approach
  - Based on an abstract model of the system, a discrete planner computes a discrete plan satisfying $\varphi$ regardless of what the environment does.
  - Based on the system dynamics, a continuous controller continuously implements the abstract plan.
Abstraction of System Dynamics

A finite transition system $\mathcal{D} = \{\mathcal{V}, \rightarrow\}$

- Serve as an abstract model of the system
- Partition the state space such that it is proposition preserving.
  - For any atomic proposition $\pi$ in $\varphi$ and any points $v_1$ and $v_2$ in the same cell, if $v_1$ satisfies $\pi$, then $v_2$ satisfies $\pi$.
  - Denote the discrete domain by $\mathcal{V} = S \times \mathcal{E}$.

- Simulation abstraction
  - Ensure that the continuous execution preserves the correctness of the discrete plan.
  - Enforced by restricting the transition relations $\rightarrow$ of $\mathcal{D}$ to those satisfying the reachability relation.
Reachability

- A discrete state $S_j$ is reachable from a discrete state $S_i$, written $S_i \leadsto S_j$, only if starting from any point $s_{\text{init}}$ in cell $S_i$, there exists a control law $u \in U$ which takes the system to a point $s_{\text{final}}$ in cell $S_j$ while always staying in cells $S_i$ and $S_j$.

- In general, for two discrete states $S_i$ and $S_j$, verifying the reachability relation $S_i \leadsto S_j$ is hard.

- Resort to a heuristic based on the optimal control problem.
  - allow reachability to be established by solving a multiparametric programming problem which can be solved automatically using the Multi-Parametric Toolbox (MPT).
Establishing Reachability

Optimal Control Problem Formulation

Given two cells $S_i, S_j \subseteq \text{dom}(S)$, the set of admissible control inputs $U$, the matrices $A_k$ and $B_k$, a horizon length $N \geq 0$ and the cost matrices $P_N, Q \succeq 0$ and $R \succ 0$, solve

$$\min_{u[0], \ldots, u[N-1]} \|P_N s[N]\|_2 + \sum_{t=0}^{N-1} \|Q s[t]\|_2 + \|R u[t]\|_2$$

s.t.

$$s[N] \in S_j, \quad s[0] \in \text{dom}(S)$$

$$s[t+1] = A_k s[t] + B_k u[t] \text{ if } (s[t], u[t]) \in D_k$$

$$s[t] \in S_i \cup S_j, \quad u[t] \in U$$

$$\forall t \in \{0, \ldots, N - 1\}.$$  

- Solve for a set $P_{i,j}$ such that for all $s[0] \in P_{i,j}$, (1) is feasible
  - Can be regarded as a multiparametric programming problem
  - For polytopic sets $S_i$, $S_j$ and $U$, MPT computes the explicit solution $P_{i,j} \subseteq S_i \cup S_j$.  

\[ S_i \quad S_j \]
Example

\[ s[t + 1] = \begin{bmatrix} 1 & 0.0952 \\ 0 & 0.9048 \end{bmatrix} s[t] + \begin{bmatrix} 0.0048 \\ 0.0952 \end{bmatrix} u \]

\[ u \in [-\sqrt{0.5}, \sqrt{0.5}] \]
State Space Discretization

- The reachability relation between any discrete states in the original partition may not be established through the solution of the multiparametric problem
  - constraints on $u$
  - a specific choice of the finite horizon $N$
- Want to refine the partition to increase the number of valid discrete state transitions of the resulting finite transition system.
State Space Discretization

Discretization Algorithm

\[ S' = S; \ IJ = \{(i, j) \mid i, j \in \{1, \ldots, \text{size}(S)\}\}; \]

while \( \text{size}(IJ) > 0 \)

Pick an \((i, j) \in IJ\);

Solve for \(P_{i,j}\) such that for any \(s[0] \in P_{i,j}\), the optimal control problem (1) is feasible;

if \( (\text{volume}(S_i \cap P_{i,j}) > \text{Vol}_\text{min} \text{ and } \text{volume}(S_i \setminus P_{i,j}) > \text{Vol}_\text{min}) \) then

Replace \(S_i \in S'\) with \(S_i \cap P_{i,j}\) and add \(S_i \setminus P_{i,j}\) to \(S'\);

For each \(k \in \{1, \ldots, \text{size}(S')\}\), add \((i, k), (k, i), (\text{size}(S'), k), (k, \text{size}(S'))\) to \(IJ\);

else

Remove \((i, j)\) from \(IJ\);

endif

endwhile

Nok Wongpiromsarn, Ufuk Topcu, and Richard Murray

Receding Horizon Temporal Logic Planning
Synthesizing the Discrete Planner

- Construct a finite transition system $\mathbb{D} = \{\mathcal{V}, \rightarrow\}$
  - $\mathcal{V}$ is the set of all the discrete states corresponding to the partition of $\text{dom}(\mathcal{V})$ after applying the discretization algorithm.
  - For any two states $\mathcal{V}_i = (S_{is}, E_{is})$ and $\mathcal{V}_j = (S_{js}, E_{js})$, $\mathcal{V}_i \rightarrow \mathcal{V}_j$ only if $S_{is} \leadsto S_{js}$.

- Based on this abstract model $\mathbb{D}$, a discrete planner, represented by a finite state automaton, can be automatically synthesized to satisfy $\varphi$ using a digital design synthesis tool.
Example

Desired Properties

- Visit the blue cell infinitely often.
- Eventually go to the red cell when a PARK signal is received.

Assumption

- Infinitely often, PARK signal is not received.

\[
\varphi = \Box \Diamond (\neg park) \implies (\Box \Diamond (s \in C_5) \land \\
\Box (park \implies \Diamond (s \in C_0)))
\]
Correctness of the System

Theorem

Let \( \sigma_d = \nu_0 \nu_1 \ldots \) be an infinite sequence of discrete states of \( \mathbb{D} \) where for each natural number \( k \), \( \nu_k \to \nu_{k+1}, \nu_k = (\rho_k, \epsilon_k) \), \( \rho_k \) is the discrete controlled state and \( \epsilon_k \) is the discrete environment state. If \( \sigma_d \models_d \varphi \) then by applying a sequence of control laws, each corresponding to the solution of the optimal control problem (1) with \( S_i = \rho_k \) and \( S_j = \rho_{k+1} \), the infinite sequence of continuous states \( \sigma = \nu_0 \nu_1 \nu_2 \ldots \) satisfies \( \varphi \).
Digital design synthesis tools suffer from the state explosion problem.

Under certain conditions, it is not necessary to plan for the whole execution in one shot.

Consider a subclass of Generalized Reactivity(1) formula of the form

\[(\varphi_{init} \land \Box \varphi_e) \implies (\Box \varphi_s \land \Diamond \varphi_g)\]

- \(\varphi_{init}\) and \(\varphi_g\) are propositional formulas of variables from \(V\)
- \(\varphi_e\) and \(\varphi_s\) are boolean combinations of propositional formulas of variables from \(V\) and expressions of the form \(\bigcirc \psi\)
Receding Horizon Temporal Logic Planning

Assume certain structure of the system:
Suppose there exists a collection of disjoint subsets $\mathcal{W}_0, \ldots, \mathcal{W}_p$ of $\mathcal{V}$ such that
(a) $\mathcal{W}_0 \cup \mathcal{W}_2 \cup \ldots \cup \mathcal{W}_p = \mathcal{V}$
(b) $\varphi_g$ is satisfied for any state in a cell in $\mathcal{W}_0$
(c) $(\{\mathcal{W}_0, \ldots, \mathcal{W}_p\}, \preceq_{\varphi_g})$ is a partially ordered set
Synthesizing Subautomata

- Recall that the specification of the system is given by
  \[ (\varphi_{\text{init}} \land \square \varphi_e) \implies (\square \varphi_s \land \Diamond \varphi_g) \]

- Suppose there exists a propositional formula \( \Phi \) and for each \( i \in \{1, \ldots, p\} \), there exists \( g_i \in \{1, \ldots, p\} \) and a subset \( D_i \) of \( \text{dom}(S) \) satisfying
  (a) \( \varphi_{\text{init}} \implies \Phi \) is a tautology
  (b) \( \mathcal{W}_{g_i} \preceq_{\varphi_g} \mathcal{W}_i \) and for each \( i \neq 0 \), \( \mathcal{W}_{g_i} \prec_{\varphi_g} \mathcal{W}_i \)

such that
\[
\Psi_i = ((v \in \mathcal{W}_i) \land \Phi \land \square \varphi_e) \implies (\square \varphi_s \land \Diamond (v \in \mathcal{W}_{g_i}) \land \square \Phi)
\]
is realizable with the domain of \( S \) restricted to \( D_i \).

- An automaton \( A_i \) satisfying \( \Psi_i \) can be synthesized using a digital design synthesis tool.
Receding Horizon Strategy

Starting from the state \( v_0 \), pick an automaton \( A_i \) such that \( v_0 \) is in a cell in \( W_i \) and execute \( A_i \) until the system reaches the state in a cell in \( W_j \prec \varphi W_i \), at which point, switch to the automaton \( A_j \). Keep iterating this process until \( A_0 \) is executed.

Theorem

Suppose for each \( i \in \{1, \ldots, p\} \), \( \Psi_i \) is realizable. Then the receding horizon strategy ensures the correctness of the system.
System Model

- Discretized nondimensional dynamics of a point-mass omnidirectional vehicle
  \[
  \begin{bmatrix}
  z[t+1] \\
  v_z[t+1]
  \end{bmatrix} = \begin{bmatrix}
  1 & 0.0952 \\
  0 & 0.9048
  \end{bmatrix} \begin{bmatrix}
  z[t] \\
  v_z[t]
  \end{bmatrix} + \begin{bmatrix}
  0.0048 \\
  0.0952
  \end{bmatrix} q_z
  \]
  \[|q_z| \leq \sqrt{0.5}\]
  where \( z \) represents either \( x \) or \( y \) and \( v_z \) represents the rate of change in \( z \).

- Domain: \( C = C_X \times C_Y \)
  \( C_X = [xmin, xmax] \times [-1, 1] \)
  \( C_Y = [ymin, ymax] \times [-1, 1] \).

- Partition \( C_z \) as
  \[
  C_z = \bigcup_{i \in \{zmin+1, \ldots, zmax\}} C_{z,i}
  \]
  where \( C_{z,i} = [i - 1, i] \times [-1, 1] \).

- Consider a straight road of length \( L \) with 2 lanes, each of width 1: \( x_{min} = 0 \), \( x_{max} = L \), \( y_{min} = 0 \), \( y_{max} = 2 \).
System Specification: Desired Properties

- No collision:
  \[ \square (O_{i,j} \implies \neg (x \in C_{x,i} \land y \in C_{y,j})) \]

- The vehicle stays in the right lane unless there is an obstacle blocking the lane:
  \[ \square ((\neg O_{i,1} \land x \in C_{x,i}) \implies (y \in C_{y,1})) \]

- Eventually the vehicle gets to the end of the road:
  \[ \Diamond (x \in C_{x,L}) \]
System Specification: Assumptions

- At the initial configuration, the vehicle is at least $d_{obs}$ away from any obstacle and the vehicle starts in the right lane.
  \[
  \left( x \in \bigcup_{k=i-d_{obs}}^{i+d_{obs}} C_{x,k} \implies (\neg O_{i,1} \land \neg O_{i,2}) \right) \land y \in C_{y,1}
  \]

- An obstacle is always detected before the vehicle gets too close to it. That is, there is a lower bound $d_{popup}$ on the distance from the vehicle for which obstacle is allowed to instantly pop up.
  \[
  \Box \left( \left( x \in \bigcup_{j=i-d_{popup}}^{i+d_{popup}} C_{x,j} \land \neg O_{i,k} \right) \implies \lozenge (\neg O_{i,k}) \right)
  \]

- Sensing range is limited. That is, the vehicle cannot detect an obstacle that is away from it farther than $d_{sr}$.
  \[
  \Box \left( x \in C_{x,i} \implies \bigwedge_{j>i+d_{sr}} (\neg O_{j,1} \land \neg O_{j,2}) \right)
  \]

- The road is not blocked: \( \Box (\neg O_{i,1} \lor \neg O_{i,2}) \)

- An obstacle on the right lane does not disappear. \( \Box (O_{i,1} \implies \lozenge (O_{i,1})) \)
State Space Discretization

- Apply the discretization algorithm for the $x$ and $y$ components separately for computational efficiency.

- With horizon length $N = 10$ and $Vol_{min} = 0.1$, get 11 cells $\{C_{z,i}^1, C_{z,i}^2, \ldots, C_{z,i}^{11}\}$ for each $C_{z,i}$. 
Receding Horizon Formulation

- Partial order structure: \( \mathcal{W}_i = \{(v_x, v_y, O_{1,1}, \ldots, O_{L,2}) \mid v_x \in C_{x,L-i}\} \)

- An invariant \( \Phi \) is determined by reasoning about the valid discrete transitions and the desired properties.
  - To ensure the progress property \( \Diamond \varphi_g \), the vehicle cannot be in a state with no valid transition to other state.
  - To ensure no collision, the vehicle cannot collide with an obstacle at the initial state.
  - etc

- With \( d_{popup} = 1 \) and the horizon length 2, the specifications for the subautomata are realizable.

- If \( d_{obs} > 1 \) and the possible initial states of the system are restricted to those with valid transition to other state, then \( \varphi_{init} \implies \Phi \) is a tautology.
The synthesis was performed on a Pentium 4, 3.4 GHz computer with 4 Gb of memory.

Computation time: 1230 seconds

The resulting automaton contains 2845 nodes.

During the synthesis process, 96796 nodes were generated.

- This particular computer crashes when approximately 97500 nodes are generated.
- Without the receding horizon strategy, problems with the road of length greater than 3 cannot be solved.
Conclusions

- Described how off-the-shelf tools from computer science and control can be integrated to allow automatic synthesis of complex dynamical systems.
  - The resulting systems are guaranteed, by construction, to satisfy the desired properties expressed in linear temporal logic even in the presence of adversary.
- Described a receding horizon scheme
  - Address the main limitation of the synthesis algorithm: the state explosion problem.
  - Allow more complex problems to be solved.
- The example illustrated that without the receding horizon scheme, the synthesis problem can be extremely computationally challenging.
Acknowledgments

- Hadas Kress-Gazit
- K. Mani Chandy
- Knot Pipatsrisawat
- Multi-Parametric Toolbox (MPT): http://control.ee.ethz.ch/~mpt/
- TLV Module for Synthesis: http://www.wisdom.weizmann.ac.il/~saar/synthesis/