Inductive controller synthesis for piecewise linear systems with SMT

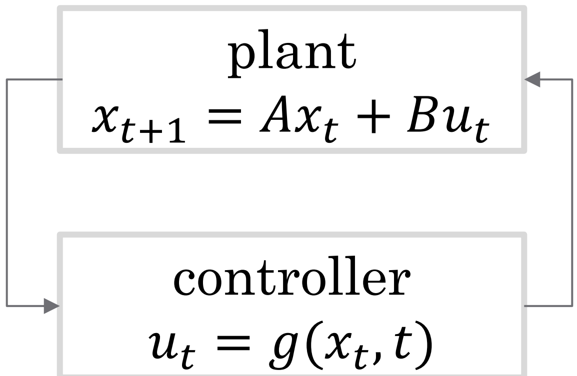
Zhenqi Huang[1], Yu Wang[1]

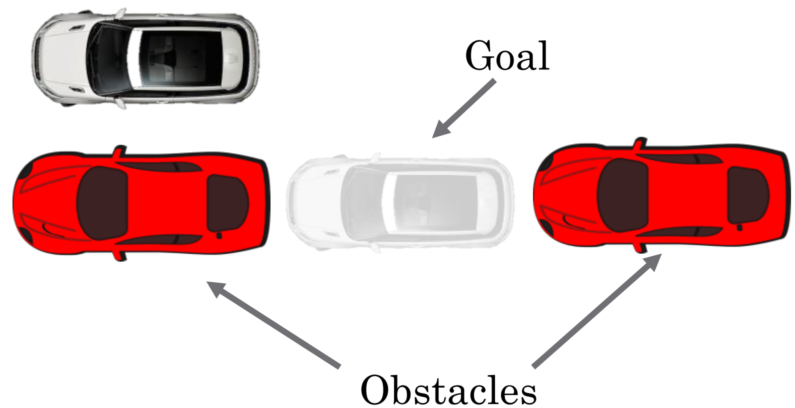
Sayan Mitra[1], Geir Dullerud[1], and Swarat Chaudhuri[2]

[1]Coordinated Science Lab
University of Illinois at Urbana-Champaign

[2]Department of Computer Science
Rice University

Typical synthesis problem: reach-avoid

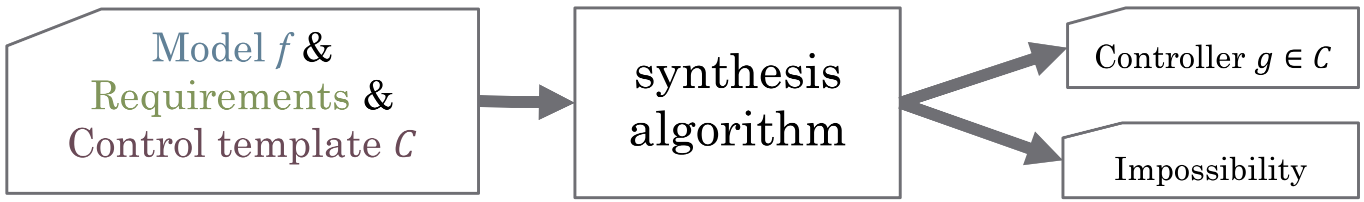




Reach-avoid problem is defined by:

- Controller class C such that $g \in C$
- Initial set $Init$ s.t. $x_0 \in Init$
- Goal set $Goal$ s.t. $x_T \in Goal$ for some T
- Safe set $Safe$ s.t. $x_t \in Safe$ for all $t \leq T$

Controller synthesis algorithm



given a system *model, safe* and *goal*, <u>find</u> control such that all behaviors are safe and reach goal

- yes (controller strategy g)
- no (impossibility certificate "no controller exists")

Existing reach-avoid synthesis

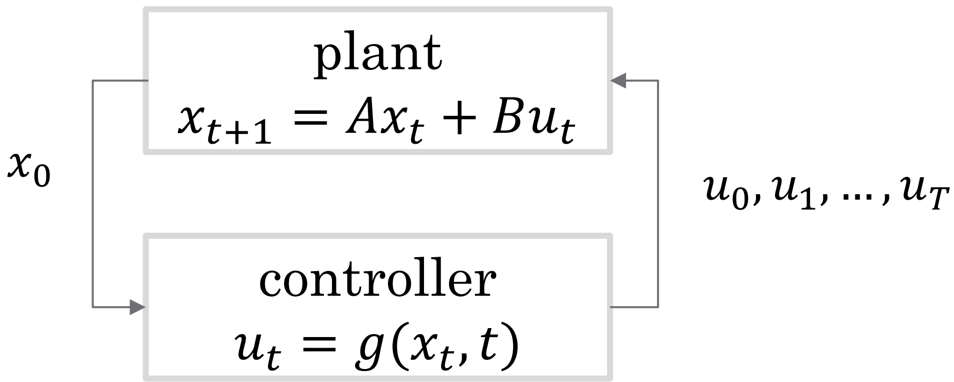
- Constraint model predictive control (e.g., [Bemporad02])
  - Cast the reach-avoid problem into a constraint optimization
  - Apply receding-horizon strategy
  - Challenges: soundness, completeness, nested constraints

- Finite automata abstraction (e.g., [Tabuada06])
  - Construct finite automata of the dynamical system and the reach-avoid property
  - Model check the product automata
  - Challenges: completeness, scalability

SMT-based synthesis: overview

- First order logic formula have quantifiers over variables
  - Example: $\exists y \forall x. (x^2 \leq y + 1) \Rightarrow (\sin x > \cos(\log y))$
- Satisfiability modulo theories (SMT) solvers
  - Finding satisfying solutions for first order logic formula, or
  - Prove no solution satisfies the formula
  - E.g. Z3, CVC4, VeriT, dReal
  - Scales up to hundreds of real variables & thousands of constraints for quantifier-free linear formula
- SMT-based synthesis: generate boolean constraints for a correct controller using the problem specifications and directly solve using SMT solvers.

Naïve SMT synthesis: open-loop control



Consider $C = \{[0, \dots, T] \rightarrow U\}$ for open-loop control with a single initial state $Init = \{x_0\}$

- $\exists u_0, u_1, \dots, u_T$:

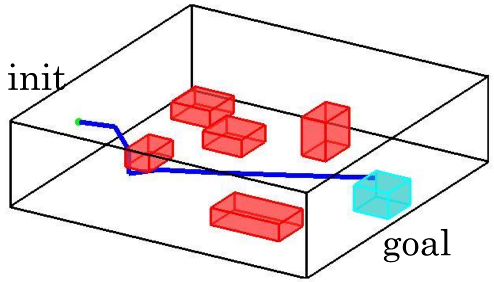
$$\left(\bigwedge_{t \leq T} x(t) \in safe\right) \wedge \; x(T) \in Goal$$

with $x(t) = A^t x_0 + \sum_{s=0}^{t-1} A^{t-s-1} B u_s$

Application: helicopter autopilot

Autonomous helicopter

- 16 dimensions, 4 inputs



Advantage

- Method is automatics, can be used by users with limited experience in control

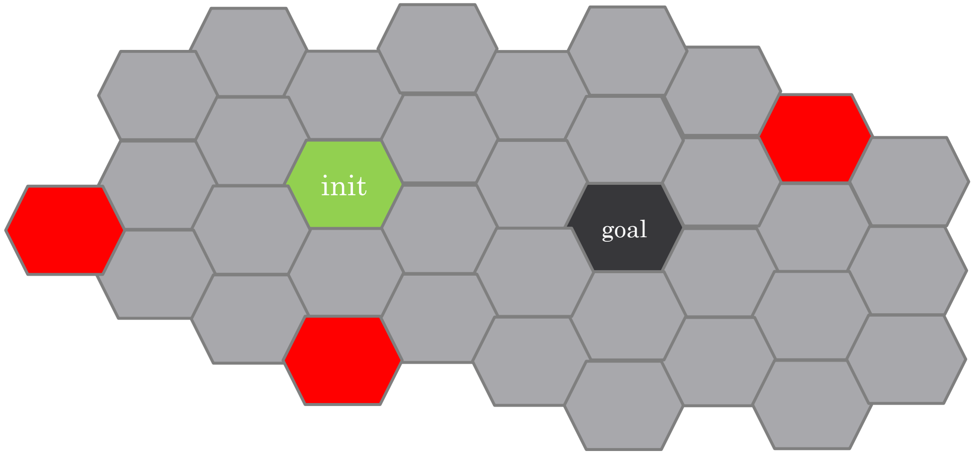
Limitations

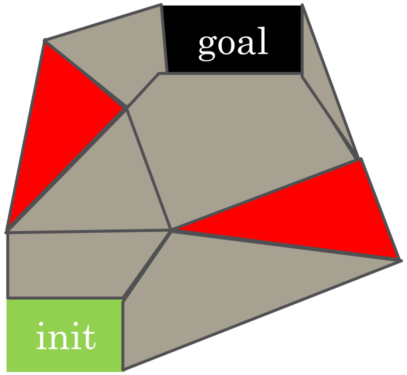
- Performance deteriorates with larger disturbances
- Relies on unrolling the system dynamics with disturbance for bounded time---does not scale beyond linear, short horizon

T	ϕ	Result	R.time (s)
9	402	Sat	24.5
12	338	Sat	60.6
15	576	Sat	158.8
18	640	--	--

Idea of inductive synthesis: (a) state feedback

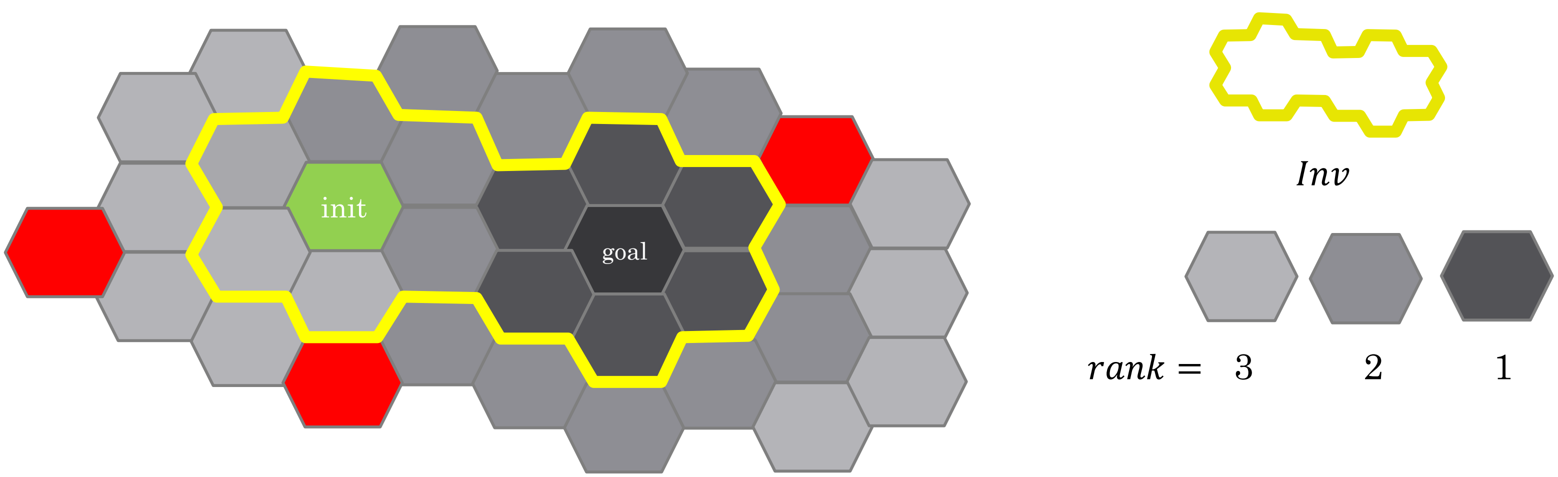
- Lookup table controller:
  - ***P***: cover of the state space, sensor quantization or heuristic
  - $C = \{ P \rightarrow U \}$
  - We denote post(p, g) as the set of partition reached in one-step from a partition p using controller g.





Idea of inductive synthesis: (b) two correctness certificates

- Safety certificate
  - An invariant set *Inv* that is reachable from *init*
- Progress certificate
  - A ranking function *rank* like a Lyapunov function



Idea of inductive synthesis: (c) inductive synthesis rules

Find $g\colon \boldsymbol{P} \to U,\ \mathrm{rank}\colon \boldsymbol{P} \to \mathbb{N},\ Inv\colon \boldsymbol{P} \to \{0,1\}$ such that:

- (initial condition) $Init \subseteq Inv$
- (control invariant) $post(Inv, g) \subseteq Inv$
- (safe) $Inv \subseteq safe$
- (goal) $p \subseteq goal \Leftrightarrow rank(p) = 0$
- (progress) $rank(p) > 0 \Rightarrow rank(p) > \max rank(\mathrm{post}^{\mathrm{k}}(p, g))$

Strengthening & relaxation of rules

The post operator is generally hard to symbolically compute, but can be over-/under-approximated

- replace *post* by over-approximated $\overline{post}$, we get a set of strengthened rules
- replace *post* by under-approximated $\underline{post}$, we get a set of relaxed rules

- If the $\overline{\text{strengthened rules}}$ are solved by control g with certificates $\overline{Inv}$, $rank$, so is the original rules.
- If the relaxed rules does not have a solution, so is the original rules.
- If the relaxed rules are solved by control g with certificates $\underline{Inv}$, $rank$, but the strengthened rules does not have a solution, the set $\underline{Inv}$ can be used to guide refinement of *post*
  - Refine in $\underline{Inv}$ helps derive progress proof, and
  - Refine in $\underline{Inv}^C$ helps derive safety proof.

Soundness & relative completeness

Given controller class C and ranking function templates R, a problem M is robust if there exists $\epsilon > 0$:

- *exists $g \in C, V \in R$ such that for any problem M' whose dynamic is ϵ-close to M, the g, V solves the inductive rules for M', OR*
- *for none of the problems M' that are ϵ-close to M, have solutions to the synthesis problem with any $g \in C, V \in R$*

Theorem. If synthesis problem M is robust, then there exists a sufficiently accurate computation of $post$ to

- (a) either find control g and proof $rank, Inv$ or
- (b) give a proof that there exists no such controller in C, R.

Application: path planning

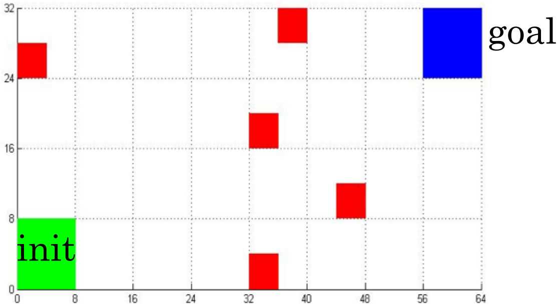
implemented using CVC4 SMT solver

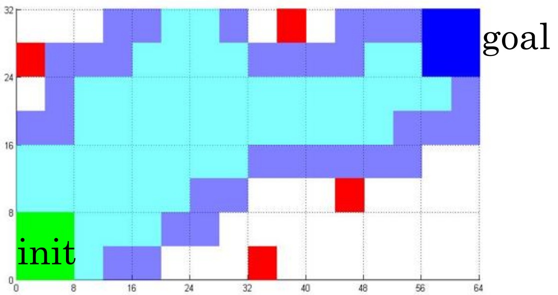
4D nonlinear vehicle navigation with noise and obstacles

P: regions in state space

$rank\colon p \rightarrow \mathbb{N}$

- 768 cells, 3072 real-valued/boolean variables, solved in less than 10 minutes





Light (under) and over (dark) approximation of post

Summary and outlook

- We propose inductive controller synthesis algorithm using SMT solvers
- Idea: synthesize an invariant set and a ranking function serving as the correctness proofs **together** with the controller actions
- Algorithms can also give impossibility certificates
- Ongoing and Future work:
  - Connect synthesis with our high-level programming language of distributed robots [Lin et al. LCTES 2015]
  - Synthesis of attacks on power networks