Reinforcement learning for formal synthesis of cyber-physical systems

Sadegh Soudjani

Newcastle University & MPI-SWS

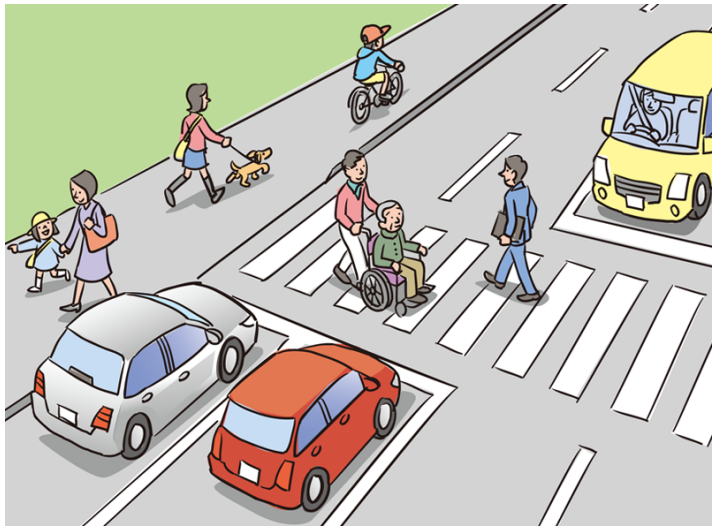
9 July 2023

Challenges

- large-scale systems
- complex interactions
- continuous variables

- uncertain environment
- lack of precise models
- complex requirements

Complex requirements



Traffic rules:

No collision: $\square\,(\text{dist}(\text{Loc}_{vi}, \text{Loc}_{vj}) \geq d_{\text{safe}} \wedge \text{dist}(\text{Loc}_{vi}, \text{Obs}) \geq d_{\text{safe}})$

speed limit: $\square\,((\text{Loc}_{vi} \in \text{Zone}_j) \rightarrow (\text{Speed}_{vi} \leq \text{limit}_j))$

Environment assumptions:

Each intersection is clear infinitely often: $\square\Diamond(\text{Intersection} = \text{empty})$

Goals:

Visit location B after location A: $\Diamond A \wedge (A \rightarrow \Diamond B)$

System under uncertainty

Automated controller synthesis to satisfy high-level logic properties

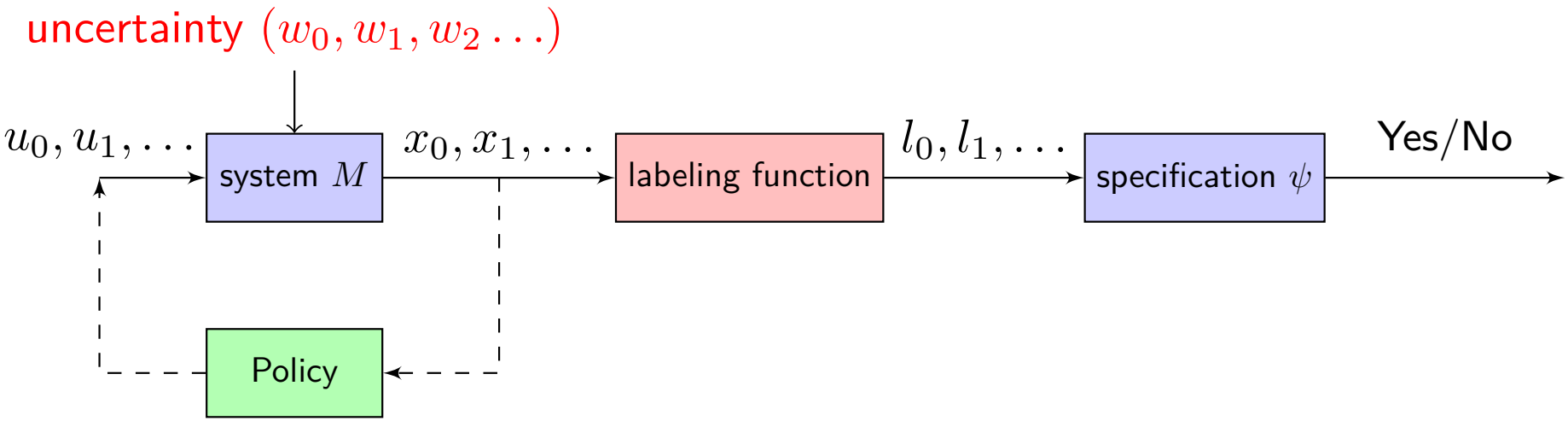
Computational complexity and uncountable sets

Outline

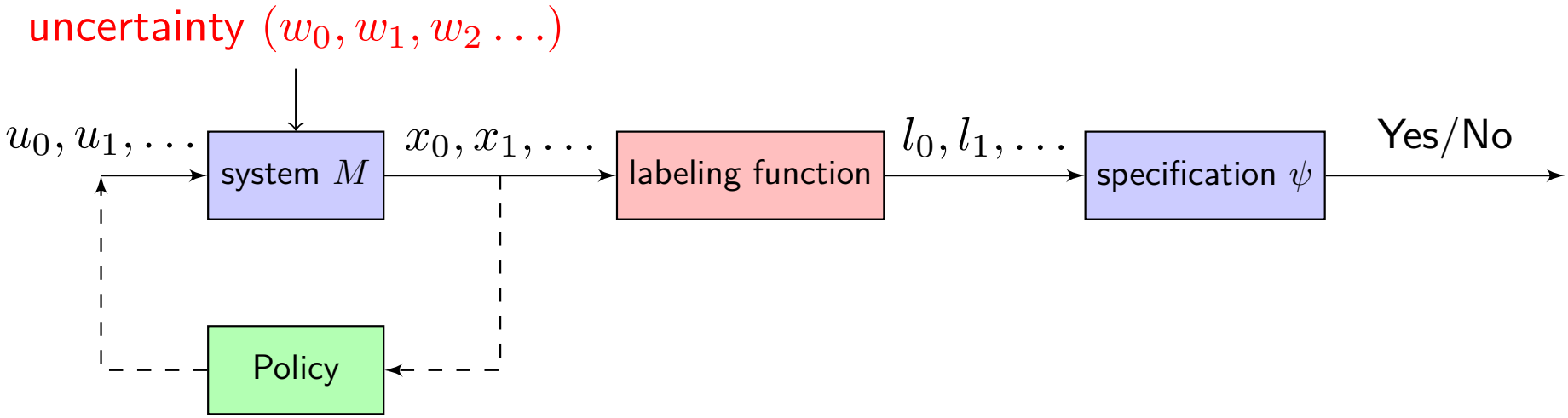
1. Model-free RL with approximate optimality guarantees
2. Model-free RL for infinite-horizon specifications
3. Translating specifications to average objectives for RL
4. Conclusion and future directions

Outline

Objective


uncertainty $(w_0, w_1, w_2 \ldots)$
$u_0, u_1, \ldots$
system M
$x_0, x_1, \ldots$
labeling function
$l_0, l_1, \ldots$
specification ψ
Yes/No
Policy

Objective



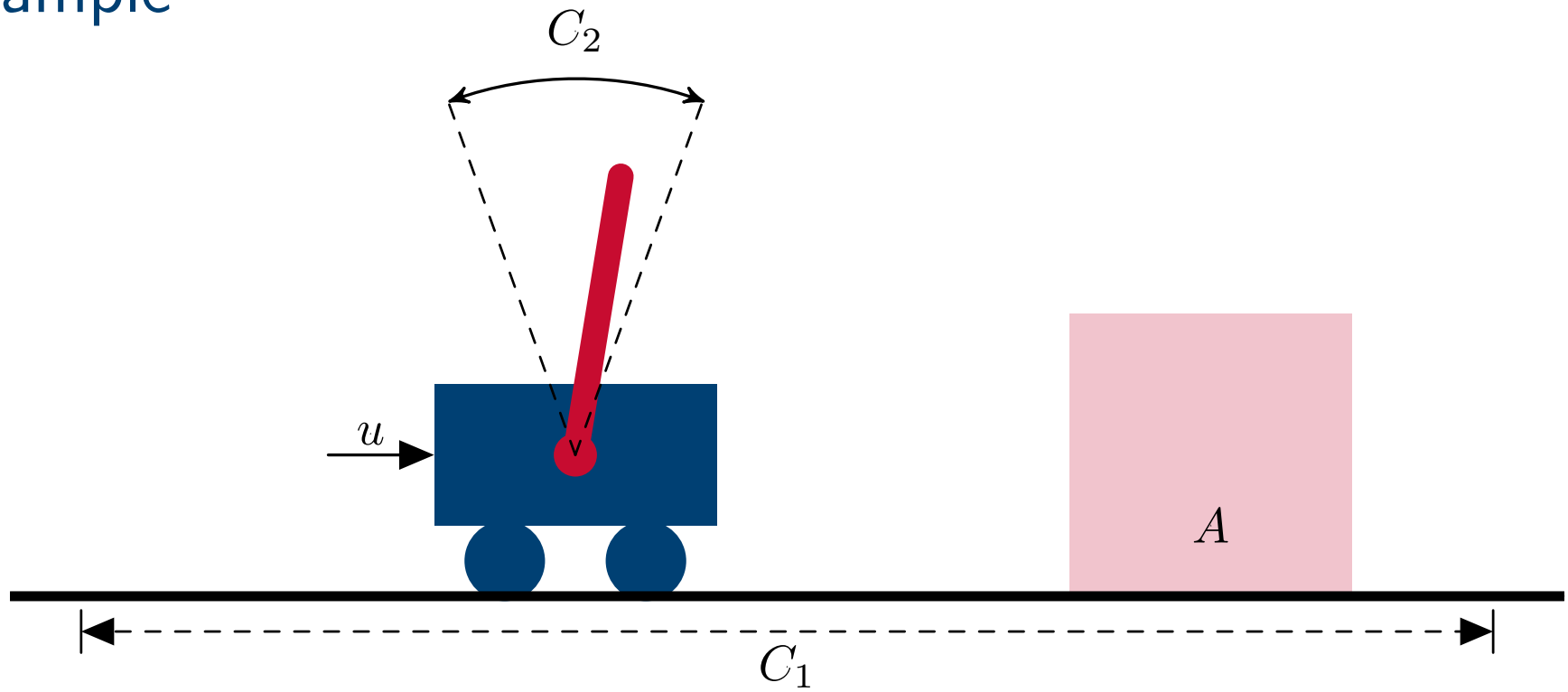
- we are looking for a policy (u as a function of current and past states)
- we do not know the model of the system M
- we can generate input/state data

Objective

Given a black-box model of the system M, specification ψ, and a labeling function, find a policy that maximises

$$\max \mathsf{Prob}(M \models \psi)$$

Example

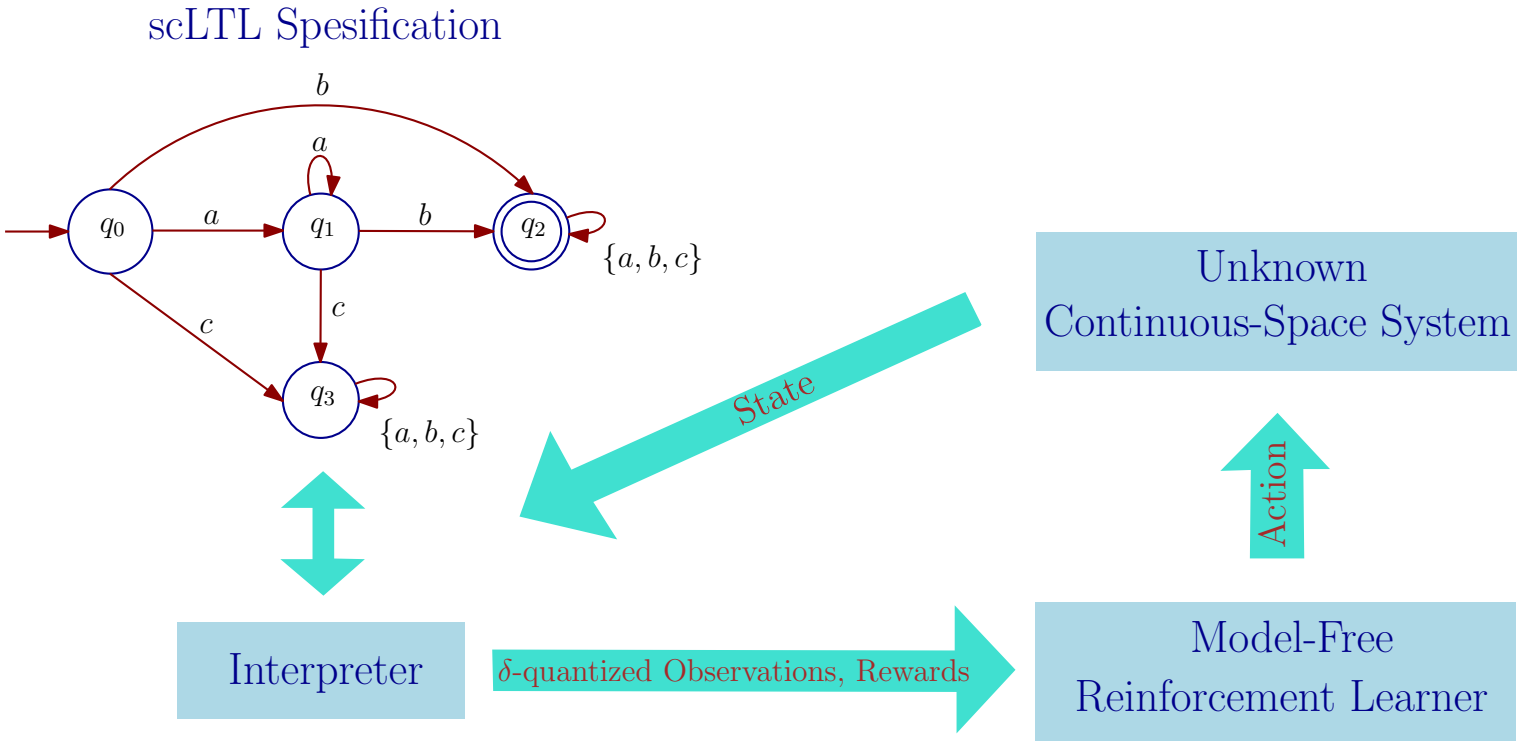


- system: cart-pole with a 4-dimentional state space

$$x_{k+1} = f(x_k, u_k, w_k), \quad x_k \in \mathbb{R}^4$$

- specification:
  - always stay within the limit C_1
  - always keep the pole upright in C_2
  - and reach the region A

Learning-based synthesis



In the first part:

- Propose model-free RL for controller synthesis
- Provide approximate optimality guarantees of synthesised policies trained by RL applying to original unknown continuous-space system
- Propose reward shaping technique to produce dense rewards

Stochastic Control Systems

$\Sigma = (X, U, \varsigma, f)$,

- $X \subseteq \mathbb{R}^n$: State set;
- U: Finite input set;
- ς: (unknown) A sequence of i.i.d. random variables;
- f: (unknown) Transition map.

Evolution of the state of Σ

$$x(k+1) = f(x(k), \nu(k), \varsigma(k)), \qquad k \in \mathbb{N}.$$

Stochastic Control Systems

$\Sigma = (X, U, \varsigma, f)$,

- $X \subseteq \mathbb{R}^n$: State set;
- U: Finite input set;
- ς: (unknown) A sequence of i.i.d. random variables;
- f: (unknown) Transition map.

Evolution of the state of Σ

$$x(k+1) = f(x(k), \nu(k), \varsigma(k)), \qquad k \in \mathbb{N}.$$

Equivalent Markov decision process (MDP)

- $\Sigma = (X, U, T_{\mathsf{x}})$
- $T_{\mathsf{x}} : \mathcal{B}(X) \times X \times U \to [0, 1]$: Conditional stochastic kernel

Stochastic Control Systems

$\Sigma = (X, U, \varsigma, f)$,

- $X \subseteq \mathbb{R}^n$: State set;
- U: Finite input set;
- ς: (unknown) A sequence of i.i.d. random variables;
- f: (unknown) Transition map.

Evolution of the state of Σ

$$x(k+1) = f(x(k), \nu(k), \varsigma(k)), \qquad k \in \mathbb{N}.$$

Equivalent Markov decision process (MDP)

- $\Sigma = (X, U, T_{\mathsf{x}})$
- $T_{\mathsf{x}} : \mathcal{B}(X) \times X \times U \to [0, 1]$: Conditional stochastic kernel
- Only information: Lipschitz constant of stochastic kernel $\mathscr{H}$

Stochastic Control Systems

$\Sigma = (X, U, \varsigma, f)$,

- $X \subseteq \mathbb{R}^n$: State set;
- U: Finite input set;
- ς: (unknown) A sequence of i.i.d. random variables;
- f: (unknown) Transition map.

Evolution of the state of Σ

$$x(k+1) = f(x(k), \nu(k), \varsigma(k)), \qquad k \in \mathbb{N}.$$

Equivalent Markov decision process (MDP)

- $\Sigma = (X, U, T_{\mathsf{x}})$
- $T_{\mathsf{x}} : \mathcal{B}(X) \times X \times U \to [0, 1]$: Conditional stochastic kernel
- Only information: Lipschitz constant of stochastic kernel $\mathscr{H}$

Markov policy: Take the input as a function of current state of the model

Stochastic Control Systems

$\Sigma = (X, U, \varsigma, f)$,

- $X \subseteq \mathbb{R}^n$: State set;
- U: Finite input set;
- ς: (unknown) A sequence of i.i.d. random variables;
- f: (unknown) Transition map.

Evolution of the state of Σ

$$x(k+1) = f(x(k), \nu(k), \varsigma(k)), \qquad k \in \mathbb{N}.$$

Equivalent Markov decision process (MDP)

- $\Sigma = (X, U, T_{\mathsf{x}})$
- $T_{\mathsf{x}} : \mathcal{B}(X) \times X \times U \to [0, 1]$: Conditional stochastic kernel
- Only information: Lipschitz constant of stochastic kernel $\mathscr{H}$

Markov policy: Take the input as a function of current state of the model

Construct finite MDP $\widehat{\Sigma} = (\hat{X}, U, \varsigma, \hat{f})$ with approximate dynamics:

$$\hat{f}(\hat{x}, \nu, \varsigma) = \Pi_x(f(\hat{x}, \nu, \varsigma)),$$

A fragment of LTL

Co-safe fragment of LTL (scLTL):

$$\phi ::= p \,|\, \neg p \,|\, \phi_1 \vee \phi_2 \,|\, \phi_1 \wedge \phi_2 \,|\, \Diamond \phi \,|\, \bigcirc \phi \,|\, \phi_1 \,\mathsf{U}\, \phi_2.$$

The negation operator ($\neg$) only occurs before atomic propositions.

A fragment of LTL

Co-safe fragment of LTL (scLTL):

$$\phi ::= p \,|\, \neg p \,|\, \phi_1 \vee \phi_2 \,|\, \phi_1 \wedge \phi_2 \,|\, \Diamond \phi \,|\, \bigcirc \phi \,|\, \phi_1 \,\mathsf{U}\, \phi_2.$$

The negation operator ($\neg$) only occurs before atomic propositions.

A *deterministic finite automaton (DFA)* is a tuple $\mathcal{A} = (Q, \Sigma_{\mathsf{a}}, \mathsf{t}, q_0, F_{\mathsf{a}})$ where Q is a finite set of states, Σ_{a} is an alphabet, $\mathsf{t} : Q \times \Sigma_{\mathsf{a}} \to Q$ is a transition function, $q_0 \in Q$ is the initial state, and $F_{\mathsf{a}} \subseteq Q$ are accepting states.

A fragment of LTL

Co-safe fragment of LTL (scLTL):

$$\phi ::= p \,|\, \neg p \,|\, \phi_1 \vee \phi_2 \,|\, \phi_1 \wedge \phi_2 \,|\, \Diamond \phi \,|\, \bigcirc \phi \,|\, \phi_1 \,\mathsf{U}\, \phi_2.$$

The negation operator ($\neg$) only occurs before atomic propositions.

A *deterministic finite automaton (DFA)* is a tuple $\mathcal{A} = (Q, \Sigma_{\mathsf{a}}, \mathsf{t}, q_0, F_{\mathsf{a}})$ where Q is a finite set of states, Σ_{a} is an alphabet, $\mathsf{t} : Q \times \Sigma_{\mathsf{a}} \to Q$ is a transition function, $q_0 \in Q$ is the initial state, and $F_{\mathsf{a}} \subseteq Q$ are accepting states.

Observation

Every scLTL formula φ can be translated to a DFA $\mathcal{A}_\varphi$ that accepts the same language as φ.

Control Policy Synthesis Problem

Objective

Given the specification φ and the black-box system $\Sigma = (X, U, \varsigma, f)$ with unknown f and ς, but known Lipschitz constant $\mathscr{H}$.

Synthesise a policy for Σ to satisfy φ while providing optimality guarantees of the synthesised policy trained by RL applying to original unknown system.

Control Policy Synthesis Problem

Objective

Given the specification φ and the black-box system $\Sigma = (X, U, \varsigma, f)$ with unknown f and ς, but known Lipschitz constant $\mathscr{H}$.

Synthesise a policy for Σ to satisfy φ while providing optimality guarantees of the synthesised policy trained by RL applying to original unknown system.

Example: For linear systems $x(k+1) = Ax(k) + B\nu(k) + \varsigma(k)$ with $A = [a_{ij}]$ and $\varsigma(k)$ having normal distribution with covariance matrix $diag(\sigma_1, \ldots, \sigma_n)$,

$$\mathscr{H} = \sum_{i,j} \frac{2|a_{ij}|}{\sigma_i \sqrt{2\pi}}.$$

Control Policy Synthesis Problem

Objective

Given the specification φ and the black-box system $\Sigma = (X, U, \varsigma, f)$ with unknown f and ς, but known Lipschitz constant $\mathscr{H}$.

Synthesise a policy for Σ to satisfy φ while providing optimality guarantees of the synthesised policy trained by RL applying to original unknown system.

Example: For linear systems $x(k+1) = Ax(k) + B\nu(k) + \varsigma(k)$ with $A = [a_{ij}]$ and $\varsigma(k)$ having normal distribution with covariance matrix $diag(\sigma_1, \ldots, \sigma_n)$,

$$\mathscr{H} = \sum_{i,j} \frac{2|a_{ij}|}{\sigma_i \sqrt{2\pi}}.$$

Estimate Lipschitz constant $\mathscr{H}$ from sample trajectories

- Construct an estimation of the conditional density function
- Compute the Lipschitz constant numerically using the derivative of the estimated conditional density function

Formal Optimality Guarantee

Theorem

Let Σ be a continuous-space MDP and $\widehat{\Sigma}$ be its finite abstraction. For a given scLTL specification φ, and for any policy $\hat{\nu}(\cdot) \in \hat{\mathcal{U}}$, the probabilistic closeness between two systems can be acquired as

$$|\mathbb{P}(\Sigma_{\hat{\nu}} \vDash \varphi) - \mathbb{P}(\widehat{\Sigma}_{\hat{\nu}} \vDash \varphi)| \leq \varepsilon, \quad \text{with } \varepsilon := T\delta\mathscr{H}\mathscr{L},$$

where T is the finite time horizon, δ is the state discretization parameter, $\mathscr{H}$ is the Lipschitz constant of the stochastic kernel, and $\mathscr{L}$ is the Lebesgue measure of the specification set.

Formal Optimality Guarantee

Theorem

Let Σ be a continuous-space MDP and $\widehat{\Sigma}$ be its finite abstraction. For a given scLTL specification φ, and for any policy $\hat{\nu}(\cdot) \in \hat{\mathcal{U}}$, the probabilistic closeness between two systems can be acquired as

$$|\mathbb{P}(\Sigma_{\hat{\nu}} \vDash \varphi) - \mathbb{P}(\widehat{\Sigma}_{\hat{\nu}} \vDash \varphi)| \leq \varepsilon, \quad \text{with } \varepsilon := T\delta\mathscr{H}\mathscr{L},$$

where T is the finite time horizon, δ is the state discretization parameter, $\mathscr{H}$ is the Lipschitz constant of the stochastic kernel, and $\mathscr{L}$ is the Lebesgue measure of the specification set.

Theorem

For a discretization parameter δ satisfying $T\delta\mathscr{H}\mathscr{L} \leq \varepsilon$, a convergent model-free reinforcement learning algorithm over $\widehat{\Sigma}^r$ with a reward function guided by the DFA $\mathcal{A}_\varphi$, converges to a 2ε-optimal policy over Σ.

Reward Shaping: Overcoming Sparse Rewards

Let $d(q)$ be the minimum distance of the state q to the unique accepting state and $d_{\texttt{max}} = 1 + \max_q\{d(q) : d(q) < \infty\}$. We define the potential function $P : \mathbb{N} \to \mathbb{R}$ as

$$P(d) = \begin{cases} \kappa \frac{d - d(q_0)}{1 - d_{\texttt{max}}}, & \text{for } d > 0, \\ 1, & \text{for } d = 0. \end{cases}$$

Then the "shaped" reward function $\rho_\kappa : \hat{X} \times \hat{U} \times \hat{X} \to \mathbb{R}$ is proposed as

$$\rho_\kappa((x, q), \nu, (x', q')) = P(d(q')) - P(d(q)).$$

Reward Shaping: Overcoming Sparse Rewards

Let $d(q)$ be the minimum distance of the state q to the unique accepting state and $d_{\texttt{max}} = 1 + \max_q\{d(q) : d(q) < \infty\}$. We define the potential function $P : \mathbb{N} \to \mathbb{R}$ as

$$P(d) = \begin{cases} \kappa \frac{d - d(q_0)}{1 - d_{\texttt{max}}}, & \text{for } d > 0, \\ 1, & \text{for } d = 0. \end{cases}$$

Then the "shaped" reward function $\rho_\kappa : \hat{X} \times \hat{U} \times \hat{X} \to \mathbb{R}$ is proposed as

$$\rho_\kappa((x, q), \nu, (x', q')) = P(d(q')) - P(d(q)).$$

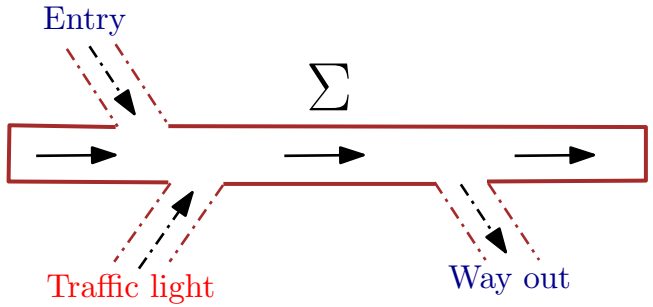
Theorem

(Correctness of Reward Shaping) There exists a $\kappa_\star > 0$ such that for all $\kappa < \kappa_\star$ the set of optimal policies is the same with reward ρ_κ and the actual reward associated with the specification.

Case Study: Road Traffic Control

Model:

$$\Sigma : x(k+1) = (1 - \frac{\tau v}{l} - q)x(k) + 6\nu(k) + 1.9\varsigma(k) + 3,$$



- v: Flow speed of the vehicles,
- l: Length of the cell,
- τ: Sampling time,
- q: Ratio of vehicles goes out on the exit of the cell,
- $\nu(k)$ is taking values in $\{0, 1\}$.

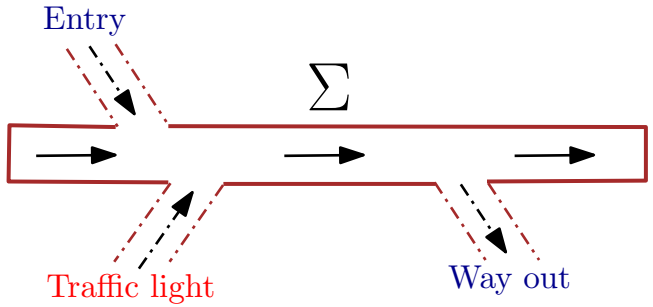
Specification

- Synthesize a controller using model-free RL such that the *density of the traffic* is lower than 20 vehicles.

Case Study: Road Traffic Control

Model:

$$\Sigma : x(k+1) = (1 - \frac{\tau v}{l} - q)x(k) + 6\nu(k) + 1.9\varsigma(k) + 3,$$



- v: Flow speed of the vehicles,
- l: Length of the cell,
- τ: Sampling time,
- q: Ratio of vehicles goes out on the exit of the cell,
- $\nu(k)$ is taking values in $\{0, 1\}$.

Specification

- Synthesize a controller using model-free RL such that the *density of the traffic* is lower than 20 vehicles.

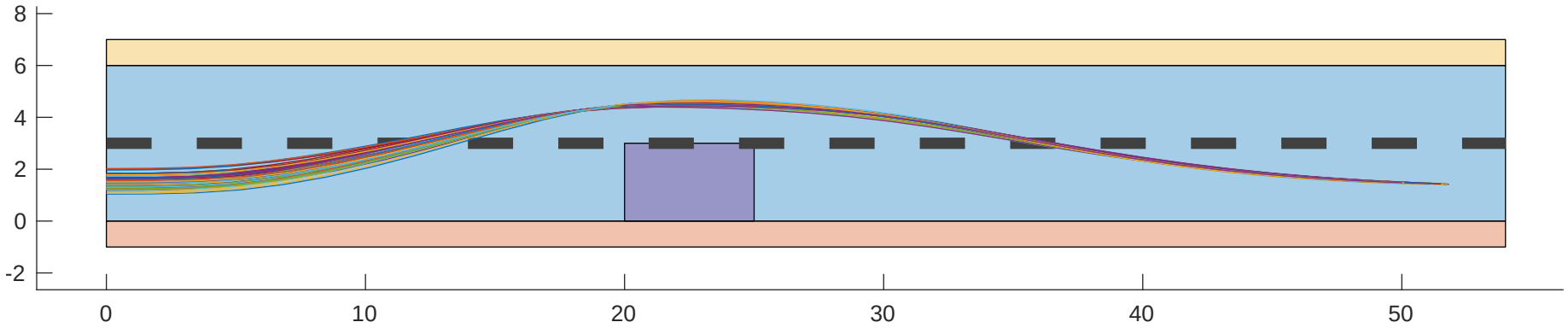
δ	p_r	p_*	ε	p_l	p_u
0.01	0.9856	0.9995	0.0160	0.9835	1.0
0.02	0.9975	0.9995	0.0319	0.9676	1.0
0.05	0.9993	0.9995	0.0798	0.9197	1.0
0.1	0.9992	0.9995	0.1596	0.8399	1.0
0.2	0.9991	0.9995	0.3193	0.6802	1.0

Case Study: 7-Dimensional model of a BMW 320i car

- 7-Dimensional model with two inputs: Steering angle and heading velocity
- Nonlinear terms including $\sin(x), \cos(x), \tan(x)$, etc.
- Deep deterministic policy gradient (DDPG)

Specification

- Consider a two-lane highway when an accident suddenly happens on the traveling. The vehicle's controller should find a safe maneuver to avoid the crash with the next-appearing obstacle.



Outline

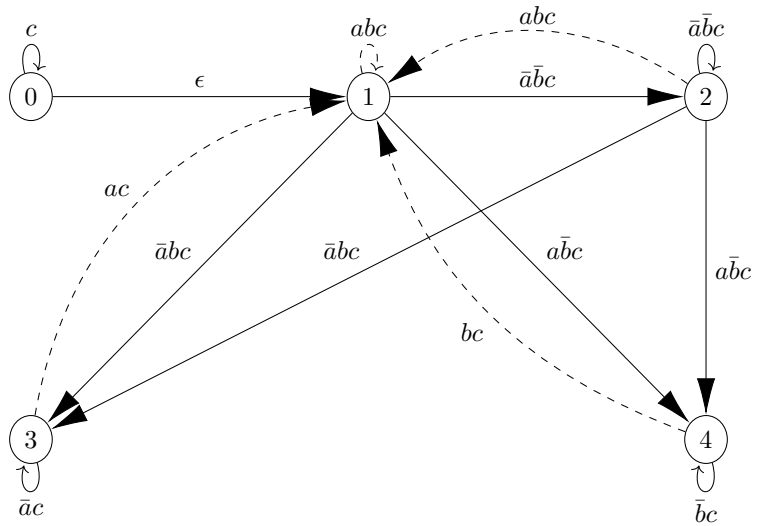
Class of specifications

- Linear Temporal Logic (LTL): safety ($\Box\varphi$), reachability ($\Diamond\varphi$), infinitely often ($\Box\Diamond\varphi$), eventually always ($\Diamond\Box\varphi$), Boolean combinations
- example: the cart pole visits A infinitely often and visits B infinitely often and always operates in the safe region C: $\psi = \Box\Diamond a \wedge \Box\Diamond b \wedge \Box c$.

Class of specifications

- Linear Temporal Logic (LTL): safety ($\Box\varphi$), reachability ($\Diamond\varphi$), infinitely often ($\Box\Diamond\varphi$), eventually always ($\Diamond\Box\varphi$), Boolean combinations
- example: the cart pole visits A infinitely often and visits B infinitely often and always operates in the safe region C: $\psi = \Box\Diamond a \wedge \Box\Diamond b \wedge \Box c$.

- large body of literature on verifying LTL specs on finite-state models (TS, MC, MDP)
- generally requires translation to an automaton
- non-deterministic automata are not suitable for probabilistic systems
- deterministic automata has complex accepting condition (e.g., Rabin)
- recent results on Limit Deterministic Büchi Auomaton



[Hahn et al., CONCUR15], [Sickert et al., CAV16], [Kretinsky et al., ATVA18]

Proposed solution

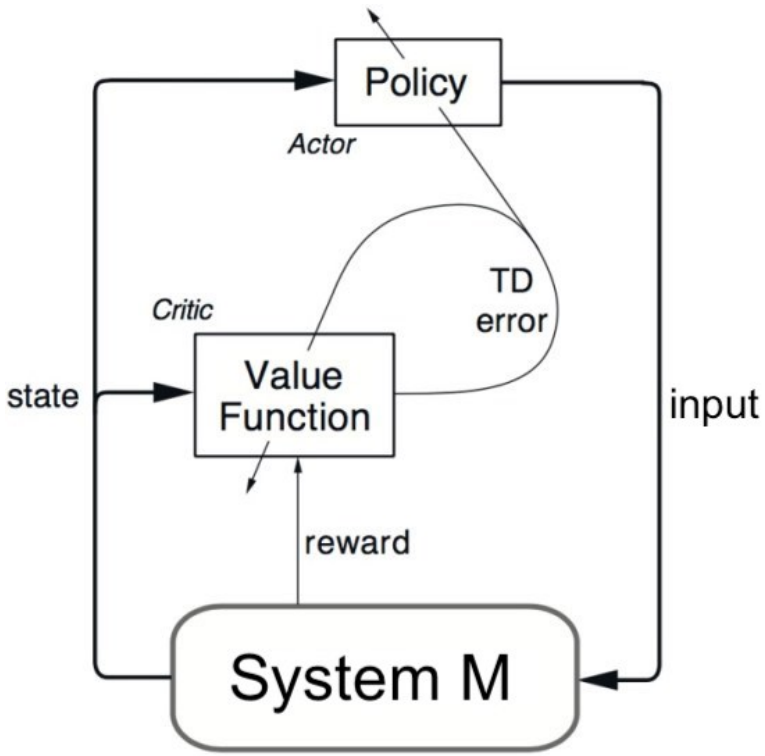
Objective

Given a black-box model M, and the LTL spec ψ, find a policy that maximises

$$\max \mathsf{Prob}(M \models \psi)$$

Reinforcement Learning (RL):

- used extensively in ML community
- handles objectives with additive rewards



Proposed solution

Objective

Given a black-box model M, and the LTL spec ψ, find a policy that maximises

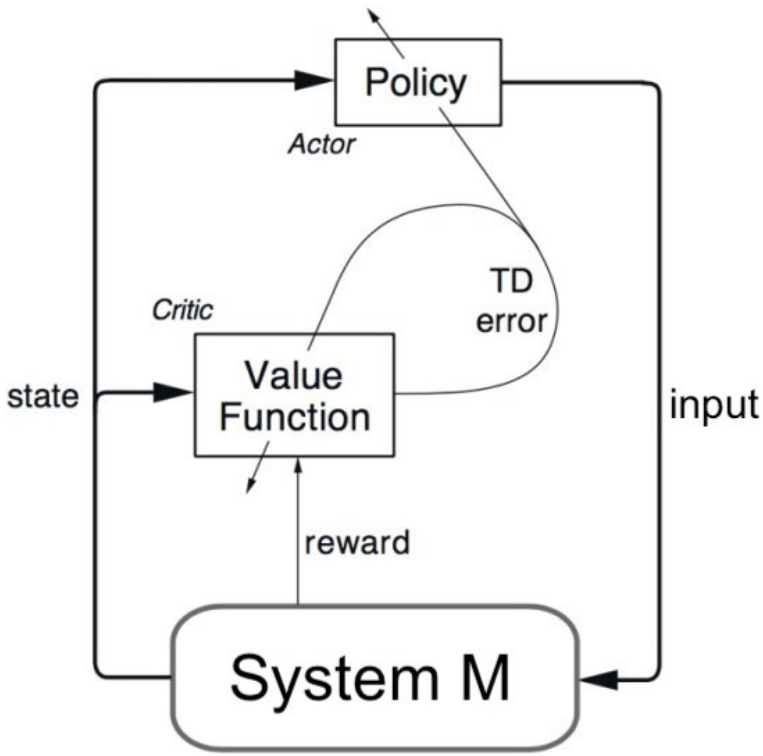
$$\max \mathsf{Prob}(M \models \psi)$$

Reinforcement Learning (RL):

- used extensively in ML community
- handles objectives with additive rewards

Our approach:

- approximate satisfaction probability with reachability
- cast reachability as an objective function with additive reward
- apply model-free RL



[Hahn et al., TACAS19], [Bozkurt et al.,ICRA20], [Hasanbeig et al., FORMATS20]

Approximation with reachability

1. modify the automaton:
   - add a sink state ϕ to the automaton
   - whenever the system wants to take an accepting transition, it jumps to the sink state ϕ with probability $(1 - \zeta)$
2. solve reachability to the sink state: $\Diamond \phi$

Approximation with reachability

1. modify the automaton:
   - add a sink state ϕ to the automaton
   - whenever the system wants to take an accepting transition, it jumps to the sink state ϕ with probability $(1-\zeta)$
2. solve reachability to the sink state: $\Diamond\phi$

Assumption

Define τ_ψ as the number of times the accepting transitions are visited conditioned on having it as a finite number. The quantity $\mathbb{E}(\tau_\psi)$ is bounded.

This assumption holds for any M with finite state space.

Approximation with reachability

1. modify the automaton:
   - add a sink state ϕ to the automaton
   - whenever the system wants to take an accepting transition, it jumps to the sink state ϕ with probability $(1-\zeta)$
2. solve reachability to the sink state: $\Diamond\phi$

Assumption

Define τ_ψ as the number of times the accepting transitions are visited conditioned on having it as a finite number. The quantity $\mathbb{E}(\tau_\psi)$ is bounded.

This assumption holds for any M with finite state space.

Theorem

$$Prob(M_\zeta \models \Diamond\phi) - (1-\zeta)\mathbb{E}(\tau_\psi) \leq Prob(M \models \psi) \leq Prob(M_\zeta \models \Diamond\phi)$$

Approximation with reachability

1. modify the automaton:
   - add a sink state ϕ to the automaton
   - whenever the system wants to take an accepting transition, it jumps to the sink state ϕ with probability $(1-\zeta)$
2. solve reachability to the sink state: $\Diamond\phi$

Assumption

Define τ_ψ as the number of times the accepting transitions are visited conditioned on having it as a finite number. The quantity $\mathbb{E}(\tau_\psi)$ is bounded.

This assumption holds for any M with finite state space.

Theorem

$$Prob(M_\zeta \models \Diamond\phi) - (1-\zeta)\mathbb{E}(\tau_\psi) \leq Prob(M \models \psi) \leq Prob(M_\zeta \models \Diamond\phi)$$

- *Corollary 1: Solution of the learning algorithm converges to the optimal solution when $\zeta \to 1$.*
- *Corollary 2: Using the automaton of the negation of ψ, the learning algorithm provides a lower bounds for the optimal satisfaction probability.*

Specification-guided learning

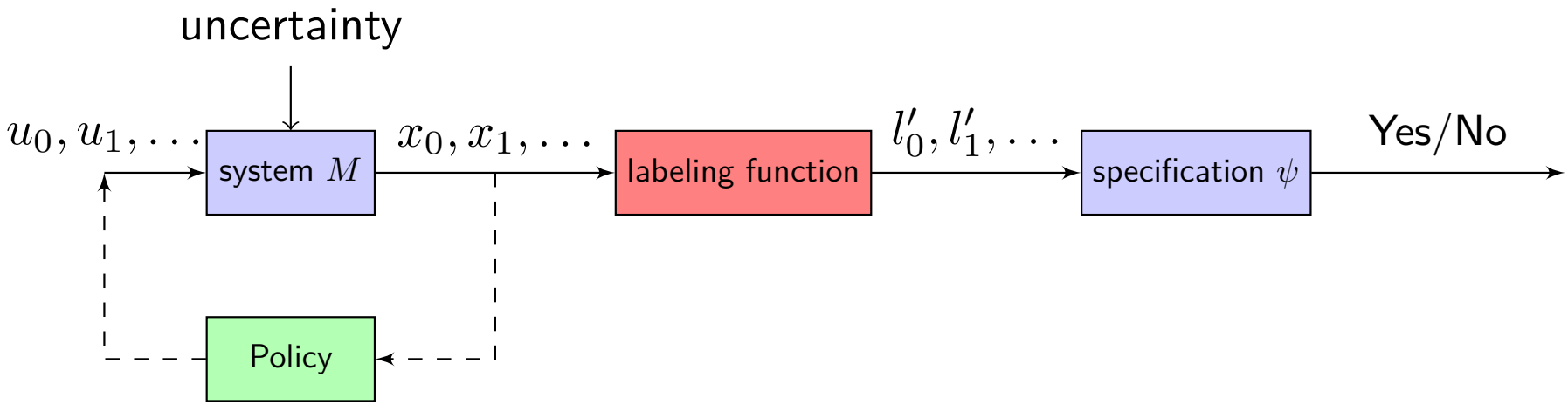
RL might require a substantial time to converge:

- need to see paths that reach the sink state ϕ
- the number of such paths become very small when $\zeta \to 1$
- very complex, sensitive, or contradicting tasks

Specification-guided learning

RL might require a substantial time to converge:

- need to see paths that reach the sink state ϕ
- the number of such paths become very small when $\zeta \to 1$
- very complex, sensitive, or contradicting tasks



- keep the structure of the specification the same
- consider a sequence of relaxed labeling functions
  - e.g., start from an enlarged safe set and make it smaller
  - generally use the positive normal form of the specification
- start with a small ζ and sequentially make it closer to one

Implementation results: cart-pole system

- spec: always in the safe position $C_1 = [-1, 1]$
  pole always upright $C_2 = [-12^\circ, 12^\circ]$
  reach location $A = [0.4, 1]$
- expanded sets for A: $[-1, 1], [0.01, 1], [0.4, 1]$
- Actor network with 7 inputs (4 real states and 3 discrete states of the automaton), 2 outputs, and two hidden layers each with 7 nodes
- $\zeta = 0.999$, learning rate 8×10^{-4}, and episode horizon $N = 500$

$$\begin{cases} x^1_{k+1} = x^1_k + \Delta x^2_k \\ x^2_{k+1} = x^2_k + \Delta a_3 \\ x^3_{k+1} = x^3_k + \Delta x^2_k \\ x^4_{k+1} = x^4_k + \Delta a_2 + w_k, \end{cases}$$

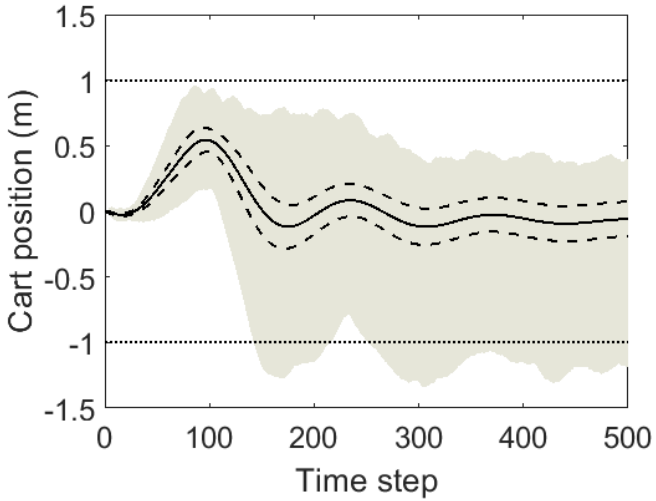
$$\begin{cases} a_3 := a_1 - \dfrac{l a_2 \cos(x^3_k)}{(M+m)} \\ a_2 := \dfrac{g \sin(x^3_k) - \cos(x^3_k) a_1}{l(\frac{4}{3} - m(\cos(x^3_k))^2/(M+m))} \\ a_1 := \dfrac{u_k + l(x^4_k)^2 \sin(x^3_k)}{M+m}. \end{cases}$$

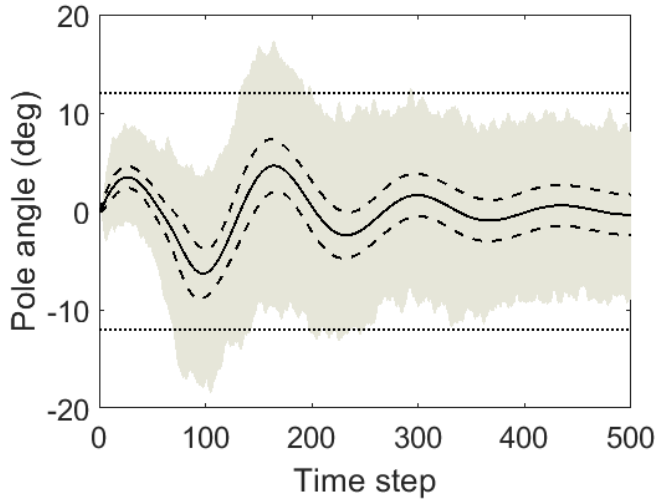
Implementation results: cart-pole system

- spec: always in the safe position $C_1 = [-1, 1]$
  pole always upright $C_2 = [-12^\circ, 12^\circ]$
  reach location $A = [0.4, 1]$
- expanded sets for A: $[-1, 1], [0.01, 1], [0.4, 1]$
- Actor network with 7 inputs (4 real states and 3 discrete states of the automaton), 2 outputs, and two hidden layers each with 7 nodes
- $\zeta = 0.999$, learning rate 8×10^{-4}, and episode

$$\begin{cases} x^1_{k+1} = x^1_k + \Delta x^2_k \\ x^2_{k+1} = x^2_k + \Delta a_3 \\ x^3_{k+1} = x^3_k + \Delta x^2_k \\ x^4_{k+1} = x^4_k + \Delta a_2 + w_k, \end{cases}$$

$$\begin{cases} a_3 := a_1 - \dfrac{l a_2 \cos(x^3_k)}{(M+m)} \\ a_2 := \dfrac{g \sin(x^3_k) - \cos(x^3_k) a_1}{l(\frac{4}{3} - m(\cos(x^3_k))^2/(M+m))} \\ a_1 := \dfrac{u_k + l(x^4_k)^2 \sin(x^3_k)}{M+m}. \end{cases}$$





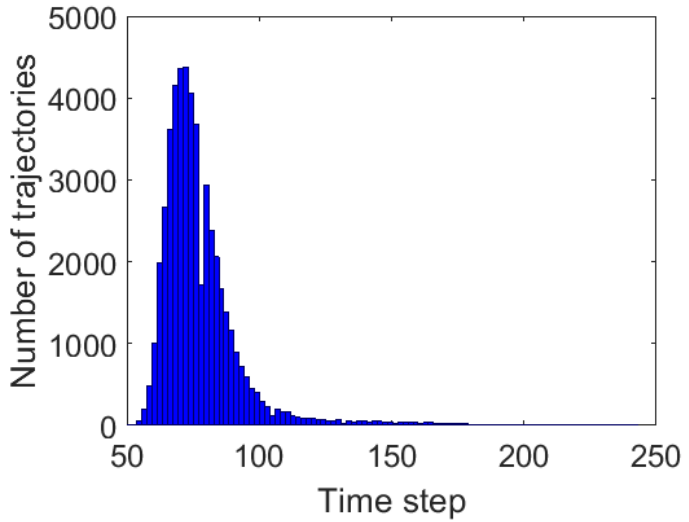
- RL learns the policy within 44 minutes and gives the lower bound 95.26%

Implementation results

Cart-pole system:

histogram of the first time paths reach $A = [0.4, 1]$

most of the paths reach A within 150 steps

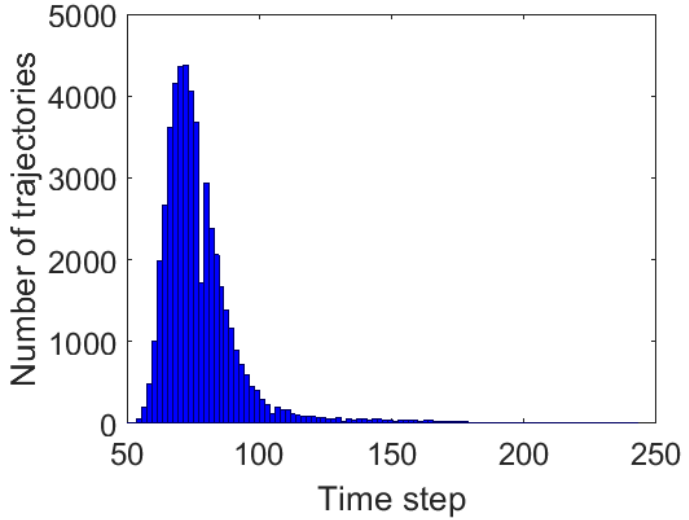


Implementation results

Cart-pole system:

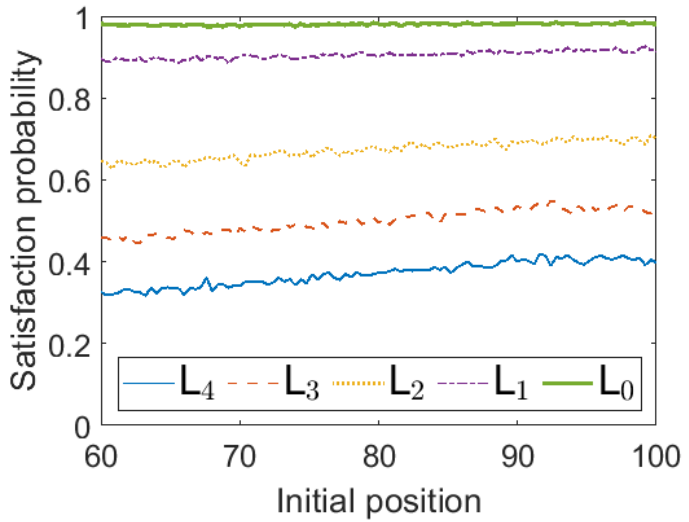
histogram of the first time paths reach $A = [0.4, 1]$

most of the paths reach A within 150 steps



Boat driving:

$$
\begin{aligned}
x_{k+1} &= \min(200, \max(0, x_k + v_{k+1}\cos(\delta_{k+1}))) \\
y_{k+1} &= \min(200, \max(0, y_k - v_{k+1}\sin(\delta_{k+1}) - E(x_{k+1}, w_k)) \\
\delta_{k+1} &= \delta_k + I\Omega_{k+1} \\
\Omega_{k+1} &= \Omega_k + (\omega_{k+1} - \Omega_k)(v_{k+1}/v_{\max}) \\
v_{k+1} &= v_k + I(v_{\text{desired}} - v_k) \\
\omega_{k+1} &= \min(\max(p(u_k - \delta_k), -45^\circ), 45^\circ),
\end{aligned}
$$



Spec: reach from the left bank to the right bank

- target range $[50, 150]$, $[80, 120]$, $[85, 115]$, $[90, 110]$, and $[95, 105]$
- adaptively increase the value of ζ : $0.9950, 0.9965, 0.9980, 0.9995, 0.9999$

Limitations

- What we provide:
  - policy synthesis for systems under uncertainty
  - handling systems with continuous state spaces
  - using RL to satisfy a specification
  - convergence guarantee under suitable assumption
  - guaranteed lower bound on the optimal satisfaction probability
  - specification-guided learning

Limitations

- What we provide:
  - policy synthesis for systems under uncertainty
  - handling systems with continuous state spaces
  - using RL to satisfy a specification
  - convergence guarantee under suitable assumption
  - guaranteed lower bound on the optimal satisfaction probability
  - specification-guided learning

- Limitation:
  - episodic learning: trajectories of finite length, horizon of the learning is sufficiently large
  - discounted objective with the discounting factor converges to 1 from below

Outline

Discounted vs average RL

Discounted-reward RL: $\mathbb{E}\left[\sum_{n=0}^{\infty} \lambda^n r(x_n)\right]$

– value short-term over long-term performance
– slow convergence and instability for discount factor close to 1
– solution depends on the distribution of the initial state

Average RL: $\lim_N \frac{1}{N}\mathbb{E}\left[\sum_{n=0}^{N} r(x_n)\right]$

– convergence requires communicating assumption
– Assumption 1: Episodic resetting is unavailable (life-long learning)
– Assumption 2: Absolute liveness properties (the satisfaction is the same by adding any finite prefix)

Objective and results

Objective

Given an unknown communicating MDP M and an absolute liveness property ψ, design an optimal policy maximising the satisfaction probability of ψ on M.

We design a reward structure R such that an optimal positional policy maximising the average reward for $M \times R$ provides a finite memory policy maximising the satisfaction probability of ψ in M:

Objective and results

Objective

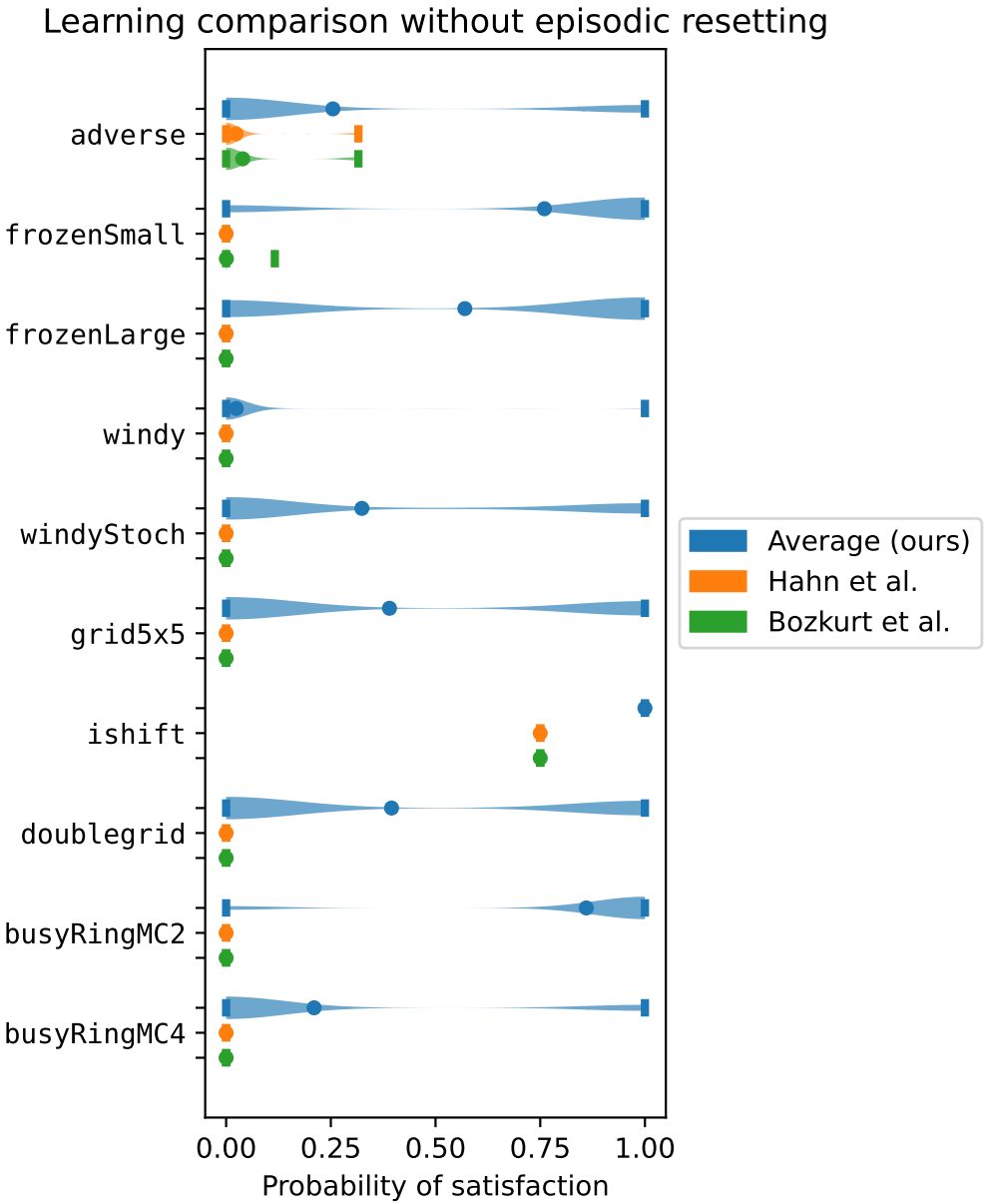
Given an unknown communicating MDP M and an absolute liveness property ψ, design an optimal policy maximising the satisfaction probability of ψ on M.

We design a reward structure R such that an optimal positional policy maximising the average reward for $M \times R$ provides a finite memory policy maximising the satisfaction probability of ψ in M:

- Translate the absolute liveness property into an automaton
- Modify the automaton by adding reset actions ϵ
- Construct the reward machine:

$$r(q, a, q') = \begin{cases} c & \text{if } a = \epsilon \\ 1 & \text{if } (q, a, q') \in F \\ 0 & \text{otherwise.} \end{cases}$$

Experiments



How do previous approaches perform in continuing setting?

- We select a wide distribution over hyperparameters for each method
- We sample 200 hyperparameter combination for each method
- We train for 10 million training step for each combination

[Hahn et al., TACAS19], [Bozkurt et al., ICRA20]

Experiments

Name	states	prod.	time (s)	time†	time‡	c	ε	α	η	train-steps
`adverse`	202	507	8.51	7.09	12.56	-150		0.2		10M
`frozenSmall`	16	64	0.99	20.23	9.88					500k
`frozenLarge`	64	256	4.07	3.88	8.79			0.02	0.02	3M
`windy`	123	366	1.40	1.81	2.61		0.95	0.5	0.05	1M
`windyStoch`	130	390	2.97	3.91	2.53			0.5		2M
`grid5x5`	25	100	0.62	1.12	1.02			0.5		200k
`ishift`	4	29	0.03	0.01	0.02					10k
`doublegrid`	1296	5183	16.43	3.45	3.09	-2	0.5	0.05	0.01	12M
`busyRingMC2`	72	288	0.03	0.03	0.03				0.01	10k
`busyRingMC4`	2592	15426	6.08	3.94	2.33				0.01	1.5M

- The default parameters are $c = -1$, $\varepsilon = 0.1$, $\alpha = 0.1$, and $\eta = 0.1$.
- Superscript † indicates results from Q-learning with reduction from [Hahn et al., TACAS19], while superscript ‡ indicates Q-learning with reduction from [Bozkurt et al., ICRA20].
- Results for † and ‡ required episodic resetting.

Outline

Conclusion and future directions

- Conclusions:
  - Learning in discrete space for continuous models (ICCPS'21)
  - Going from LTL specification to reachability and using discounted RL (iFM'20)
  - Using Average reward instead of discounted reward (AMAS'22)

- Future directions:
  - Assume-guarantee RL with communication between agents
  - Model-free assume-guarantee learning (Learning A/G under probabilistic transitions)
  - Find average RL algorithms with convergence guarantees beyond (weakly) communicating MDPs

Supports from

EIC Horizon Europe, ERC, EPSRC, NU, MPI-SWS

European Research Council

Established by the European Commission

European Innovation Council

Engineering and Physical Sciences Research Council

MAX PLANCK INSTITUTE FOR SOFTWARE SYSTEMS

Thanks for your attention!

sadegh.soudjani@ncl.ac.uk

https://hycodev.com/

HyCoDeV Lab

AMBER Group