Planning under Partial Observability

A Betrothal of Formal Verification and Machine Learning

Nils Jansen
LearnAut, June 23, 2019
me
joint work with:

Sebastian Junges, Joost-Pieter Katoen, Tim Quatmann, Bernd Becker, Ralf Wimmer, Leonore Winterer, Steve Carr, Ufuk Topcu, Mohamedreza Ahmadi, Roderick Bloem, Bettina Koenighofer, Alexandru Serban
Autonomous (Cyber-physical) Systems

- Safety specification
- Performance specification
- System model
  - Formal verification
  - Model-based Testing
  - Controller synthesis
Autonomous (Cyber-physical) Systems

- Safety specification
- Performance specification
- System model
- Real system
- Formal verification
- Model-based Testing
- Controller synthesis
- Machine learning
Autonomous (Cyber-physical) Systems

System model

Safety specification

Performance specification

Real system

Formal verification

Model-based Testing

Controller synthesis

Machine learning
Autonomous (Cyber-physical) Systems

- Safety specification
- Performance specification

System model

Real system

- Formal verification
- Model-based Testing
- Controller synthesis
- Machine learning

Solutions at the interfaces of domains
Help the Robot

Find the best way to the airbag
Help the Robot

Find the best way to the airbag
Help the Robot

Find the best way to the airbag

Take expensive surfaces into account
Help the Robot

Find the best way to the airbag

Take expensive surfaces into account

Avoid randomly moving dust storm
Help the Robot

Find the best way to the airbag
Take expensive surfaces into account
Avoid randomly moving dust storm

Find safe and/or cost-optimal policy to get to the airbag
Help the Robot

Find the best way to the airbag

Take expensive surfaces into account

Avoid randomly moving dust storm

Find safe and/or cost-optimal policy to get to the airbag

Underlying Model: Markov Decision Process

multi-objective model checking for MDPs
\[ Pr_{\text{max}}(\Diamond s_7) \]
\[ EC_{\text{min}}(\Diamond s_7) \]
MDPs

\[ Pr_{\text{max}}(\Diamond s_7) \]

\[ EC_{\text{min}}(\Diamond s_7) \]

- **efficient** model checking
  - memoryless deterministic strategies suffice for single objectives

  randomized strategies may be needed for multiple objectives
Partial Observability

It's a well known fact that you must spin a USB **three times** before it will fit. From this, we can gather that a USB has three states:

- **Up position**
- **Down position**
- **Superposition**

Until the USB is observed it will stay in the superposition. Therefore it will not fit until observed - except for in cases of USB tunnelling.
Help the Robot with Partial Observability

Robot has restricted range of vision
Help the Robot with Partial Observability

Robot has restricted range of vision

Storm is only observable when near
Help the Robot with Partial Observability

Robot has restricted range of vision

Storm is only observable when near

For robot, storm is either near or far
Help the Robot with Partial Observability

Robot has restricted range of vision

Storm is only observable when near

For robot, storm is either near or far

Belief state: Likelihood of the actual position of the storm

infinite belief MDP
Help the Robot with Partial Observability

<table>
<thead>
<tr>
<th>Robot has restricted range of vision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storm is only observable when near</td>
</tr>
<tr>
<td>For robot, storm is either near or far</td>
</tr>
<tr>
<td>Find strategy that induces</td>
</tr>
<tr>
<td>$Pr_{max}(\neg BUG)$</td>
</tr>
</tbody>
</table>

Belief state: Likelihood of the actual position of the storm

Find safe and/or cost-optimal policy to get to the airbag

infinite belief MDP
POMDPs

\[ Pr_{\text{max}}(\Diamond s_7) \]
\[ EC_{\text{min}}(\Diamond s_7) \]

Choices at observation ‘blue’:
POMDPs

\[ Pr_{max}(\Diamond s_7) \]

\[ EC_{min}(\Diamond s_7) \]

Choices at observation ‘blue’:

- Choose ‘up’ at each state: prob 2/3 to reach \( s_7 \)

memoryless deterministic
POMDPs

\[ Pr_{max}(\Diamond s_7) \]

\[ EC_{min}(\Diamond s_7) \]

Choices at observation ‘blue’:

- Choose ‘up’ at each state: prob $2/3$ to reach $s_7$
  memoryless deterministic

- Choose ‘up’ with prob $0 < p < 1$ and ‘down’ with prob $1 - p$: \[ \frac{2}{3} + \frac{1}{3}p < 1 \]
  memoryless randomized
POMDPs

\[ Pr_{\text{max}}(\diamond s_7) \]

\[ EC_{\text{min}}(\diamond s_7) \]

Choices at observation ‘blue’:

- Choose ‘up’ at each state: prob \( \frac{2}{3} \) to reach \( s_7 \) memoryless deterministic

- Choose ‘up’ with prob \( 0 < p < 1 \) and ‘down’ with prob \( 1 - p \) : \[
\frac{2}{3} + \frac{1}{3}p < 1
\]
memoryless randomized

- Choose ‘up’ if predecessor is ‘yellow’. Otherwise, choose ‘up’ if ‘blue’ observed even number of times, ‘down’ otherwise: prob 1 deterministic with memory
POMDPs - Applications

- Stock
- Surveying
- Health
- Wireless
- Autonomous
- Machine
And by the way: POMDPs and their subclasses form the operational semantics to probabilistic programs with discrete probability distributions.
Computing Policies for POMDPs

- Randomized with infinite memory: undecidable, optimal results.
Computing Policies for POMDPs

- **Randomized with infinite memory**: undecidable, optimal results.
- **Randomized with finite memory**: NP-hard, SQRT-SUM-hard, in PSPACE, not optimal in general, but sufficient for many applications.
Computing Policies for POMDPs

- Randomized with infinite memory: undecidable, optimal results.

- Randomized with finite memory: NP-hard, SQRT-SUM-hard, in PSPACE, not optimal in general, but sufficient for many applications.

- Intuitively: Randomization can often trade off memory.
Stories - Policy Synthesis for POMDPs

1. Game-based abstraction
2. Finite-memory controllers
3. Recurrent neural networks
4. Fun: Humans in the loop

If time permits: Teaser on safe reinforcement learning.
Game-based Abstraction for POMDPs

Robot has restricted range of vision

Storm is only observable when near

For robot, storm is either near or far
Game-based Abstraction for POMDPs

Robot has restricted range of vision

Storm is only **observable** when near

For robot, storm is either **near** or **far**

Abstract possible positions into nondeterministic choices
Game-based Abstraction for POMDPs

Robot has restricted range of vision

Storm is only observable when near

For robot, storm is either near or far

Abstract possible positions into nondeterministic choices

Instead of infinitely many distributions, finite number of choices
Game-based Abstraction for POMDPs

Robot has restricted range of vision

Storm is only **observable** when near

For robot, storm is either *near* or *far*

Abstract possible positions into nondeterministic choices

Instead of infinitely many distributions, finite number of choices

Probabilistic Two-Player Game
Concept: Game-based Abstraction for POMDPs
Concept: Game-based Abstraction for POMDPs

- **Merge states** that share an observation into an abstract state
  - probabilistic movements of storm outside of the visible area
Concept: Game-based Abstraction for POMDPs

- **Merge states** that share an observation into an abstract state
  - probabilistic movements of storm outside of the visible area

- **Introduce choice** over those states
  - position of storm is now determined nondeterministically
Concept: Game-based Abstraction for POMDPs

- **Merge states** that share an observation into an abstract state
  - probabilistic movements of storm outside of the visible area

- **Introduce choice** over those states
  - position of storm is now determined nondeterministically

- Additional level of nondeterminism: **2-Player game**
  - player 2 chooses position of storm
Concept: Game-based Abstraction for POMDPs

- **Merge states** that share an observation into an abstract state
  - probabilistic movements of storm outside of the visible area

- **Introduce choice** over those states
  - position of storm is now determined nondeterministically

- Additional level of nondeterminism: **2-Player game**
  - player 2 chooses position of storm

- **Worst case** analysis
  - opponent can jump → storm is strengthened, spurious movements
Story - Game-based Abstraction for POMDPs

Safety Specification → POMDP
Story - Game-based Abstraction for POMDPs

- Safety Specification
- POMDP
- PG

abstract
Story - Game-based Abstraction for POMDPs

- Safety Specification
- POMDP
- PG
  - abstract
  - model checking/policy synthesis
  - Optimal PG Strategy
Story - Game-based Abstraction for POMDPs

Safety Specification → POMDP → abstract → PG → model checking/policy synthesis → Optimal PG Strategy → on PG → on POMDP → optimum → full observability

\[ [p, p + \tau, Pr_{max}(\neg B U G), u] \]
Refinement

- Usual state splitting as for MDPs is **not possible**
  - we need a one-to-one correspondence with the POMDP

- Remove **spurious movements**

- **History-based** refinement
  - 1-step, multi-step
  - region-based, magnifying lens abstraction

- Alternative: Refine environment → increase range of vision
Correctness and Completeness

Correct, as (refined) PG strategy is an overapproximation.

Not complete, as abstraction may be too coarse.
### Experiments - Comparison to PRISM-POMDP

<table>
<thead>
<tr>
<th>Grid size</th>
<th>POMDP solution</th>
<th>PG solution</th>
<th>Lifting Result</th>
<th>MDP Result</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>States Result</td>
<td>Sol. Time</td>
<td>States Result</td>
<td>Sol. Time</td>
</tr>
<tr>
<td>3 × 3</td>
<td>299 0.8323</td>
<td>0.26</td>
<td>396 0.8323</td>
<td>0.040</td>
</tr>
<tr>
<td>4 × 4</td>
<td>983 0.9556</td>
<td>1.81</td>
<td>1344 0.9556</td>
<td>0.078</td>
</tr>
<tr>
<td>5 × 5</td>
<td>2835 0.9882</td>
<td>175.94</td>
<td>6016 0.9740</td>
<td>0.452</td>
</tr>
<tr>
<td>5 × 6</td>
<td>4390 0.9945</td>
<td>4215.06</td>
<td>7986 0.9785</td>
<td>0.534</td>
</tr>
<tr>
<td>6 × 6</td>
<td>6705 ?</td>
<td>– MO –</td>
<td>10544 0.9830</td>
<td>1.414</td>
</tr>
<tr>
<td>8 × 8</td>
<td>24893 ?</td>
<td>– MO –</td>
<td>23128 0.9897</td>
<td>6.349</td>
</tr>
<tr>
<td>10 × 10</td>
<td>66297 ?</td>
<td>– MO –</td>
<td>40464 0.9914</td>
<td>12.652</td>
</tr>
<tr>
<td>20 × 20</td>
<td>– Time out while modelling –</td>
<td>199144 0.9921</td>
<td>127.356</td>
<td>0.9999</td>
</tr>
<tr>
<td>30 × 30</td>
<td>– Time out while modelling –</td>
<td>477824 0.9921</td>
<td>489.369 – MO –</td>
<td>0.9999</td>
</tr>
<tr>
<td>40 × 40</td>
<td>– Time out while modelling –</td>
<td>876504 0.9921</td>
<td>1726.489 – MO –</td>
<td>0.9999</td>
</tr>
<tr>
<td>50 × 50</td>
<td>– Time out while modelling –</td>
<td>1395184 0.9921</td>
<td>3963.281 – MO –</td>
<td>– MO –</td>
</tr>
</tbody>
</table>
## Experiments - Comparison to PRISM-POMDP

<table>
<thead>
<tr>
<th>Grid size</th>
<th>POMDP solution</th>
<th>PG solution</th>
<th>Lifting</th>
<th>MDP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>States</td>
<td>Result</td>
<td>Sol. Time</td>
<td>States</td>
</tr>
<tr>
<td>3 × 3</td>
<td>299</td>
<td>0.8323</td>
<td>0.26</td>
<td>396</td>
</tr>
<tr>
<td>4 × 4</td>
<td>983</td>
<td>0.9556</td>
<td>1.81</td>
<td>1344</td>
</tr>
<tr>
<td>5 × 5</td>
<td>2835</td>
<td>0.9882</td>
<td>175.94</td>
<td>6016</td>
</tr>
<tr>
<td>5 × 6</td>
<td>4390</td>
<td>0.9945</td>
<td>4215.06</td>
<td>7986</td>
</tr>
<tr>
<td>6 × 6</td>
<td>6705</td>
<td>?</td>
<td>– MO –</td>
<td>10544</td>
</tr>
<tr>
<td>8 × 8</td>
<td>24893</td>
<td>?</td>
<td>– MO –</td>
<td>23128</td>
</tr>
<tr>
<td>10 × 10</td>
<td>66297</td>
<td>?</td>
<td>– MO –</td>
<td>40464</td>
</tr>
<tr>
<td>20 × 20</td>
<td>– Time out while modelling –</td>
<td>199144</td>
<td>0.9921</td>
<td>127.356</td>
</tr>
<tr>
<td>30 × 30</td>
<td>– Time out while modelling –</td>
<td>477824</td>
<td>0.9921</td>
<td>489.369</td>
</tr>
<tr>
<td>40 × 40</td>
<td>– Time out while modelling –</td>
<td>876504</td>
<td>0.9921</td>
<td>1726.489</td>
</tr>
<tr>
<td>50 × 50</td>
<td>– Time out while modelling –</td>
<td>1395184</td>
<td>0.9921</td>
<td>3963.281</td>
</tr>
</tbody>
</table>
# Experiments - Comparison to PRISM-POMDP

<table>
<thead>
<tr>
<th>Grid size</th>
<th>States</th>
<th>Result</th>
<th>Sol. Time</th>
<th>POMDP solution</th>
<th>States</th>
<th>Result</th>
<th>Sol. Time</th>
<th>PG solution</th>
<th>Lifting Result</th>
<th>MDP Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 × 3</td>
<td>299</td>
<td>0.8323</td>
<td>0.26</td>
<td></td>
<td>396</td>
<td>0.8323</td>
<td>0.040</td>
<td>0.8323</td>
<td>0.8323</td>
<td>0.8323</td>
</tr>
<tr>
<td>4 × 4</td>
<td>983</td>
<td>0.9556</td>
<td>1.81</td>
<td></td>
<td>1344</td>
<td>0.9556</td>
<td>0.078</td>
<td>0.9556</td>
<td>0.9556</td>
<td>0.9556</td>
</tr>
<tr>
<td>5 × 5</td>
<td>2835</td>
<td>0.9882</td>
<td>175.94</td>
<td></td>
<td>6016</td>
<td>0.9740</td>
<td>0.452</td>
<td>0.9825</td>
<td>0.9882</td>
<td>0.9882</td>
</tr>
<tr>
<td>5 × 6</td>
<td>4390</td>
<td>0.9945</td>
<td>4215.06</td>
<td></td>
<td>7986</td>
<td>0.9785</td>
<td>0.534</td>
<td>0.9893</td>
<td>0.9945</td>
<td>0.9945</td>
</tr>
<tr>
<td>6 × 6</td>
<td>6705</td>
<td>?</td>
<td>– MO –</td>
<td></td>
<td>10544</td>
<td>0.9830</td>
<td>1.414</td>
<td>0.9933</td>
<td>0.9970</td>
<td>0.9970</td>
</tr>
<tr>
<td>8 × 8</td>
<td>24893</td>
<td>?</td>
<td>– MO –</td>
<td></td>
<td>23128</td>
<td>0.9897</td>
<td>6.349</td>
<td>0.9992</td>
<td>0.9998</td>
<td>0.9998</td>
</tr>
<tr>
<td>10 × 10</td>
<td>66297</td>
<td>?</td>
<td>– MO –</td>
<td></td>
<td>40464</td>
<td>0.9914</td>
<td>12.652</td>
<td>0.9999</td>
<td>0.9999</td>
<td>0.9999</td>
</tr>
<tr>
<td>20 × 20</td>
<td>– Time out while modelling –</td>
<td>199144</td>
<td>0.9921</td>
<td></td>
<td>127.356</td>
<td>0.9999</td>
<td>0.9999</td>
<td>0.9999</td>
<td>0.9999</td>
<td>0.9999</td>
</tr>
<tr>
<td>30 × 30</td>
<td>– Time out while modelling –</td>
<td>477824</td>
<td>0.9921</td>
<td></td>
<td>489.369</td>
<td>– MO –</td>
<td>0.9999</td>
<td>0.9999</td>
<td>0.9999</td>
<td>0.9999</td>
</tr>
<tr>
<td>40 × 40</td>
<td>– Time out while modelling –</td>
<td>876504</td>
<td>0.9921</td>
<td></td>
<td>1726.489</td>
<td>– MO –</td>
<td>0.9999</td>
<td>0.9999</td>
<td>0.9999</td>
<td>0.9999</td>
</tr>
<tr>
<td>50 × 50</td>
<td>– Time out while modelling –</td>
<td>1395184</td>
<td>0.9921</td>
<td></td>
<td>3963.281</td>
<td>– MO –</td>
<td>– MO –</td>
<td>0.9999</td>
<td>0.9999</td>
<td>0.9999</td>
</tr>
</tbody>
</table>
### Experiments - Comparison to PRISM-POMDP

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3 × 3</td>
<td>299</td>
<td>0.8323</td>
<td>0.26</td>
<td>396</td>
<td>0.8323</td>
<td>0.040</td>
<td>0.8323</td>
<td>0.8323</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 × 4</td>
<td>983</td>
<td>0.9556</td>
<td>1.81</td>
<td>1344</td>
<td>0.9556</td>
<td>0.078</td>
<td>0.9556</td>
<td>0.9556</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 × 5</td>
<td>2835</td>
<td>0.9882</td>
<td>175.94</td>
<td>6016</td>
<td>0.9740</td>
<td>0.452</td>
<td>0.9825</td>
<td>0.9882</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 × 6</td>
<td>4390</td>
<td>0.9945</td>
<td>4215.06</td>
<td>7986</td>
<td>0.9785</td>
<td>0.534</td>
<td>0.9893</td>
<td>0.9945</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 × 6</td>
<td>6705</td>
<td>?</td>
<td>– MO –</td>
<td>10544</td>
<td>0.9830</td>
<td>1.414</td>
<td>0.9933</td>
<td>0.9970</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 × 8</td>
<td>24893</td>
<td>?</td>
<td>– MO –</td>
<td>23128</td>
<td>0.9897</td>
<td>6.349</td>
<td>0.9992</td>
<td>0.9998</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 × 10</td>
<td>66297</td>
<td>?</td>
<td>– MO –</td>
<td>40464</td>
<td>0.9914</td>
<td>12.652</td>
<td>0.9999</td>
<td>0.9999</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 × 20</td>
<td>– Time out while modelling –</td>
<td></td>
<td></td>
<td>199144</td>
<td>0.9921</td>
<td>127.356</td>
<td>0.9999</td>
<td>0.9999</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30 × 30</td>
<td>– Time out while modelling –</td>
<td></td>
<td></td>
<td>477824</td>
<td>0.9921</td>
<td>489.369</td>
<td>– MO –</td>
<td>0.9999</td>
<td></td>
<td></td>
</tr>
<tr>
<td>40 × 40</td>
<td>– Time out while modelling –</td>
<td></td>
<td></td>
<td>876504</td>
<td>0.9921</td>
<td>1726.489</td>
<td>– MO –</td>
<td>0.9999</td>
<td></td>
<td></td>
</tr>
<tr>
<td>50 × 50</td>
<td>– Time out while modelling –</td>
<td></td>
<td></td>
<td>1395184</td>
<td>0.9921</td>
<td>3963.281</td>
<td>– MO –</td>
<td>– MO –</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Experiments - Policies

![Graphs showing experiment results and time taken with different numbers of obstacles.](image-url)
Conclusion - Story 1

- First approach to **game-based abstraction** for POMDPs
- Superior **scalability**, no **completeness**
- Future: **Automatic refinement**


Stories - Policy Synthesis for POMDPs

1. Game-based abstraction
2. Finite-memory controllers
3. Recurrent neural networks
4. Fun: Humans in the loop
Story - Finite-state Controllers (FSCs)

- POMDP
- Specification

\( M \)

\( \varphi \)

Probabilistic Temporal Logic Constraints
Story - Finite-state Controllers (FSCs)

POMDP \( \mathcal{M} \)

Policy Synthesis for \( k \) memory states

\( \phi \)

Specification

Probabilistic Temporal Logic Constraints
Story - Finite-state Controllers (FSCs)

- POMDP
  - Policy Synthesis for $k$ memory states
  - Model Checking
    - $\mathcal{M}^\sigma \models \varphi$?

- Specification
  - Probabilistic Temporal Logic Constraints

$\mathcal{M} \models \varphi$
Story - Finite-state Controllers (FSCs)

- POMDP
- Specification
- Probabilistic Temporal Logic Constraints
- Parametric Markov Chain

Policy Synthesis for $k$ memory states

$\mathcal{M} \sigma \models \varphi$?
Story - Finite-state Controllers (FSCs)

POMDP

$\mathcal{M}$

Policy Synthesis for $k$
memory states

$\sigma$

Model Checking

$\mathcal{M}^\sigma \models \varphi$?

Specification

$\varphi$

Probabilistic Temporal Logic Constraints

Parametric Markov Chain

$\mathcal{P}$

Parameter Synthesis

$u$
Story - Finite-state Controllers (FSCs)

POMDP

\( \mathcal{M} \)

Policy Synthesis for \( k \) memory states

\( \sigma \)

Model Checking

\( \mathcal{M}^\sigma \models \varphi ? \)

Specification

\( \varphi \)

Probabilistic Temporal Logic Constraints

Parametric Markov Chain

\( \mathcal{P} \)

Parameter Synthesis

\( u \)

Model Checking

\( \mathcal{P}[u] \models \varphi ? \)
Finite-Memory Strategies for POMDPs
Finite-Memory Strategies for POMDPs
Finite-Memory Strategies for POMDPs

\[ s_5 \rightarrow_{0.3} a_2 \rightarrow_{0.6} s_2 \]
\[ s_4 \rightarrow_{0.7} a_1 \rightarrow_{0.4} s_3 \]

\[ \langle n_1 \rangle \]

\[ z_0? \]
\[ z_1? \]

\[ \langle n_2 \rangle \]

\[ \ldots \]
Finite-Memory Strategies for POMDPs

- Encode memory using finite state controller (FSCs)
- On the product, policy is memoryless
Dependencies for Randomized Policies
Dependencies for Randomized Policies

Map observation-action pairs to randomized choices

- $s_0$ to $s_1$: $a_1$ with probability 0.5, $a_2$ with probability 0.5
- $s_1$ to $s_2$: $a_1$ with probability 1
- $s_2$ to $s_3$: $a_2$ with probability 0.5, $a_3$ with probability 0.5
- $s_3$ to $s_3$: $a_1$ with probability 1

- $a_1$ to 1
- $a_2$ to $q$
- $a_3$ to $1 - p_1 - p_2$
Dependencies for Randomized Policies

Map observation-action pairs to randomized choices

Randomized observation-based policy is sufficiently described by these probabilities!
Randomized Policies as Parametric MCs
Randomized Policies as Parametric MCs

\[
\begin{align*}
\text{State } s_0 & \quad \text{Transitions:} \\
\quad & a_1 \rightarrow s_1 \\
\quad & a_2 \rightarrow s_2 \\
\quad & a_3 \rightarrow s_3 \\
\end{align*}
\]

\[
\begin{align*}
\text{State } s_1 & \quad \text{Transitions:} \\
\quad & a_1 \rightarrow s_1 \\
\quad & p_1 \cdot 1 \\
\end{align*}
\]

\[
\begin{align*}
\text{State } s_2 & \quad \text{Transitions:} \\
\quad & p_2 \cdot 0.5 \\
\quad & (1 - p_1 - p_2) \cdot 1 \\
\quad & 1 - q \\
\end{align*}
\]

\[
\begin{align*}
\text{State } s_3 & \quad \text{Transitions:} \\
\quad & a_1 \rightarrow s_3 \\
\quad & q \\
\end{align*}
\]
Parameter Synthesis - Outputs

Parameter Space Partitioning
- concise description of parameter values that yield (un)satisfactory results

Rational Function
- generalization of non-parametric model checking

\[ p \cdot (1-p) \cdot \frac{1-q}{1-pq} \]
Parameter Synthesis - Outputs

Parameter Space Partitioning
- concise description of parameter values that yield (un)satisfactory results

Feasible Solution
- one parameter valuation that is satisfactory

Rational Function
- generalization of non-parametric model checking

\[ p \cdot (1-p) \cdot \frac{1-q}{1-pq} \]
Parameter Synthesis - Outputs

Parameter Space Partitioning
- concise description of parameter values that yield (un)satisfactory results

Feasible Solution
- one parameter valuation that is satisfactory

Rational Function
- generalization of non-parametric model checking

\[ p \cdot (1-p) \cdot \frac{1-q}{1-pq} \]

Already the feasibility problem is ETR-hard! (existential theory of the reals)

Parameter Synthesis Approaches
Parameter Synthesis Approaches

- Tool support: PRISM, PARAM, PROPhESY, Storm
Parameter Synthesis Approaches

- Tool support: PRISM, PARAM, PROPhESY, Storm

- Used to be restricted to a few parameters
Parameter Synthesis Approaches

- Tool support: PRISM, PARAM, PROPhESY, Storm
- Used to be restricted to a few parameters
- Utilize convex optimization:
  - Sequential Convex Optimization for the Efficient Verification of Parametric MDPs
  - Synthesis in pMDPs: A Tale of 1001 Parameters
Nonlinear Program - Parameter Synthesis

minimize \( p_{s_I} \)

subject to

\( \forall s \in T. \quad p_s = 1 \)

\( \forall s, s' \in S. \forall \alpha \in Act. \quad \mathcal{P}(s, \alpha, s') \geq \epsilon \)

\( \forall s \in S. \forall \alpha \in Act. \quad \sum_{s' \in S} \mathcal{P}(s, \alpha, s') = 1 \)

\( \lambda \geq p_{s_I} \)

\( \forall s \in S \setminus T. \forall \alpha \in Act. \quad p_s \geq \sum_{s' \in S} \mathcal{P}(s, \alpha, s') \cdot p_{s'} \)
Nonlinear Program - Parameter Synthesis

minimize \( p_{sI} \)

subject to

\( \forall s \in T. \quad p_s = 1 \)

\( \forall s, s' \in S. \forall \alpha \in Act. \quad \mathcal{P}(s, \alpha, s') \geq \epsilon \)

\( \forall s \in S. \forall \alpha \in Act. \quad \sum_{s' \in S} \mathcal{P}(s, \alpha, s') = 1 \)

\( \lambda \geq p_{sI} \)

\( \forall s \in S \setminus T. \forall \alpha \in Act. \quad p_s \geq \sum_{s' \in S} \mathcal{P}(s, \alpha, s') \cdot p_{s'} \)
Convexify it!
Story - Convex-concave Procedure

Specification

General parametric MDP

Parametric MDP restricted to affine functions

Nonlinear Program
Story - Convex-concave Procedure

- Specification
- General parametric MDP
  - Parametric MDP restricted to affine functions
  - Nonlinear Program
  - Quadratically Constrained Quadratic Program (QCQP)
**Story - Convex-concave Procedure**

1. **Specification**
2. **General parametric MDP**
3. **Parametric MDP restricted to affine functions**
4. **Nonlinear Program**
5. **Split quadratic functions into convex and concave part**
6. **Quadratically Constrained Quadratic Program (QCQP)**
Story - Convex-concave Procedure

1. Specification
2. General parametric MDP
3. Parametric MDP restricted to affine functions
4. Nonlinear Program
5. Split quadratic functions into convex and concave part
6. Quadratically Constrained Quadratic Program (QCQP)
7. Linearize concave part, introduce penalty for violation
**Story - Convex-concave Procedure**

1. **Specification**
2. **General parametric MDP**
3. **Parametric MDP restricted to affine functions**
4. **Nonlinear Program**
5. **Split quadratic functions into convex and concave part**
6. **Quadratically Constrained Quadratic Program (QCQP)**
7. **Linearize concave part, introduce penalty for violation**
8. **Minimize violations using Model Checking**
9. **Feasible solution**
Why is This Helpful?

- **POMDP**
  - $\mathcal{M}$
  - Policy Synthesis for $k$ memory states
  - $\sigma$
- **Specification**
  - $\varphi$
  - Probabilistic Temporal Logic Constraints
- **Parametric Markov Chain**
  - $\mathcal{P}$
  - Parameter Synthesis
  - $u$
  - Model Checking
  - $\mathcal{P}[u] \models \varphi$?
- **Model Checking**
  - $\mathcal{M}^\sigma \not\models \varphi$?

Nils Jansen

Radboud University
Why is This Helpful?

- All algorithms and complexity results carry over
- Extensive and mature tool-support for parameter synthesis
- Superior performance to state-of-the-art POMDP solvers

http://stormchecker.org

https://github.com/moves-rwth/prophesy
# Experiments

<table>
<thead>
<tr>
<th>Problem</th>
<th>Set</th>
<th>Inst</th>
<th>Spec</th>
<th>Info States</th>
<th>Trans.</th>
<th>Par.</th>
<th>PSO tmin</th>
<th>tmax</th>
<th>tavg</th>
<th>SMT t</th>
<th>CCP t</th>
<th>solv</th>
<th>iter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brp</td>
<td>16,2</td>
<td>$P_{\leq 0.1}$</td>
<td>98</td>
<td>194</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>40</td>
<td>0 30%</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Brp</td>
<td>512,5</td>
<td>$P_{\leq 0.1}$</td>
<td>6146</td>
<td>12290</td>
<td>2</td>
<td>24</td>
<td>36</td>
<td>28</td>
<td>TO</td>
<td>33 24%</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Crowds</td>
<td>10,5</td>
<td>$P_{\leq 0.1}$</td>
<td>42</td>
<td>82</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>8</td>
<td>4 2%</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nand</td>
<td>5,10</td>
<td>$P_{\leq 0.05}$</td>
<td>10492</td>
<td>20982</td>
<td>2</td>
<td>21</td>
<td>51</td>
<td>28</td>
<td>TO</td>
<td>22 21%</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zeroconf</td>
<td>10000</td>
<td>$E_{\leq 10010}$</td>
<td>10003</td>
<td>20004</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>TO</td>
<td>57 81%</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GridA</td>
<td>4</td>
<td>$P_{\geq 0.84}$</td>
<td>1026</td>
<td>2098</td>
<td>72</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>TO</td>
<td>22 81%</td>
<td>11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GridB</td>
<td>8,5</td>
<td>$P_{\geq 0.84}$</td>
<td>8653</td>
<td>17369</td>
<td>700</td>
<td>409</td>
<td>440</td>
<td>427</td>
<td>TO</td>
<td>213 84%</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GridB</td>
<td>10,6</td>
<td>$P_{\geq 0.84}$</td>
<td>16941</td>
<td>33958</td>
<td>1290</td>
<td>533</td>
<td>567</td>
<td>553</td>
<td>TO</td>
<td>426 84%</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GridC</td>
<td>6</td>
<td>$E_{\leq 4.8}$</td>
<td>1665</td>
<td>305</td>
<td>168</td>
<td>261</td>
<td>274</td>
<td>267</td>
<td>TO</td>
<td>169 90%</td>
<td>23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maze</td>
<td>5</td>
<td>$E_{\leq 14}$</td>
<td>1303</td>
<td>2658</td>
<td>590</td>
<td>213</td>
<td>230</td>
<td>219</td>
<td>TO</td>
<td>67   89%</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maze</td>
<td>5</td>
<td>$E_{\leq 6}$</td>
<td>1303</td>
<td>2658</td>
<td>590</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>TO</td>
<td>422 85%</td>
<td>97</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maze</td>
<td>7</td>
<td>$E_{\leq 6}$</td>
<td>2580</td>
<td>5233</td>
<td>1176</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>TO</td>
<td>740 90%</td>
<td>60</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Netw</td>
<td>5,2</td>
<td>$E_{\leq 11.5}$</td>
<td>21746</td>
<td>63158</td>
<td>2420</td>
<td>312</td>
<td>523</td>
<td>359</td>
<td>TO</td>
<td>207 39%</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Netw</td>
<td>5,2</td>
<td>$E_{\leq 10.5}$</td>
<td>21746</td>
<td>63158</td>
<td>2420</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>TO</td>
<td>210 38%</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Netw</td>
<td>4,3</td>
<td>$E_{\leq 11.5}$</td>
<td>38055</td>
<td>97335</td>
<td>4545</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>TO</td>
<td>MO   –</td>
<td>–</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Repud</td>
<td>8,5</td>
<td>$P_{\geq 0.1}$</td>
<td>1487</td>
<td>3002</td>
<td>360</td>
<td>16</td>
<td>22</td>
<td>18</td>
<td>TO</td>
<td>4 36%</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Repud</td>
<td>8,5</td>
<td>$P_{\leq 0.05}$</td>
<td>1487</td>
<td>3002</td>
<td>360</td>
<td>273</td>
<td>324</td>
<td>293</td>
<td>TO</td>
<td>14 72%</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Repud</td>
<td>16,2</td>
<td>$P_{\leq 0.01}$</td>
<td>790</td>
<td>1606</td>
<td>96</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>TO</td>
<td>15 78%</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Repud</td>
<td>16,2</td>
<td>$P_{&gt;0.062}$</td>
<td>790</td>
<td>1606</td>
<td>96</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>TO</td>
<td>TO   –</td>
<td>–</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Numerical Experiments

Performance analysis

Simple Grid.

POMDP with 17 states, 62 branches, 3 observations
Numerical Experiments

Performance analysis

Simple Grid.

POMDP with 17 states, 62 branches, 3 observations

Find a policy such that we arrive at destination within T steps

Actual optimum (arbitrary K) 4.13
Numerical Experiments

Performance analysis

Simple Grid.

POMDP with 17 states, 62 branches, 3 observations

Find a policy such that we arrive at destination within T steps

<table>
<thead>
<tr>
<th>K</th>
<th>states</th>
<th>parameters</th>
<th>time for T=4.15</th>
<th>time for T=5.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>K=1</td>
<td>47</td>
<td>3</td>
<td>not possible</td>
<td>&lt;1 s</td>
</tr>
<tr>
<td>K=2</td>
<td>183</td>
<td>15</td>
<td>7.4 s</td>
<td>&lt;1 s</td>
</tr>
</tbody>
</table>

Actual optimum (arbitrary K) 4.13
Numerical Experiments

Performance analysis

Simple Grid.

POMDP with 17 states, 62 branches, 3 observations

Find a policy such that we arrive at destination within T steps

<table>
<thead>
<tr>
<th>K</th>
<th>states</th>
<th>parameters</th>
<th>time for T=4.15</th>
<th>time for T=5.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>K=1</td>
<td>47</td>
<td>3</td>
<td>not possible</td>
<td>&lt;1 s</td>
</tr>
<tr>
<td>K=2</td>
<td>183</td>
<td>15</td>
<td>7.4 s</td>
<td>&lt;1 s</td>
</tr>
</tbody>
</table>

Automatically proven: For K=1, 5 is a lower bound. (<1 s)

Actual optimum (arbitrary K) 4.13
Larger Numerical Experiments

Practical implications

Network protocol:
Optimally assign packets to slots.

POMDP with 2729 states, 4937 branches, 361 observations

Find a policy such that the expected packet loss is below T packets

Actual optimum (arbitrary K) around 9
Larger Numerical Experiments

Practical implications

Network protocol:
Optimally assign packets to slots.

POMDP with 2729 states, 4937 branches, 361 observations

Find a policy such that the expected packet loss is below T packets

<table>
<thead>
<tr>
<th>K</th>
<th>states</th>
<th>parameters</th>
<th>time for T=10</th>
<th>time for T=15</th>
</tr>
</thead>
<tbody>
<tr>
<td>K=1</td>
<td>3268</td>
<td>276</td>
<td>43 s</td>
<td>4 s</td>
</tr>
<tr>
<td>K=2</td>
<td>16004</td>
<td>1783</td>
<td>877 s</td>
<td>28 s</td>
</tr>
</tbody>
</table>

SolvePOMDP and PRISM-POMDP: Time outs (> 3600 seconds)

Actual optimum (arbitrary K) around 9
Larger Numerical Experiments

Network protocol: Optimally assign packets to slots.

POMDP with 2729 states, 4937 branches, 361 observations

Find a policy such that the expected packet loss is below T packets

Actual optimum (arbitrary K) around 9

<table>
<thead>
<tr>
<th>K</th>
<th>states</th>
<th>parameters</th>
<th>time for T=10</th>
<th>time for T=15</th>
</tr>
</thead>
<tbody>
<tr>
<td>K=1</td>
<td>3268</td>
<td>276</td>
<td>43 s</td>
<td>4 s</td>
</tr>
<tr>
<td>K=2</td>
<td>16004</td>
<td>1783</td>
<td>877 s</td>
<td>28 s</td>
</tr>
</tbody>
</table>

SolvePOMDP and PRISM-POMDP: Time outs (> 3600 seconds)

For K=4, 5 is a lower bound. (183 s)
Conclusion to Story 2

- Novel way to generate **provably correct** POMDP strategies
- **Good scalability**, **not optimal**
- Future: **More principled approach** to permissive strategies

Stories - Policy Synthesis for POMDPs

1. Game-based abstraction
2. Finite-memory controllers
3. Recurrent neural networks
4. Fun: Humans in the loop
Be Lazy: Guess a Policy and Verify!

M \rightarrow \phi

POMDP \rightarrow Specification

Probabilistic Temporal Logic Constraints

Probabilistic Temporal Logic Constraints
Be Lazy: Guess a Policy and Verify!

- POMDP
- Specification
  - Probabilistic Temporal Logic Constraints
- Guess Candidate Policy
Be Lazy: Guess a Policy and Verify!

- POMDP
- Specification
- Probabilistic Temporal Logic Constraints

- Guess Candidate Policy
- Apply Policy to POMDP

\[ M, \varphi, \sigma, M^\sigma \]
Be Lazy: Guess a Policy and Verify!

POMDP \( \mathcal{M} \) \( \vdash \phi \) Specification

Guess Candidate Policy

Apply Policy to POMDP \( \mathcal{M}^\sigma \)

Model Checking \( \mathcal{M}^\sigma \vdash \phi ? \)

Probabilistic Temporal Logic Constraints

\( \mathcal{M} \)

\( \phi \)

\( \sigma \)

\( \mathcal{M}^\sigma \)

SAT

UNSAT

Nils Jansen

Radboud University
Be Lazy: Guess a Policy and Verify!

- **POMDP**
- **Specification**
- **Probabilistic Temporal Logic Constraints**

**Diagram:**
1. **Guess Candidate Policy**
2. **Apply Policy to POMDP**
3. **Model Checking**
   - $\mathcal{M}^\sigma \models \varphi$?
   - SAT
   - UNSAT

**Annotations:**
- $\mathcal{M}$
- $\varphi$
- $\sigma$
- $\mathcal{M}^\sigma$
- $\mathcal{M}^\sigma \models \varphi$?
- efficient
Be Lazy: Guess a Policy and Verify!

Guess Candidate Policy

Apply Policy to POMDP

Model Checking

Guess a Policy and Verify!

how to guess a good policy?

efficient
Let Machine Learning do the Guessing?

Let Machine Learning do the Guessing?

Let Machine Learning do the Guessing?

Let Machine Learning do the Guessing?

Let Machine Learning do the Guessing?

Let Machine Learning do the Guessing?
Let Machine Learning do the Guessing?

- **POMDP**
- **Specification**

$\mathcal{M} \models \varphi$?

Apply Policy to POMDP

Model Checking

$\mathcal{M}^{\sigma} \models \varphi$?

how to employ a neural network?

$\mathcal{M}^{\sigma} \not\models \varphi$?

UNSAT

SAT

Nils Jansen
RNN Strategy Improvement

\[ M \xrightarrow{\sigma} M^\sigma \xrightarrow{\phi} \text{Model Checking} \xrightarrow{\text{SAT}} \]

Apply Policy to POMDP

specification

strategy network

Nils Jansen

Radboud University
RNN Strategy Improvement

POMDP

Specification

$\mathcal{M}$

$\varphi$

strategy network

$\sigma$

Apply Policy to POMDP

$\mathcal{M}^\sigma$

$\mathcal{M}^\sigma \models \varphi$?

Model Checking

SAT

UNSAT

Counterexamples

$S' \subseteq S$
RNN Strategy Improvement

1. **POMDP**
2. **Specification**
   - Apply Policy to POMDP
   - Model Checking
   - Counterexamples
   - Local Improvement of the Policy

- $\mathcal{M}$
- $\phi$
- $\sigma$
- $\mathcal{M}^\sigma$
- $\mathcal{M}^\sigma \models \phi$?
- UNSAT
- $S' \subseteq S$
RNN Strategy Improvement

\( \mathcal{M} \) \( \varphi \) \( \sigma \) 

Apply Policy to POMDP 

Training Data 

Observation-Action Sequences 

Model Checking 

Counterexamples 

\( \mathcal{M}^\sigma \) \( \models \varphi \) 

\( \mathcal{M}^\sigma \models \varphi \) ? 

\( S' \subseteq S \) 

Local Improvement of the Policy 

POMDP Specification 

Strategy network

Training Data

Counterexamples

\( S' \subseteq S \)

Nils Jansen
Learning Strategies with RNNs
Learning Strategies with RNNs

Recurrent Neural Network

- long short-term memory (LSTM) architecture to learn dependencies in sequential data
- trained with observation-action sequences \( \text{ObsSeq}_{\text{fin}}^M \)
- policy network \( \sigma: \text{ObsSeq}_{\text{fin}}^M \rightarrow \text{Distr}(\text{Act}) \)
Learning Strategies with RNNs

Recurrent Neural Network
- long short-term memory (LSTM) architecture to learn dependencies in sequential data
- trained with observation-action sequences $\text{ObsSeq}_{\text{fin}}^m$
- policy network $\sigma: \text{ObsSeq}_{\text{fin}}^m \rightarrow Distr(\text{Act})$

predictor for a (memoryless) randomized policy
Learning Strategies with RNNs

Recurrent Neural Network
• long short-term memory (LSTM) architecture to learn dependencies in sequential data
• trained with observation-action sequences $\text{ObsSeq}_{\text{fin}}^M$
• policy network $\sigma: \text{ObsSeq}_{\text{fin}}^M \rightarrow \text{Distr}(\text{Act})$

Training
• Compute optimal MDP policy
• Generate (possible) observation-action sequences
• Observations are input labels, actions are output labels
Learning Strategies with RNNs

Recurrent Neural Network

- long short-term memory (LSTM) architecture to learn dependencies in sequential data
- trained with observation-action sequences $\text{ObsSeq}_{\text{fin}}^M$
- policy network $\sigma: \text{ObsSeq}_{\text{fin}}^M \rightarrow \text{Distr(Act)}$

Training

- Compute optimal MDP policy
- Generate (possible) observation-action sequences
- Observations are input labels, actions are output labels

Large Environments

- Train on smaller environments that share observations and actions
Improving the Policy
Improving the Policy

Identify critical decisions $\sigma(z)(\alpha) > 0$ that lead to states with high probability of violating the specification.
Improving the Policy

Identify **critical decisions** $\sigma(z)(\alpha) > 0$ that lead to states with high probability of **violating** the specification.

For each observation $z \in (O)$ with critical decision, minimize the number of different critical actions.
Improving the Policy

Identify critical decisions $\sigma(z)(\alpha) > 0$ that lead to states with high probability of violating the specification.

For each observation $z \in (O)$ with critical decision, minimize the number of different critical actions.

Retrain with the new (locally improved) policy.
Improving the Policy

Identify **critical decisions** $\sigma(z)(\alpha) > 0$ that lead to states with high probability of **violating** the specification.

For each observation $z \in (O)$ with critical decision, minimize the number of different critical actions.

Retrain with the new (locally improved) policy.

Local linear program

\[
\begin{align*}
\max_{\gamma(z)(a) \in \text{Act}} \min_{s \in S} p_s \\
\text{subject to} \\
\forall s \in O^{-1}(z). \quad p_s = \sum_{a \in \text{Act}} \gamma(z)(a) \cdot \sum_{s' \in S} \mathcal{P}(s, a, s') \cdot p^*(s')
\end{align*}
\]
Improving the Policy

Identify **critical decisions** $\sigma(z)(\alpha) > 0$ that lead to states with high probability of **violating** the specification.

For each observation $z \in (O)$ with critical decision, minimize the number of different critical actions.

Retrain with the new (locally improved) policy.

Even if specification is satisfied, there may be critical states and decisions!
Finite-memory Strategies

- Encode finite memory:
Finite-memory Strategies

• Encode finite memory:

- Policy network is of the form $\sigma: \text{ObsSeq}^{\mathcal{M}}_{\text{fin}} \rightarrow \text{Distr(Act)}$
Finite-memory Strategies

• Encode finite memory:

• Policy network is of the form $\sigma: \text{ObsSeq}_{\text{fin}}^M \rightarrow \text{Distr}(\text{Act})$

• But: How to infer a memory-update function to construct a finite-state controller?
Finite-memory Strategies

• Encode finite memory:

Policy network is of the form \( \sigma: \text{ObsSeq}_{\text{fin}}^{M} \rightarrow \text{Distr}(\text{Act}) \)

But: How to infer a memory-update function to construct a finite-state controller?

First solution: Predefine memory update, for instance (deterministic) transition upon repetition of an observation.
 Finite-memory Strategies

• Encode finite memory:

\[ \langle s_5, n_1 \rangle \xrightarrow{0.075} \langle s_1, n_1 \rangle \xrightarrow{0.15} \langle s_2, n_1 \rangle \]
\[ \langle s_5, n_2 \rangle \xrightarrow{0.075} \langle s_1, n_2 \rangle \xrightarrow{0.15} \langle s_2, n_2 \rangle \]
\[ \langle s_4, n_1 \rangle \xrightarrow{0.175} \langle s_3, n_1 \rangle \]
\[ \langle s_4, n_2 \rangle \xrightarrow{0.175} \langle s_3, n_2 \rangle \]

• Policy network is of the form \( \sigma : \text{ObsSeq}^M_{\text{fin}} \rightarrow \text{Distr}(\text{Act}) \)

• But: How to infer a memory-update function to construct a finite-state controller?

• First solution: Predefine memory update, for instance (deterministic) transition upon repetition of an observation.

• Compute product of FSC and POMDP and compute memoryless policy as before.
Correctness and Completeness?

\[ \mathcal{M} \]

\[ \mathcal{M} \models \phi \]

\[ \mathcal{M}^\sigma \models \phi ? \]

\[ S' \subseteq S \]

POMDP

Specification

Apply Policy to POMDP

Model Checking

Training Data

Local proof

Counterexamples

\[ \sigma \]

\[ \mathcal{M}^\sigma \]

\[ \mathcal{M}^\sigma \models \phi ? \]
Correctness and Completeness?

Correct, as each policy prediction is evaluated using model checking.
Correctness and Completeness?

Correct, as each policy prediction is evaluated using model checking.

Not complete, as we may never find a feasible policy. Also, problem is undecidable (or hard) anyways :).
# Experiments - LTL

<table>
<thead>
<tr>
<th>Problem</th>
<th>States</th>
<th>Type, $\varphi$</th>
<th>RNN-based Synthesis</th>
<th>PRISM-POMDP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Res.</td>
<td>Time (s)</td>
<td>Res.</td>
</tr>
<tr>
<td>Navigation (3)</td>
<td>333</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td>0.74</td>
<td><strong>14.16</strong></td>
</tr>
<tr>
<td>Navigation (4)</td>
<td>1088</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td>0.82</td>
<td><strong>22.67</strong></td>
</tr>
<tr>
<td>Navigation (4) [2-FSC]</td>
<td>13373</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td>0.91</td>
<td>47.26</td>
</tr>
<tr>
<td>Navigation (4) [4-FSC]</td>
<td>26741</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td>0.92</td>
<td>59.42</td>
</tr>
<tr>
<td>Navigation (4) [8-FSC]</td>
<td>53477</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td>0.92</td>
<td>85.26</td>
</tr>
<tr>
<td>Navigation (5)</td>
<td>2725</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td>0.91</td>
<td><strong>34.34</strong></td>
</tr>
<tr>
<td>Navigation (5) [2-FSC]</td>
<td>33357</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td>0.92</td>
<td>115.16</td>
</tr>
<tr>
<td>Navigation (5) [4-FSC]</td>
<td>66709</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td>0.92</td>
<td>159.61</td>
</tr>
<tr>
<td>Navigation (5) [8-FSC]</td>
<td>133413</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td><strong>0.92</strong></td>
<td>250.91</td>
</tr>
<tr>
<td>Navigation (10)</td>
<td>49060</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td>0.79</td>
<td><strong>822.87</strong></td>
</tr>
<tr>
<td>Navigation (10) [2-FSC]</td>
<td>475053</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td>0.83</td>
<td>1185.41</td>
</tr>
<tr>
<td>Navigation (10) [4-FSC]</td>
<td>950101</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td><strong>0.85</strong></td>
<td>1488.77</td>
</tr>
<tr>
<td>Navigation (10) [8-FSC]</td>
<td>1900197</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td>0.81</td>
<td>1805.22</td>
</tr>
<tr>
<td>Navigation (15)</td>
<td>251965</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td>0.91</td>
<td><strong>1271.80</strong></td>
</tr>
<tr>
<td>Navigation (20)</td>
<td>798040</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td>0.96</td>
<td><strong>4712.25</strong></td>
</tr>
<tr>
<td>Navigation (30)</td>
<td>4045840</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td><strong>0.95</strong></td>
<td>25191.05</td>
</tr>
<tr>
<td>Navigation (40)</td>
<td>–</td>
<td>$P_{\text{max}}$, $\varphi_1$</td>
<td>TO</td>
<td>TO</td>
</tr>
<tr>
<td>Delivery (4) [2-FSC]</td>
<td>80</td>
<td>$P_{\text{min}}$, $\varphi_2$</td>
<td>6.02</td>
<td>35.35</td>
</tr>
<tr>
<td>Delivery (5) [2-FSC]</td>
<td>125</td>
<td>$P_{\text{min}}$, $\varphi_2$</td>
<td>8.11</td>
<td><strong>78.32</strong></td>
</tr>
<tr>
<td>Delivery (10) [2-FSC]</td>
<td>500</td>
<td>$P_{\text{min}}$, $\varphi_2$</td>
<td><strong>18.13</strong></td>
<td><strong>120.34</strong></td>
</tr>
<tr>
<td>Slippery (4) [2-FSC]</td>
<td>460</td>
<td>$P_{\text{max}}$, $\varphi_3$</td>
<td>0.78</td>
<td>67.51</td>
</tr>
<tr>
<td>Slippery (5) [2-FSC]</td>
<td>730</td>
<td>$P_{\text{max}}$, $\varphi_3$</td>
<td>0.89</td>
<td>84.32</td>
</tr>
<tr>
<td>Slippery (10) [2-FSC]</td>
<td>2980</td>
<td>$P_{\text{max}}$, $\varphi_3$</td>
<td><strong>0.98</strong></td>
<td>119.14</td>
</tr>
<tr>
<td>Slippery (20) [2-FSC]</td>
<td>11980</td>
<td>$P_{\text{max}}$, $\varphi_3$</td>
<td><strong>0.99</strong></td>
<td>1580.42</td>
</tr>
</tbody>
</table>
Experiments - LTL

| Problem         | $|S|$  | $|Act|$ | $|Z|$ | RNN-based Synthesis | PRISM-POMDP |
|-----------------|------|--------|------|---------------------|-------------|
| Navigation (c)  | $c^4$| 4      | 256  | 0.74 | 14.16 | 0.84 | 73.88 |
| Delivery (c)    | $c^2$| 4      | 256  | 0.92 | 115.16 | -   | -    |
| Slippery (c)    | $c^2$| 4      | 256  | 0.92 | 159.61 | -   | -    |
| Maze(c)         | $3c + 8$| 4      | 7    | 0.92 | 250.91 | -   | -    |
| Grid(c)         | $c^2$| 4      | 2    | TO   | -     | TO   | -    |
| Delivery (4)    | 80   | $E_{min}$ | $\varphi_2$ | 6.02 | 35.35 | 6.0  | 28.53 |
| Delivery (5)    | 125  | $E_{min}$ | $\varphi_2$ | 8.11 | 78.32 | 8.0  | 102.41 |
| Delivery (10)   | 500  | $E_{min}$ | $\varphi_2$ | 18.13 | 120.34 | -   | -    |
| Slippery (4)    | 460  | $E_{max}$ | $\varphi_3$ | 0.78 | 67.51 | 0.98 | 5.10 |
| Slippery (5)    | 730  | $E_{max}$ | $\varphi_3$ | 0.89 | 84.32 | 0.98 | 5.10 |
| Slippery (10)   | 2980 | $E_{max}$ | $\varphi_3$ | 119.14 | MO    | MO   | MO   |
| Slippery (20)   | 11980| $E_{max}$ | $\varphi_3$ | 1580.42 | MO    | MO   | MO   |
### Experiments - LTL

| Problem               | $|S|$ | $|Act|$ | $|Z|$ | RNN-based Synthesis | PRISM-POMDP |
|-----------------------|-----|--------|-----|---------------------|-------------|
|                       |     |        |     | Res.    | Time (s) | Res.   | Time (s) |
| Navigation (3)        |     |        |     | 0.74    | 14.16   | 0.84   | 73.88   |
| Navigation (4)        |     |        |     | 0.82    | 22.67   | 0.93   | 1034.64 |
| Navigation (4) [2-FSC]|     |        |     | 0.91    | 47.26   | -      | -       |
| Navigation (4) [4-FSC]|     |        |     | 0.92    | 59.42   | -      | -       |
| Navigation (4) [8-FSC]|     |        |     | 0.92    | 85.26   | -      | -       |
| Navigation (5)        |     |        |     | 0.91    | 34.34   | MO     | MO      |
| Navigation (5) [2-FSC]|     |        |     | 0.92    | 115.16  | -      | -       |
| Navigation (5) [4-FSC]|     |        |     | 0.92    | 159.61  | -      | -       |
| Navigation (5) [8-FSC]|     |        |     | 0.92    | 250.91  | -      | -       |
| Navigation (10)       |     |        |     | 0.79    | 822.87  | MO     | MO      |
| Navigation (10) [2-FSC]|    |        |     | 0.83    | 1185.41 | -      | -       |
| Navigation (10) [4-FSC]|    |        |     | 0.85    | 1488.77 | -      | -       |
| Navigation (10) [8-FSC]|    |        |     | 0.81    | 1805.22 | -      | -       |
| Navigation (15)       |     |        |     | 0.91    | 1271.80*| MO     | MO      |
| Navigation (20)       |     |        |     | 0.96    | 4712.25*| MO     | MO      |
| Navigation (30)       |     |        |     | 0.95    | 25191.05*| MO | MO      |
| Navigation (40)       |     |        |     | TO      | TO      | MO     | MO      |
| Delivery (4) [2-FSC]  |     |        |     | 6.02    | 35.35   | 6.0    | 28.53   |
| Delivery (5) [2-FSC]  |     |        |     | 8.11    | 78.32   | 8.0    | 102.41  |
| Delivery (10) [2-FSC] |     |        |     | 18.13   | 120.34  | MO     | MO      |
| Slippery (4) [2-FSC]  |     |        |     | 0.78    | 67.51   | 0.90   | 5.10    |
| Slippery (5) [2-FSC]  |     |        |     | 0.89    | 84.32   | 0.93   | 83.24   |
| Slippery (10) [2-FSC] |     |        |     | 0.98    | 119.14  | MO     | MO      |
| Slippery (20) [2-FSC] |     |        |     | 0.99    | 1580.42 | MO     | MO      |
# Experiments - Standard POMDPs

<table>
<thead>
<tr>
<th>Problem</th>
<th>Type</th>
<th>RNN-based Synthesis</th>
<th>PRISM-POMDP</th>
<th>pomdpSolve</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>States</td>
<td>Res</td>
<td>Time (s)</td>
</tr>
<tr>
<td>Maze (1)</td>
<td>$E_{\text{min}}^M$</td>
<td>68</td>
<td>4.31</td>
<td>31.70</td>
</tr>
<tr>
<td>Maze (2)</td>
<td>$E_{\text{min}}^M$</td>
<td>83</td>
<td>5.31</td>
<td>46.65</td>
</tr>
<tr>
<td>Maze (3)</td>
<td>$E_{\text{min}}^M$</td>
<td>98</td>
<td>8.10</td>
<td>58.75</td>
</tr>
<tr>
<td>Maze (4)</td>
<td>$E_{\text{min}}^M$</td>
<td>113</td>
<td>11.53</td>
<td>58.09</td>
</tr>
<tr>
<td>Maze (5)</td>
<td>$E_{\text{min}}^M$</td>
<td>128</td>
<td>14.40</td>
<td>68.09</td>
</tr>
<tr>
<td>Maze (6)</td>
<td>$E_{\text{min}}^M$</td>
<td>143</td>
<td>22.34</td>
<td>71.89</td>
</tr>
<tr>
<td>Maze (10)</td>
<td>$E_{\text{min}}^M$</td>
<td>203</td>
<td>100.21</td>
<td>158.33</td>
</tr>
<tr>
<td>Grid (3)</td>
<td>$E_{\text{min}}^M$</td>
<td>165</td>
<td>2.90</td>
<td>38.94</td>
</tr>
<tr>
<td>Grid (4)</td>
<td>$E_{\text{min}}^M$</td>
<td>381</td>
<td>4.32</td>
<td>79.99</td>
</tr>
<tr>
<td>Grid (5)</td>
<td>$E_{\text{min}}^M$</td>
<td>727</td>
<td>6.623</td>
<td>91.42</td>
</tr>
<tr>
<td>Grid (10)</td>
<td>$E_{\text{min}}^M$</td>
<td>5457</td>
<td>13.630</td>
<td>268.40</td>
</tr>
<tr>
<td>RockSample[4, 4]</td>
<td>$E_{\text{max}}^M$</td>
<td>2432</td>
<td>17.71</td>
<td>35.35</td>
</tr>
<tr>
<td>RockSample[5, 5]</td>
<td>$E_{\text{max}}^M$</td>
<td>8320</td>
<td>18.40</td>
<td>43.74</td>
</tr>
<tr>
<td>RockSample[7, 8]</td>
<td>$E_{\text{max}}^M$</td>
<td>166656</td>
<td>20.32</td>
<td>860.53</td>
</tr>
</tbody>
</table>
# Experiments - Standard POMDPs

<table>
<thead>
<tr>
<th>Problem</th>
<th>Type</th>
<th>RNN-based Synthesis</th>
<th>PRISM-POMDP</th>
<th>pomdpSolve</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>States</td>
<td>Res</td>
<td>Time (s)</td>
</tr>
<tr>
<td>Maze (1)</td>
<td>Eₘ</td>
<td>68</td>
<td>4.31</td>
<td>31.70</td>
</tr>
<tr>
<td>Maze (2)</td>
<td>Eₘ</td>
<td>83</td>
<td>5.31</td>
<td>46.65</td>
</tr>
<tr>
<td>Maze (3)</td>
<td>Eₘ</td>
<td>98</td>
<td>8.10</td>
<td>58.75</td>
</tr>
<tr>
<td>Maze (4)</td>
<td>Eₘ</td>
<td>113</td>
<td>11.53</td>
<td>58.09</td>
</tr>
<tr>
<td>Maze (5)</td>
<td>Eₘ</td>
<td>128</td>
<td>14.40</td>
<td>68.09</td>
</tr>
<tr>
<td>Maze (6)</td>
<td>Eₘ</td>
<td>143</td>
<td>22.34</td>
<td>71.89</td>
</tr>
<tr>
<td>Maze (10)</td>
<td>Eₘ</td>
<td>203</td>
<td>100.21</td>
<td>158.33</td>
</tr>
<tr>
<td>Grid (3)</td>
<td>Eₘ</td>
<td>165</td>
<td>2.90</td>
<td>38.94</td>
</tr>
<tr>
<td>Grid (4)</td>
<td>Eₘ</td>
<td>381</td>
<td>4.32</td>
<td>79.99</td>
</tr>
<tr>
<td>Grid (5)</td>
<td>Eₘ</td>
<td>727</td>
<td>6.623</td>
<td>91.42</td>
</tr>
<tr>
<td>Grid (10)</td>
<td>Eₘ</td>
<td>5457</td>
<td>13.630</td>
<td>268.40</td>
</tr>
<tr>
<td>RockSample [4, 4]</td>
<td>Eₘ</td>
<td>2432</td>
<td>17.71</td>
<td>35.35</td>
</tr>
<tr>
<td>RockSample [5, 5]</td>
<td>Eₘ</td>
<td>8320</td>
<td>18.40</td>
<td>43.74</td>
</tr>
<tr>
<td>RockSample [7, 8]</td>
<td>Eₘ</td>
<td>166656</td>
<td>20.32</td>
<td>860.53</td>
</tr>
</tbody>
</table>
## Experiments - Standard POMDPs

<table>
<thead>
<tr>
<th>Problem</th>
<th>Type</th>
<th>RNN-based Synthesis</th>
<th>PRISM-POMDP</th>
<th>pomdpSolve</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>States</td>
<td>Res</td>
<td>Time (s)</td>
</tr>
<tr>
<td>Maze (1)</td>
<td>$E_1^M$</td>
<td>68</td>
<td>4.31</td>
<td>31.70</td>
</tr>
<tr>
<td>Maze (2)</td>
<td>$E_1^M$</td>
<td>83</td>
<td>5.31</td>
<td>46.65</td>
</tr>
<tr>
<td>Maze (3)</td>
<td>$E_1^M$</td>
<td>98</td>
<td>8.10</td>
<td>58.75</td>
</tr>
<tr>
<td>Maze (4)</td>
<td>$E_1^M$</td>
<td>113</td>
<td>11.53</td>
<td>58.09</td>
</tr>
<tr>
<td>Maze (5)</td>
<td>$E_1^M$</td>
<td>128</td>
<td>14.40</td>
<td>68.09</td>
</tr>
<tr>
<td>Maze (6)</td>
<td>$E_1^M$</td>
<td>143</td>
<td>22.34</td>
<td>71.89</td>
</tr>
<tr>
<td>Maze (10)</td>
<td>$E_1^M$</td>
<td>203</td>
<td>100.21</td>
<td>158.33</td>
</tr>
<tr>
<td>Grid (3)</td>
<td>$E_1^M$</td>
<td>165</td>
<td>2.90</td>
<td>38.94</td>
</tr>
<tr>
<td>Grid (4)</td>
<td>$E_1^M$</td>
<td>381</td>
<td>4.32</td>
<td>79.99</td>
</tr>
<tr>
<td>Grid (5)</td>
<td>$E_1^M$</td>
<td>727</td>
<td>6.623</td>
<td>91.42</td>
</tr>
<tr>
<td>Grid (10)</td>
<td>$E_1^M$</td>
<td>5457</td>
<td>13.630</td>
<td>268.40</td>
</tr>
<tr>
<td>RockSample[4, 4]</td>
<td>$E_1^M$</td>
<td>2432</td>
<td>17.71</td>
<td>35.35</td>
</tr>
<tr>
<td>RockSample[5, 5]</td>
<td>$E_1^M$</td>
<td>8320</td>
<td>18.40</td>
<td>43.74</td>
</tr>
<tr>
<td>RockSample[7, 8]</td>
<td>$E_1^M$</td>
<td>166656</td>
<td>20.32</td>
<td>860.53</td>
</tr>
</tbody>
</table>
# Experiments - Standard POMDPs

<table>
<thead>
<tr>
<th>Problem</th>
<th>Type</th>
<th>RNN-based Synthesis</th>
<th>PRISM-POMDP</th>
<th>pomdpSolve</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>States</td>
<td>Res</td>
<td>Time (s)</td>
</tr>
<tr>
<td>Maze (1)</td>
<td>$E_{\min}^M$</td>
<td>68</td>
<td>4.31</td>
<td>31.70</td>
</tr>
<tr>
<td>Maze (2)</td>
<td>$E_{\min}^M$</td>
<td>83</td>
<td>5.31</td>
<td>46.65</td>
</tr>
<tr>
<td>Maze (3)</td>
<td>$E_{\min}^M$</td>
<td>98</td>
<td>8.10</td>
<td>58.75</td>
</tr>
<tr>
<td>Maze (4)</td>
<td>$E_{\min}^M$</td>
<td>113</td>
<td>11.53</td>
<td>58.09</td>
</tr>
<tr>
<td>Maze (5)</td>
<td>$E_{\min}^M$</td>
<td>128</td>
<td>14.40</td>
<td>68.09</td>
</tr>
<tr>
<td>Maze (6)</td>
<td>$E_{\min}^M$</td>
<td>143</td>
<td>22.34</td>
<td>71.89</td>
</tr>
<tr>
<td>Maze (10)</td>
<td>$E_{\min}^M$</td>
<td>203</td>
<td>100.21</td>
<td>158.33</td>
</tr>
<tr>
<td>Grid (3)</td>
<td>$E_{\min}^M$</td>
<td>165</td>
<td>2.90</td>
<td>38.94</td>
</tr>
<tr>
<td>Grid (4)</td>
<td>$E_{\min}^M$</td>
<td>381</td>
<td>4.32</td>
<td>79.99</td>
</tr>
<tr>
<td>Grid (5)</td>
<td>$E_{\min}^M$</td>
<td>727</td>
<td>6.623</td>
<td>91.42</td>
</tr>
<tr>
<td>Grid (10)</td>
<td>$E_{\min}^M$</td>
<td>5457</td>
<td>13.630</td>
<td>268.40</td>
</tr>
<tr>
<td>RockSample [4, 4]</td>
<td>$E_{\max}^M$</td>
<td>2432</td>
<td>17.71</td>
<td>35.35</td>
</tr>
<tr>
<td>RockSample [5, 5]</td>
<td>$E_{\max}^M$</td>
<td>8320</td>
<td>18.40</td>
<td>43.74</td>
</tr>
<tr>
<td>RockSample [7, 8]</td>
<td>$E_{\max}^M$</td>
<td>166656</td>
<td>20.32</td>
<td>860.53</td>
</tr>
</tbody>
</table>
Conclusion Story 3

- Novel way to generate **provably correct** POMDP policies
- **Good** scalability, **not optimal**
- Results **transferrable**
- Future work: *More principled* approach to finite-memory strategies
Conclusion Story 3

- Novel way to generate **provably correct** POMDP policies
- Good scalability, **not optimal**
- Results **transferrable**
- Future work: **More principled** approach to finite-memory strategies

Steven Carr, Nils Jansen, Ralf Wimmer, Alexandru Constantin Serban, Bernd Becker, Ufuk Topcu:
Counterexample-Guided Strategy Improvement for POMDPs Using Recurrent Neural Networks. IJCAI (2019)

Steven Carr, Nils Jansen, Ralf Wimmer, Alexandru Constantin Serban, Bernd Becker, Ufuk Topcu:
Stories - Policy Synthesis for POMDPs

1. Game-based abstraction
2. Finite-memory controllers
3. Recurrent neural networks
4. Fun: Humans in the loop
Computers do not deal well with ambiguity. We have to tell them PRECISELY what we want them to do. Thus, computer science requires precise thinking from us.

The challenge of precise thinking attracted me to study computer science.

Moshe Vardi
NAS Member
Idea: Human-in-the-loop Synthesis for POMDPs
Idea: Human-in-the-loop Synthesis for POMDPs

Turn scenario into an arcade game

Underlying (family of) POMDPs
Idea: Human-in-the-loop Synthesis for POMDPs

Turn scenario into an arcade game
Collect data of human playing

Underlying (family of) POMDPs
Idea: Human-in-the-loop Synthesis for POMDPs

- Turn scenario into an arcade game
- Collect data of human playing
- From data, infer a strategy

Underlying (family of) POMDPs

Applying strategy yields DTMC, efficient verification
Idea: Human-in-the-loop Synthesis for POMDPs

- Turn scenario into an arcade game
- Collect data of human playing
- From data, infer a strategy
- Put human in critical situations

Underlying (family of) POMDPs

Applying strategy yields DTMC, efficient verification

Counterexamples point to critical parts
Story: HiL Synthesis for POMDPs
Story: HiL Synthesis for POMDPs

- Safety Specification
- POMDP
- Training Environment
- gamify
Story: HiL Synthesis for POMDPs

Safety Specification → POMDP → Training Environment → Strategy Computation

- gamify
- collect user data
Story: HiL Synthesis for POMDPs

1. Safety Specification
2. POMDP
3. Training Environment
4. Model Checking
5. Strategy Computation
6. Collect user data
7. Gamify
8. Apply strategy

Nils Jansen
Radboud University
Story: HiL Synthesis for POMDPs

1. Safety Specification
2. POMDP
3. Training Environment
4. Model Checking
5. Strategy Computation
6. SAT

- Gamify
- Collect user data
- Apply strategy
- SAT

ACC 2018
Story: HiL Synthesis for POMDPs

1. Safety Specification
2. POMDP
3. Training Environment
4. Model Checking
5. Strategy Computation
6. Counterexample

- Safety Specification → POMDP
- POMDP → gamify
- Training Environment
- Training Environment → collect user data
- Counterexample → UNSAT
- UNSAT → Model Checking
- Model Checking → apply strategy
- apply strategy → Strategy Computation
- Strategy Computation → SAT
- SAT → ✔️

ACC 2018
Story: HiL Synthesis for POMDPs

Safety Specification → POMDP → Training Environment

counterexample-based refinement

collect user data

Counterexample → UNSAT → Model Checking

apply strategy

Strategy Computation → SAT

gamify

Counterexample-based refinement
Data Augmentation

- Strategy is trained on randomly generated environments
- Training set needs samples until further environments wouldn’t likely change the strategy

- To reduce training set, similar observations are handled similar
## Experiments

<table>
<thead>
<tr>
<th>Iteration</th>
<th>$\Pr(\neg B U G)$</th>
<th>Expected Cost ($EC_{\gamma}[C]$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.225</td>
<td>13.57</td>
</tr>
<tr>
<td>1</td>
<td>0.503</td>
<td>9.110</td>
</tr>
<tr>
<td>2</td>
<td>0.592</td>
<td>7.154</td>
</tr>
<tr>
<td>3</td>
<td>0.610</td>
<td>6.055</td>
</tr>
<tr>
<td>4</td>
<td>0.636</td>
<td>5.923</td>
</tr>
<tr>
<td>Optimal</td>
<td>$-n. a.-$</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>grid</th>
<th>HiL states</th>
<th>Synth time (s)</th>
<th>PRISM-POMDP states</th>
<th>PRISM-POMDP time (s)</th>
<th>PBVI states</th>
<th>PBVI time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$3 \times 3$</td>
<td>277</td>
<td>43.74</td>
<td>303</td>
<td>2.20</td>
<td>81</td>
<td>3.86</td>
</tr>
<tr>
<td>$4 \times 4$</td>
<td>990</td>
<td>121.74</td>
<td>987</td>
<td>4.64</td>
<td>256</td>
<td>2431.05</td>
</tr>
<tr>
<td>$5 \times 5$</td>
<td>2459</td>
<td>174.90</td>
<td>2523</td>
<td>213.53</td>
<td>625</td>
<td>$-MO-$</td>
</tr>
<tr>
<td>$6 \times 6$</td>
<td>5437</td>
<td>313.50</td>
<td>5743</td>
<td>$-MO-$</td>
<td>1296</td>
<td>$-MO-$</td>
</tr>
<tr>
<td>$10 \times 10$</td>
<td>44794</td>
<td>1668.30</td>
<td>54783</td>
<td>$-MO-$</td>
<td>$-MO-$</td>
<td>$-MO-$</td>
</tr>
<tr>
<td>$11 \times 11$</td>
<td>$-MO-$</td>
<td>$-MO-$</td>
<td>81663</td>
<td>$-MO-$</td>
<td>$-MO-$</td>
<td>$-MO-$</td>
</tr>
</tbody>
</table>
Conclusion and Future Work

1. Game-based abstraction
2. Finite-memory controllers
3. Recurrent neural networks
4. Fun: Humans in the loop

• Several directions to compute provably correct finite-memory policies for POMDPs
• Work on the intersection of AI, Machine Learning, and Formal Methods
• Future work: A toolbox!
• Call for collaboration:
  • Extract finite-state controllers from recurrent neural networks
  • Automata learning for stochastic systems
  • Whatever is fun!