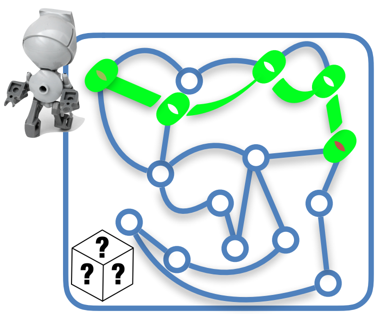
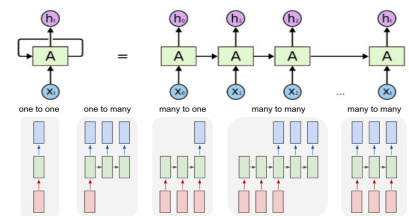
Automation and Planning under Uncertainty and Partial Observability

Machine Learning and Formal Verification
(and Humans) Join Forces?





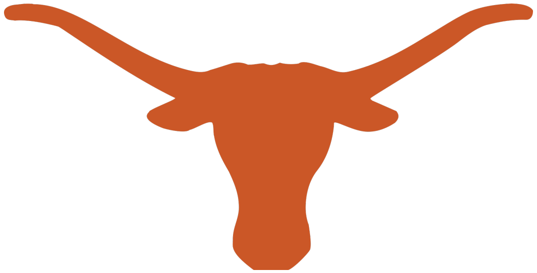
Nils Jansen

CM Labs, April 16, 2019

RWTH Aachen University, Germany

Lehrstuhl für Informatik 2 Softwaremodellierung und Verifikation | RWTH AACHEN UNIVERSITY

UT Austin, TX, USA

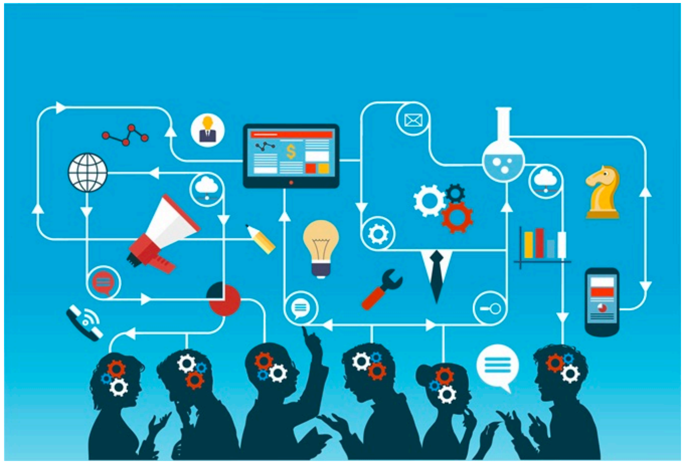


ICES

Radboud University, Nijmegen

Radboud University

joint work with:



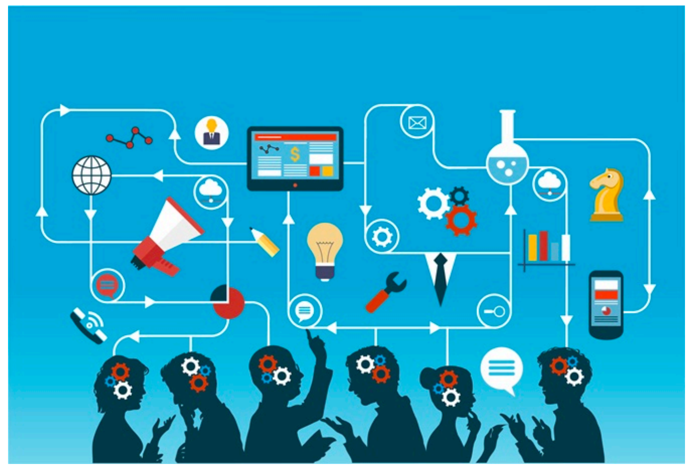
joint work with:

Caltech

UNIVERSITY OF OXFORD

KTH VETENSKAP OCH KONST

TU Graz



AIR FORCE RESEARCH LABORATORY

Lehrstuhl für Informatik 2 Softwaremodellierung und Verifikation

RWTH AACHEN UNIVERSITY

océ

A CANON COMPANY

UNI FREIBURG

TEXAS
The University of Texas at Austin

PHILIPS
Healthcare

Autonomous Systems

- How to affordably build trustworthy systems?

Autonomous Systems

- How to affordably build trustworthy systems?
- Use a reliable system model

Autonomous Systems

- How to affordably build trustworthy systems?
- Use a reliable system model

Formal verification

Autonomous Systems

- How to affordably build trustworthy systems?
- Use a reliable system model

Formal verification

Autonomous Systems

- How to affordably build trustworthy systems?
- Use a reliable system model

humans

- Operate along uncontrollable agents, in uncertain, or partially observable environments

Formal verification

Autonomous Systems

- How to affordably build trustworthy systems?
- Use a reliable system model humans
- Operate along uncontrollable agents, in uncertain, or partially observable environments

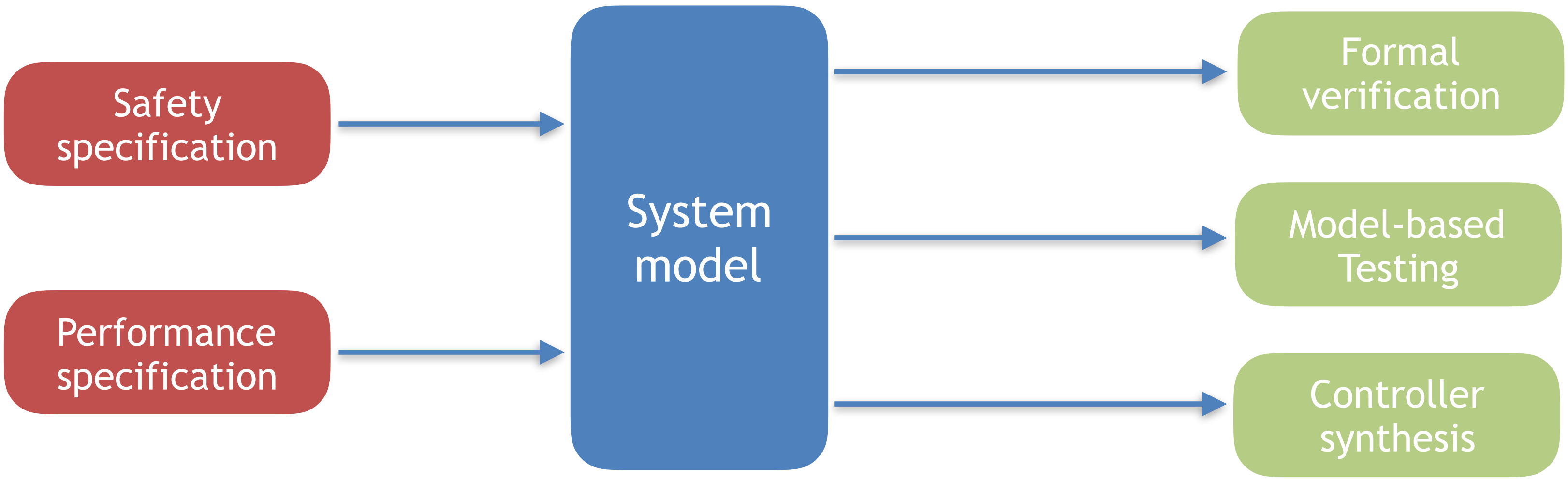
Autonomous Systems

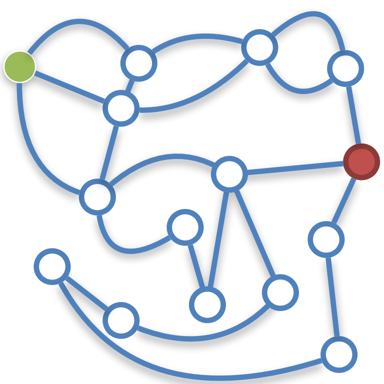
- How to affordably build trustworthy systems?
- Use a reliable system model
- Operate along uncontrollable agents, in uncertain, or partially observable environments

humans

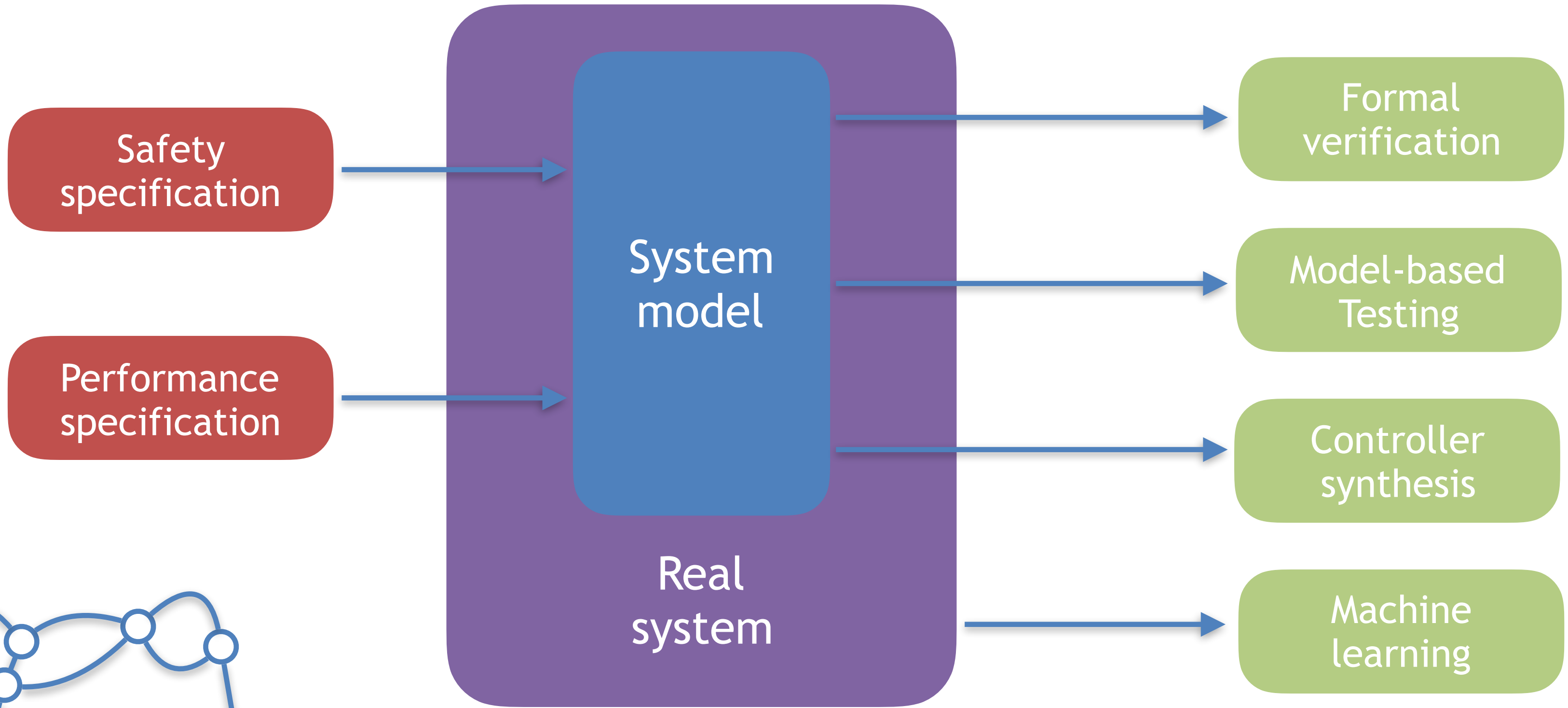
Formal verification needs to account for these factors.

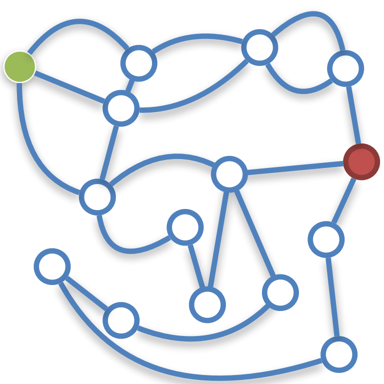
Autonomous (Cyber-physical) Systems



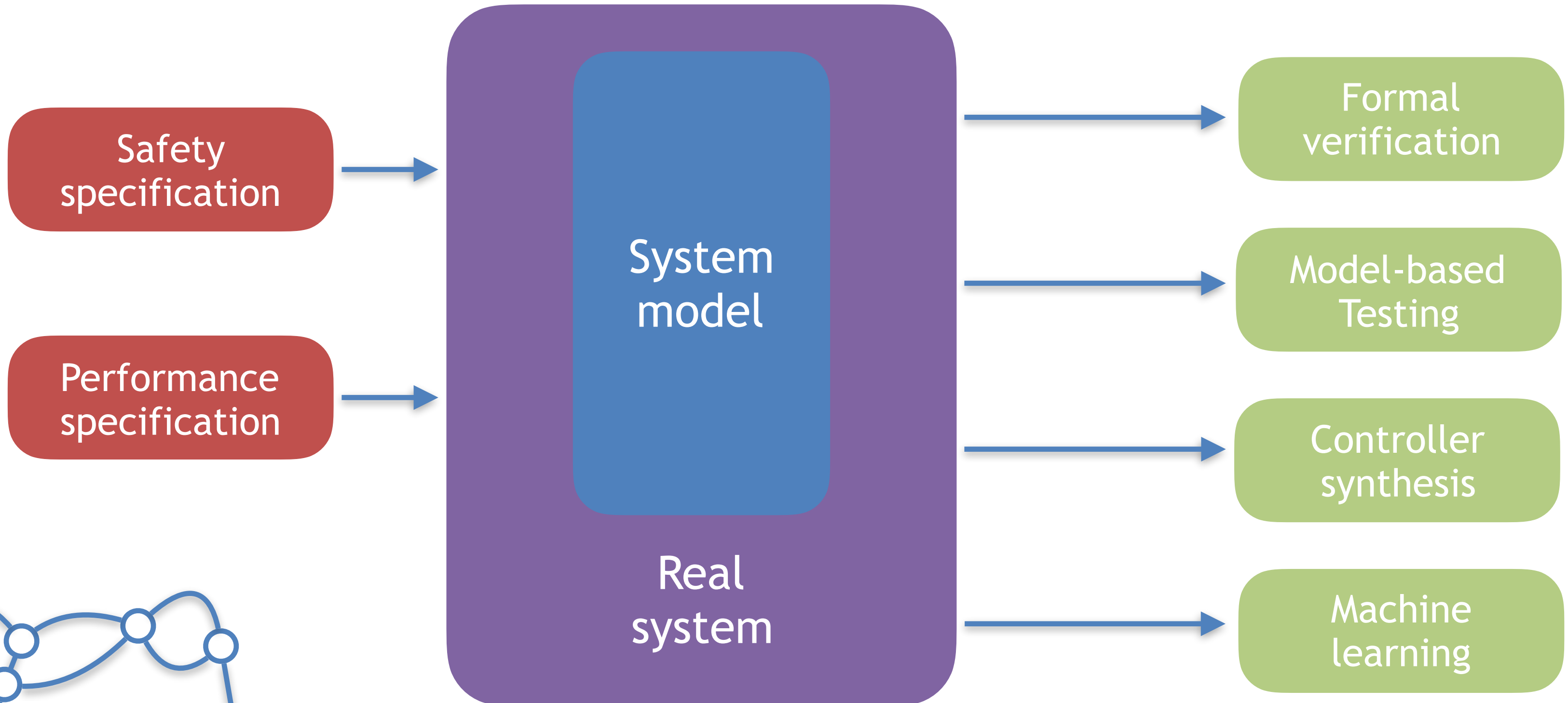


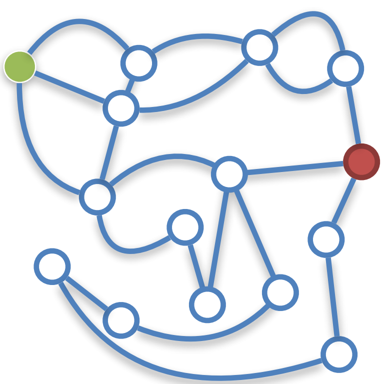
Autonomous (Cyber-physical) Systems



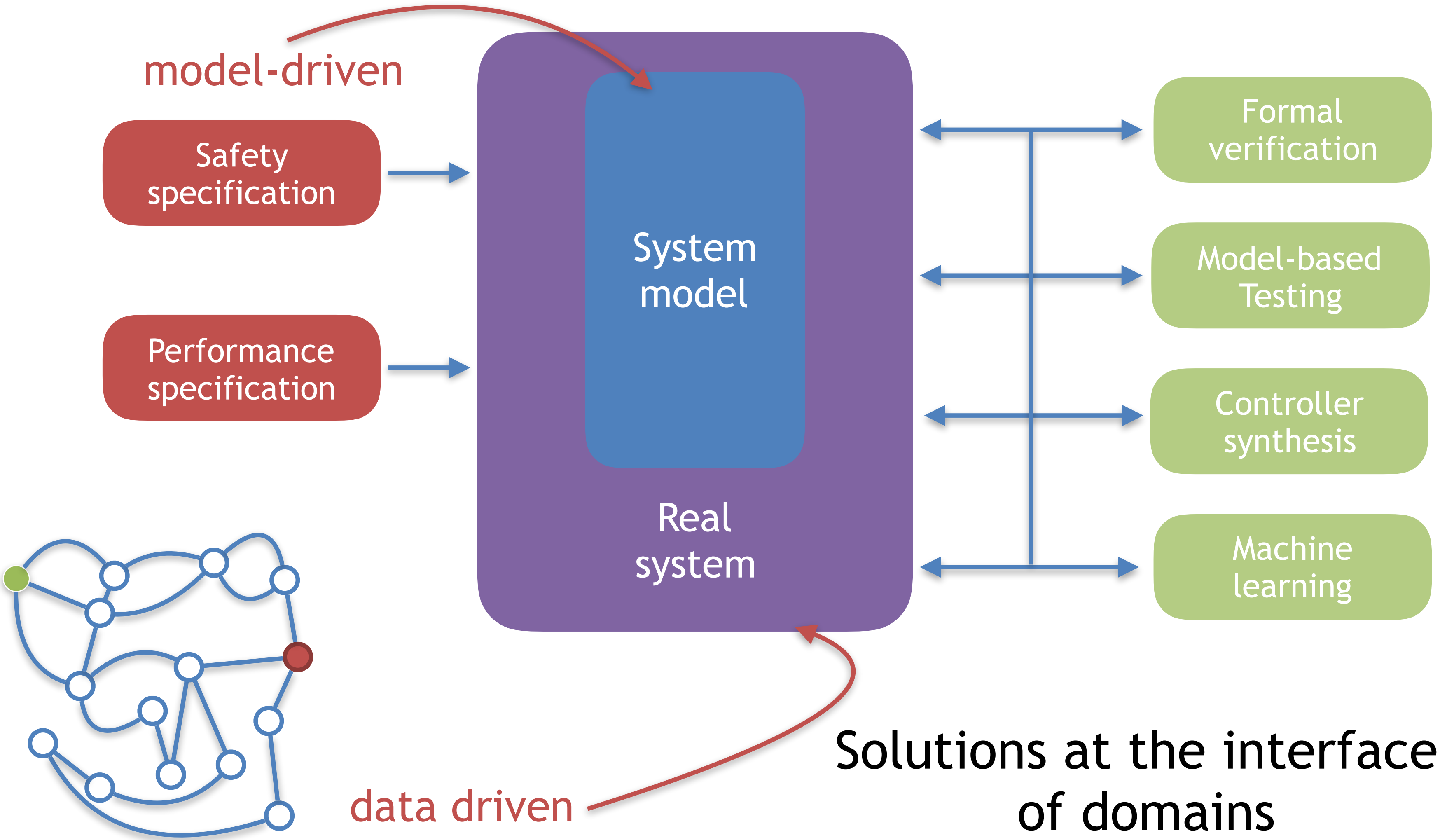


Autonomous (Cyber-physical) Systems





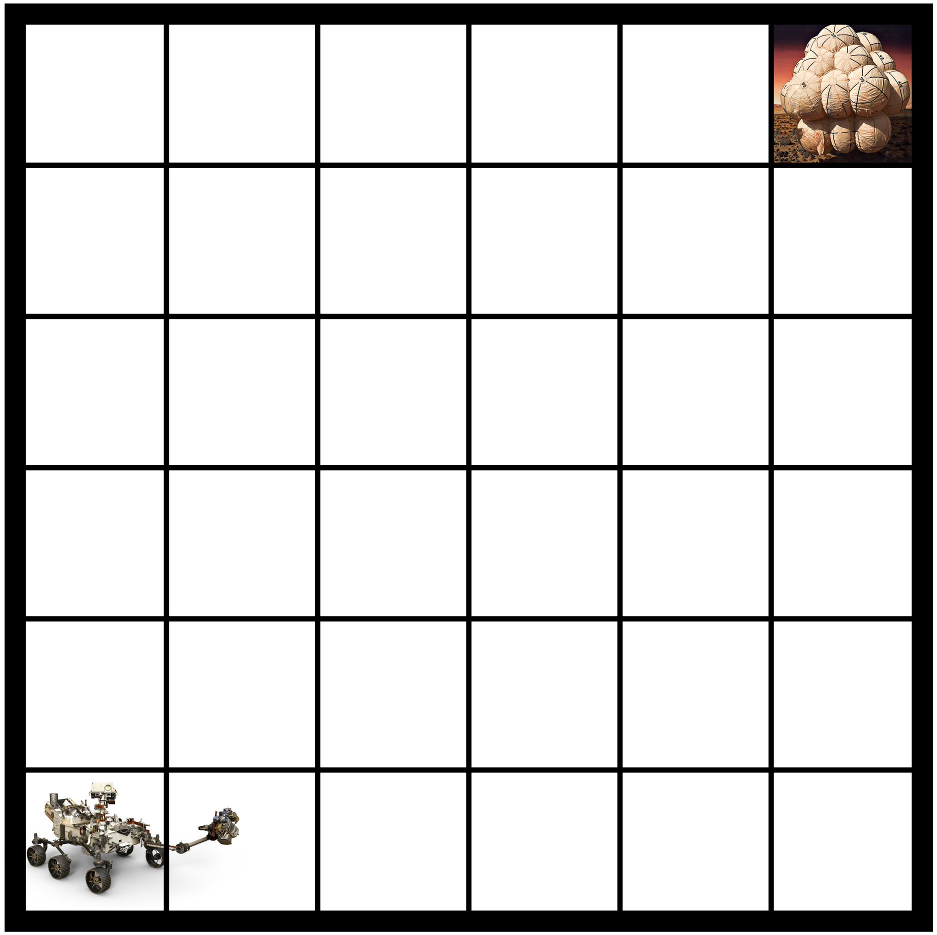
Autonomous (Cyber-physical) Systems



Solutions at the interfaces of domains

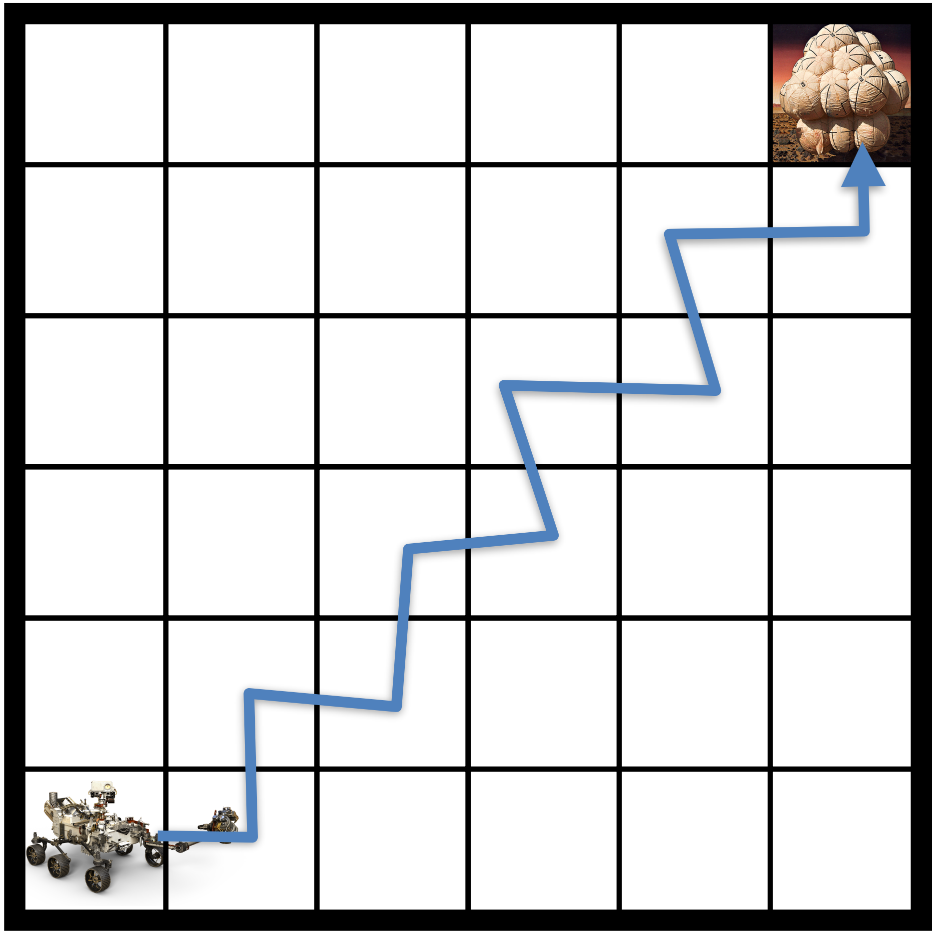
The “Help the Robot” Planning Problem

Find the best way to the airbag



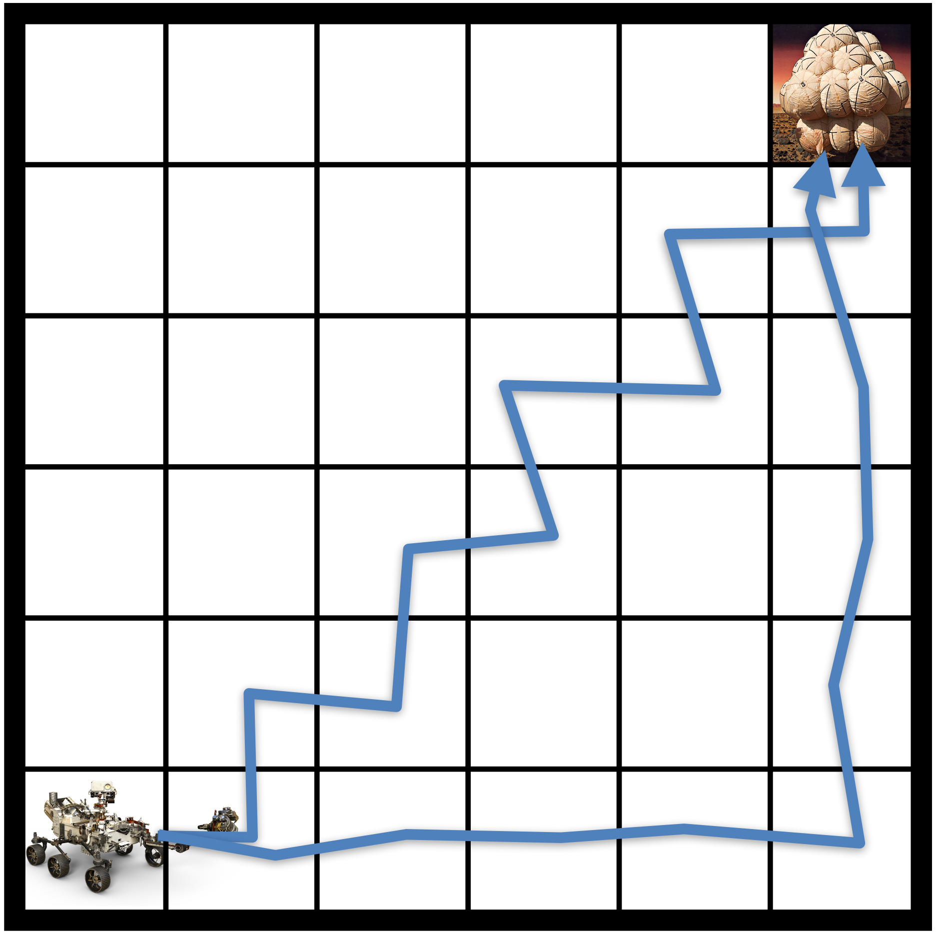
The "Help the Robot" Planning Problem

Find the best way to the airbag



The “Help the Robot” Planning Problem

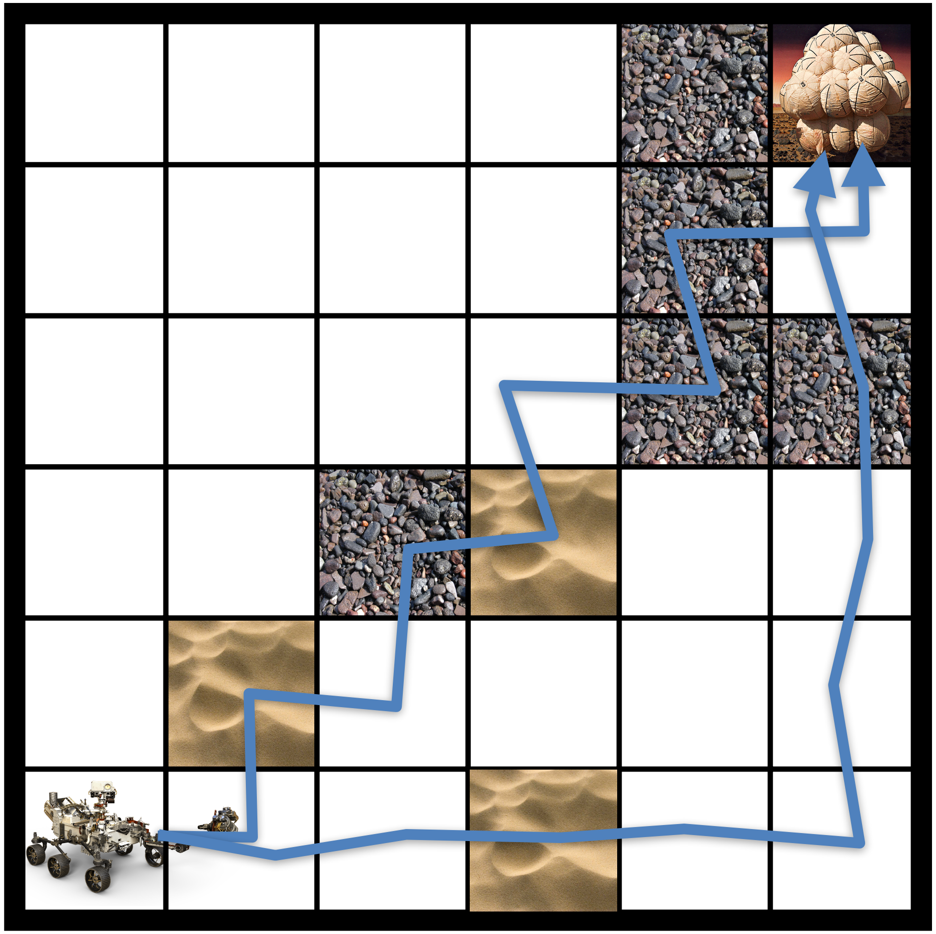
Find the best way to the airbag



The “Help the Robot” Planning Problem

Find the best way to the airbag

While moving, robot discovers expensive surfaces

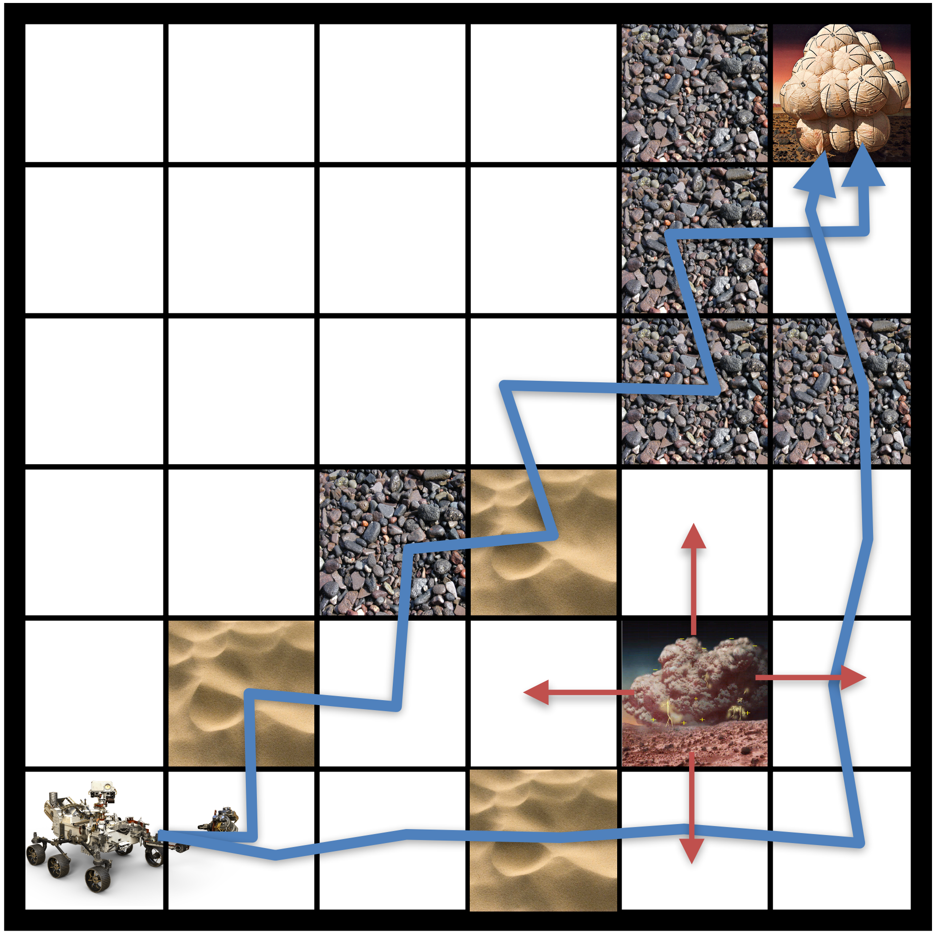


The “Help the Robot” Planning Problem

Find the best way to the airbag

While moving, robot discovers expensive surfaces

Avoid randomly moving dust storm

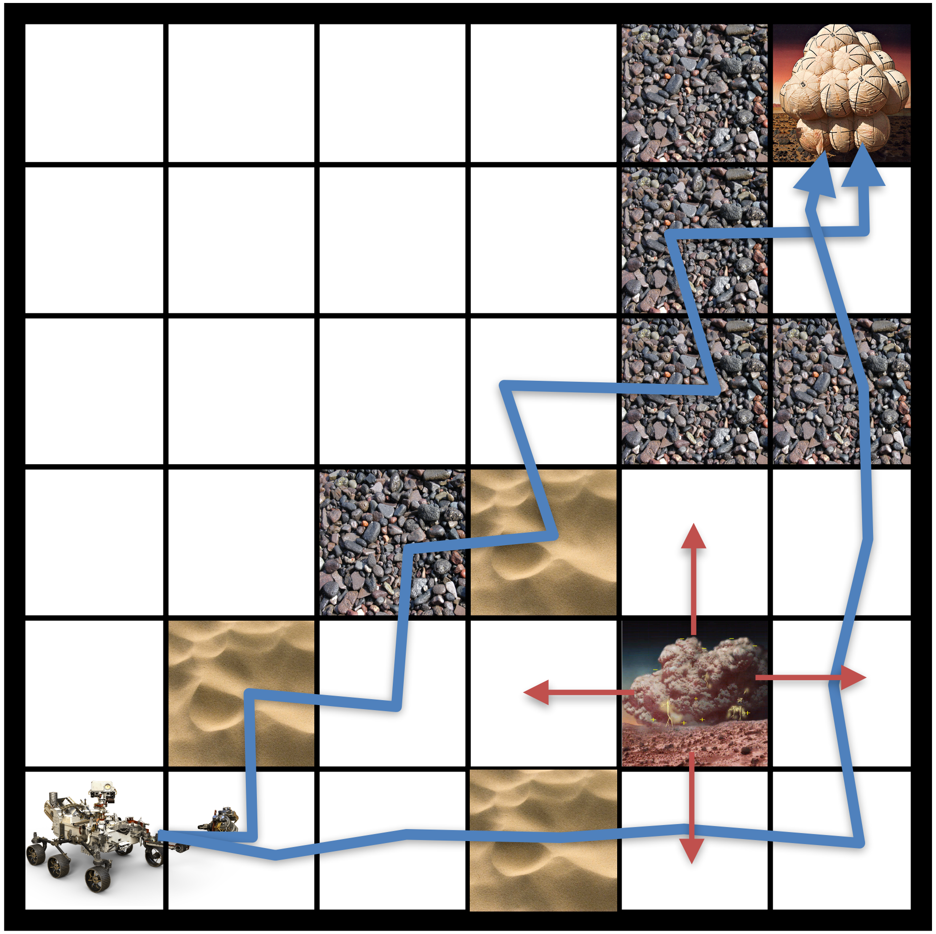


The “Help the Robot” Planning Problem

Find the best way to the airbag

While moving, robot discovers expensive surfaces

Avoid randomly moving dust storm



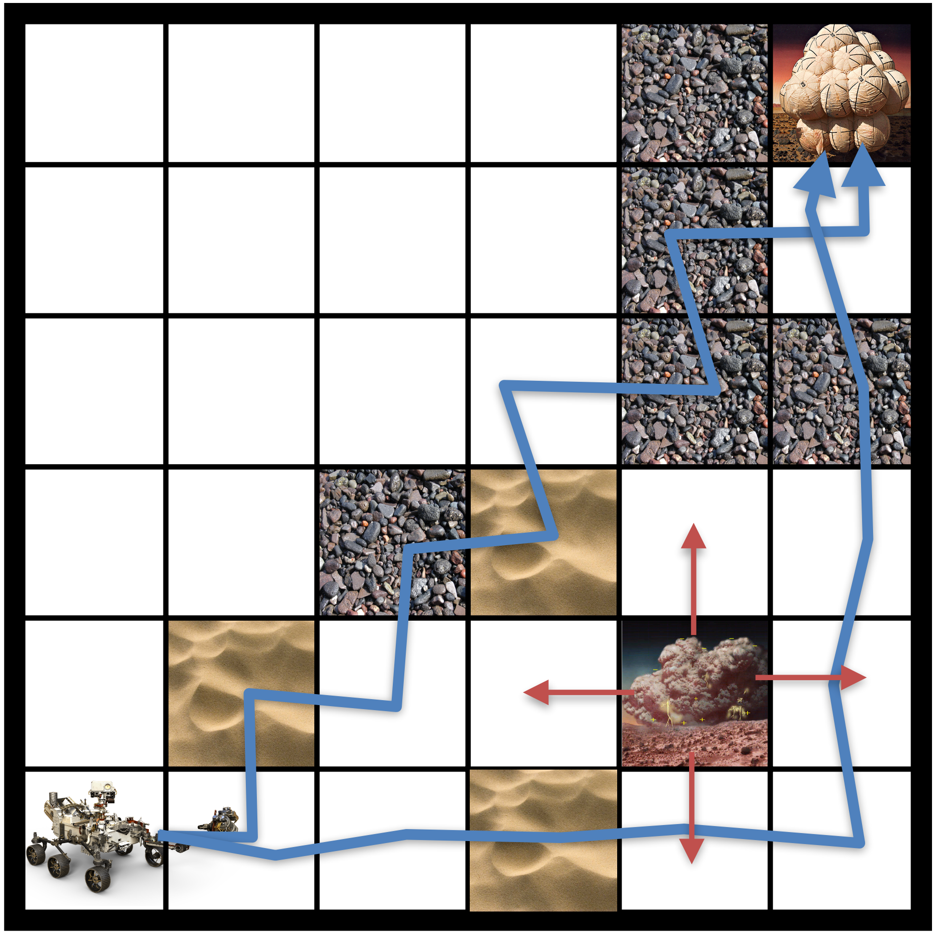
Find **safe and cost-optimal** strategy to get to the airbag

The "Help the Robot" Planning Problem

Find the best way to the airbag

While moving, robot discovers expensive surfaces

Avoid randomly moving dust storm



Find **safe and cost-optimal** strategy to get to the airbag

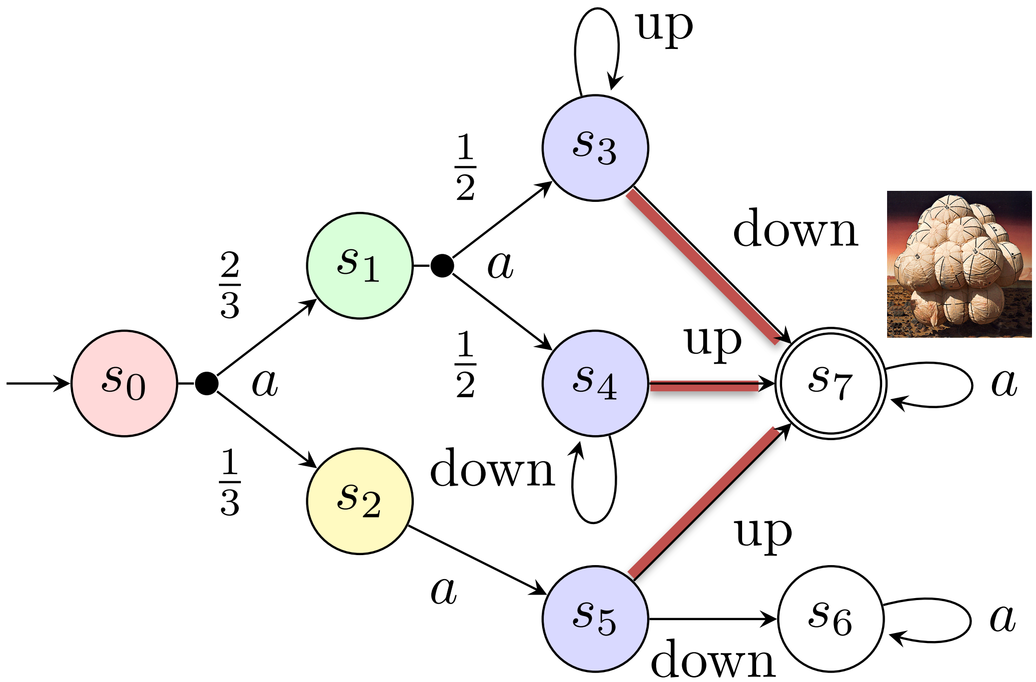
Underlying Model: Markov Decision Process

Obtained via Reinforcement Learning

Markov Decision Process

$$Pr_{max}(\Diamond s_7)$$

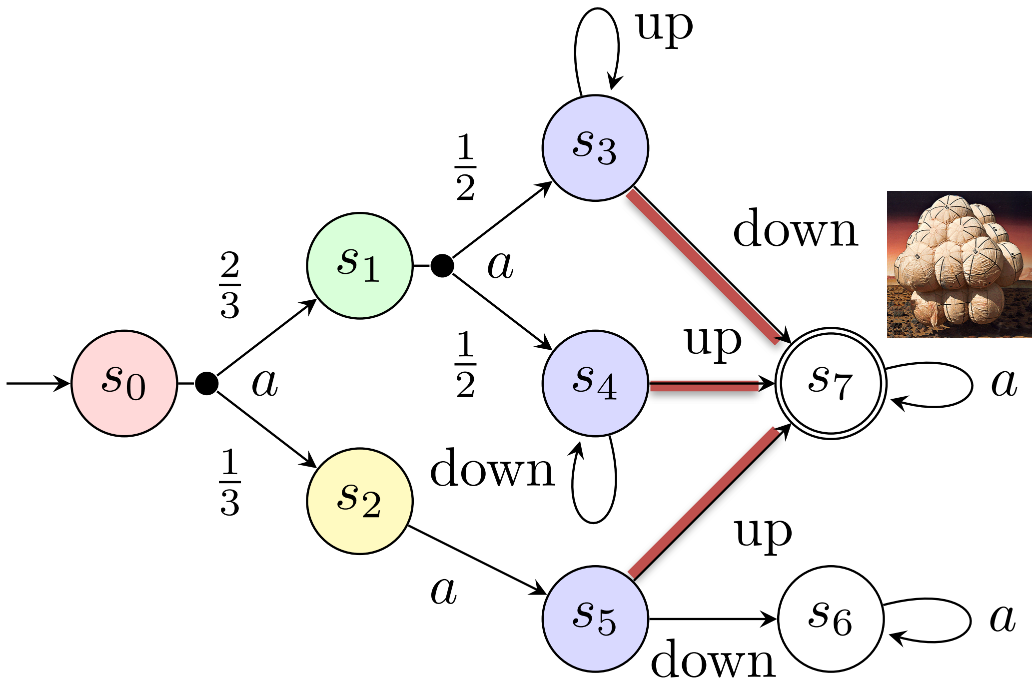
$$EC_{min}(\Diamond s_7)$$



Markov Decision Process

$$Pr_{max}(\Diamond s_7)$$

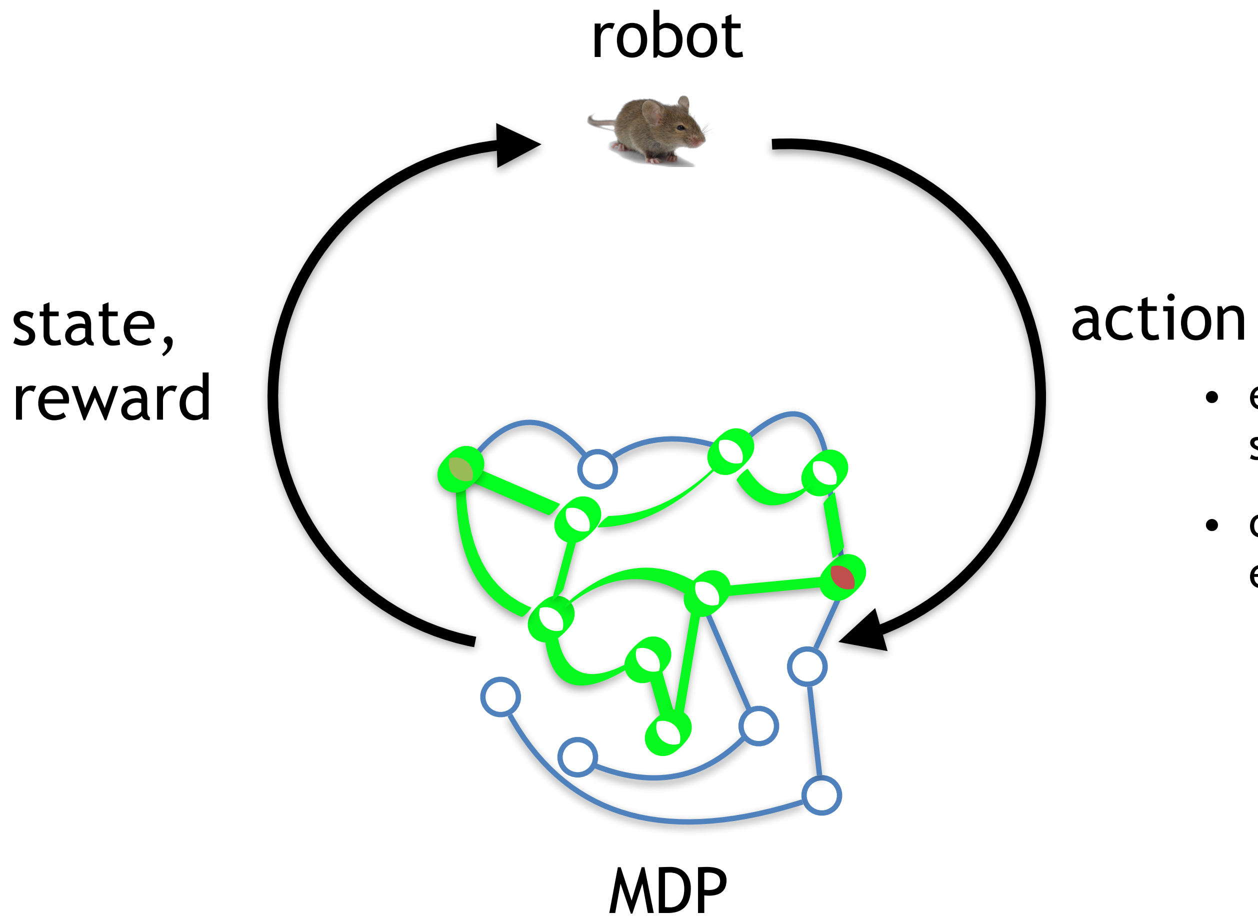
$$EC_{min}(\Diamond s_7)$$



billions of states

efficient verification, however, model may not be fully known

Reinforcement Learning



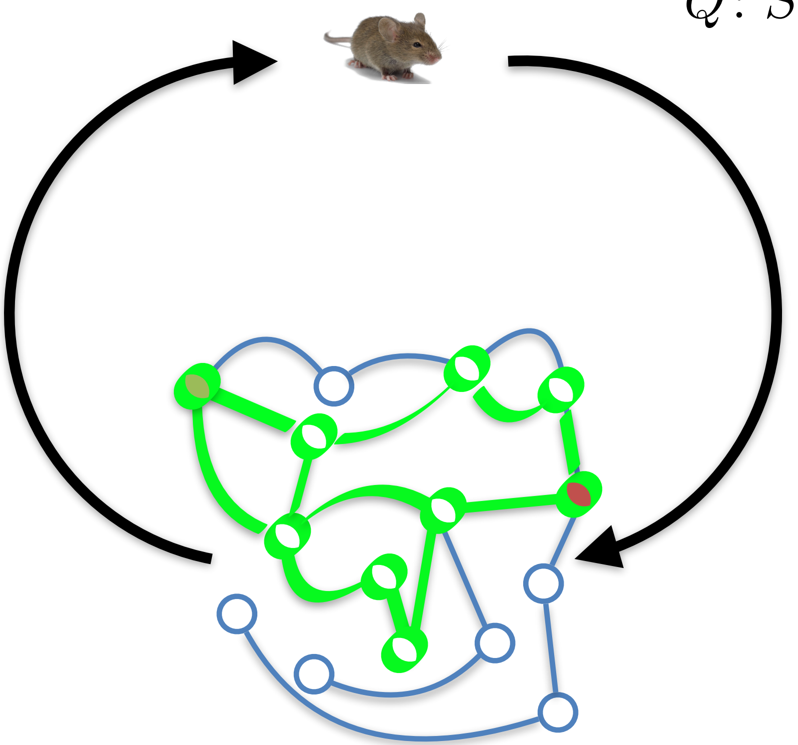
- episodic exploration of the state space
- collect information about environment by interaction

Reinforcement Learning

robot

maintains
$Q : S \times Act \to \mathbb{R}$

state,
reward



action

MDP

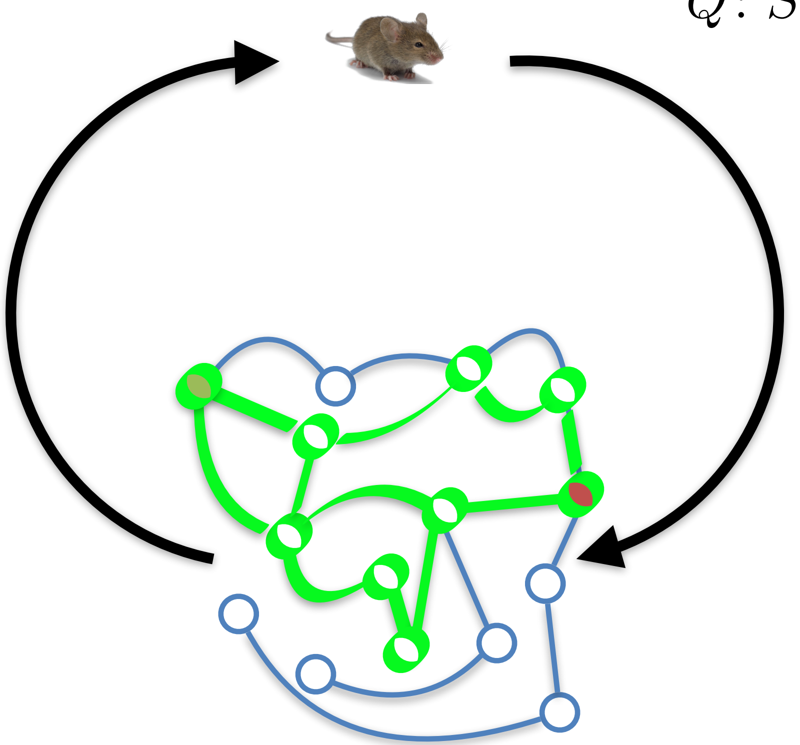
- episodic exploration of the state space
- collect information about environment by interaction
- Q-learning approximates optimal value function
- Q-matrix contains values of taking each action at each state

Reinforcement Learning

robot

maintains
$Q\colon S \times Act \to \mathbb{R}$

state, reward



action

MDP

- episodic exploration of the state space
- collect information about environment by interaction
- Q-learning approximates optimal value function
- Q-matrix contains values of taking each action at each state

Maximizes expected reward but neglects safety!

Stories

1. Safe Reinforcement Learning via Formal Verification and Behavior Models
2. Planning under Partial Observability
   I. Human-in-the-loop Planning via Gamification
   II. Planning via Recurrent Neural Networks

Stories

1. Safe Reinforcement Learning via Formal Verification and Behavior Models

2. Planning under Partial Observability

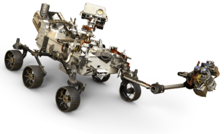
    I. Human-in-the-loop Planning via Gamification

    II. Planning via Recurrent Neural Networks

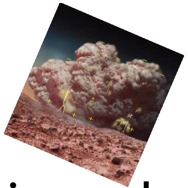
Partially Controllable Multi-agent Systems

Partially Controllable Multi-agent Systems

Autonomous agent amongst uncontrollable agents



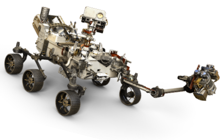
- find a control strategy for the autonomous agent (**Avatar**)
- provably adhering to safety and performance specifications



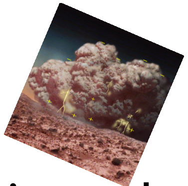
- account for uncontrollable agents (**Adversaries**)
- self driving cars, autonomous trading agents, service robots

Partially Controllable Multi-agent Systems

Autonomous agent amongst uncontrollable agents



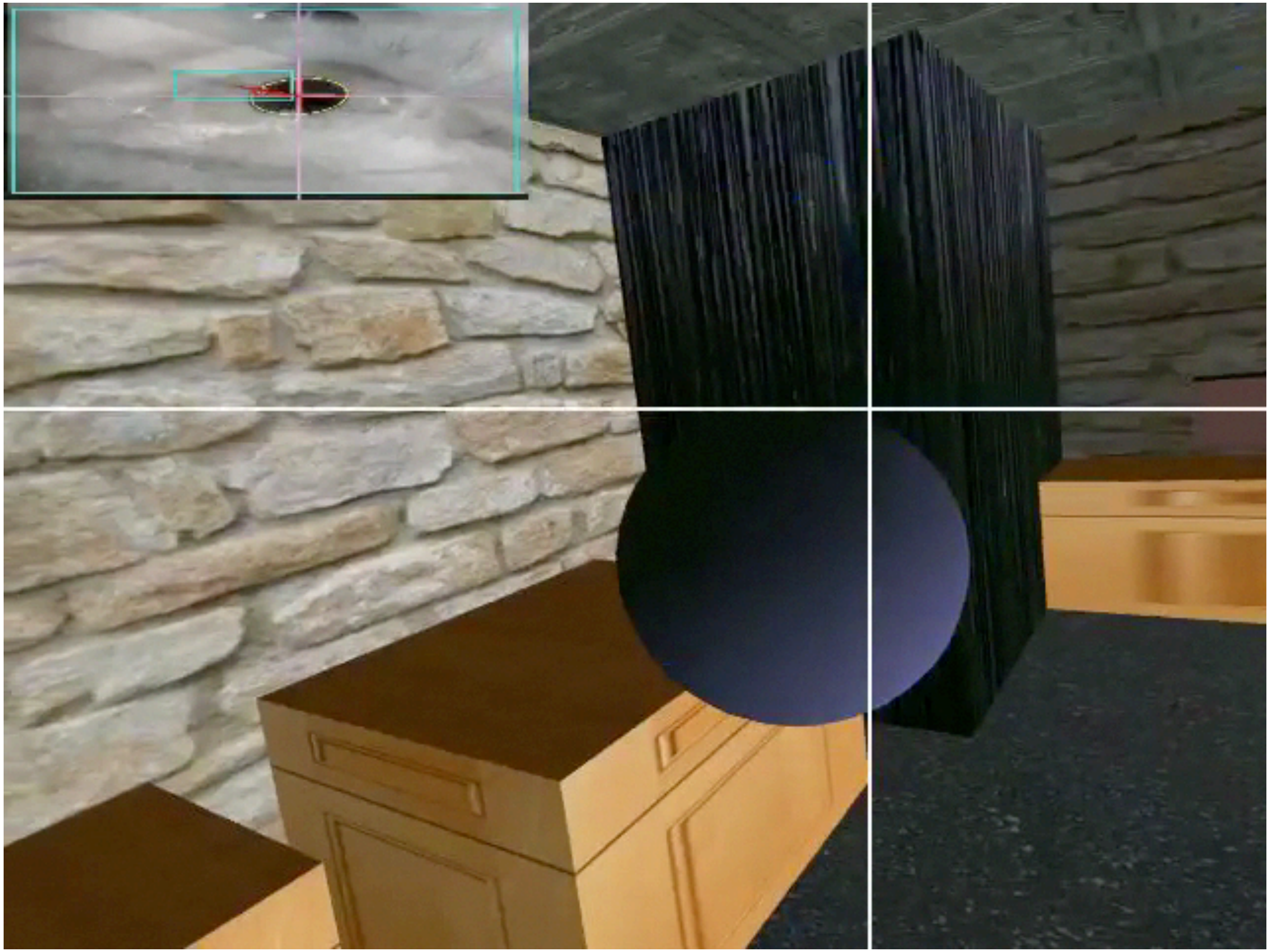
- find a control strategy for the autonomous agent (**Avatar**)
- provably adhering to safety and performance specifications
- account for uncontrollable agents (**Adversaries**)

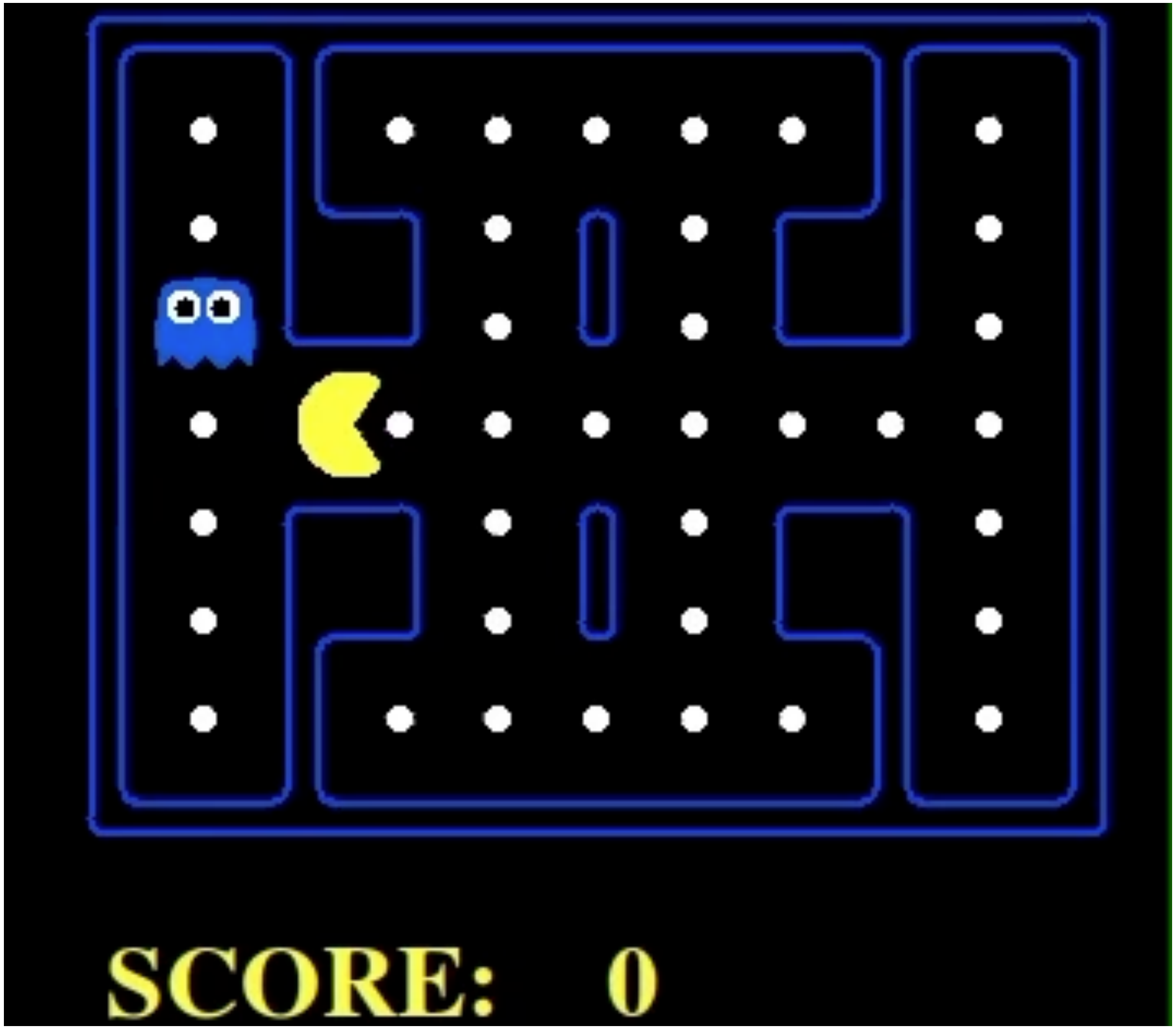


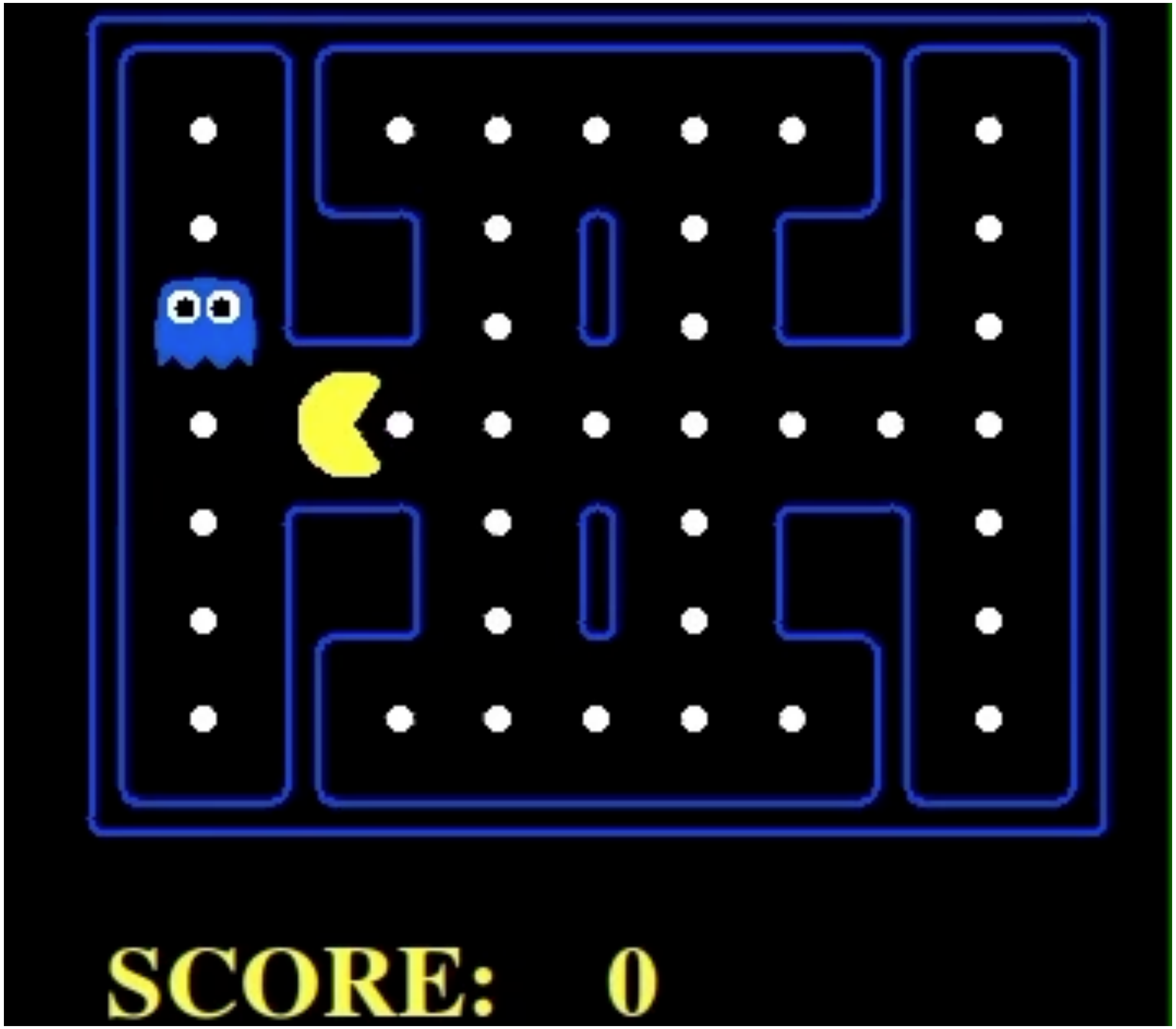
- self driving cars, autonomous trading agents, service robots

Behavior model for uncontrollable agents

- using reinforcement learning
- encode data from observations
- cast behavior model into Markov chain, accounting for likelihoods of choices




SCORE: 0


SCORE: 0

Formal Setting

Game Arena

- Finite graph $G = (V, E, d)$
- Edges $E \subseteq V \times V$
- Distance $d: E \rightarrow \mathbb{N}_{\geq 0}$
- Tokens $\circ: E \rightarrow \{0,1\}$
- Reward associated with tokens

Multi-agent Setting

- Controllable avatar
- Uncontrollable adversaries

Formal Setting

Game Arena

- Finite graph $G = (V, E, d)$
- Edges $E \subseteq V \times V$
- Distance $d\colon E \to \mathbb{N}_{\geq 0}$
- Tokens $\circ\colon E \to \{0,1\}$
- Reward associated with tokens

Safety:
Avatar and adversaries do not collide

Multi-agent Setting

- Controllable avatar
- Uncontrollable adversaries

Formal Setting

Game Arena

- Finite graph $G = (V, E, d)$
- Edges $E \subseteq V \times V$
- Distance $d \colon E \to \mathbb{N}_{\geq 0}$
- Tokens $\circ \colon E \to \{0,1\}$
- Reward associated with tokens

Safety:
Avatar and adversaries do not collide

Performance:
Collect as much reward as possible

Multi-agent Setting

- Controllable avatar
- Uncontrollable adversaries

Formal Setting

Game Arena

- Finite graph $G = (V, E, d)$
- Edges $E \subseteq V \times V$
- Distance $d\colon E \to \mathbb{N}_{\geq 0}$
- Tokens $\circ\colon E \to \{0,1\}$
- Reward associated with tokens

Multi-agent Setting

- Controllable avatar
- Uncontrollable adversaries

Safety:
Avatar and adversaries do not collide

Performance:
Collect as much reward as possible

Tokens and rewards not relevant for safety!

Formal Setting

Game Arena

- Finite graph $G = (V, E, d)$
- Edges $E \subseteq V \times V$
- Distance $d \colon E \to \mathbb{N}_{\geq 0}$
- Tokens $\circ \colon E \to \{0,1\}$
- Reward associated with tokens

Multi-agent Setting

- Controllable avatar
- Uncontrollable adversaries

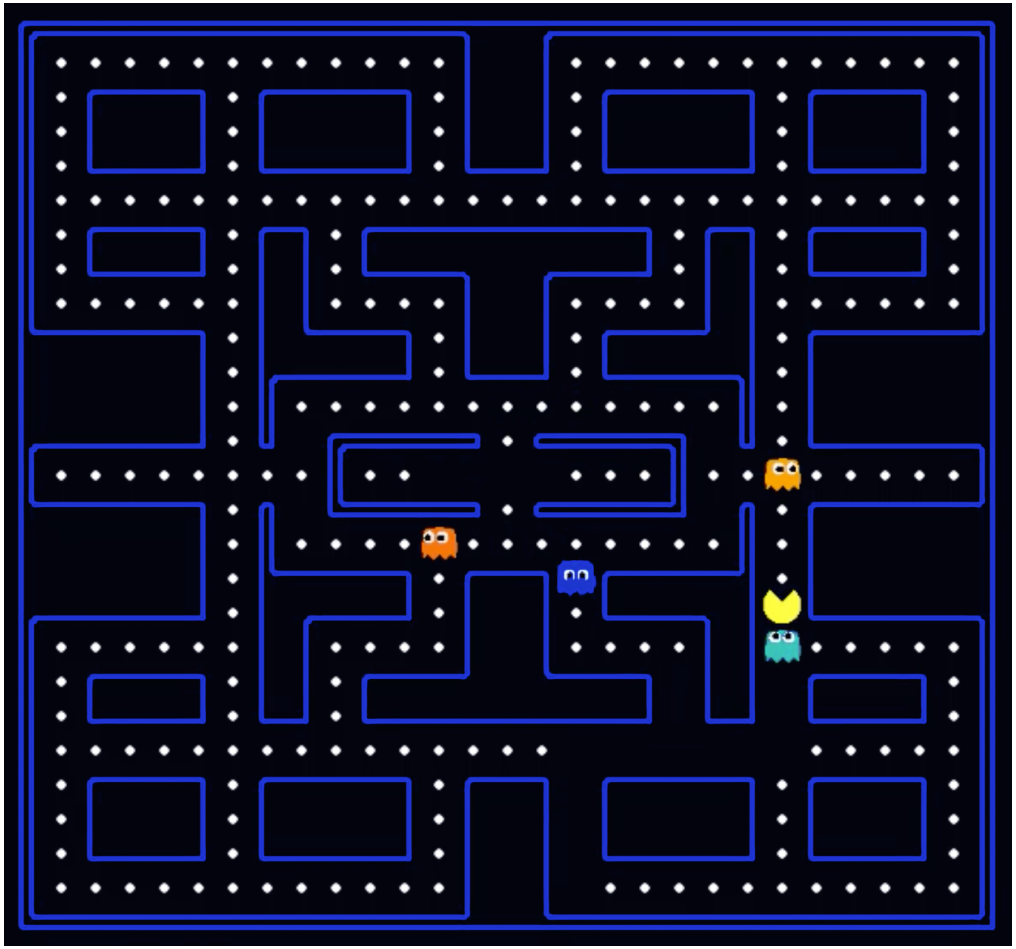
Safety:
Avatar and adversaries do not collide

Performance:
Collect as much reward as possible

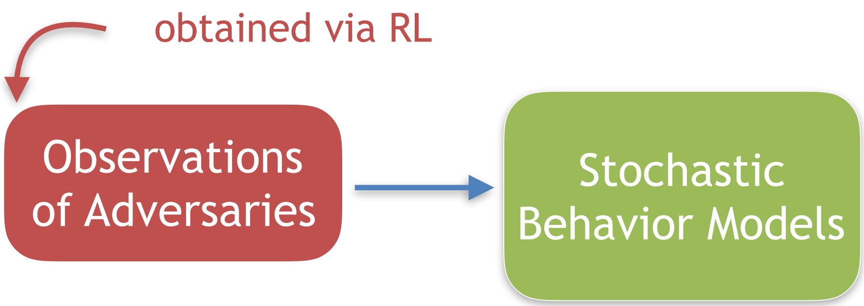
Tokens and rewards not relevant for safety!

Arena plus behavior model for adversaries yields MDP!

Model is Huge!

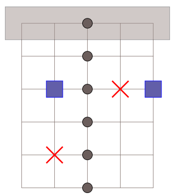


Story - Safety Shield



Story - Safety Shield

concrete scenario



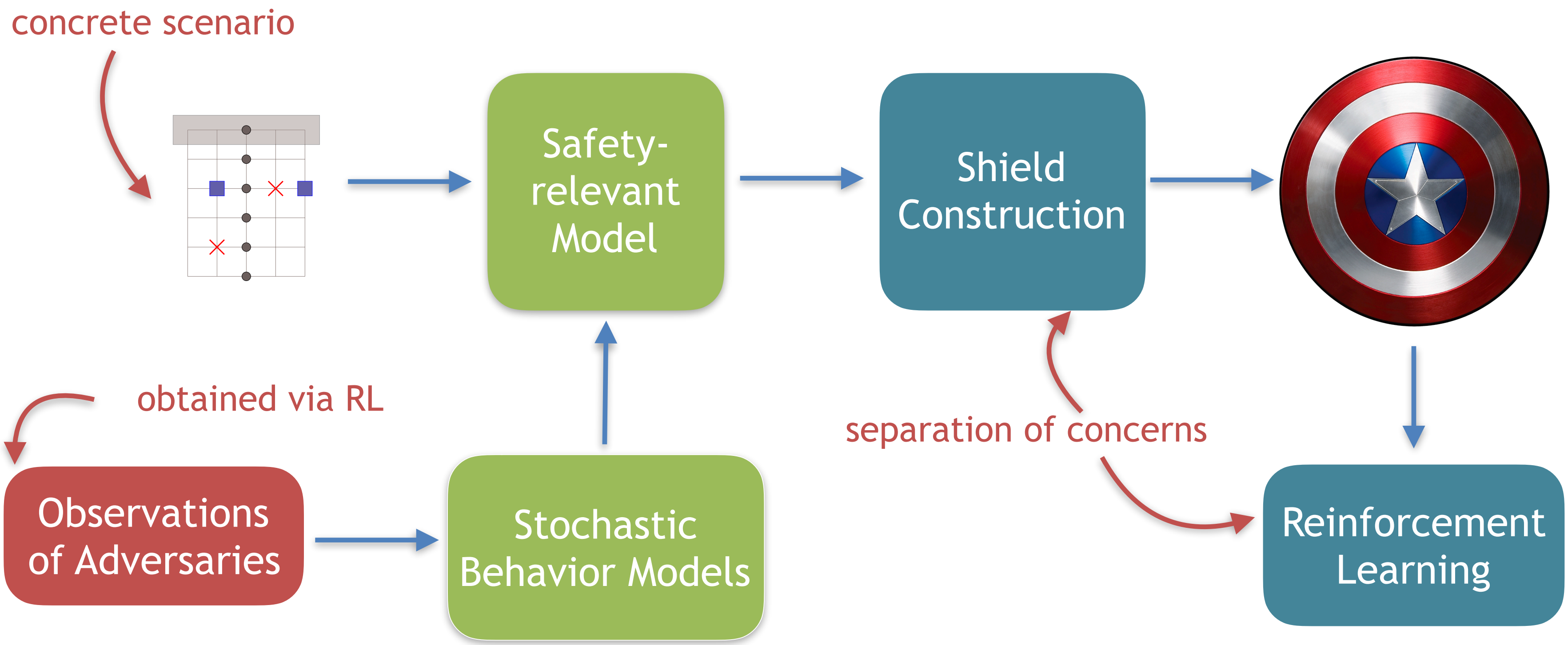
Safety-relevant Model

obtained via RL

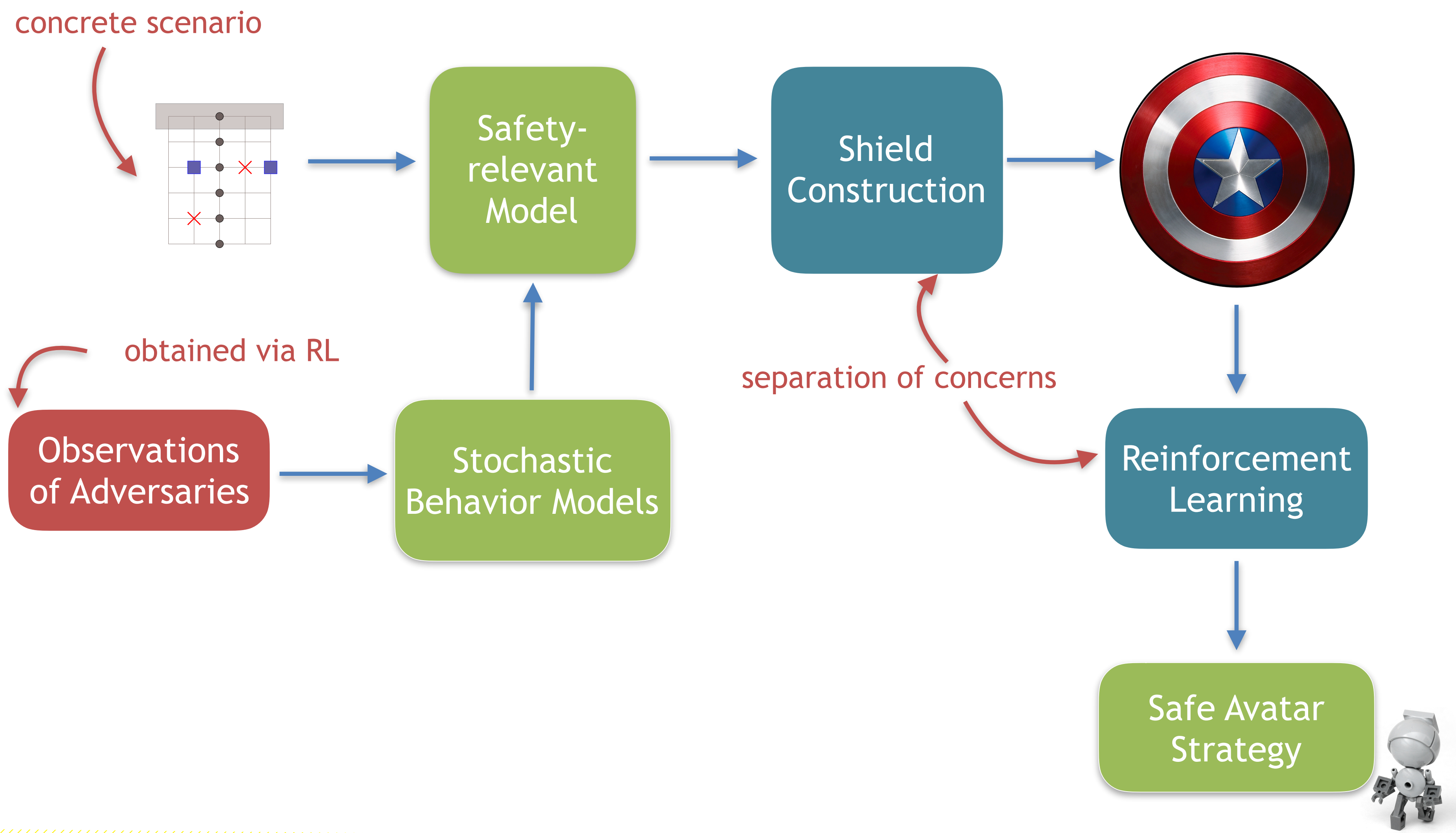
Observations of Adversaries

Stochastic Behavior Models

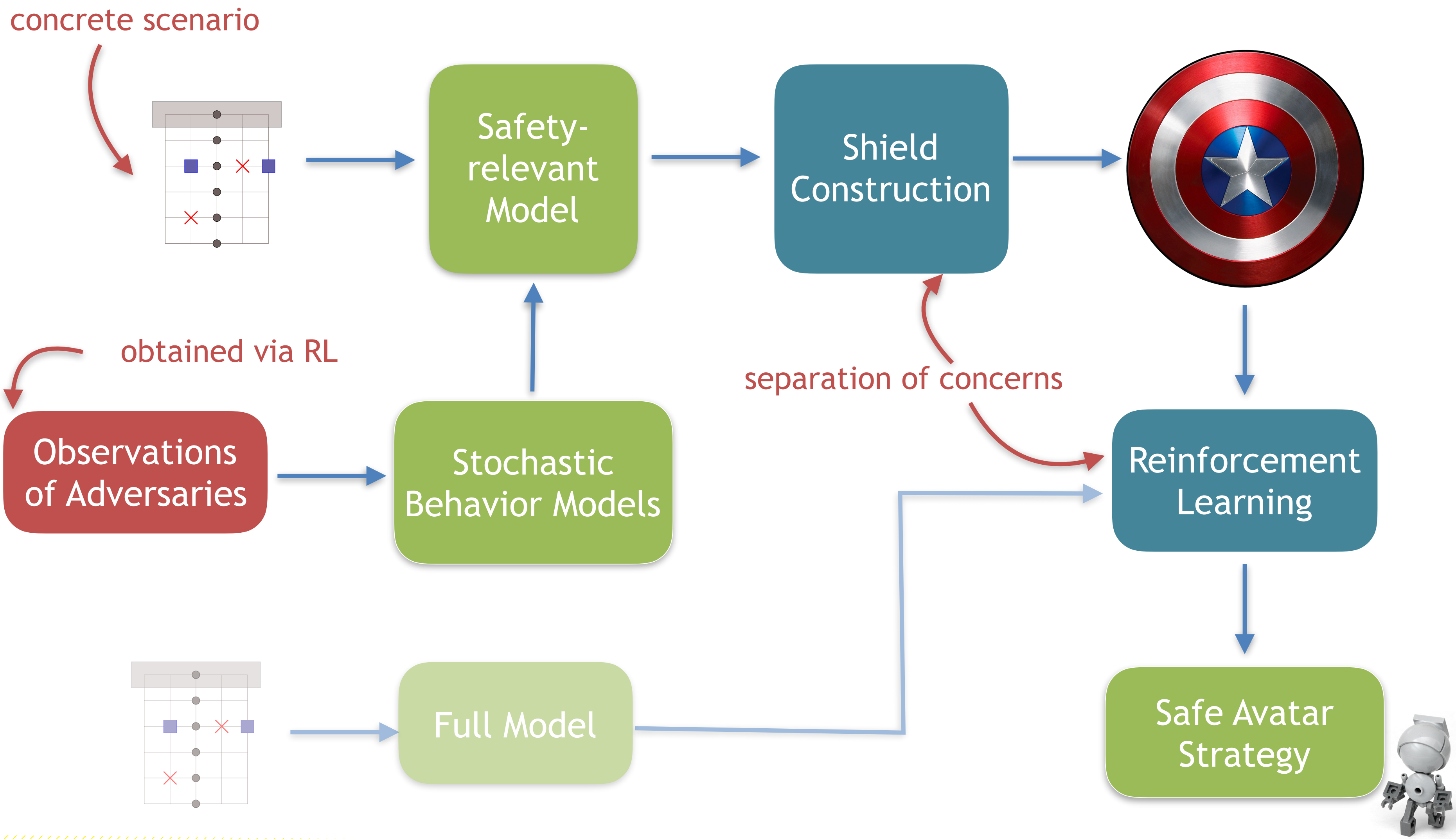
Story - Safety Shield



Story - Safety Shield



Story - Safety Shield

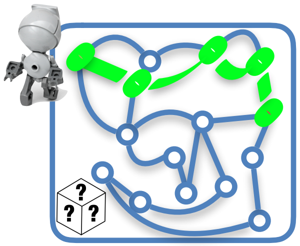


But Does it Scale?

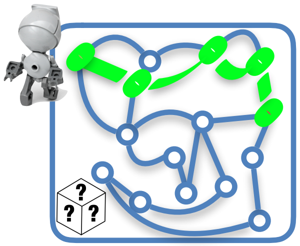
But Does it Scale?

Finite Horizon

- safety for finite number of steps
- infinite horizon may cause large errors anyways



But Does it Scale?



Finite Horizon

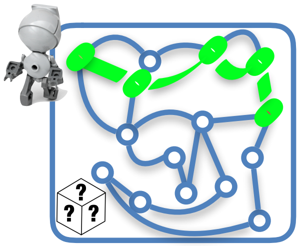
- safety for finite number of steps
- infinite horizon may cause large errors anyways

Piecewise Construction

- compute shield for each state independently
- in parellel!

But Does it Scale?

Finite Horizon



- safety for finite number of steps
- infinite horizon may cause large errors anyways

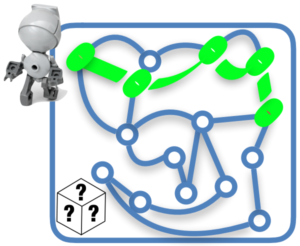
Piecewise Construction

- compute shield for each state independently
- in parellel!

Independent Agents

- crashing probabilities for different agents are stochastically independent
- compute individually, compose shields

But Does it Scale?



Finite Horizon

- safety for finite number of steps
- infinite horizon may cause large errors anyways

Piecewise Construction

- compute shield for each state independently
- in parellel!

Independent Agents

- crashing probabilities for different agents are stochastically independent
- compute individually, compose shields

Abstractions

- adversaries may be far away
- neglect adversary positions that are not relevant

Shielding vs Progress

Guaranteed Safety vs Sufficient Progress

Shielding vs Progress

Guaranteed Safety vs Sufficient Progress

Iterative Weakening

- When progress of avatar is decreasing, weaken shield
- No new computation, on-the-fly based on computed values

Shielding vs Progress

Guaranteed Safety vs Sufficient Progress

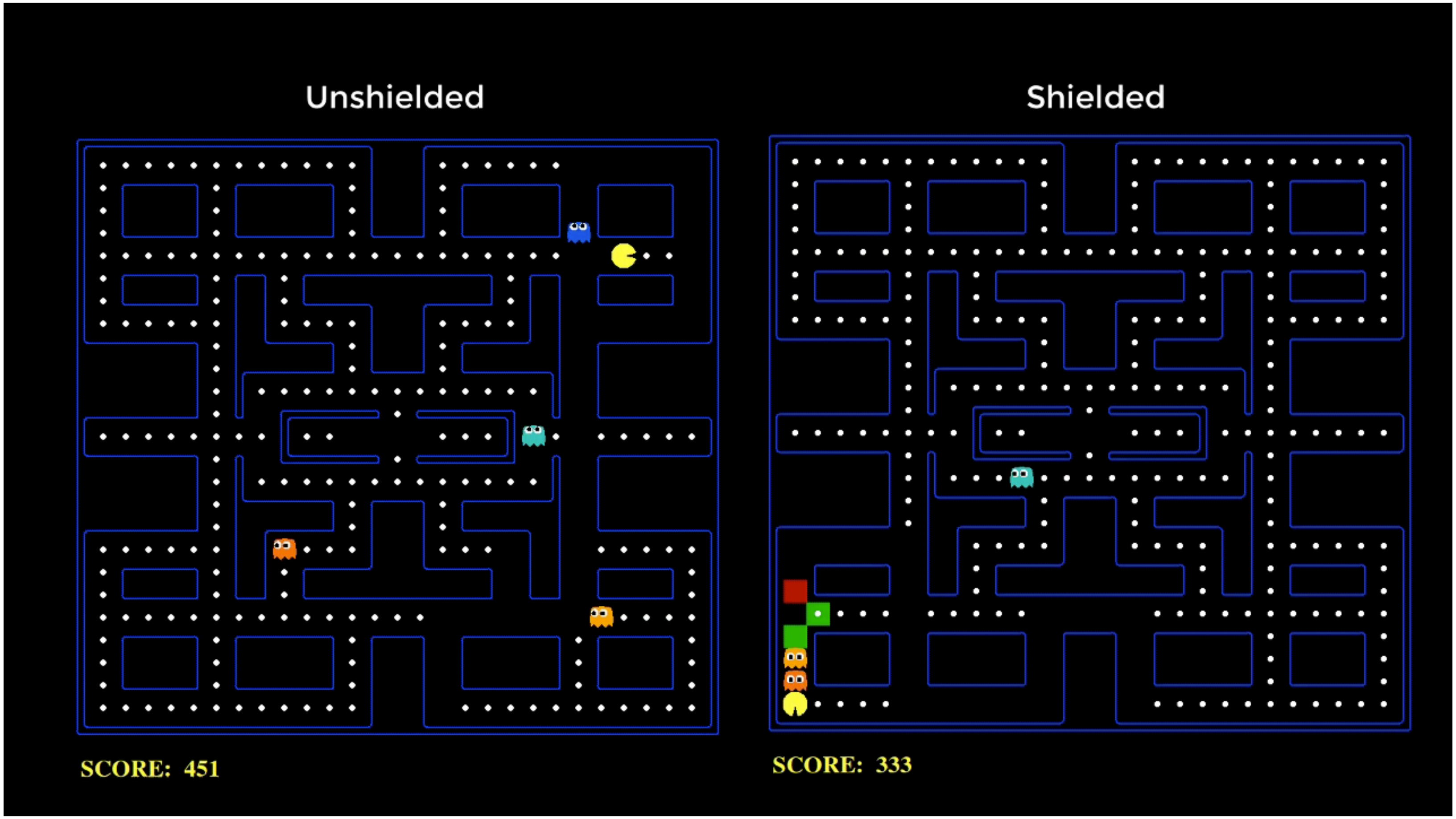
Iterative Weakening

- When progress of avatar is decreasing, weaken shield
- No new computation, on-the-fly based on computed values

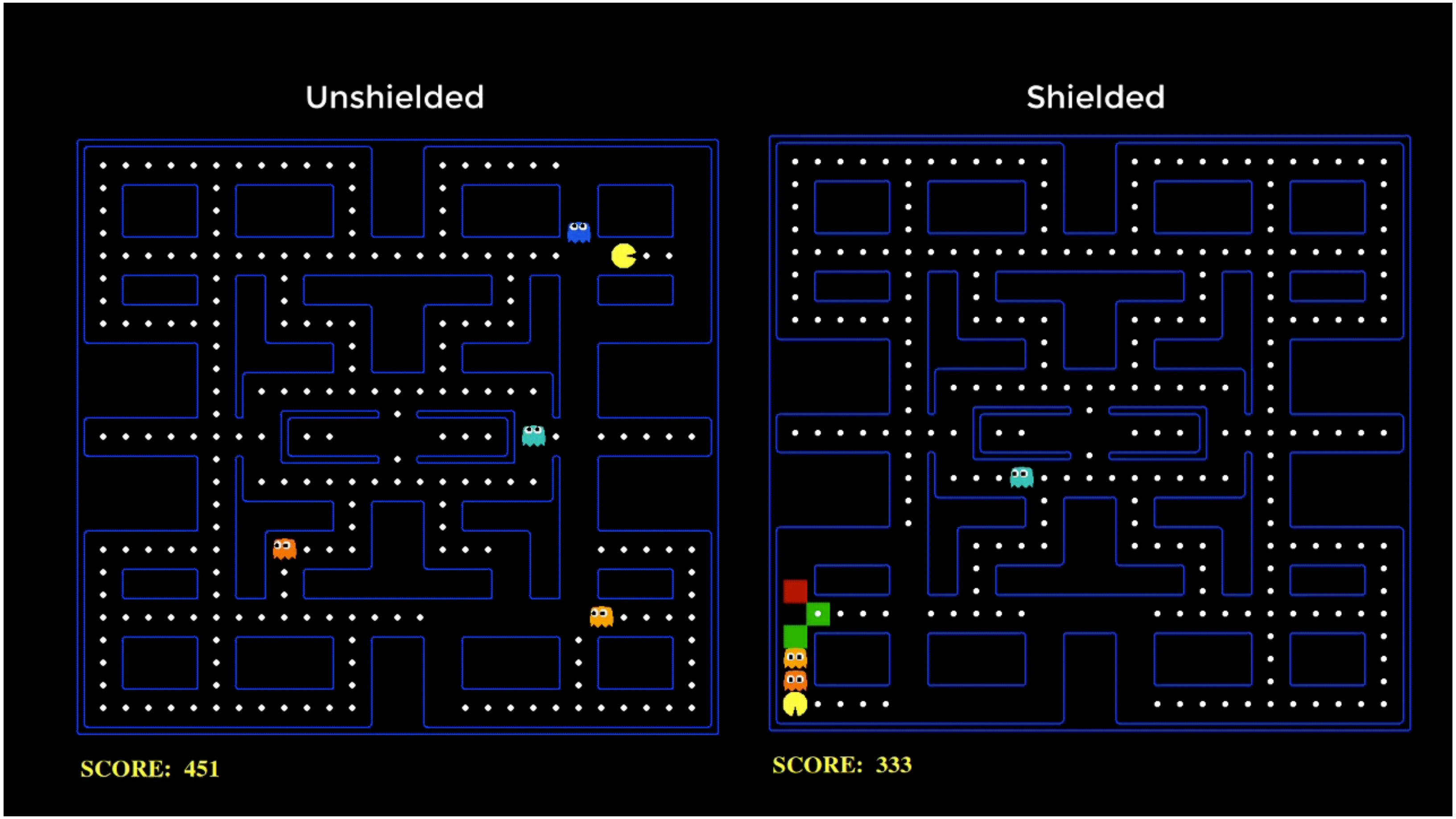
Adapted Specifications

- Capture tradeoff between safety and progress in the specifications
- Conditional probabilities

The Video!

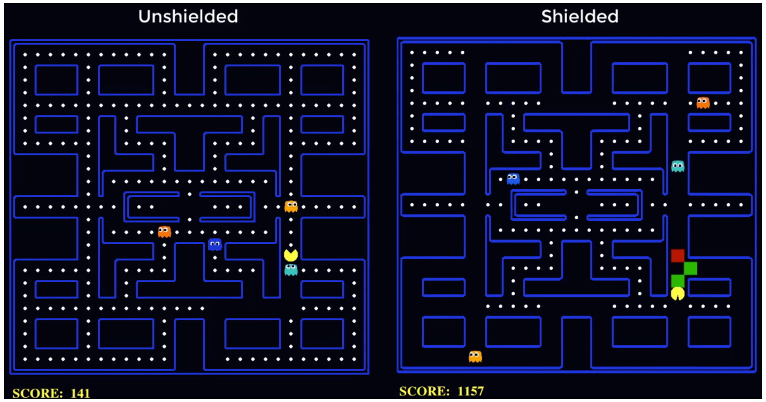


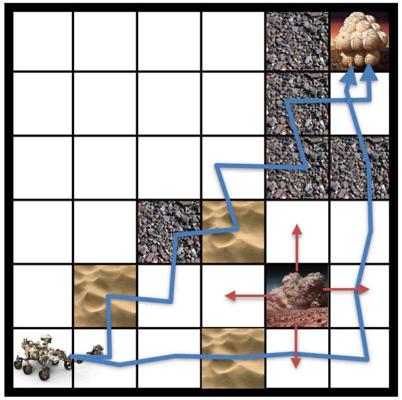
The Video!



(First) Conclusion

- Safe planning under uncertainty
- Runtime-Shield for Reinforcement Learning via Behavior Models

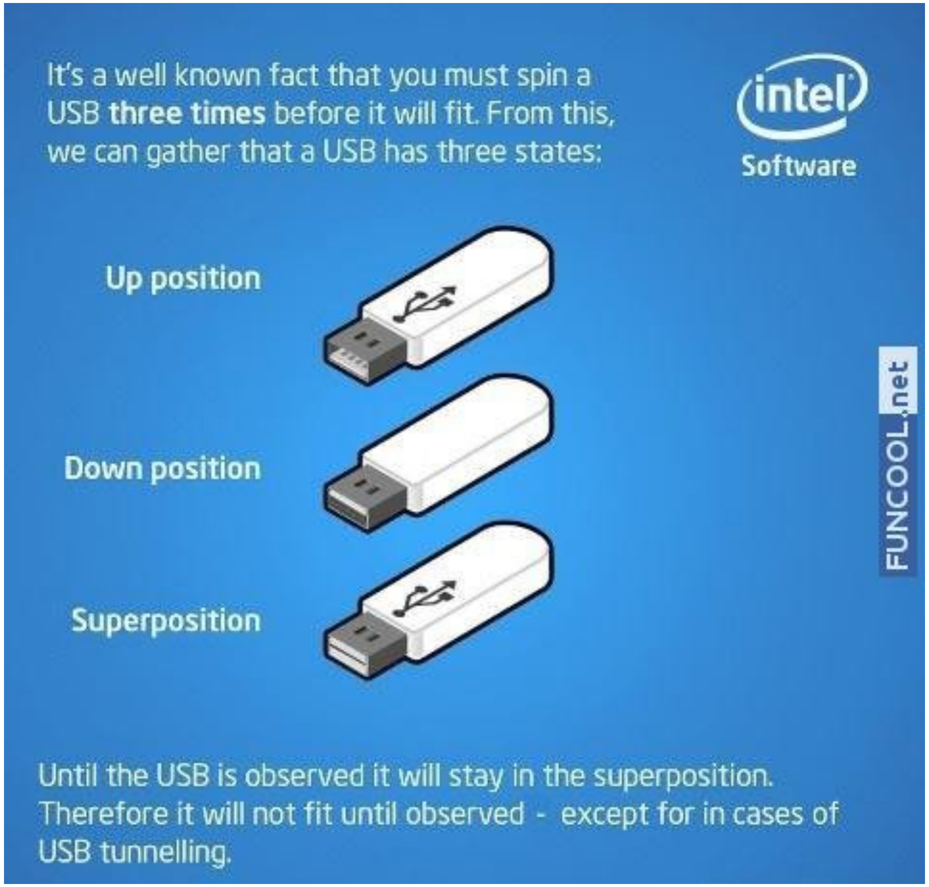




Stories

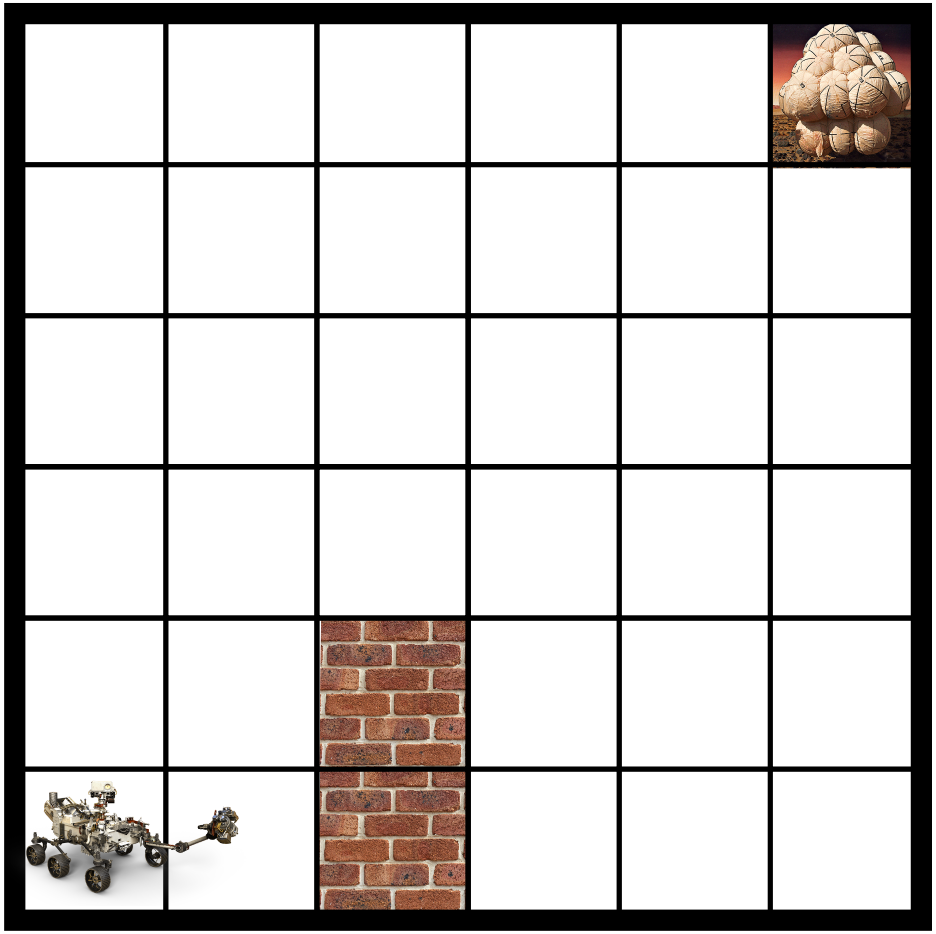
1. Safe Reinforcement Learning via Formal Verification and Behavior Models
2. Planning under Partial Observability
   I. Human-in-the-loop Planning via Gamification
   II. Planning via Recurrent Neural Networks

Partial Observability



Help the Robot with Partial Observability

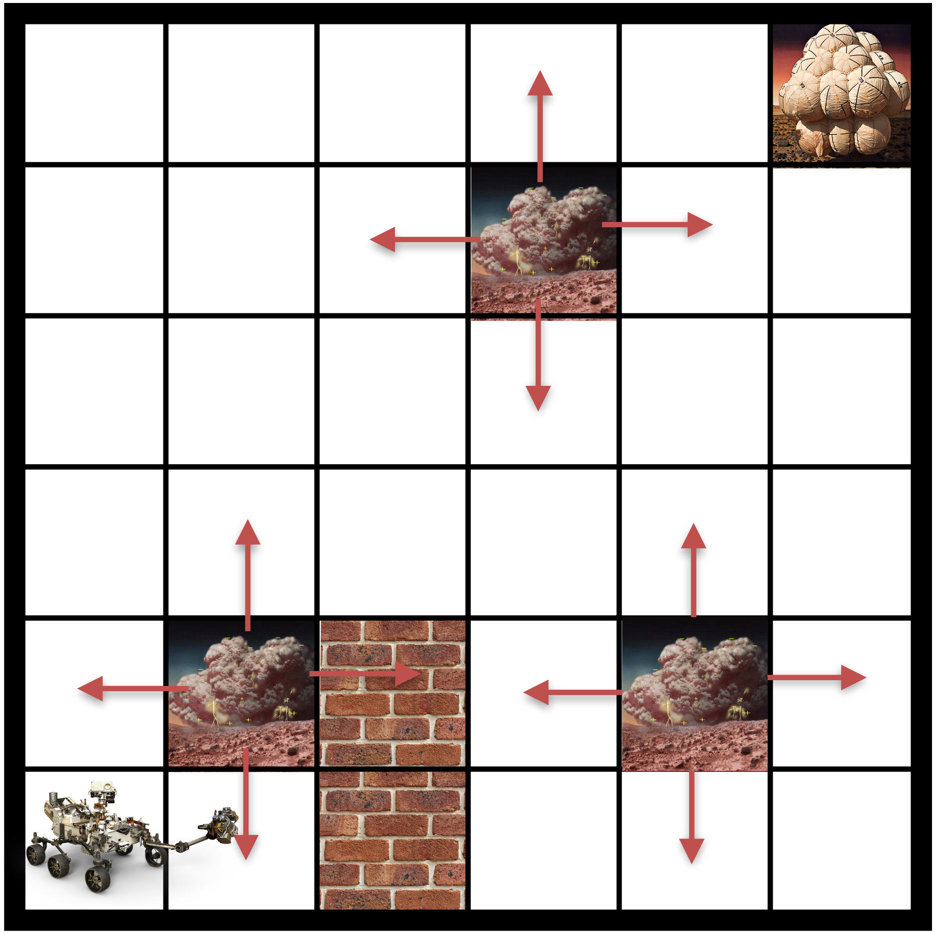
Robot has restricted range of vision



Help the Robot with Partial Observability

Robot has restricted range of vision

Storm is only **observable** when near

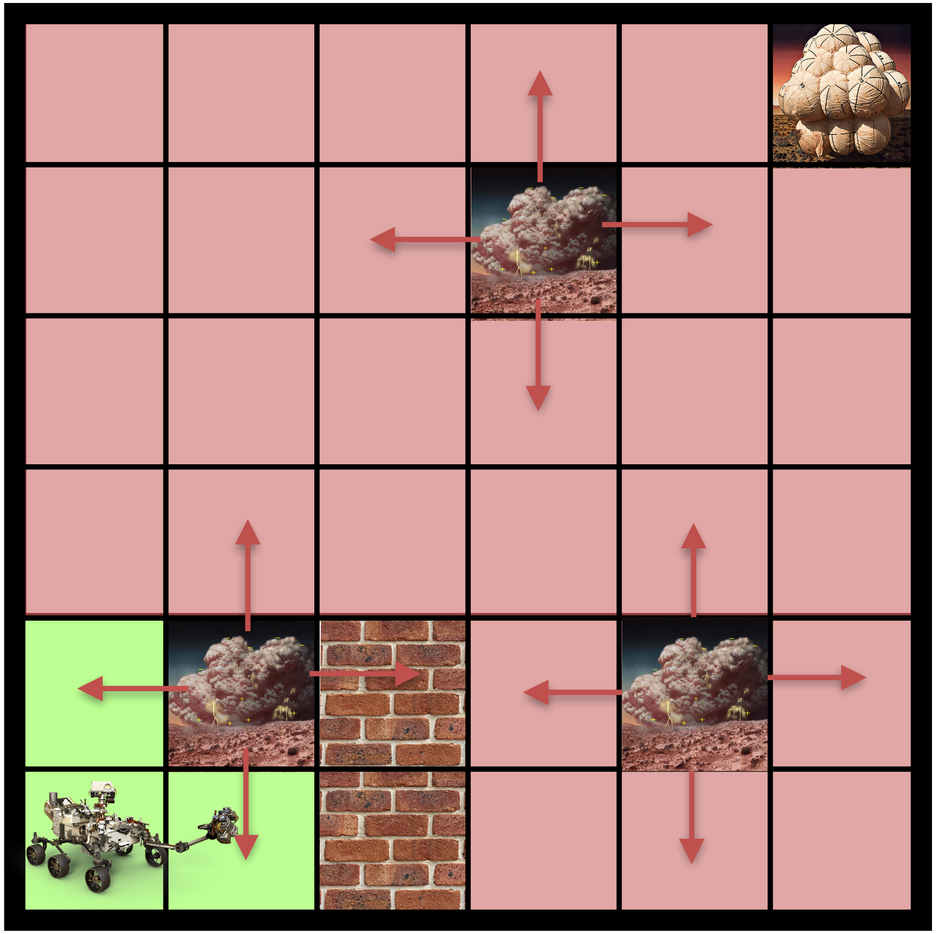


Help the Robot with Partial Observability

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

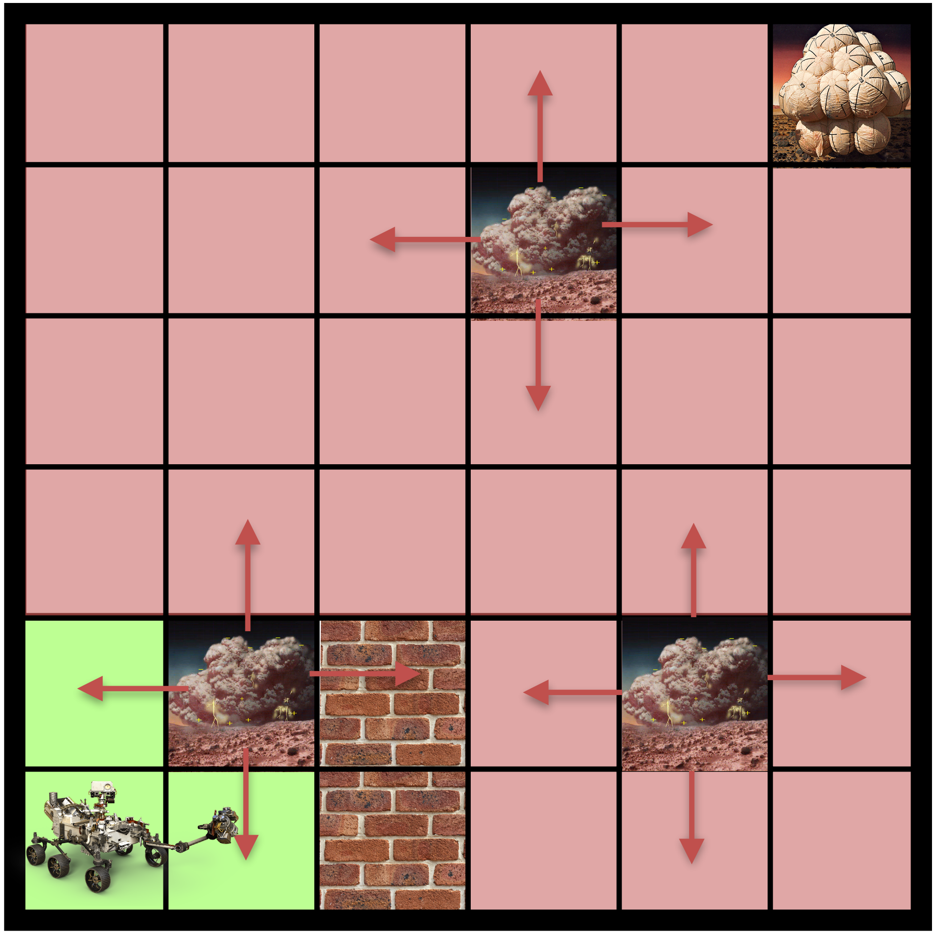
Robot has restricted range of vision

Storm is only **observable** when near

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

Find strategy that induces

$$Pr_{max}(\neg B U G)$$



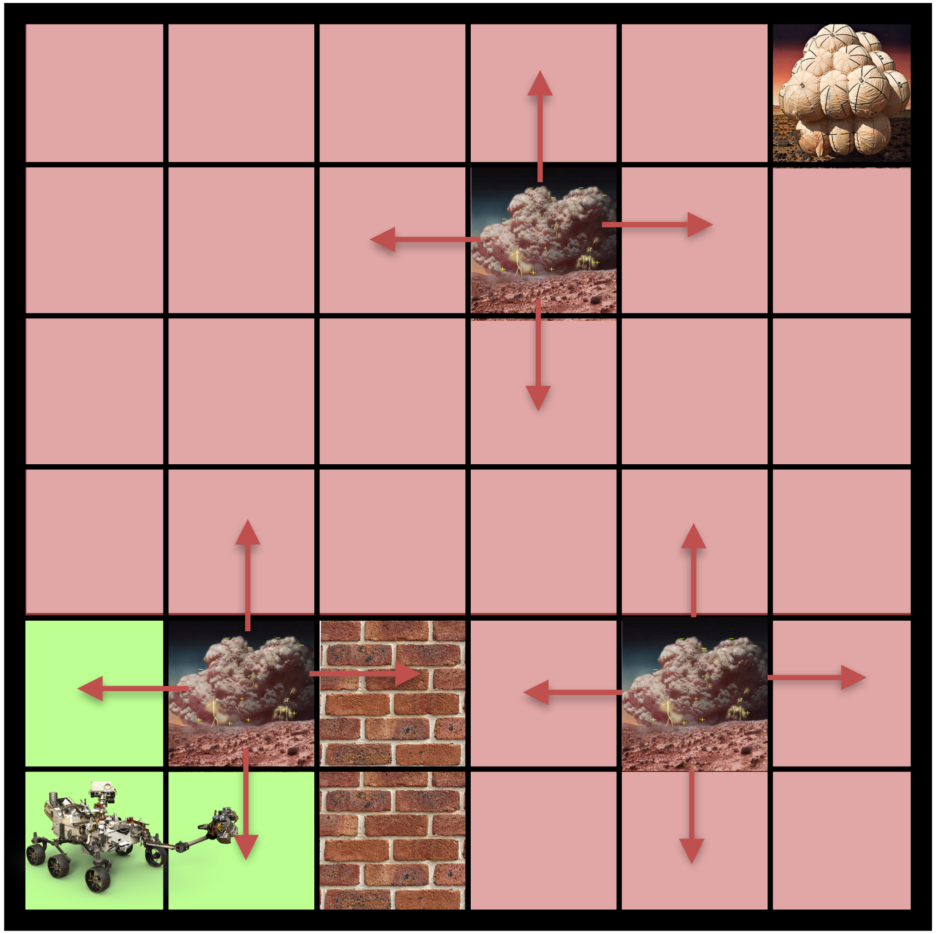
Help the Robot with Partial Observability

Robot has restricted range of vision

Storm is only **observable** when near

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

Find strategy that induces $Pr_{max}(\neg BUG)$



Find **safe and cost-optimal** strategy to get to the airbag

Help the Robot with Partial Observability

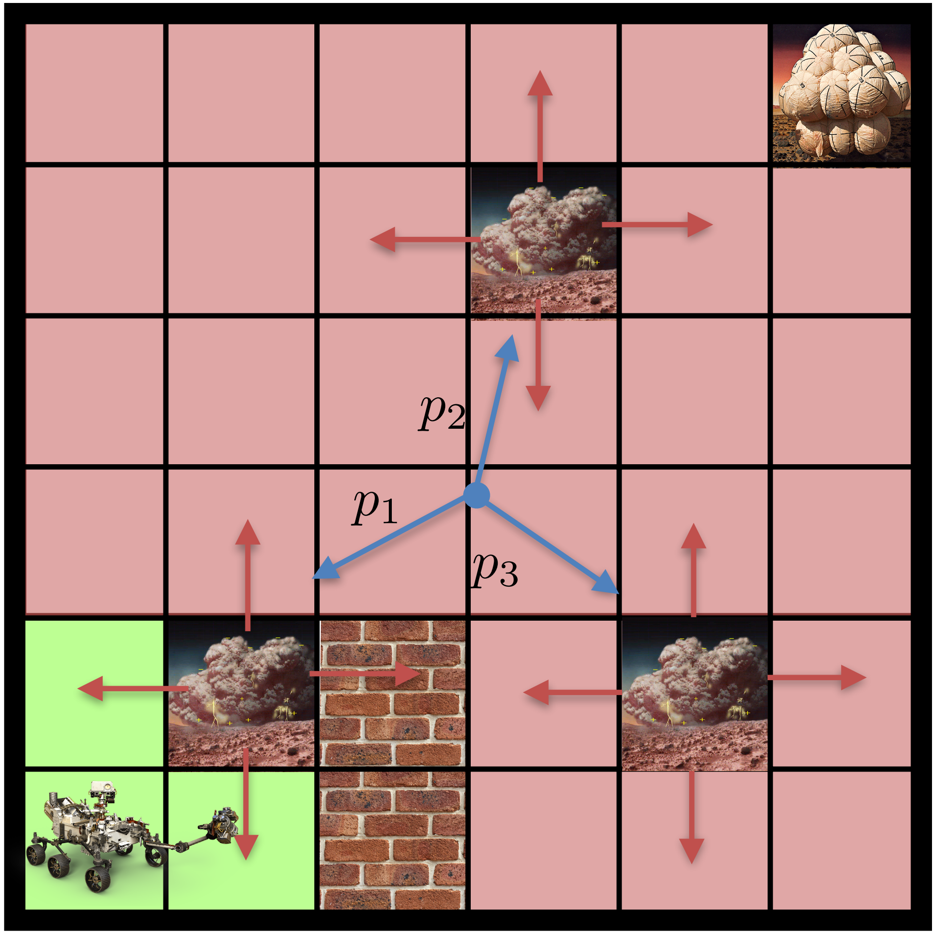
Robot has restricted range of vision

Storm is only **observable** when near

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

Find strategy that induces

$Pr_{max}(\neg BUG)$



Find **safe and cost-optimal** strategy to get to the airbag

Belief state: Likelihood of the actual position of the storm

infinite belief MDP

Help the Robot with Partial Observability

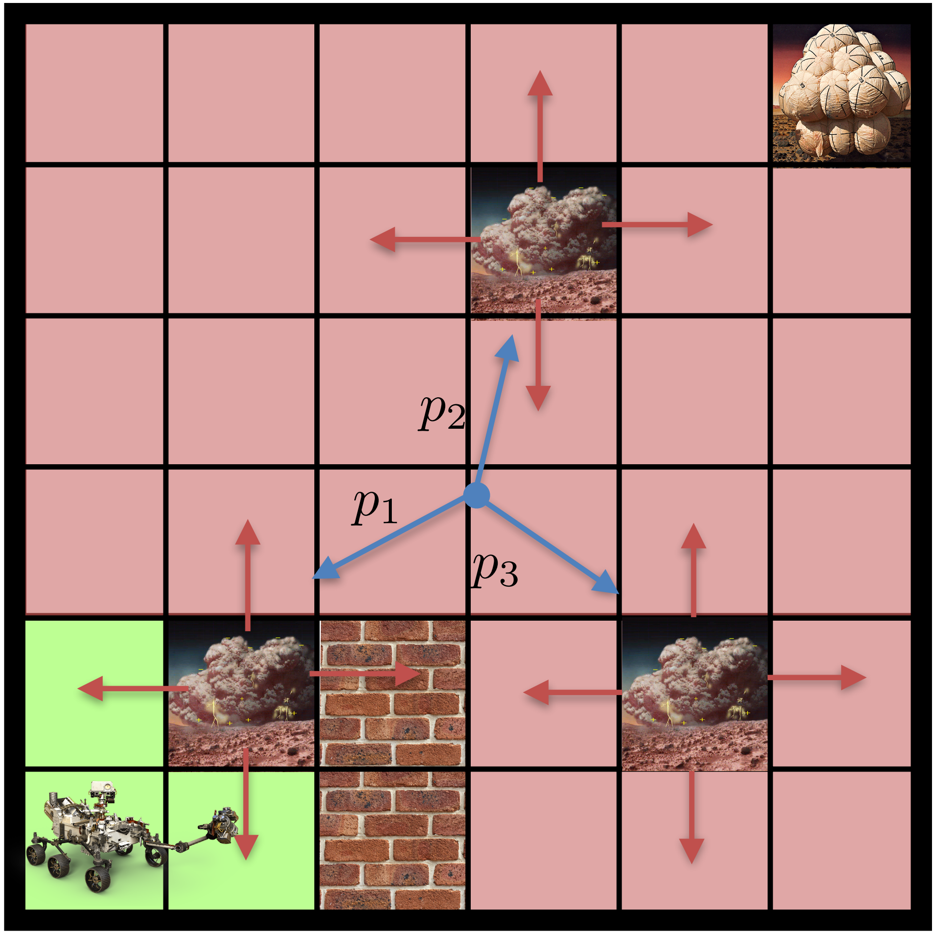
Robot has restricted range of vision

Storm is only **observable** when near

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

Find strategy that induces

$$Pr_{max}(\neg BUG)$$



Find **safe and cost-optimal** strategy to get to the airbag

Belief state: Likelihood of the actual position of the storm

infinite belief MDP

Verification undecidable

Computing Strategies for POMDPs

Computing Strategies for POMDPs

- Randomized with infinite memory: undecidable, but needed for optimal results.

Computing Strategies for POMDPs

- Randomized with infinite memory: undecidable, but needed for optimal results.

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

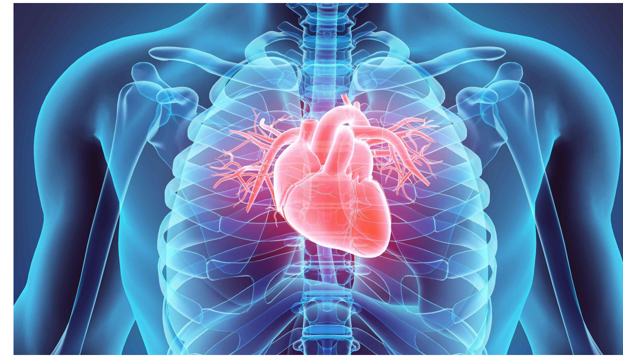
Computing Strategies for POMDPs

- Randomized with infinite memory: undecidable, but needed for 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.

POMDPs - Applications

Stock Market

Surveying Threatened Species



Health Care

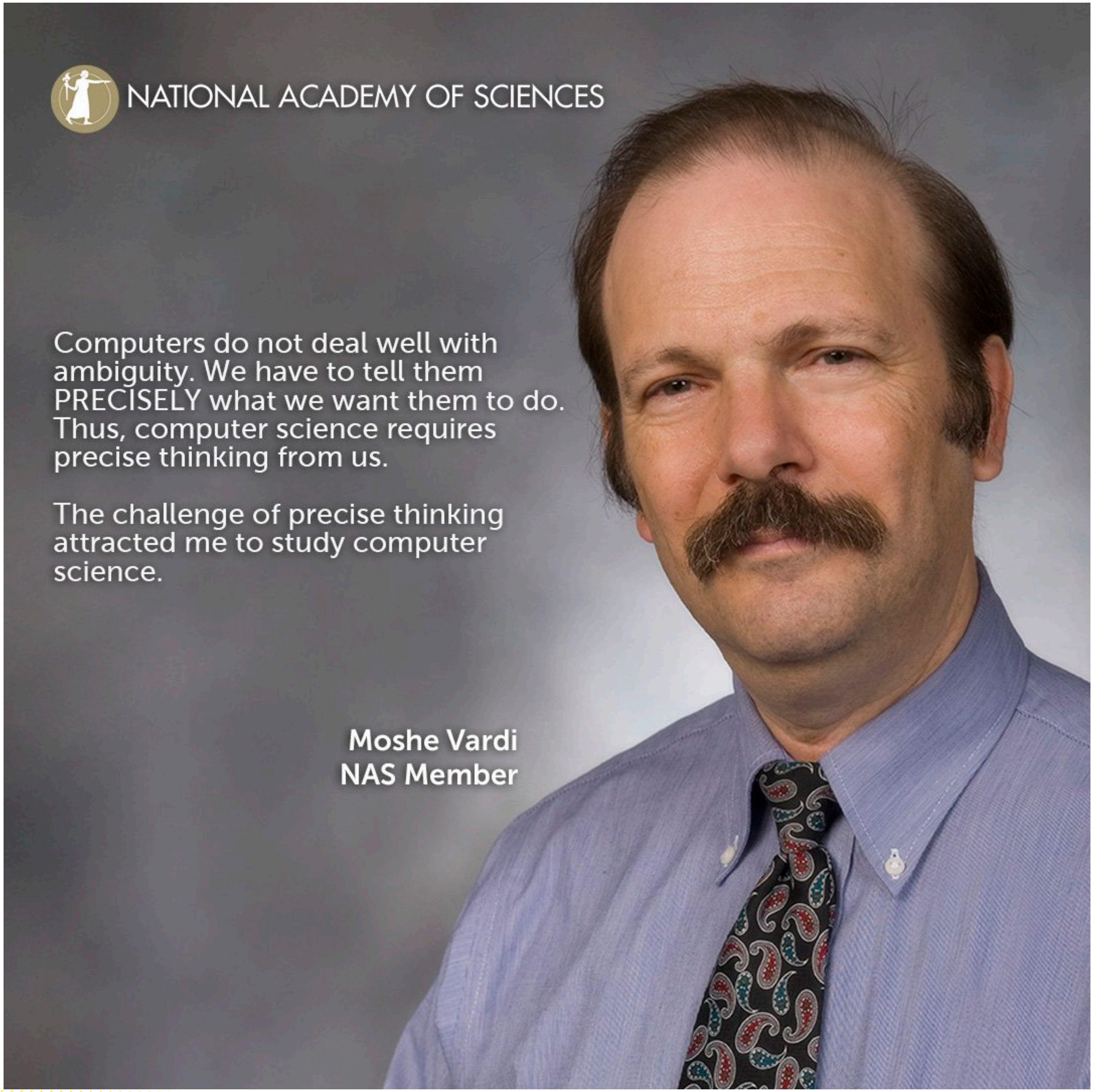
Wireless Sensor Networks

Autonomous Systems

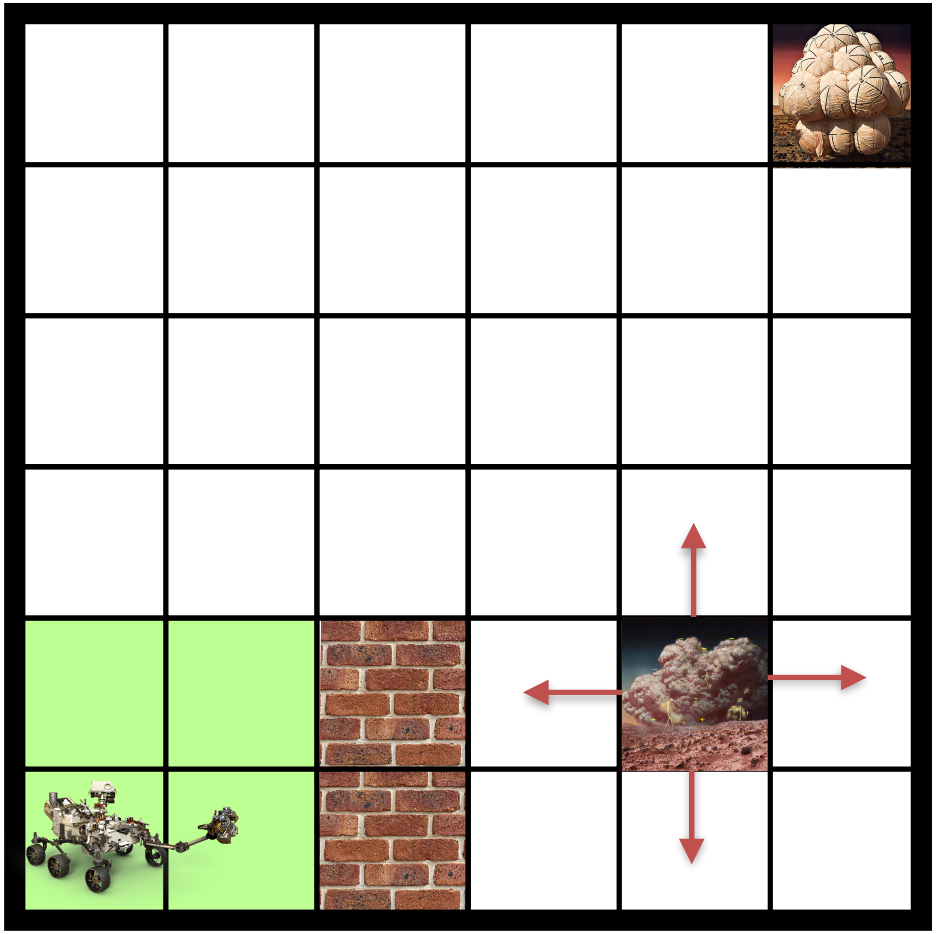
Machine Vision

Stories

1. Safe Reinforcement Learning via Formal Verification and Behavior Models
2. Planning under Partial Observability
   I. Human-in-the-loop Planning via Gamification
   II. Planning via Recurrent Neural Networks

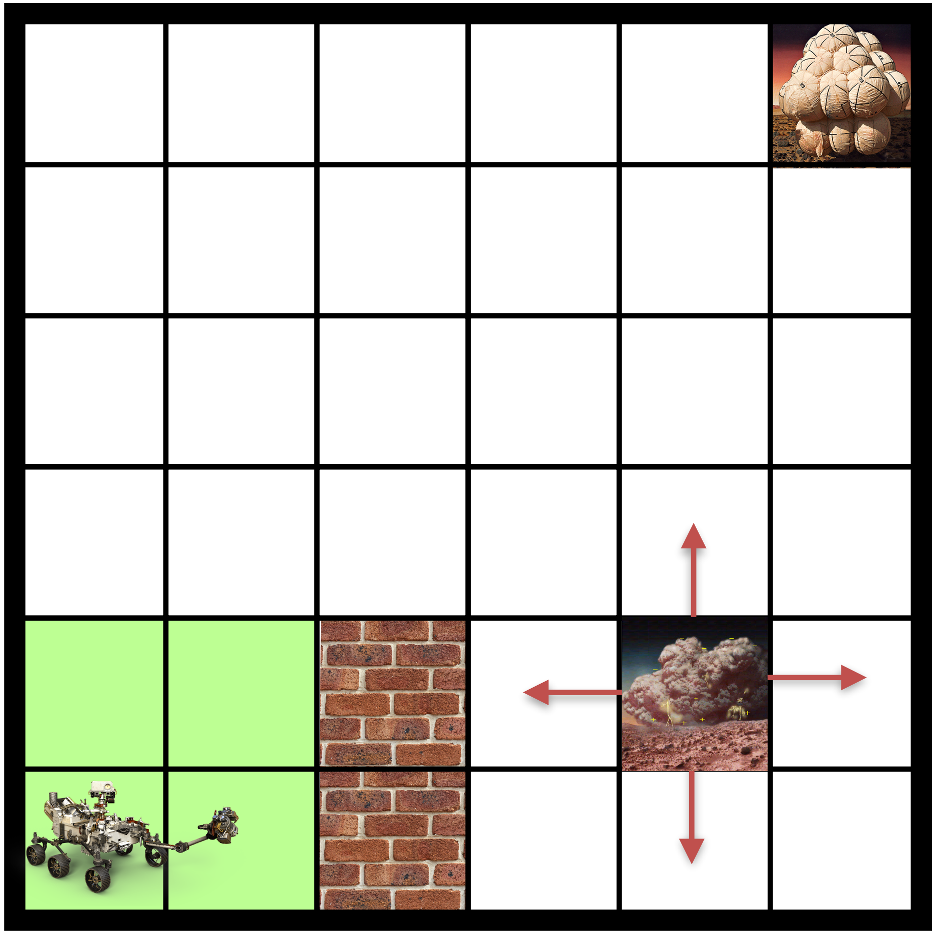

NATIONAL ACADEMY OF SCIENCES
Computers do not deal well with ambiguity. We have to tell them PRECISELY what we want them to do. Thus, computer science requires precise thinking from us.
The challenge of precise thinking attracted me to study computer science.
Moshe Vardi
NAS Member

Idea: Human-in-the-loop Synthesis for POMDPs



Idea: Human-in-the-loop Synthesis for POMDPs

Turn scenario into an arcade game

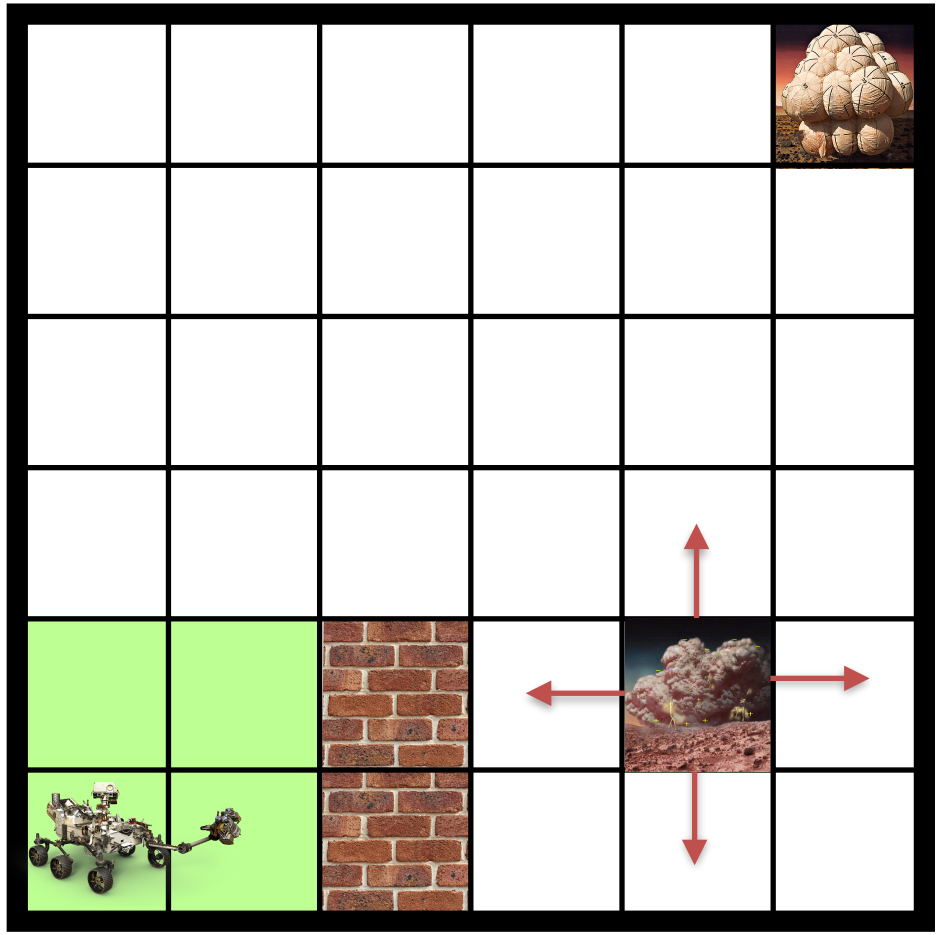


Underlying (family of) POMDPs

Idea: Human-in-the-loop Synthesis for POMDPs

Turn scenario into an arcade game

Collect data of human playing



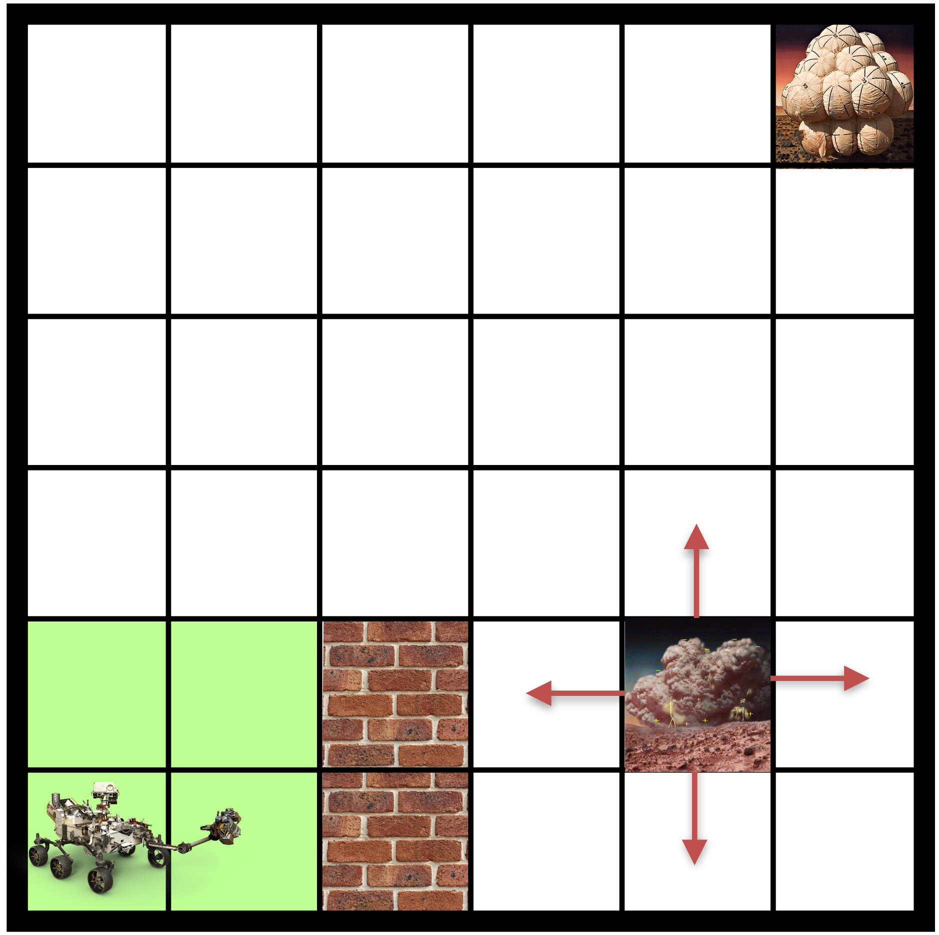
Underlying (family of) POMDPs

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 restricted model, efficient verification

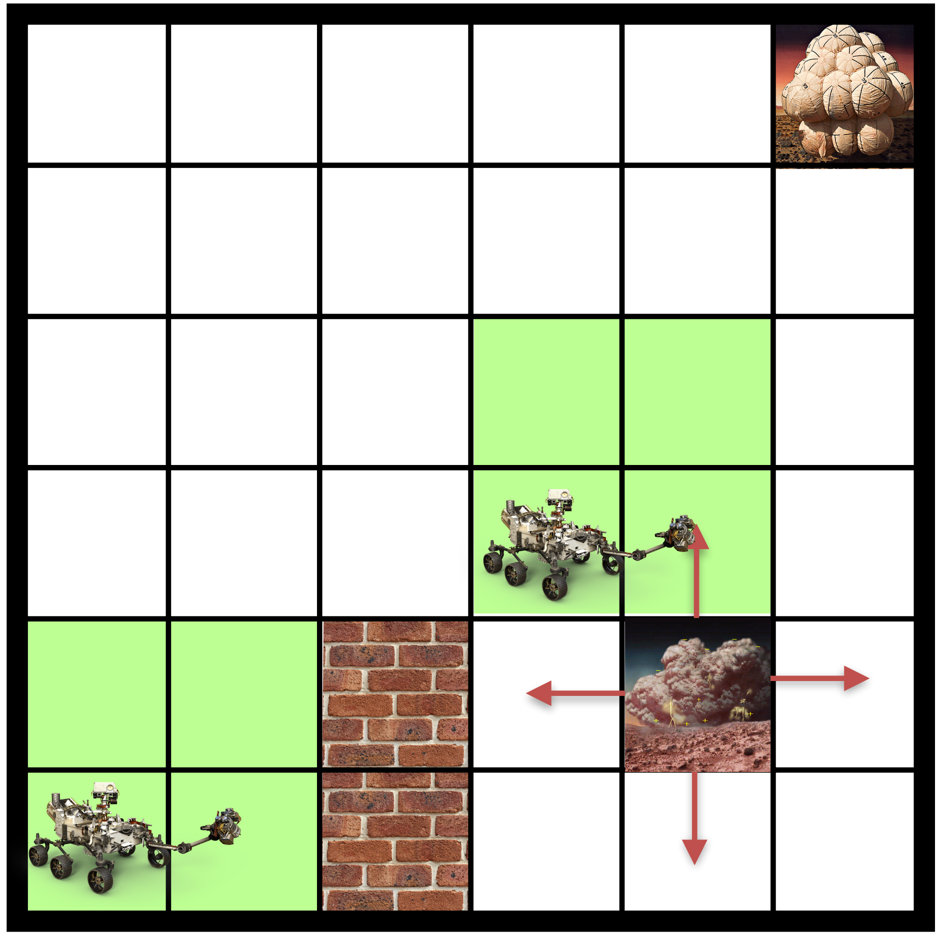
Idea: Human-in-the-loop Synthesis for POMDPs

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

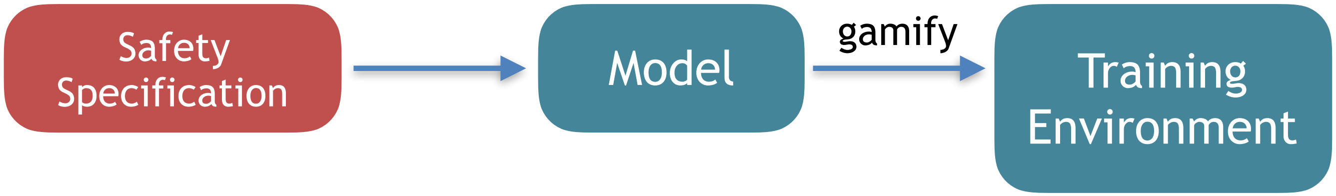
Applying strategy yields restricted model, efficient verification

Counterexamples point to critical parts

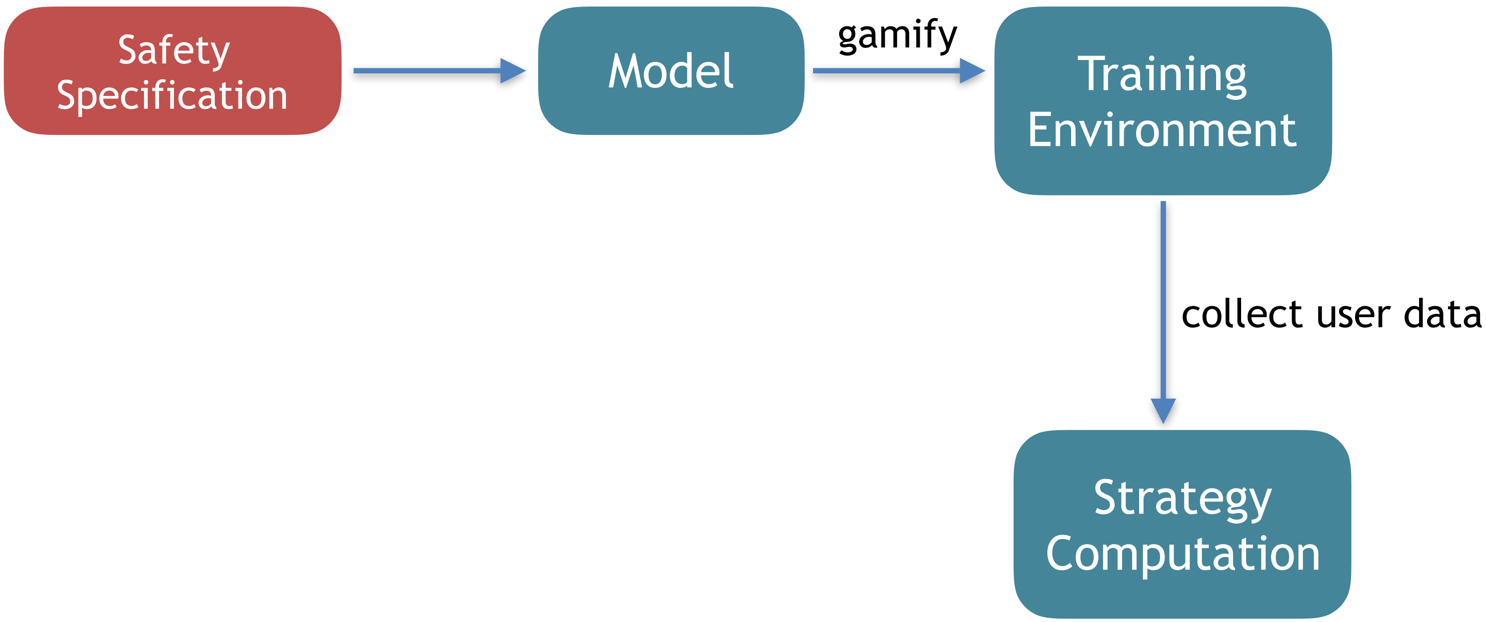
Story: HiL Synthesis for POMDPs

Safety Specification → Model

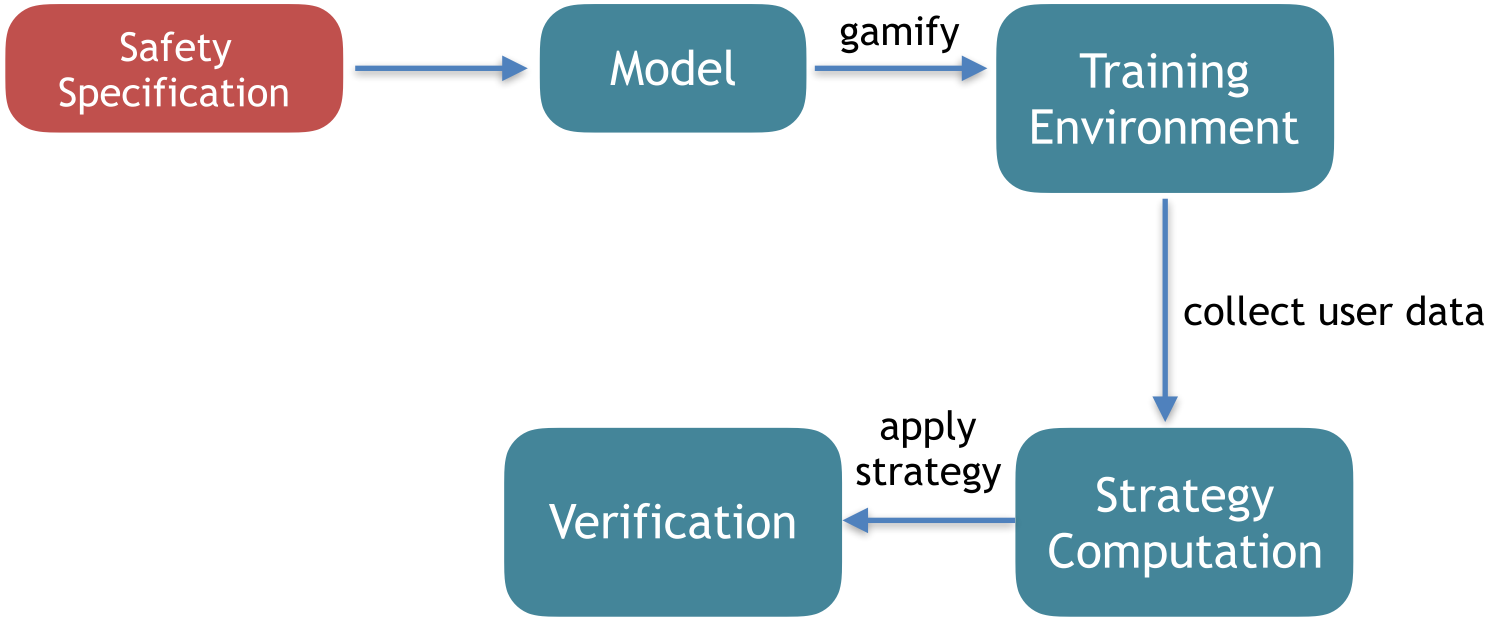
Story: HiL Synthesis for POMDPs



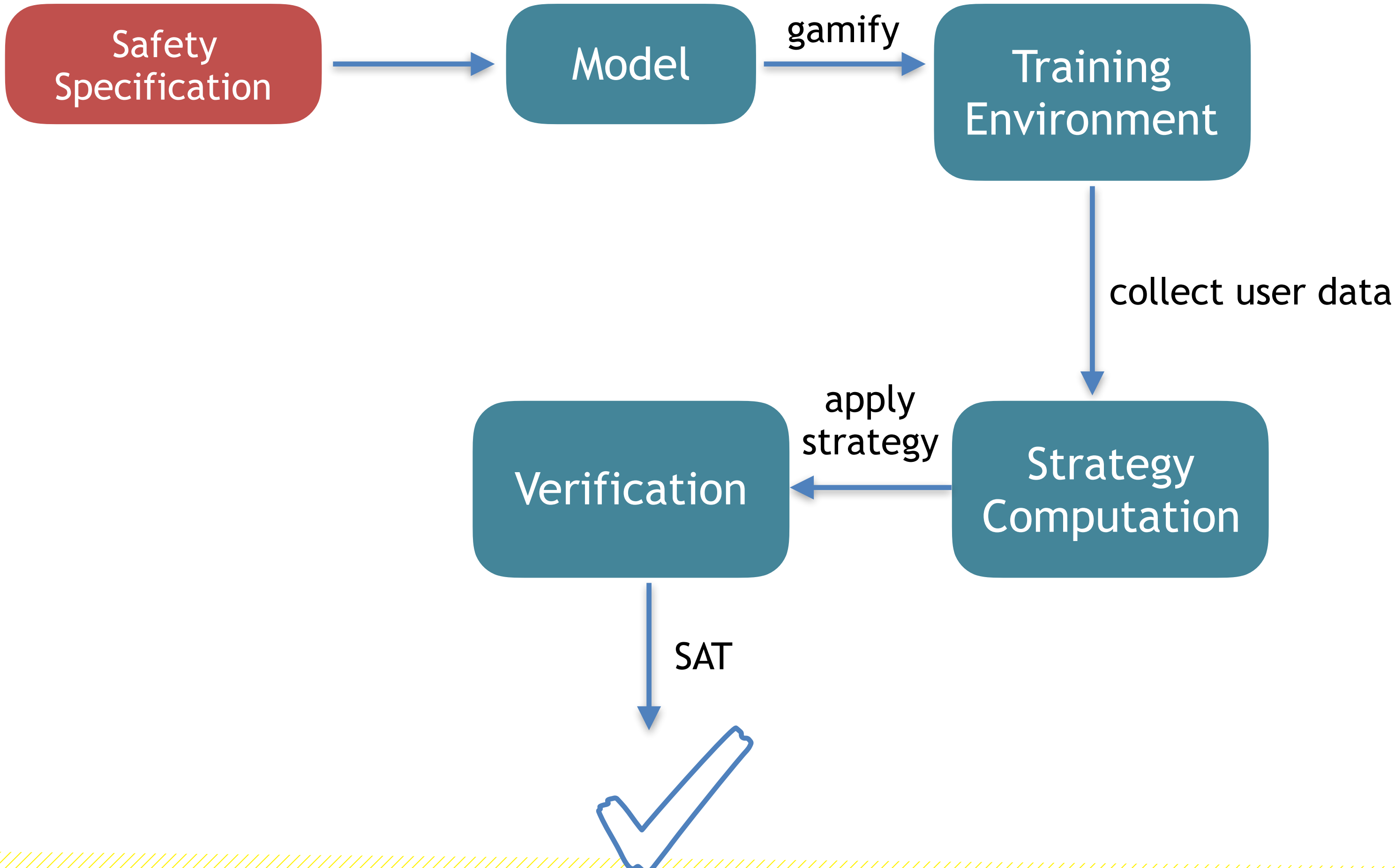
Story: HiL Synthesis for POMDPs



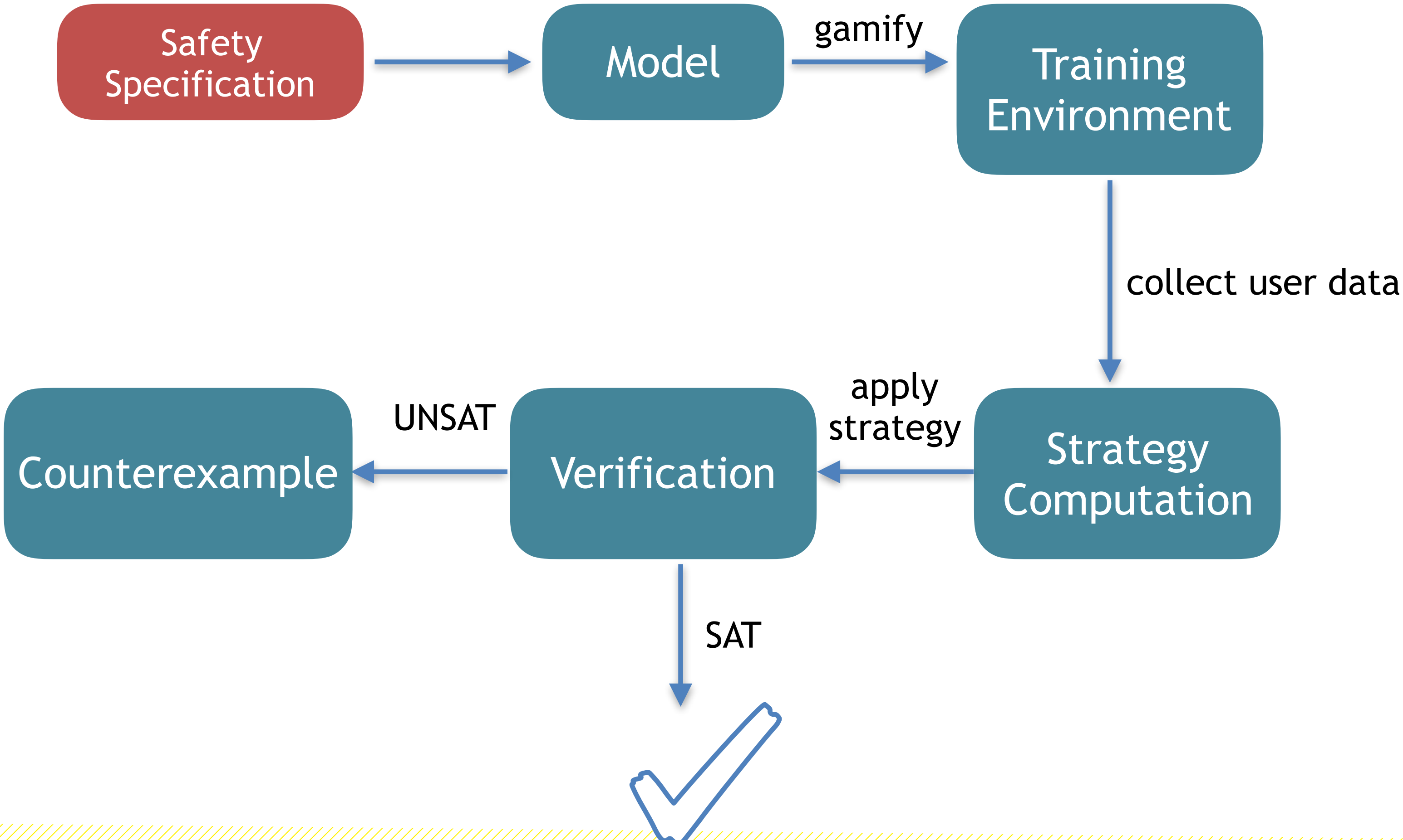
Story: HiL Synthesis for POMDPs



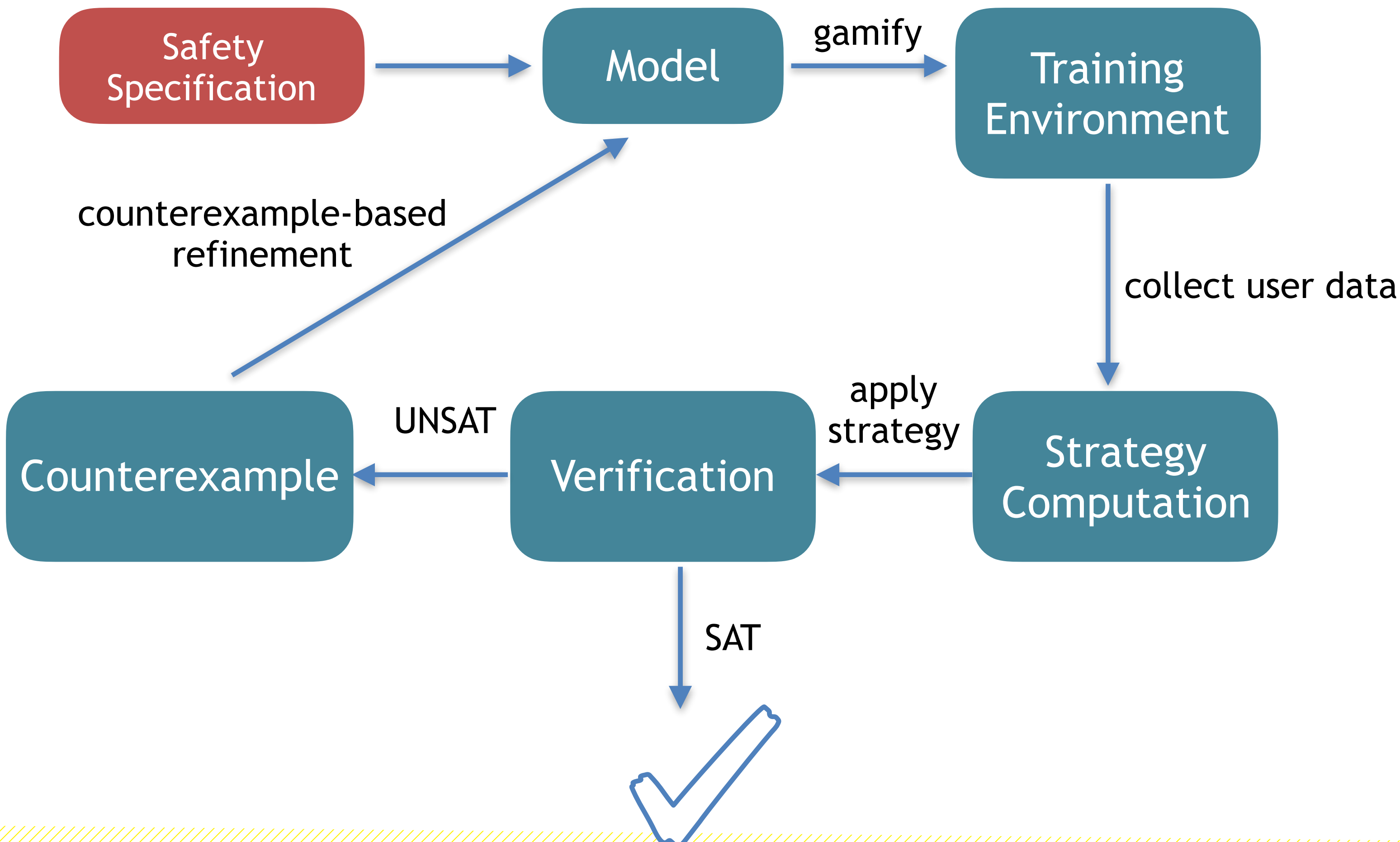
Story: HiL Synthesis for POMDPs



Story: HiL Synthesis for POMDPs

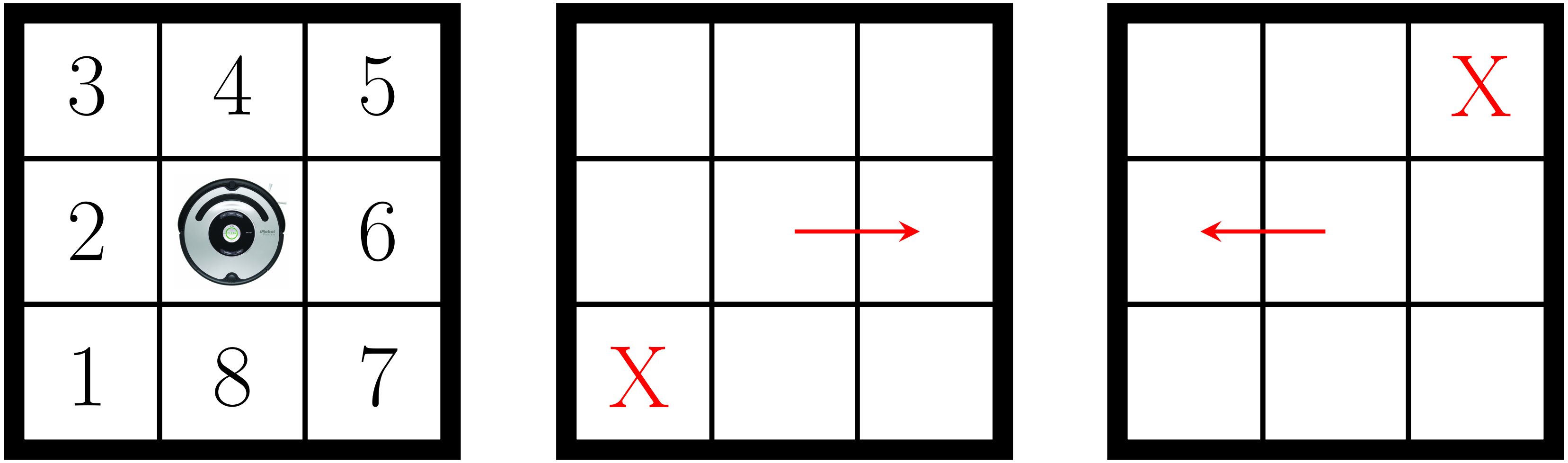


Story: HiL Synthesis for POMDPs



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

Stories

1. Safe Reinforcement Learning via Formal Verification and Behavior Models
2. Planning under Partial Observability
    I. Human-in-the-loop Planning via Gamification
    II. Planning via Recurrent Neural Networks

Guess a Strategy and Verify!

Model

$\mathcal{M}$

Specification

φ

Guess a Strategy and Verify!

Model

$\mathscr{M}$

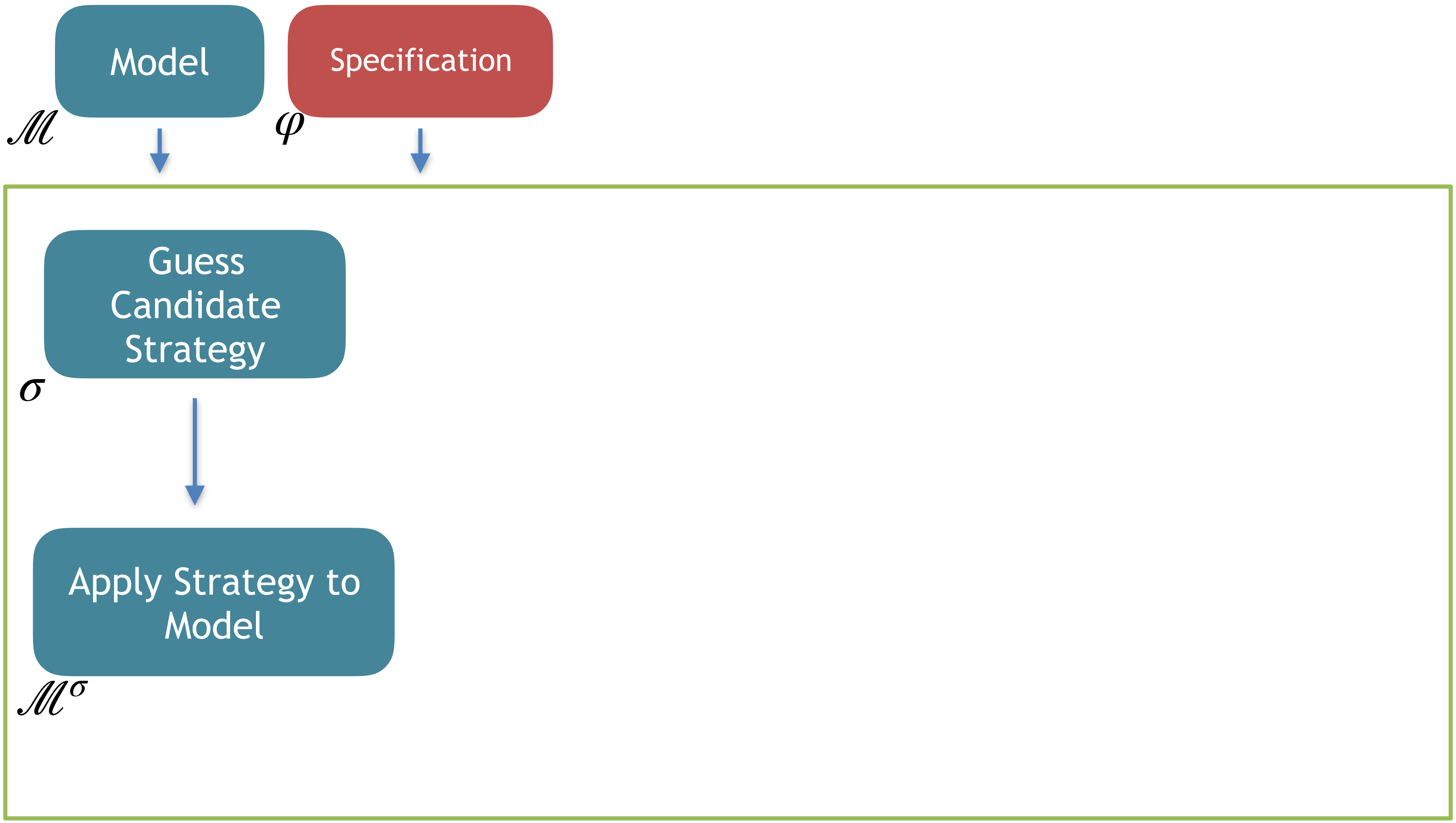
Specification

φ

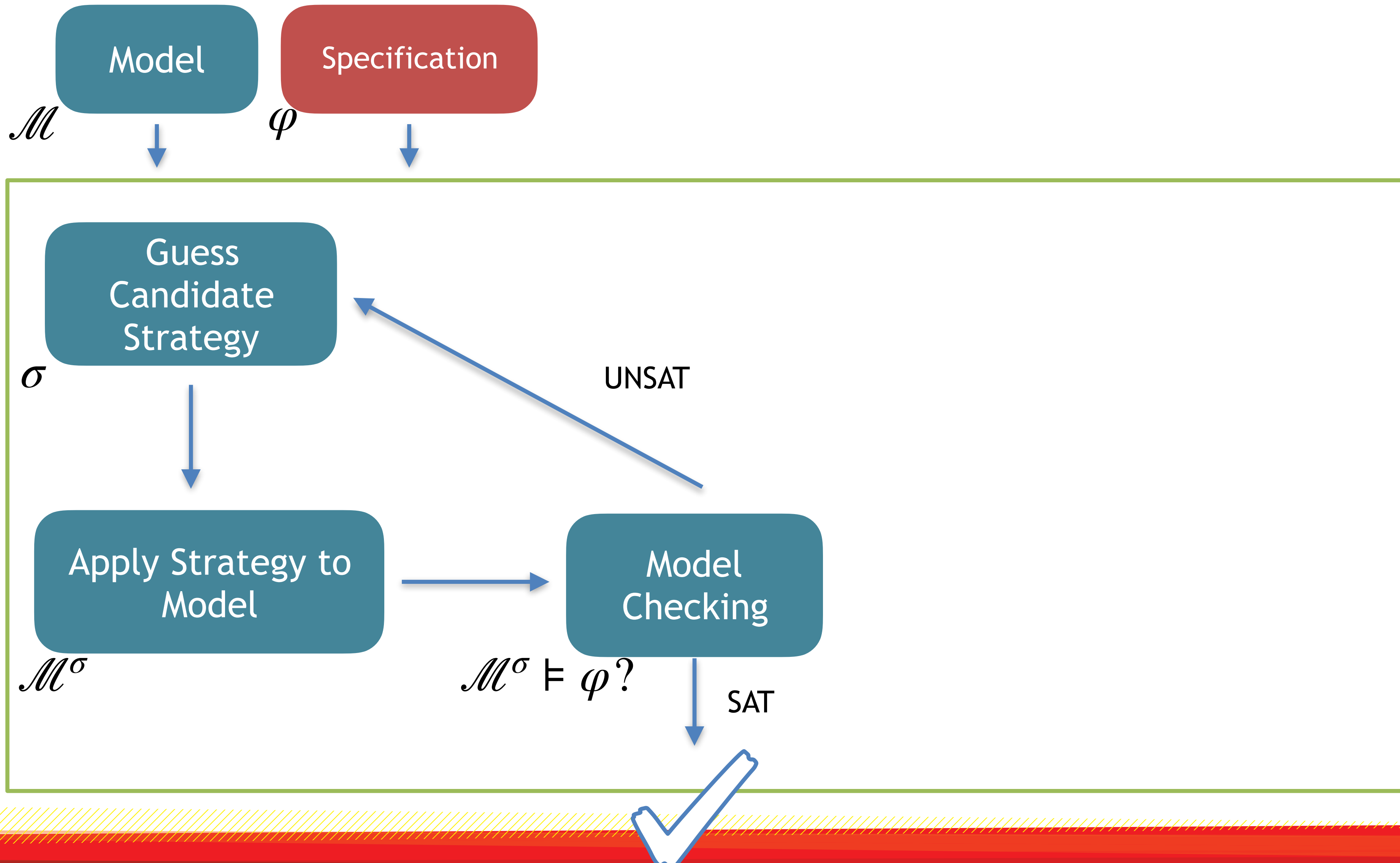
Guess Candidate Strategy

σ

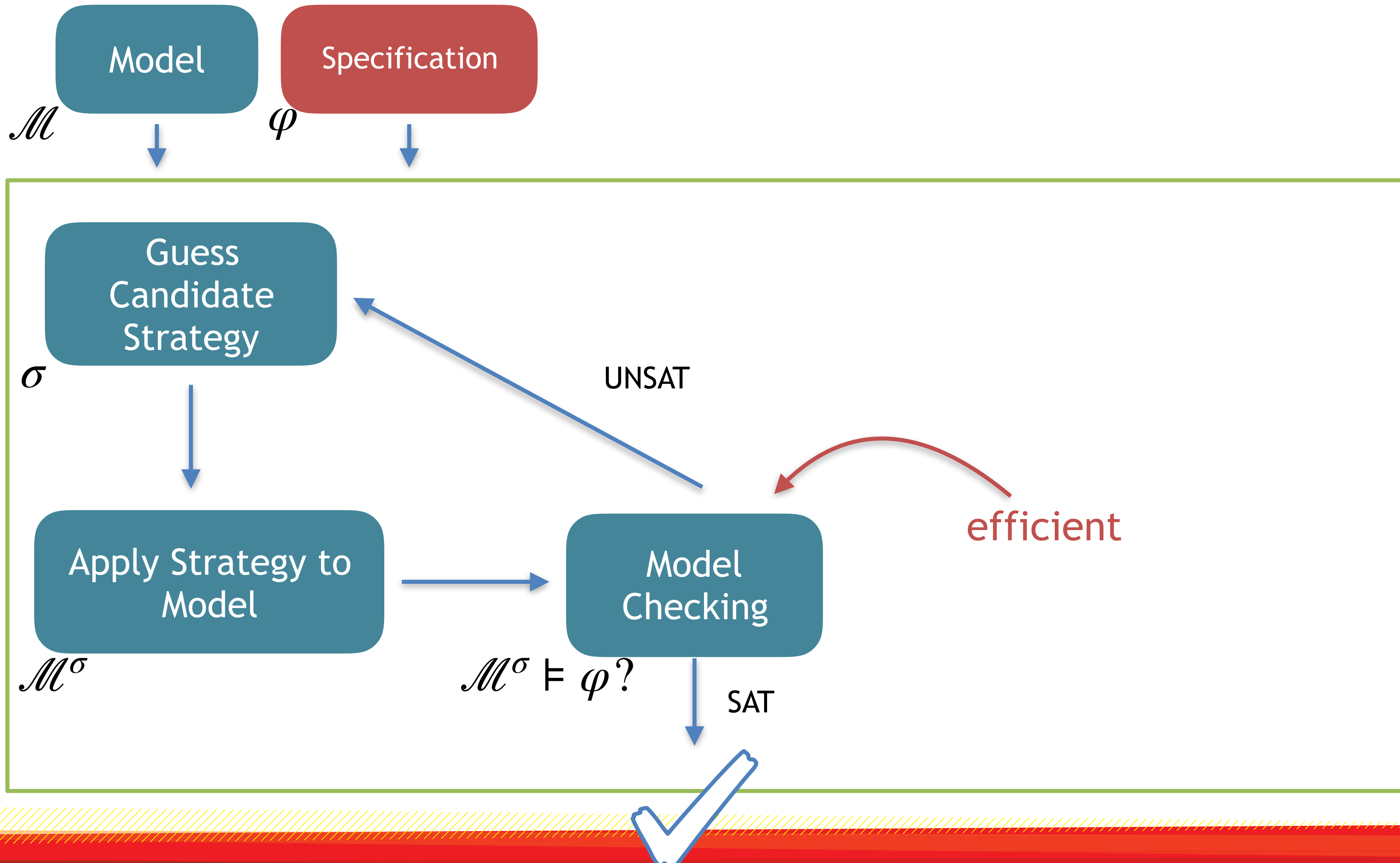
Guess a Strategy and Verify!



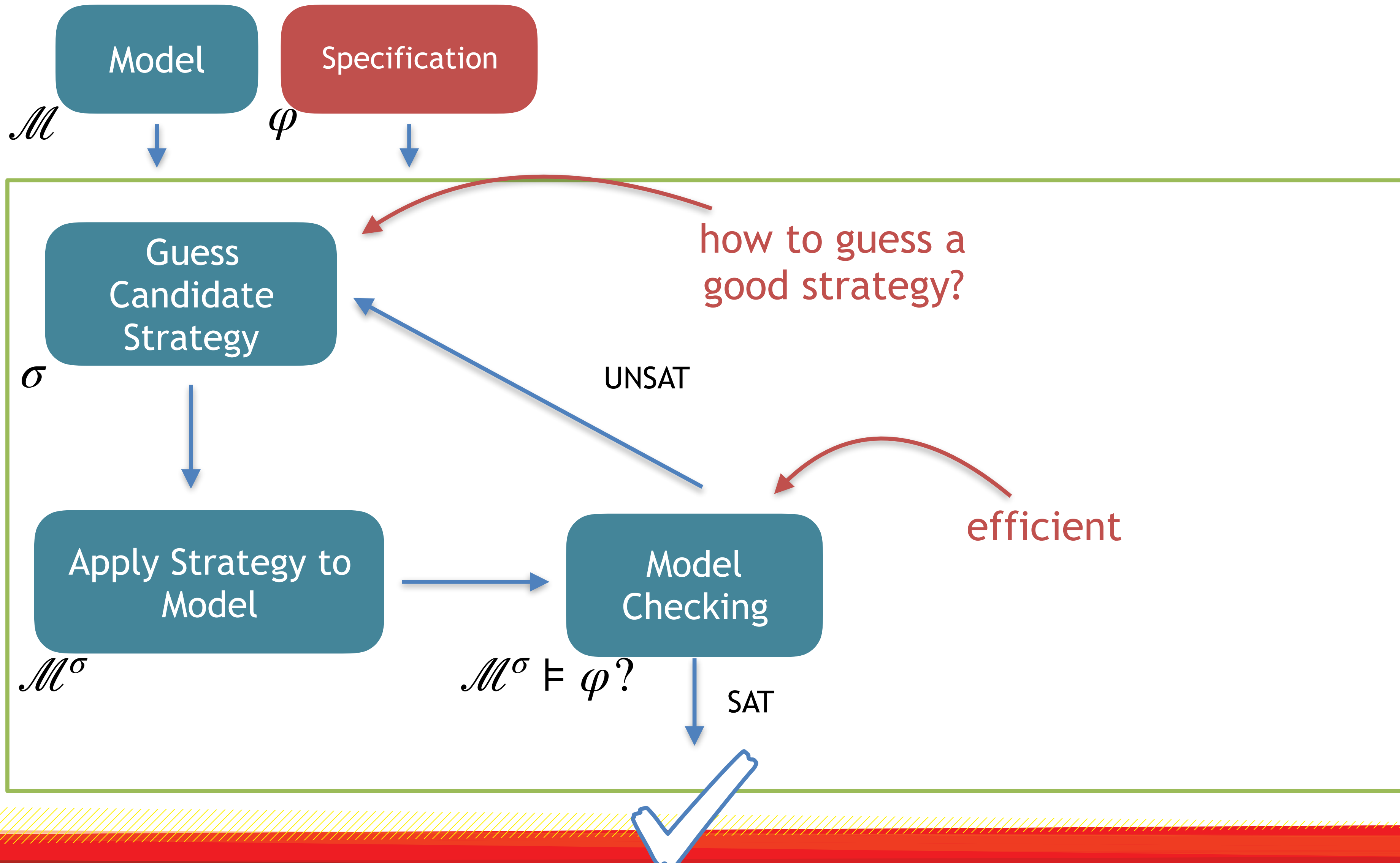
Guess a Strategy and Verify!



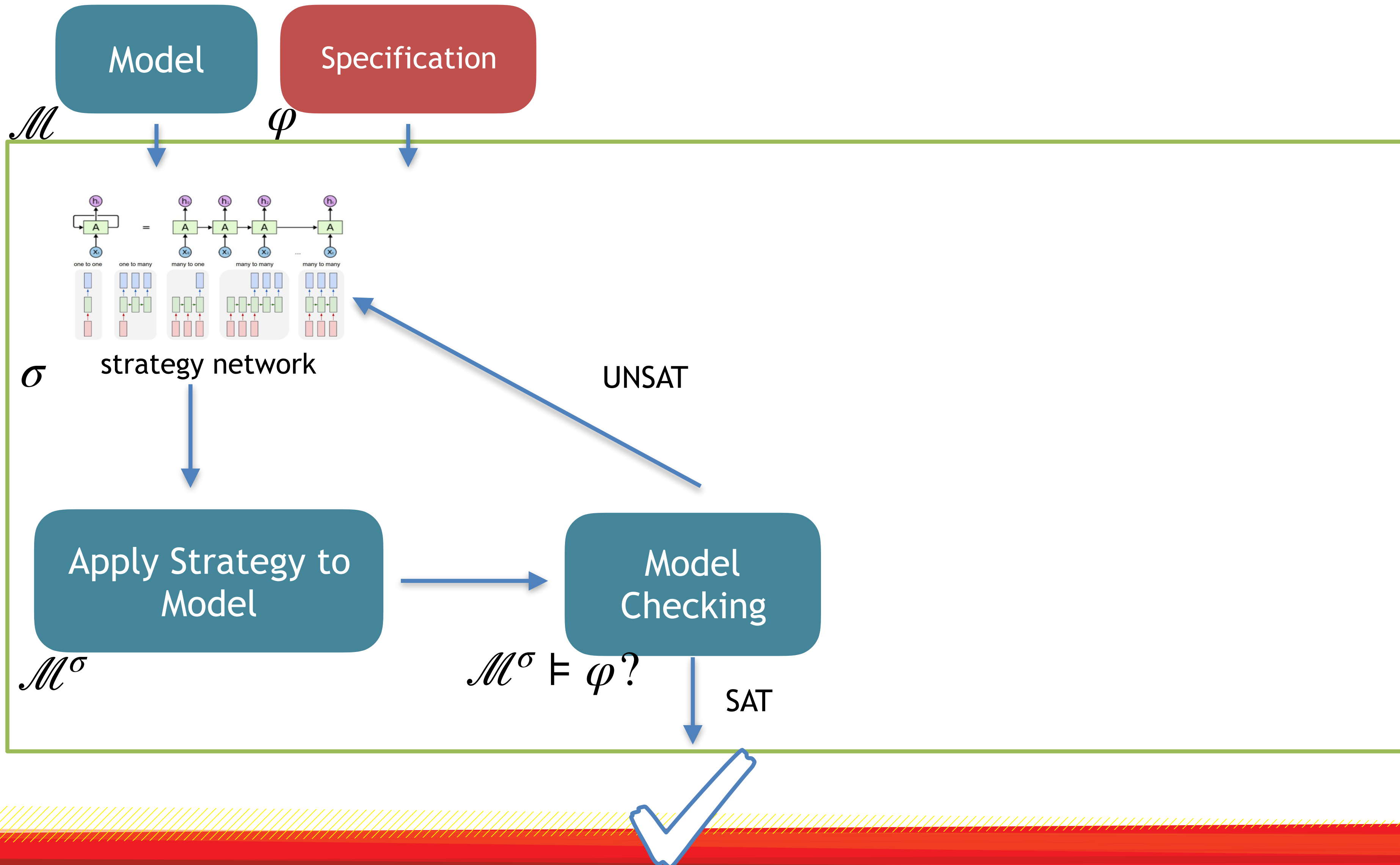
Guess a Strategy and Verify!



Guess a Strategy and Verify!



Let Machine Learning do the Guessing?



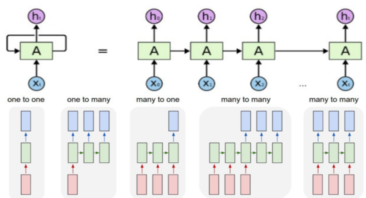
Let Machine Learning do the Guessing?

Model

Specification

$\mathscr{M}$

φ



σ strategy network

how to employ a neural network?

UNSAT

Apply Strategy to Model

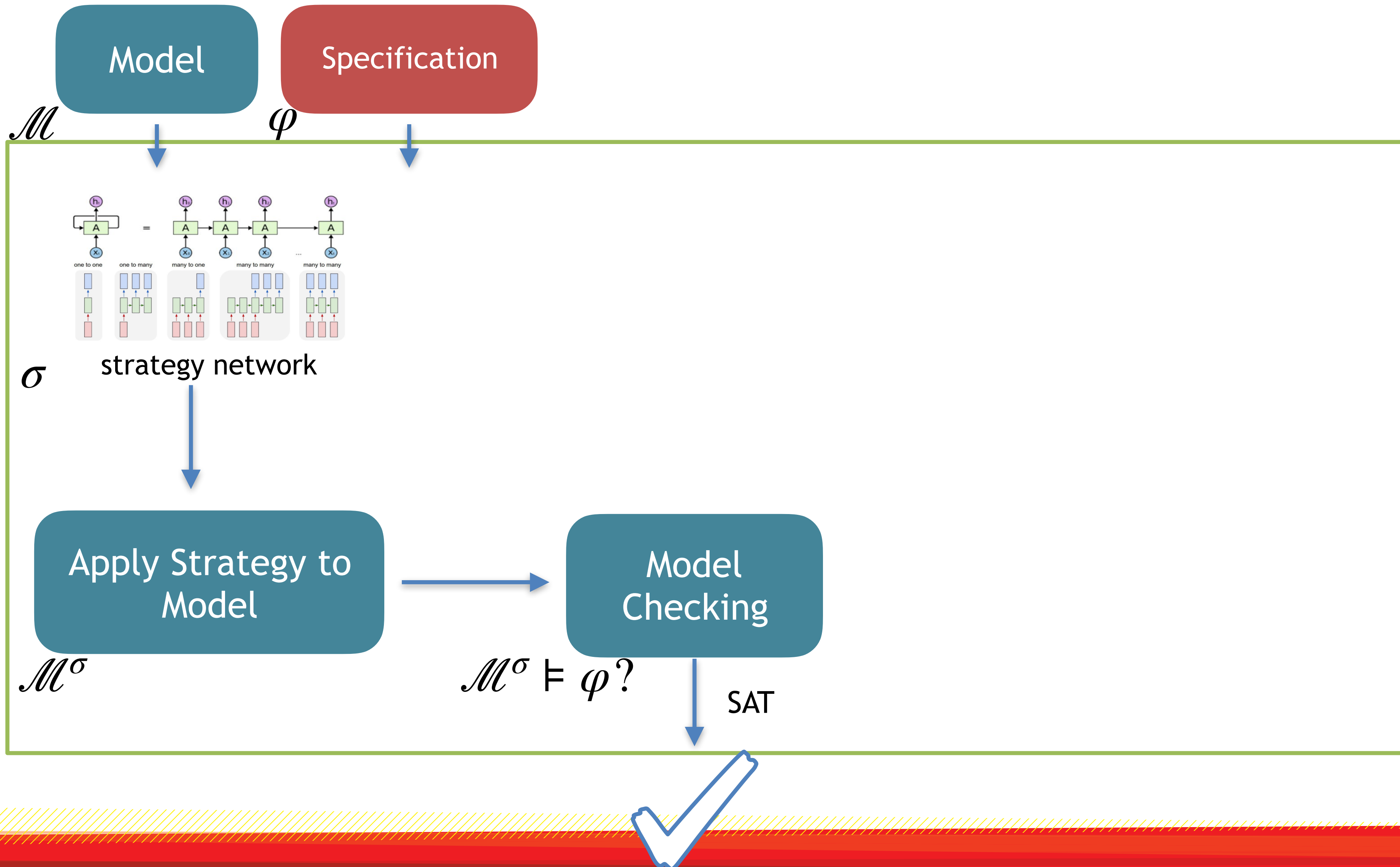
Model Checking

$\mathscr{M}^\sigma$

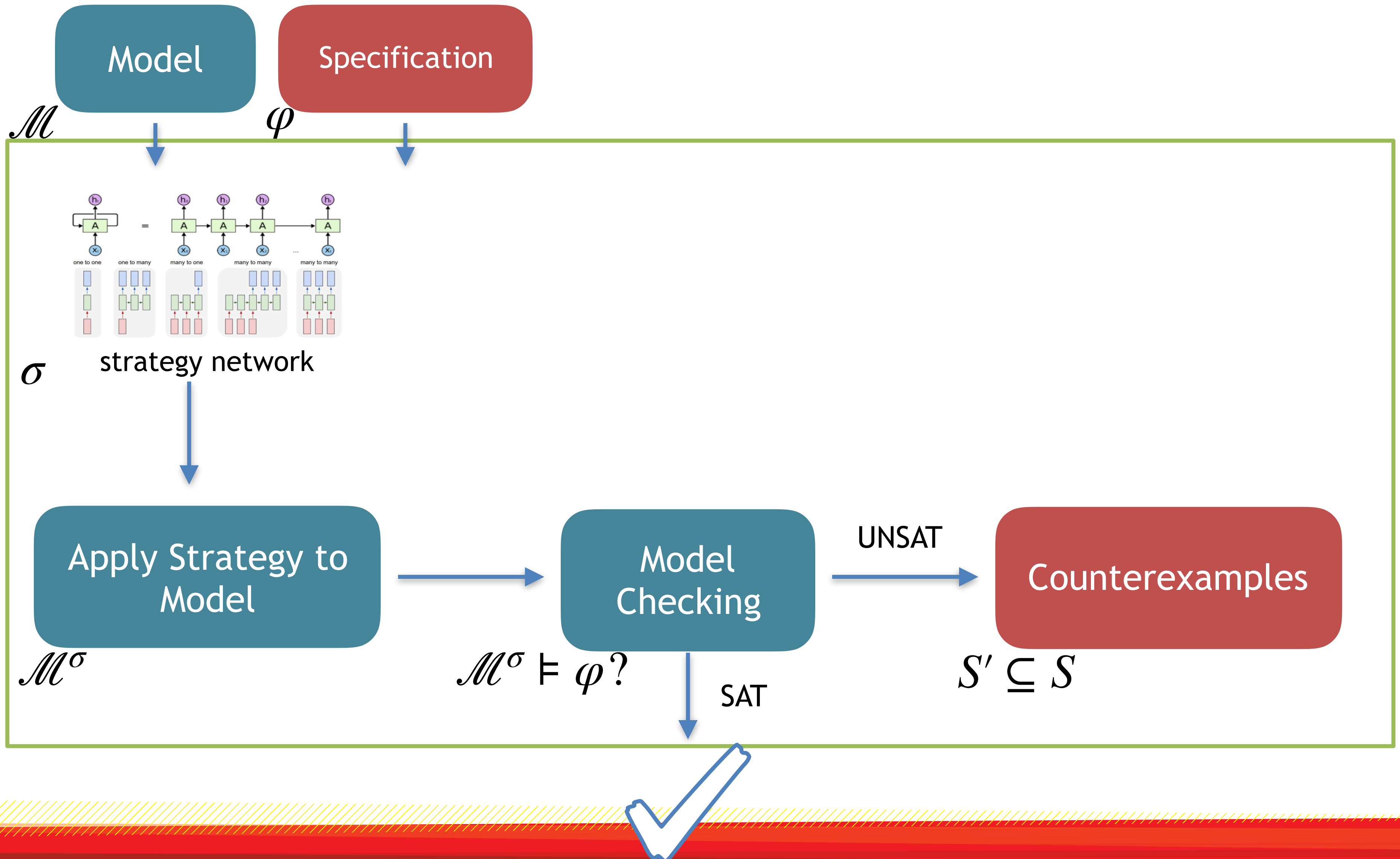
$\mathscr{M}^\sigma \models \varphi$?

SAT

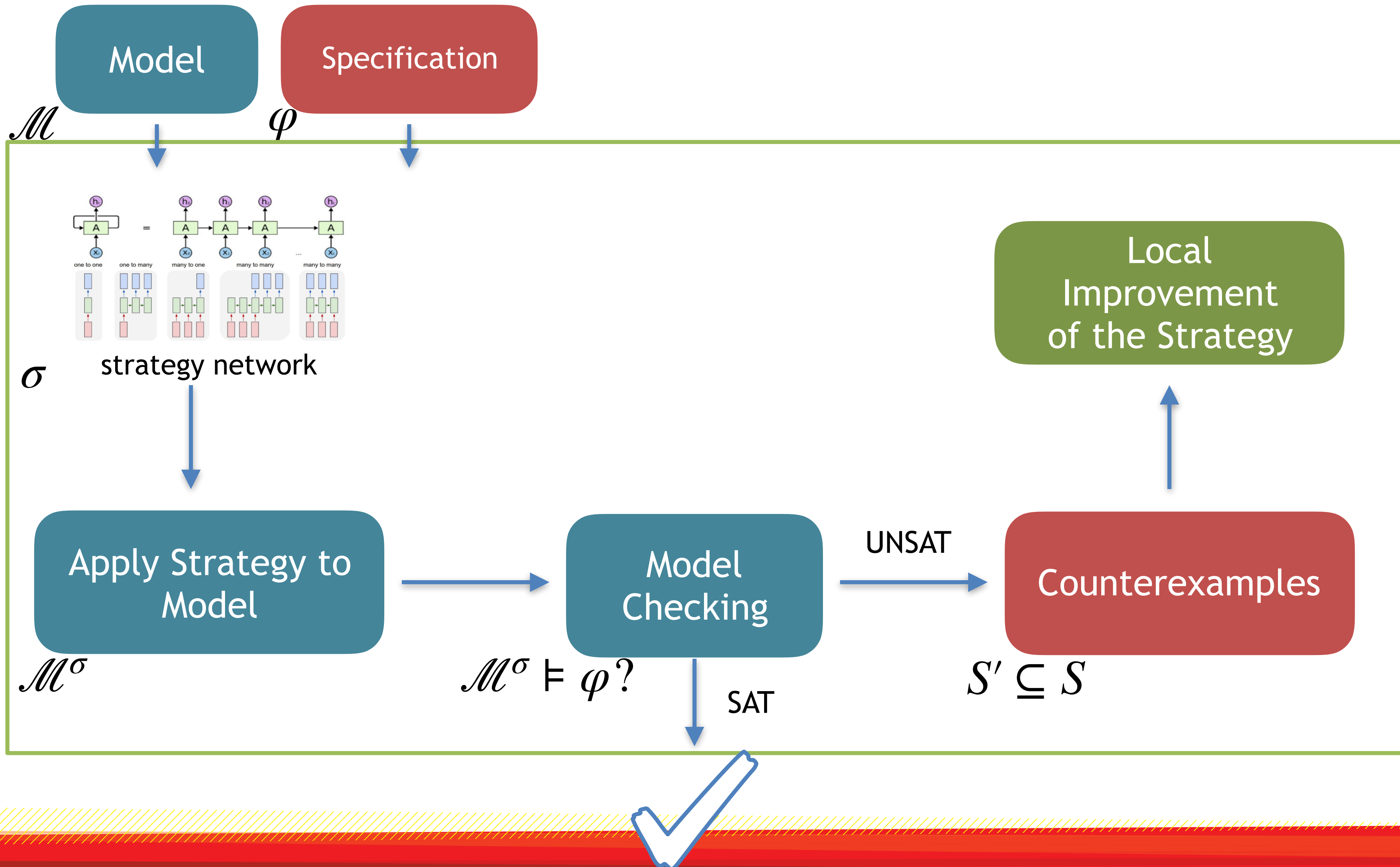
RNN Strategy Improvement



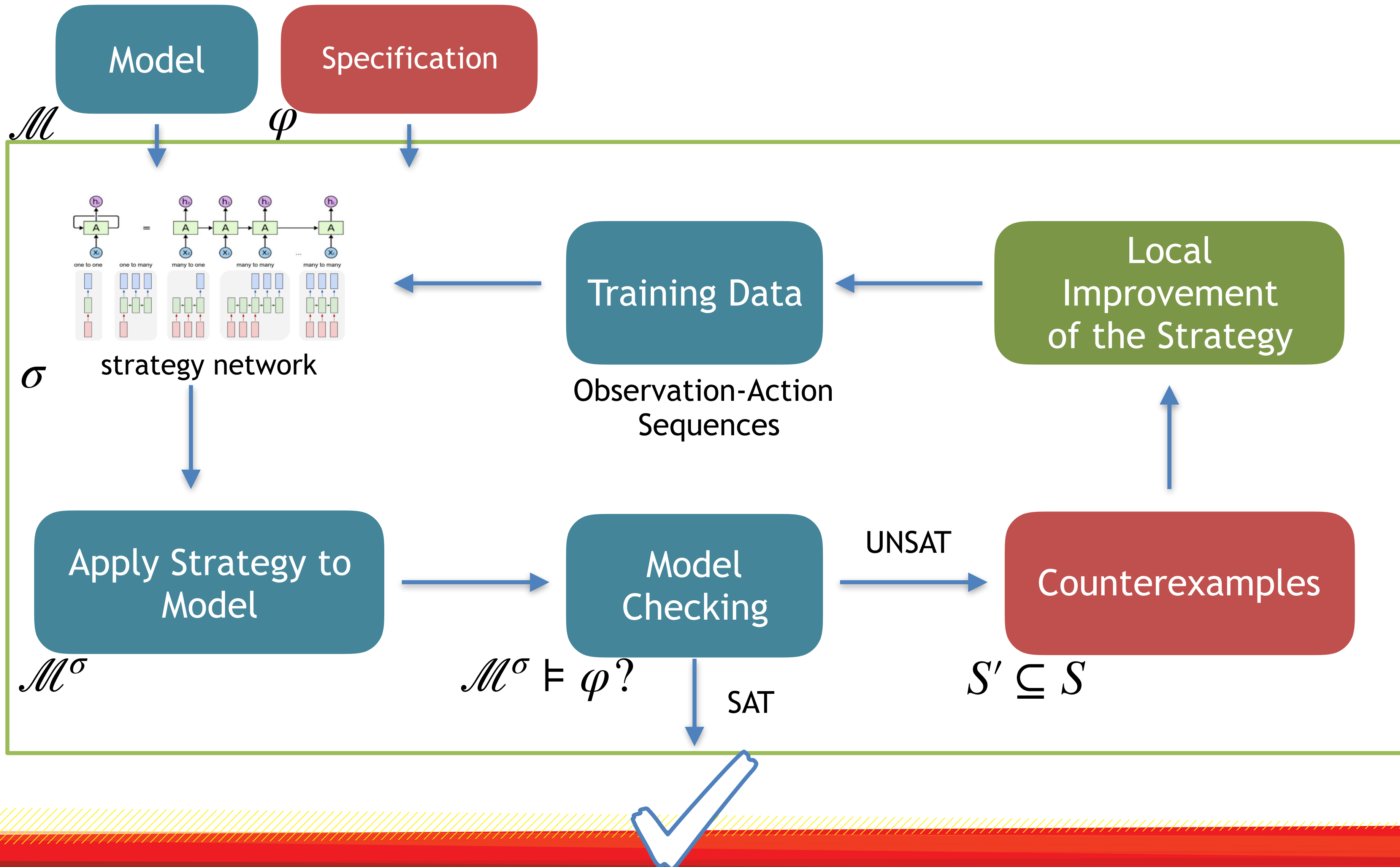
RNN Strategy Improvement



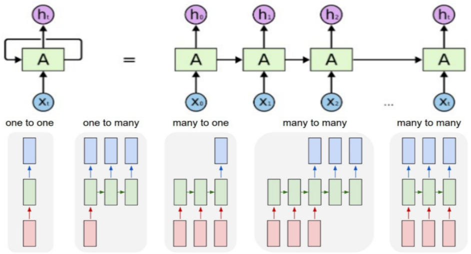
RNN Strategy Improvement



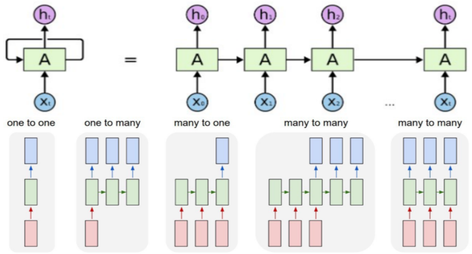
RNN Strategy Improvement



Learning Strategies with RNNs



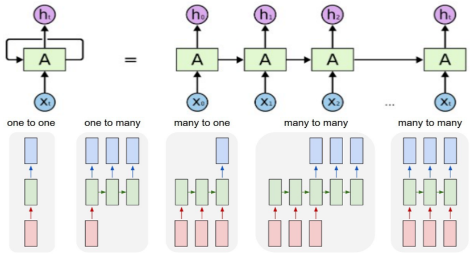
Learning Strategies with RNNs



Recurrent Neural Network

- long short-term memory (LSTM) architecture to learn dependencies in sequential data
- trained with observation-action sequences $ObsSeq_{fin}^{\mathscr{M}}$
- strategy network $\sigma\colon ObsSeq_{fin}^{\mathscr{M}} \to Distr(Act)$

Learning Strategies with RNNs

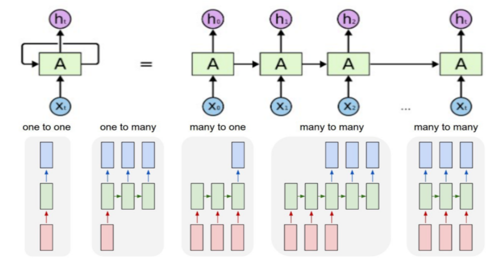


Recurrent Neural Network

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

predictor for a (memoryless) randomized strategy

Learning Strategies with RNNs



Recurrent Neural Network

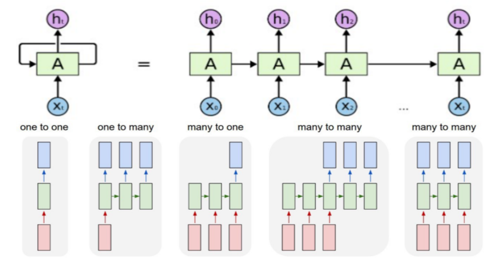
- long short-term memory (LSTM) architecture to learn dependencies in sequential data
- trained with observation-action sequences $ObsSeq^{\mathcal{M}}_{fin}$
- strategy network $\sigma\colon ObsSeq^{\mathcal{M}}_{fin} \rightarrow Distr(Act)$

predictor for a (memoryless) randomized strategy

Training

- Compute optimal MDP strategy
- 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 $ObsSeq^{\mathcal{M}}_{fin}$
- strategy network $\sigma\colon ObsSeq^{\mathcal{M}}_{fin} \rightarrow Distr(Act)$

predictor for a (memoryless) randomized strategy

Training

- Compute optimal MDP strategy
- 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 Strategy

Improving the Strategy

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

Improving the Strategy

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) strategy.

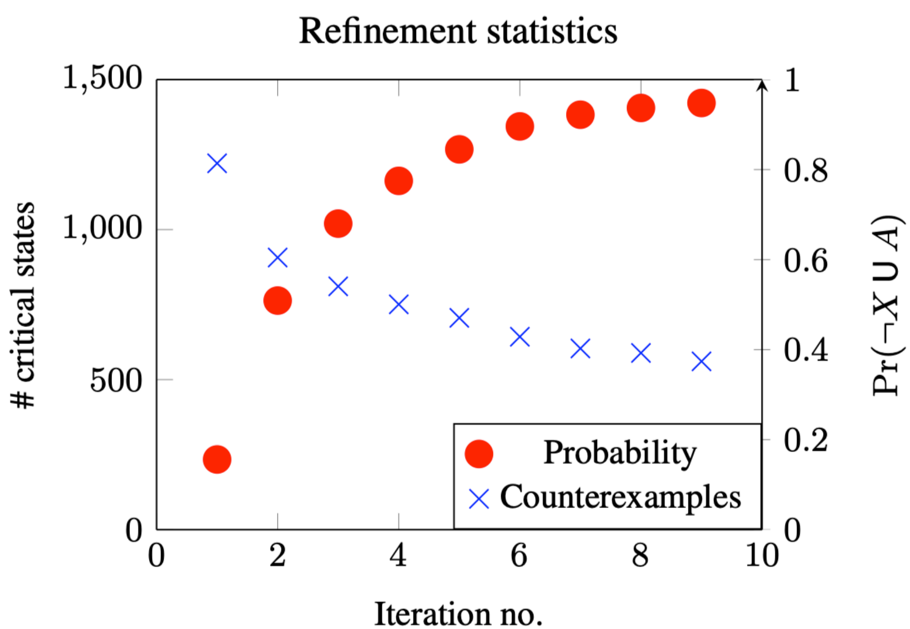
Improving the Strategy

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.

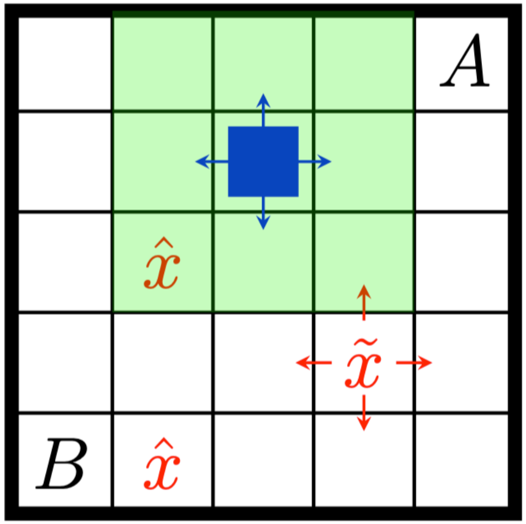
Local linear program

Retrain with the new (locally improved) strategy.



Even if specification is satisfied,
there may be critical states and decisions!

Experiments - LTL



Problem	$\lvert S\rvert$	$\lvert Act\rvert$	$\lvert Z\rvert$
Navigation (c)	c^4	4	256
Delivery (c)	c^2	4	256
Slippery (c)	c^2	4	256
Maze(c)	$3c+8$	4	7
Grid(c)	c^2	4	2
RockSample$[4,4]$	257	9	2
RockSample$[5,5]$	801	10	2
RockSample$[7,8]$	12545	13	2

Problem	States	Type, φ	RNN-based Synthesis Res.	RNN-based Synthesis Time (s)	PRISM-POMDP Res.	PRISM-POMDP Time (s)
Navigation (3)	333	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	0.74	**14.16**	**0.84**	73.88
Navigation (4)	1088	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	0.82	**22.67**	**0.93**	1034.64
Navigation (4) [2-FSC]	13373	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	0.91	47.26	–	–
Navigation (4) [4-FSC]	26741	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	0.92	59.42	–	–
Navigation (4) [8-FSC]	53477	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	**0.92**	85.26	–	–
Navigation (5)	2725	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	0.91	**34.34**	MO	MO
Navigation (5) [2-FSC]	33357	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	0.92	115.16	–	–
Navigation (5) [4-FSC]	66709	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	0.92	159.61	–	–
Navigation (5) [8-FSC]	133413	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	**0.92**	250.91	–	–
Navigation (10)	49060	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	0.79	**822.87**	MO	MO
Navigation (10) [2-FSC]	475053	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	0.83	1185.41	–	–
Navigation (10) [4-FSC]	950101	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	**0.85**	1488.77	–	–
Navigation (10) [8-FSC]	1900197	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	0.81	1805.22	–	–
Navigation (15)	251965	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	**0.91**	**1271.80***	MO	MO
Navigation (20)	798040	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	**0.96**	**4712.25***	MO	MO
Navigation (30)	4045840	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	**0.95**	**25191.05***	MO	MO
Navigation (40)	–	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_1$	TO	TO	MO	MO
Delivery (4) [2-FSC]	80	$\mathbb{E}^{\mathcal{M}}_{\min}, \varphi_2$	6.02	35.35	**6.0**	**28.53**
Delivery (5) [2-FSC]	125	$\mathbb{E}^{\mathcal{M}}_{\min}, \varphi_2$	8.11	**78.32**	**8.0**	102.41
Delivery (10) [2-FSC]	500	$\mathbb{E}^{\mathcal{M}}_{\min}, \varphi_2$	**18.13**	**120.34**	MO	MO
Slippery (4) [2-FSC]	460	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_3$	0.78	67.51	**0.90**	**5.10**
Slippery (5) [2-FSC]	730	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_3$	0.89	84.32	**0.93**	**83.24**
Slippery (10) [2-FSC]	2980	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_3$	**0.98**	**119.14**	MO	MO
Slippery (20) [2-FSC]	11980	$\mathbb{P}^{\mathcal{M}}_{\max}, \varphi_3$	**0.99**	**1580.42**	MO	MO

Experiments - Standard POMDP

Problem	Type	RNN-based Synthesis States	RNN-based Synthesis Res	RNN-based Synthesis Time (s)	PRISM-POMDP Res	PRISM-POMDP Time (s)	pomdpSolve Res	pomdpSolve Time (s)
Maze (1)	$\mathbb{E}^{\mathcal{M}}_{\mathrm{min}}$	68	4.31	31.70	**4.30**	**0.09**	4.30	0.30
Maze (2)	$\mathbb{E}^{\mathcal{M}}_{\mathrm{min}}$	83	5.31	46.65	5.23	2.176	**5.23**	**0.67**
Maze (3)	$\mathbb{E}^{\mathcal{M}}_{\mathrm{min}}$	98	8.10	58.75	7.13	38.82	**7.13**	**2.39**
Maze (4)	$\mathbb{E}^{\mathcal{M}}_{\mathrm{min}}$	113	11.53	58.09	8.58	543.06	**8.58**	**7.15**
Maze (5)	$\mathbb{E}^{\mathcal{M}}_{\mathrm{min}}$	128	14.40	**68.09**	13.00	4110.50	**12.04**	132.12
Maze (6)	$\mathbb{E}^{\mathcal{M}}_{\mathrm{min}}$	143	22.34	**71.89**	MO	MO	**18.52**	1546.02
Maze (10)	$\mathbb{E}^{\mathcal{M}}_{\mathrm{min}}$	203	100.21	**158.33**	MO	MO	MO	MO
Grid (3)	$\mathbb{E}^{\mathcal{M}}_{\mathrm{min}}$	165	2.90	38.94	2.88	2.332	**2.88**	**0.07**
Grid (4)	$\mathbb{E}^{\mathcal{M}}_{\mathrm{min}}$	381	4.32	79.99	4.13	1032.53	**4.13**	**0.77**
Grid (5)	$\mathbb{E}^{\mathcal{M}}_{\mathrm{min}}$	727	6.623	91.42	MO	MO	**5.42**	**1.94**
Grid (10)	$\mathbb{E}^{\mathcal{M}}_{\mathrm{min}}$	5457	**13.630**	**268.40**	MO	MO	MO	MO
RockSample[4, 4]	$\mathbb{E}^{\mathcal{M}}_{\mathrm{max}}$	2432	17.71	35.35	N/A	N/A	**18.04**	**0.43**
RockSample[5, 5]	$\mathbb{E}^{\mathcal{M}}_{\mathrm{max}}$	8320	18.40	**43.74**	N/A	N/A	**19.23**	621.28
RockSample[7, 8]	$\mathbb{E}^{\mathcal{M}}_{\mathrm{max}}$	166656	20.32	**860.53**	N/A	N/A	**21.64**	20458.41

Conclusion

- Novel ways to generate provably correct strategies
- Good scalability, not optimal
- Marriage of Machine Learning and Verification