Assuring Autonomy in Contested Environments

Attack-Resilient Design

Miroslav Pajic
Cyber-Physical Systems Lab (CPSL)
Pratt School of Engineering
Duke University
Security-Aware Design of Autonomous Systems

• Physical world abides by the laws of physics!
• Physical interfaces introduce new attack vectors!

• How can we exploit limited knowledge of laws of physics (system model) for control and attack detection/identification

• Attack-Resilient design with uncertainty, resource/platform constraints, as well as varying (especially high) levels of autonomy
  – How much can the attacker exploit modeling limitation?
  – How can we effectively exploit physics to improve guarantees in the presence of attacks?
Security-Aware Control for Autonomous Systems

<table>
<thead>
<tr>
<th>Control Stack</th>
<th>Control view</th>
<th>Modeling view</th>
<th>Adding Resiliency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mission Planner</td>
<td>Long-horizon views</td>
<td></td>
<td>[ICRA19, ICRA20a, ICRA20b*, CAV’19a, THMS19]</td>
</tr>
<tr>
<td>Tactical Planner</td>
<td>Short-horizon views</td>
<td></td>
<td>[CDC19a, CDC19b, TAC19*, TII19*]</td>
</tr>
</tbody>
</table>
| Low-level Control| Continuous/discrete control with constraints | $f_r(x(t)) = \int_0^T q(x(t), h(t)) dt + \int_0^T \|x(t)\|^2 dt$, \begin{align*}
\min & \quad f_r(x_r(t)) + f_h(x_h(t)) \\
\text{s.t.} & \quad x_r(t) = x_h(t), \quad u_r(t) = u_h(t),
\end{align*}$ | [TAC19a, TAC19b, TCPS20*, ACC20*, AUT20a*, AUT19*, AUT18, TECS17, RTSS17, TCNS17, CSM17, CDC17, CDC18, ...] |
| Vehicle         |                                   |                                |                                                                                  |

Our Goal: Add resiliency to controls across different/all levels of control stack
Attack-resilient State Estimation

• Attack-resilient control of Cyber-Physical Systems
  – Idea: Design attack-resilient state estimators

• Attack model
  – Goal: force the system into an unsafe state by creating a discrepancy between states and the estimates
  – Attacker has the ability to inject any signal using the compromised sensors
  – Attacker has full system knowledge and unlimited computational power

• Attacks on sensors in $\mathcal{K} = \{s_{i_1}, \ldots, s_{i_q}\} \subseteq S$
  – modeled with attack vector $e_k$
  – $e_{k,i} \neq 0 \iff$ sensor $s_i$ is under attack at time $k$
Attack-resilient State Estimation

- Attack-resilient control of Cyber-Physical Systems
  - Idea: Design attack-resilient state estimators

- Attack model
  - Goal: force the system into an unsafe state by creating a discrepancy between states and the estimates
  - Attacker has the ability to inject any signal using the compromised sensors
  - Attacker has full system knowledge and unlimited computational power

- Attacks on sensors in $\mathcal{K} = \{s_1, ..., s_n\} \subseteq S$
  - modeled with attack vector $e_k$
  - $e_{k,i} \neq 0 \iff$ sensor $s_i$ is under attack at time $k$
Attack-Resilient State Estimation for Noisy Dynamical Systems

- Consider an initial state $x_0$ and attack vectors from $\tilde{e}$

$$\begin{align*}
P_0 : & \quad \min_{\tilde{e}, x} \|\tilde{e}\|_{l_2, l_0} \\
& \quad s.t. \quad \tilde{y} - Ox_0 - \tilde{e} = 0
\end{align*}$$

- Goal: guarantees for $P_{0,\omega}$ and $P_{1,\omega}$ based estimators
  - Bounds on the state estimation errors
  - Sound attacked sensor identification

$$\begin{align*}
P_{0,\omega} : & \quad \min_{\tilde{e}, x} \|\tilde{e}\|_{l_2, l_0} \\
& \quad s.t. \quad \tilde{y} - O x_0 - \tilde{e} = \tilde{w} \\
& \quad \quad \tilde{w} \in \Omega
\end{align*}$$

$$\begin{align*}
P_{1,\omega} : & \quad \min_{\tilde{e}, x} \|\tilde{e}\|_{l_2, l_1} \\
& \quad s.t. \quad \tilde{y} - O x_0 - \tilde{e} = \tilde{w} \\
& \quad \quad \tilde{w} \in \Omega
\end{align*}$$

[ICCPS’14 – Best paper award, CDC15, IEEE CSM’17, IEEE TCNS’17]
Scalable and Optimal Graph-Search Method for RSE

Consider an initial state $x_0$ and attack vectors from $\tilde{e}$

$P_0 : \min_{\tilde{e}, x} \| \tilde{e} \|_{l_2, l_0}$

$s.t. \quad \tilde{y} - Ox_0 - \tilde{e} = 0$

$P_{0,\omega} : \min_{\tilde{e}, x} \| \tilde{e} \|_{l_2, l_0}$

$s.t. \quad \tilde{y} - Ox_0 - \tilde{e} = \tilde{w}$

$\tilde{w} \in \Omega$

Graph capturing possible sensor attack assignments

System Model With Attacks

Plant

\[ x_{k+1} = f(x_k, u_k) + w_k \]
\[ y_k = g(x_k) + e_k + v_k \]

Controller

Estimator

Network

Intrusion Detector

Alarm

Network attack equation:

\[ e_{ik} = x_{ik} - \hat{x}_{ik} \]
Can Attacker Reach Any State?

$$x_{k+1} = Ax_k + Bu_k + w_k$$
$$y_k = Cx_k + a_k + v_k$$

$$supp(a_k) = \mathcal{K}$$
$$a_{k,i} = 0, \forall i \in \mathcal{K}^C$$

Theorem 1 [1,2,3,4*]:
A system presented above is perfectly attackable if and only if it is unstable, and at least one eigenvector $v$ corresponding to an unstable mode satisfies $supp(Cv) \subseteq \mathcal{K}$ and $v$ is a reachable state of the dynamic system.

Physical detectors cannot always protect us from an intelligent attacker...

Can data authentication help?

Can Attacker Reach Any State?

\[ x_{k+1} = Ax_k + Bu_k + w_k \]
\[ y_k = Cx_k + a_k + v_k \]

**Theorem 1 [1,2,3,4*]:**
A system presented above is perfectly attackable if and only if it is unstable, and at least one eigenvector \( v \) corresponding to an unstable mode satisfies \( supp(a_k) = \mathcal{K} \) and \( v \) is a reachable state of the dynamic system.

**Theorem:** A system \( \Sigma \) with a global data integrity police \( \mu(L) \) is not perfectly attackable.
Reachable region of the state estimation error under attack \([1,2,3]\)

\[ \mathcal{R}[k] = \left\{ e \in \mathbb{R}^n \left| ee^T \leq E[e^a[k]]E[e^a[k]]^T + \gamma \text{Cov}(e^a_k) \right\} \]

\[ e^a[k] = e^a_k(a_{1...k}), a_{1...k} \in \mathcal{A}_k \]

\( \mathcal{A}_k \) is the set of all stealthy attacks

\[ \mathbf{a}_{1...k} = [a[1]^T ... a[k]^T]^T \]

\( e^a_k(a_{1...k}) \) is the estimation error evolution due to attack \( a_{1...k} \)
Integrity enforcement policy ensures attacker’s influence is zeroed at enforcement points

Data integrity enforcement policy \((\mu, l)\) where \(\mu = \{t_k\}_{k=0}^{\infty}\) with \(t_{k-1} < t_k, \forall k > 0\) and \(l = \sup_{k>0} t_k - t_{k-1}\) ensures that \(a_{1\ldots k} = 0, \forall k \geq 0\)

This means that at points of authentication \(y_{i}^{\text{net,a}}[k] = y_{i}^{a}[k]\)
Security-Aware Design Framework

Physical Model: \( \Sigma_i, (f_i, g_i) \)

Attack Model: \( \mu(l) \)

Attack impact evaluation: \( R^l[k] \)

QoC Degradation \( \rightarrow \) Security overhead: \( J_i(l) \)

Platform Model: Task model

Resource allocation / Scheduling

QoC guarantees under attack
Platform-aware Execution/Integration of Cyber-Physical Security Components

Constrained computation and communication resources limit the full use of developed cyber-physical techniques.

Our Goal: Provide quantitative tradeoff procedure to map security-aware modules onto available architecture.

[CMS17, TECS/EMSOFT17, RTSS17, TCPS*19]
Our Goal: Add resiliency to controls across different/all levels of control stack
On the higher level, CPS is abstracted by *discrete event systems*, namely, *finite state models* driven by *discrete events*.

**Desired Model** $\mathcal{D} \subseteq \mathcal{P}$ is *controllable* without attacks.

- **Attack on Sensors**
  - Violation if $F$ is replaced by $S$
- **Attack on Actuators**
  - Violation if an attacker removes $L$
- **Attack on Communications**
  - Empty symbol to start

**Idea:** Model Attacks as *Finite State Transducers (FSTs)*

Input: $LRLR \ldots$
Using FSTs to Model Attacks

1. Attacks usually have patterns.
2. All possible attacks captured with nondeterminism
3. FST models can be built from partial information on the attackers to overapproximate.
4. Attack models even unknown, may be inferred from executions.
Attack-Resiliency $\iff$ Controllability Under Attacks

Not any desired model $\mathcal{D}$ is controllable!

**Controllability Theorem:** For desired Model $\mathcal{D} \subseteq \mathcal{P}$

1. The minimal controllable model containing $\mathcal{D}$ is
   \[ \mathbf{\tilde{D}} = \mathcal{A}^{-1}_I \circ \mathcal{A}_I \circ \mathcal{D} \]
   achieved by the supervisor when observable
   \[ S = \mathcal{A}^{-1}_O \circ \mathcal{D} \circ \mathcal{A}_I^{-1}. \]

2. The maximal controllable model contained in $\mathcal{D}$ is
   \[ \mathcal{D} = \mathcal{D} \setminus \mathcal{A}^{-1}_I \circ \mathcal{A}_I \circ (\mathcal{A}^{-1}_I \circ \mathcal{A}_I)^\infty \circ \mathcal{D} \setminus \mathcal{D}, \]
   achieved by the supervisor when observable
   \[ S = \mathcal{A}^{-1}_O \circ \mathcal{D} \circ \mathcal{A}_I^{-1}. \]

The desired model is controllable if and only if $\mathcal{D} = \mathbf{\tilde{D}} = \mathcal{D}$.

---

Model subtraction $C = A \setminus B$ if $C \subseteq A$ and $B, C$ share no common I/O sequences.

---

Controllability Theorem: For desired Model $\mathcal{D} \subseteq \mathcal{P}$

1. The minimal controllable model containing $\mathcal{D}$ is
   \[ \tilde{\mathcal{D}} = \mathcal{A}_I^{-1} \circ \mathcal{A}_I \circ \mathcal{D} \]
   achieved by the supervisor when observable
   \[ \mathcal{S} = \mathcal{A}_O^{-1} \circ \mathcal{D} \circ \mathcal{A}_I^{-1}. \]

2. The maximal controllable model contained in $\mathcal{D}$ is
   \[ \underline{\mathcal{D}} = \mathcal{D} \setminus \mathcal{A}_I^{-1} \circ \mathcal{A}_I \circ (\mathcal{A}_I^{-1} \circ \mathcal{A}_I)^\infty \circ \mathcal{D} \setminus \mathcal{D}, \]
   achieved by the supervisor when observable
   \[ \mathcal{S} = \mathcal{A}_O^{-1} \circ \mathcal{D} \circ \mathcal{A}_I^{-1}. \]

The desired model is controllable if and only if $\underline{\mathcal{D}} = \tilde{\mathcal{D}} = \mathcal{D}$.

Toolbox: ARSC for Synthesis of Attack-Resilient Supervisory Control
Modeling Intermittent Authentication in DES

- Activated by supervisor when necessary
- Not consecutively
- Transmit anchoring word $\leq l_1$ and recovering word $\leq l_2$

Can only accept or repair symbols

Received: $i_1i_5i_1i_2 \ldots$
Recovered: $i_1i_2i_3i_4i_1i_2 \ldots$

Want: $i_1i_2i_3i_4i_1i_2 \ldots$
Send attack-resilient: $i_1i_5i_1i_2 \ldots$

[CDC19b]
Resiliency with Intermittent Authentication

\((l_1, l_2)\)-Accessibility: For models \(N \subseteq M\), \(M\) is \((l_1, l_2)\)-accessible from \(N\) if

1. The graph subtraction \(M/N\) is a tree, with longest path \(\leq l_2\).
2. For any such path, there is a path \(\leq l_1\) with same start and end in the graph of \(N\).

Want: \(i_1i_3i_1i_2i_3\ ...
Send attack-resilient word: \(i_1i_2i_3i_1i_2i_3\ ...
Received: \(i_1i_2i_3i_1i_2i_3\ ...
Recovered: \(i_1i_3i_1i_2i_3\ ...

\[\text{Desired} \quad \text{Maximal controllable sub-model}\]

Controllability Theorem with Intermittent Authentication: The desired model \(\mathcal{D}\) is controllable if and only if it is \((l_1, l_2)\)-accessible from \(\mathcal{D}\). [CDC19b]
Real-Time Enforcement of Regular Specifications
Assuring safe control execution in the age of AI

Challenge 1: Given the set of possible corrupted controls $\mathcal{D}$, how to revise any corrupted control $I_2 \in \mathcal{D}$ with minimal cost to some safe control $I_3 \in \mathcal{K}$

Challenge 2: Given the attack model $\mathcal{A}$, how to repair any corrupted control $I_2 \in \mathcal{D}$ with minimal cost to some control $I_3 \in \mathcal{K}$ that is indistinguishable from $I_1$
Security-Aware Control for Autonomous Systems

Control Stack

Mission Planner
Tactical Planner
Low-level Control
Vehicle

Control view

Long-horizon views
Short-horizon views
Continuous/discrete control with constraints

Modeling view

\[ f_r(x(t)) = \int_0^T g(x(t), u(t)) dt + \int_0^T \|x(t)\|^2 dt, \]
\[ \min f_r(x_r(t)) + f_h(x_h(t)) \]
\[ \text{s.t. } x_r(t) = x_h(t), \quad u_r(t) = u_h(t) \]

Map of Duke University with marked locations:
- Duke Chapel
- Wallace Wade Stadium
- North Building
- Washington Duke
DAG | Hidden-Information Semantics

**UAV Model**

\[
\begin{align*}
pl &= uav \\
fly! &
\end{align*}
\]

\[
\begin{align*}
d &
\in A_{\text{uav}} \\
x_B &= x_B + \Delta x(d) \\
d_B &= d \\
pl &= adv
\end{align*}
\]

**Adversary Model**

\[
\begin{align*}
pl &= adv \\
\mathcal{M}_{\text{adv}} &
\end{align*}
\]

\[
\begin{align*}
x_T &= x_T + \Delta x(f(z)) \\
pl &= as
\end{align*}
\]

**Advisory System Model**

\[
\begin{align*}
pl &= as \\
pl &= uav \\
pl &= adv
\end{align*}
\]

\[
\begin{align*}
update &
\end{align*}
\]

\[
\begin{align*}
x_B &= x_f \\
pl &= uav \\
pl &= adv
\end{align*}
\]

**Information inside this box is oftentimes unknown, i.e., hidden**

Off-the-shelf model checkers do NOT support hidden variables
Strategies CANNOT be synthesized based on hidden information
Approach: Delaying Actions

Information is hidden from one player (H-UAV) by delaying the actions of the other player (ADV).
Delayed-Action Games (DAGs)

**Definition (Delayed-Action Game)**.

A DAG of an HIG \( \mathcal{G}_H = \langle S, (S_1, S_{II}, S_\circ), A, s_0, \beta, \delta \rangle \), is based on an HIG

with players \( \Gamma = \{I, \text{II}, \text{\O}\} \)

over a set of variables \( V = \{v_T, v_B\} \)

is a tuple \( \mathcal{G}_D = \langle \hat{S}, (\hat{S}_1, \hat{S}_{II}, \hat{S}_\circ), A, \hat{s}_0, \beta, \hat{\delta} \rangle \) where

- \( \hat{S} \subseteq \text{Ev}(v_T) \times \text{Ev}(v_B) \times A_{\text{II}}^* \times \mathbb{N}_0 \times \Gamma \)
  partitioned into \( \hat{S}_1, \hat{S}_{II} \) and \( \hat{S}_\circ \);

- \( \hat{s}_0 \in \hat{S}_{II} \) is the initial state;

- \( \hat{\delta}: \hat{S} \times A \times \hat{S} \rightarrow [0, 1] \) is a transition function s.t.
  \[
  \hat{\delta}(\hat{s}_{II}, a, \hat{s}_\circ) = \hat{\delta}(\hat{s}_1, a, \hat{s}_{II}) = \hat{\delta}(\hat{s}_\circ, a, \hat{s}_1) = 0, \text{ and}
  \]
  \[
  \hat{\delta}(\hat{s}_{II}, a, \hat{s}_{II}), \hat{\delta}(\hat{s}_1, a, \hat{s}_1), \hat{\delta}(\hat{s}_1, a, \hat{s}_\circ) \subseteq \{0, 1\},
  \]

- \( \hat{\delta}(\hat{s}_{II}, \theta, \hat{s}_1) \subseteq \{0, 1\} \),

for all \( \hat{s}_1 \in \hat{S}_1, \hat{s}_{II} \in \hat{S}_{II}, \hat{s}_\circ \in \hat{S}_\circ \) and \( a \in A \),

where \( \sum_{\hat{s}' \in \hat{S}_{II}} \delta(\hat{s}_\circ, a, \hat{s}') = 1 \).
DAG Properties

- DAG-HIG simulation relation

**Definition 9 (Game Proper Simulation).** A game $G_D$ properly simulates $G_H$, denoted by $G_D \rightsquigarrow G_H$, iff $\forall \varphi \in \text{Prop}(G_H), \exists \hat{\varphi} \in \text{Prop}(G_D)$ such that $\varphi \sim \hat{\varphi}$.

**Theorem 1 (Probabilistic Simulation).** For any $s_0 \simeq \hat{s}_0$ and $\varphi \in \text{Prop}(G_H)$ where $\text{first}(\varphi) = s_0$, it holds that

$$\Pr [\text{last}(\varphi) = s'] = \Pr \left[ \left( \overline{\text{move}(\varphi)} \right)(\hat{s}_0) = \hat{s}' \right] \quad \forall s', \hat{s}' \text{ s.t. } s' \simeq \hat{s}' .$$

**Theorem 2 (DAG-HIG Simulation).** For any HIG $G_H$ there exists a DAG $G_D = \mathcal{D}[G_H]$ such that $G_D \rightsquigarrow G_H$ (as defined in Def. 9).

- DAG decomposition

**Definition 10 (DAG Subgames).** The subgames of a $G_D$ are defined by the set $\{ \hat{G}_i \mid \hat{G}_i = \langle \hat{S}, (\hat{s}^{(i)}, \hat{s}^{(i)}_{H}, \hat{s}^{(i)}_{\bigcirc}), A, \hat{s}_0^{(i)}, \hat{\delta}^{(i)} \rangle, i \in \mathbb{N}_0 \}$, where $\hat{S} = \bigcup_i \hat{S}^{(i)}$; $\hat{S}_\gamma = \bigcup_i \hat{S}^{(i)}$ $\forall \gamma \in \Gamma$; and $\hat{s}_0^{(i)} = \hat{s}_0^{(i)}$ $\text{s.t. } \hat{s}_0^{(i)} \in \text{Prop}(G_D^{(i)})$, $\hat{s}_0^{(i)} \neq \hat{s}_0^{(j)} \forall i, j \in \mathbb{N}_0$. 
DAG-Based Synthesis

Primary Components $\mathcal{M}_I, \mathcal{M}_{II}, \mathcal{M}_O$

Composition (MC)

$\mathcal{G}$

Composition (MC)

$\mathcal{G}^{\pi_{II}}$

Analysis (MC, $\phi_a$)

Model Refinement

Auxiliary Components $\mathcal{M}_{mrd}, \mathcal{M}_{mwr}$

DAG Construction (Alg. 1, MC)

$\hat{\mathcal{G}}$

Strategy Synthesis (Alg. 2, MC, $\phi_s$)

$\pi_{II}$

MC: Model Checker

$\phi_s$: Synthesis query

$\phi_a$: Analysis query
Case Study

- Model Checker: PRISM-games
Case Study

- Analysis

(a) Geolocation task at stage $h$

(b) Max. no. of geolocation tasks $n$
Security-Aware Human-on-the-Loop Protocols
How can we use human context awareness (in real-time) for security?

Operator
- Set goals
- Supervise mission
- Imagery tasks

Autonomy/automation
- Target assignment
- Trajectory planning
- Attack detection

Adversary
- Effects low-level control

Security-aware protocols
- Exploit human context-awareness for security

Scenarios
- Design of experimental variables
- Generate RESCHU-SA configuration files

Experiments
- Capture HOL behaviors with varying levels of workload and fatigue

Data Mining
- HOL context awareness
- Impacts of workload and fatigue on system performance
Security-aware Human-on-the-Loop Planning
Security-aware Human-on-the-Loop Planning [ICRA’19]
Attack-Resilient Mission Design

- Develop planning methods that will improve attack-detection guarantees by allowing the deployed intrusion detection system to interact with the controller and the rest of the system.

- How to model such interactions? – MDPs, PTAs, SHAs

- Optimization based on solving stochastic games
  - How to incorporate learning?
  - How to incorporate formal guarantees?
Model-free Control Synthesis from LTLs [ICRA20a*]

Problem Statement
Given an MDP $M = (S, A, P, s_0, AP, L)$ where $P$ is fully unknown and an LTL specification $\varphi$, design a model-free RL algorithm that finds a finite-memory objective policy $\pi^\varphi$ that satisfies

$$P_{r_{\pi^\varphi}}(s \models \varphi) = P_{r_{\max}}(s \models \varphi),$$

where $P_{r_{\max}}(s \models \varphi) = \max_{\pi} P_{r_{\pi}}(s \models \varphi)$ for all $s \in S$.

Limit-Deterministic Büchi Automata (LDBA) – consist of two deterministic components the initial and accepting. The only non-de-terministic transitions are the $\epsilon$-moves from the initial component to the accepting components.

(a) A derived LDBA $A$ for the LTL formula $\varphi = \Diamond \alpha \lor \Diamond \beta$

(b) An example MDP $M$; the circles denote MDP states, rectangles denote actions, and numbers transition probabilities

(c) The obtained product MDP
Model-free Control Synthesis from LTLs

Problem Statement
Given an MDP $M = (S, A, P, s_0, AP, L)$ where $P$ is fully unknown and an LTL specification $\varphi$, design a model-free RL algorithm that finds a finite-memory objective policy $\pi^\varphi$ that satisfies

$$Pr_{\pi^\varphi}(s \models \varphi) = Pr_{\pi^{max}}(s \models \varphi),$$

where $Pr_{\pi^{max}}(s \models \varphi) = \max_{\pi}Pr_{\pi}(s \models \varphi)$ for all $s \in S$.

Learning for Büchi conditions
For a given MDP $M$ with $B \subseteq S$, the value function $v_\pi^\gamma$ for the policy $\pi$ and the discount factor $\gamma$ satisfies

$$\lim_{\gamma \to 1^-} v_\pi^\gamma(s) = Pr_{\pi^\varphi}(s \models \varphi)$$

for all states for all $s \in S$ if the return of a path is defined as

$$G_t(\sigma) := \sum_{i=0}^{\infty} R_B(\sigma[t + i]) \prod_{j=0}^{i-1} \Gamma_B(\sigma[t + j])$$

where $\prod_{j=0}^{i-1} := 1$, $R_B : S \to [0,1)$ and $\Gamma_B : S \to (0,1)$ are the reward and the discount functions defined as

$$R_B(s) := \begin{cases} 1 - \gamma_B, & s \in B, \\ 0, & s \notin B, \end{cases} \quad \Gamma_B(s) := \begin{cases} \gamma_B, & s \in B, \\ \gamma, & s \notin B. \end{cases}$$

Here, we set $\gamma_B = \gamma_B(\gamma)$ as a function of $\gamma$ such that

$$\lim_{\gamma \to 1^-} \frac{1 - \gamma}{1 - \gamma_B(\gamma)} = 0.$$
Case Studies

Robot tries to reach a safe absorbing state (states a or b in circle), while avoiding unsafe states (states c).

$$\varphi_1 = (\Diamond a \lor \Diamond b) \land \Box \neg c$$

Nursery Scenario

The robot’s objective is to repeatedly check a baby (at state b) and go back to its charger (at state c), while avoiding the danger zone (at state d).

Near the baby b, the only allowed action is left and when taken the following situations can happen

- the robot hits the wall with probability 0.1, waking up the baby
- the robot moves left with probability 0.8 or moves down with probability 0.1.
- If the baby has been woken up, which means the robot could not leave in a single time step (represented by LTL as b ∧ Ob), the robot should notify the adult (at state a);
- otherwise, the robot should directly go back to the charger (at state c).

(a) Policy c to b  (b) Policy b to c  (c) Policy b to a  (d) Policy a to b

$$\varphi_2 = \Box \left( \neg d \land (b \land \neg \Diamond b) \to \Diamond (\neg b \lor (a \lor c)) \land a \to \Diamond (\neg a \lor b) \right)$$

$$(1) \land (2) \land (3) \land (4) \land (5) \land (6)$$
Synthesis from LTL via Deep Imitative Q-Learning [ICRA20b*]

Gather sensing info, synthesize globally optimal Rabin sequence $\omega^*_1 = q_1q_2q_3\ldots$

Take instruction or not?

Unique and optimal solution! Only depends on LTL.

Yes

No

Run regular exploration policy

Generate Instruction

Gather $(s,a,s',r)$ and store them into buffer. Run RL to synthesize policy

MDP with 1600 states

Current State: $[s_1, q_2]$
Locate $q_2$ in $\omega^*_1$:
$\omega^*_1 = q_1q_2q_3\ldots$
Locate transition t1,t2... from q2 to q3 in the DRA:

Set local LTL as $\phi' = t_1 \lor t_2$
Convert $\phi'$ to DRA' and take product w/ FTS to give instruction

Deep Imitative Reinforcement Learning for Temporal Logic Robot Motion Planning with Noisy Semantic Observations

Qitong Gao, Miroslav Pajic and Michael M. Zavlanos
Duke University
ICRA 20'
Develop planning methods that will improve attack-detection guarantees by allowing the deployed intrusion detection system to interact with the controller and the rest of the system.

How to model such interactions? – MDPs, PTAs, SHAs

Optimization based on solving stochastic games

How to incorporate learning in 2-player hidden information stochastic games?

with formal guarantees...
Thank you