Cyberphysical Systems
(Autonomous Systems)
in the Intersection of
Controls, Learning and Formal Methods

Ufuk Topcu

The University of Texas at Austin

u-t-autonomous.info

autonomous
SYSTEMS GROUP

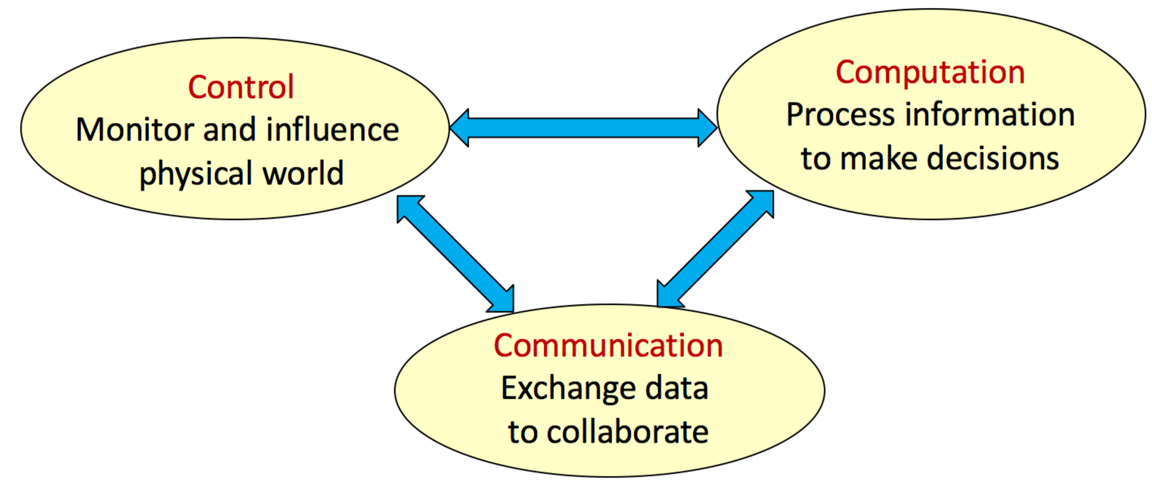
What are cyberphysical systems (CPS)?

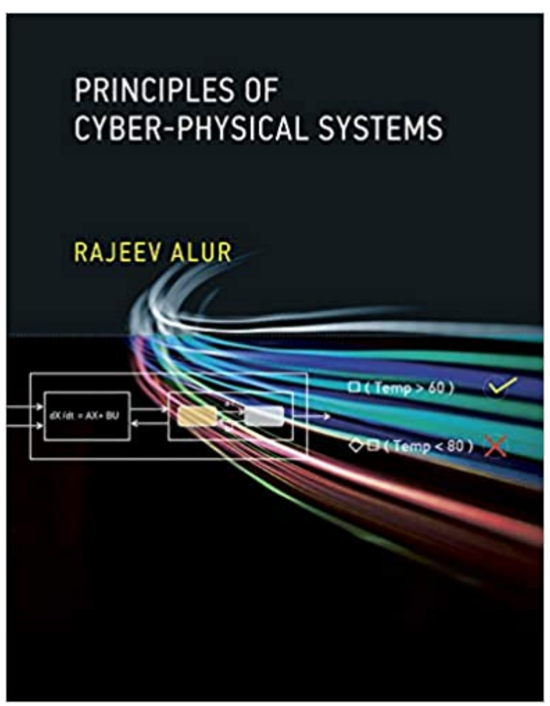
A cyberphysical system…

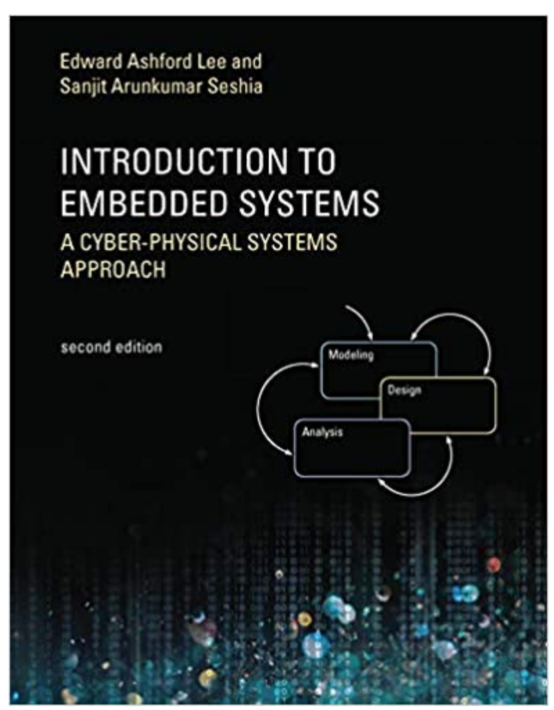
…consists of a collection of computing devices communicating with one another and…

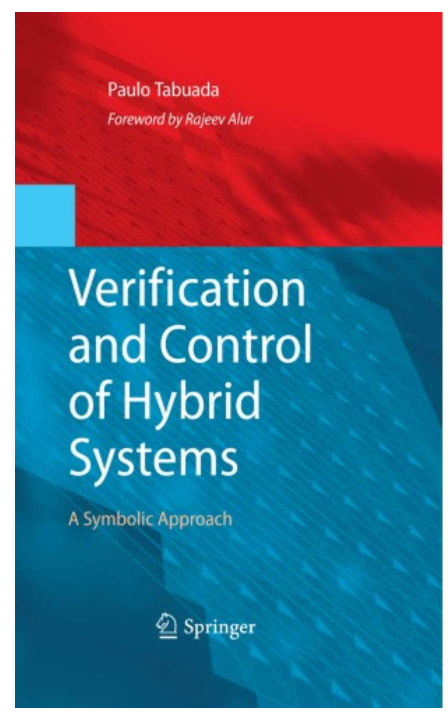
…interacting with the physical world via sensors and actuators in a feedback loop.

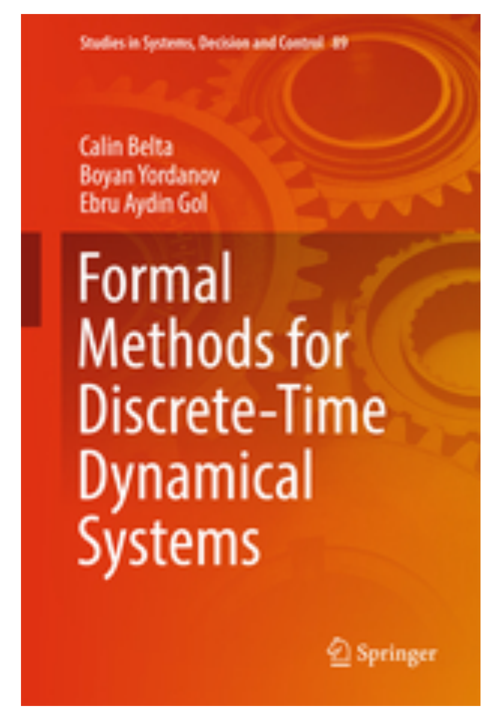
Common among definitions: convergence

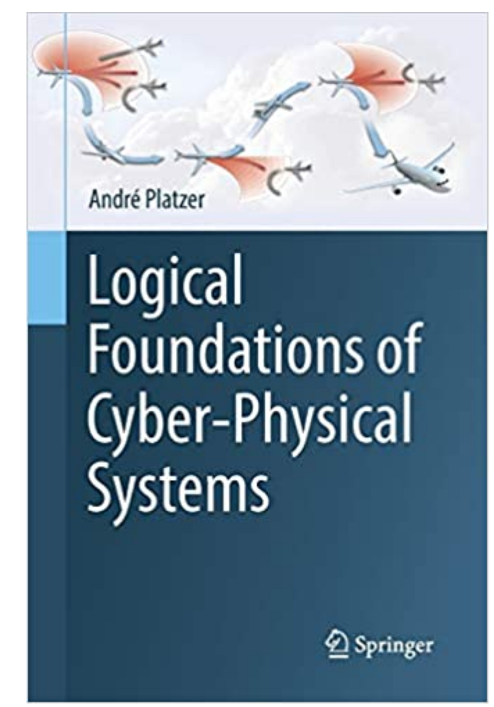












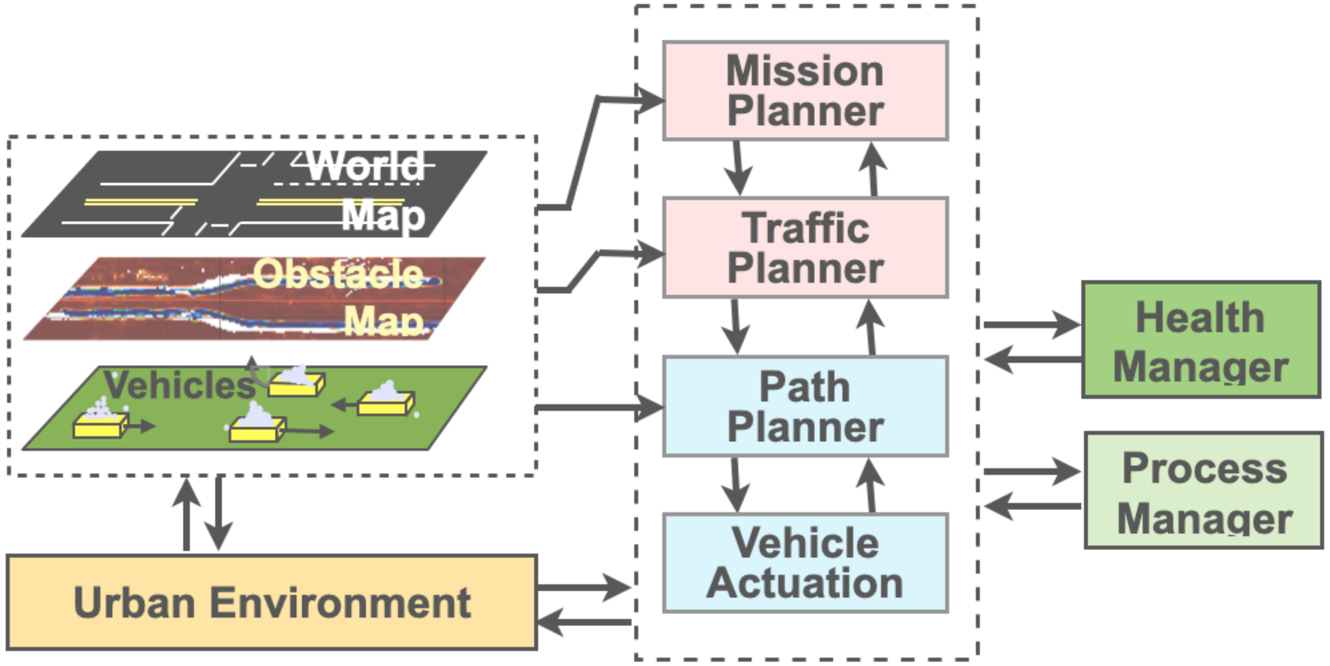
Some properties of cyberphysical systems

Reactivity and interaction with the physical world

A reactive system interacts with its environment in an ongoing manner via inputs (e.g., through sensors) and outputs (e.g., through actuators).

Heterogeneity

Multiple, integrated functionality.

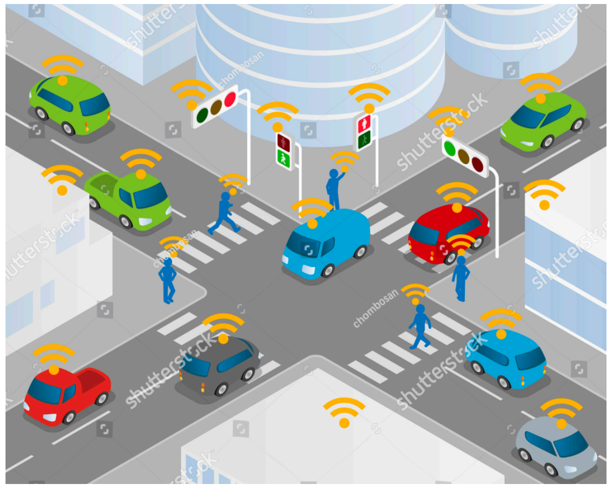


Real-time decisions

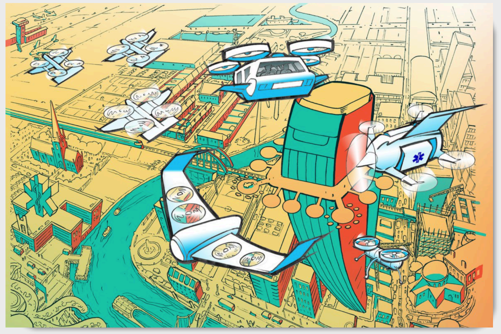
Delays in computation and communication are critical.

Concurrency

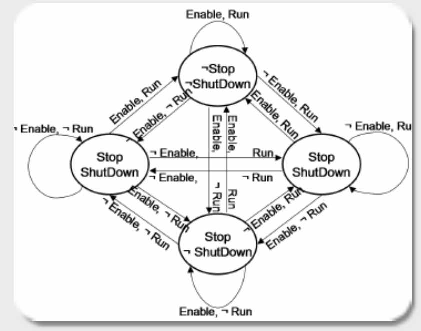
Multiple threads (components or processes) execute simultaneously, exchanging information to achieve a desired goal.



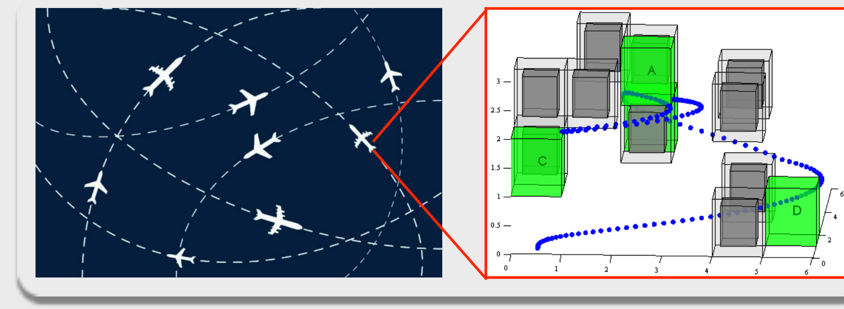
What makes autonomy hard?



dynamic environment

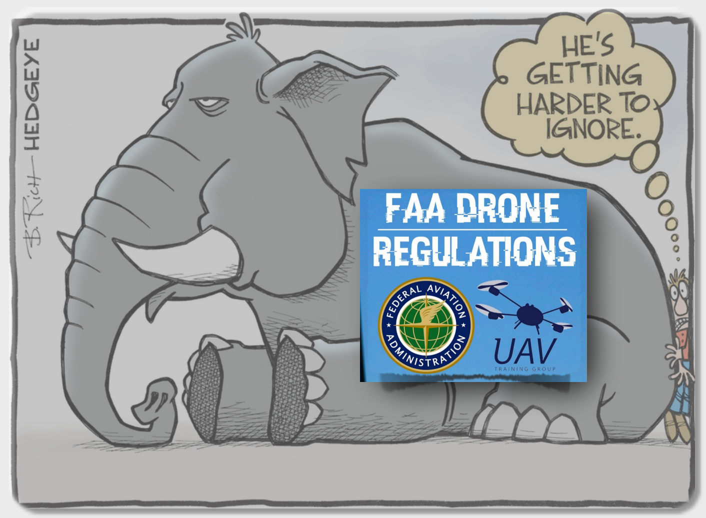


complex missions



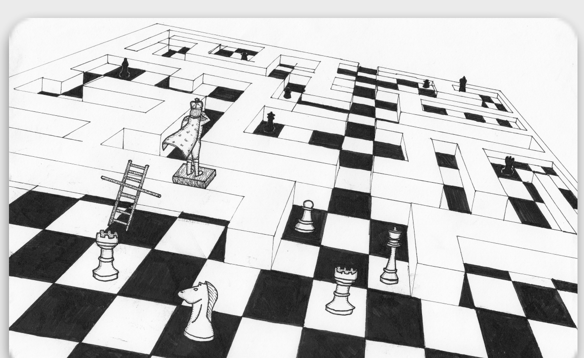
heterogenous decisions

run-time faults

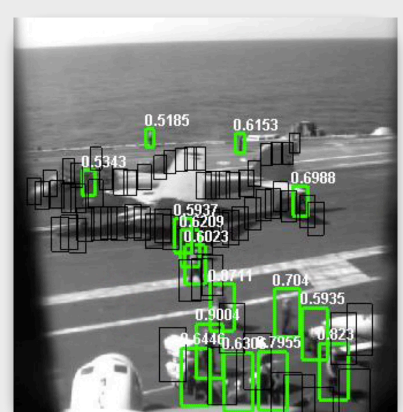


verifiability

unknown environments



integrity of critical information



imperfect perception

variations in user characteristics

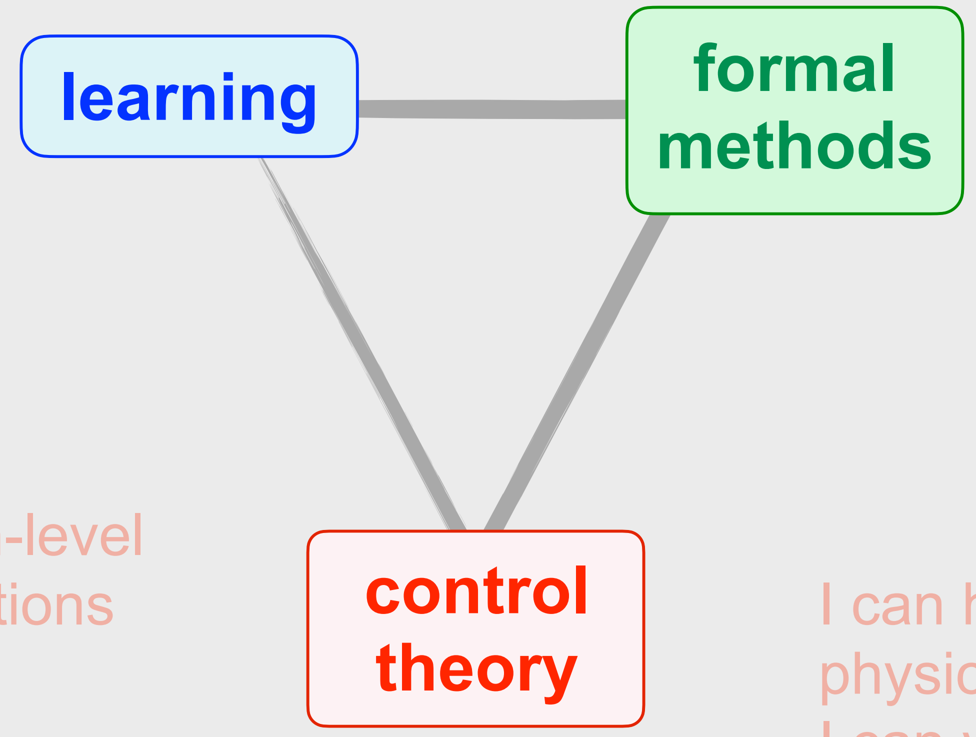
Why are we not there yet? What is still missing?

What if you don't have models, models adapt,…?

I know how to verify software.

You may, but how will you verify?

Give me data, I'll learn all you need.



Btw, an autonomous system is not only software.

Sure. How about high-level requirements, interactions with human…?

I can handle physics. I can adapt. I can verify stability.

Autonomous systems are nobody's comfort zone:

We need hybrid solutions.

Outline

Correct-by-construction synthesis of hierarchical control protocols

- formal methods ↔ controls

What would it take to construct the control software of an autonomous system **in an hour —as opposed to days or even weeks**—to deliver a mission?

Verifiable reinforcement learning

- formal methods ↔ learning

How can an autonomous system **learn** how to execute a new mission in an a priori unknown environment **efficiently** and **safely?**

Planning in POMDPs

- formal methods ↔ convex optimization
- formal methods ↔ learning

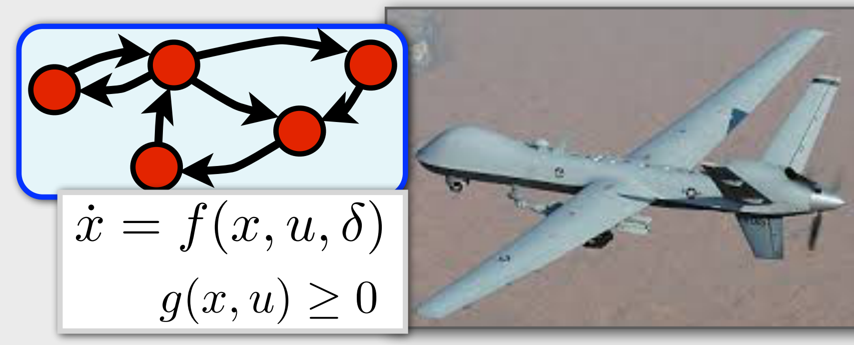
How can we cope with imperfection and/or limitations in run-time information?

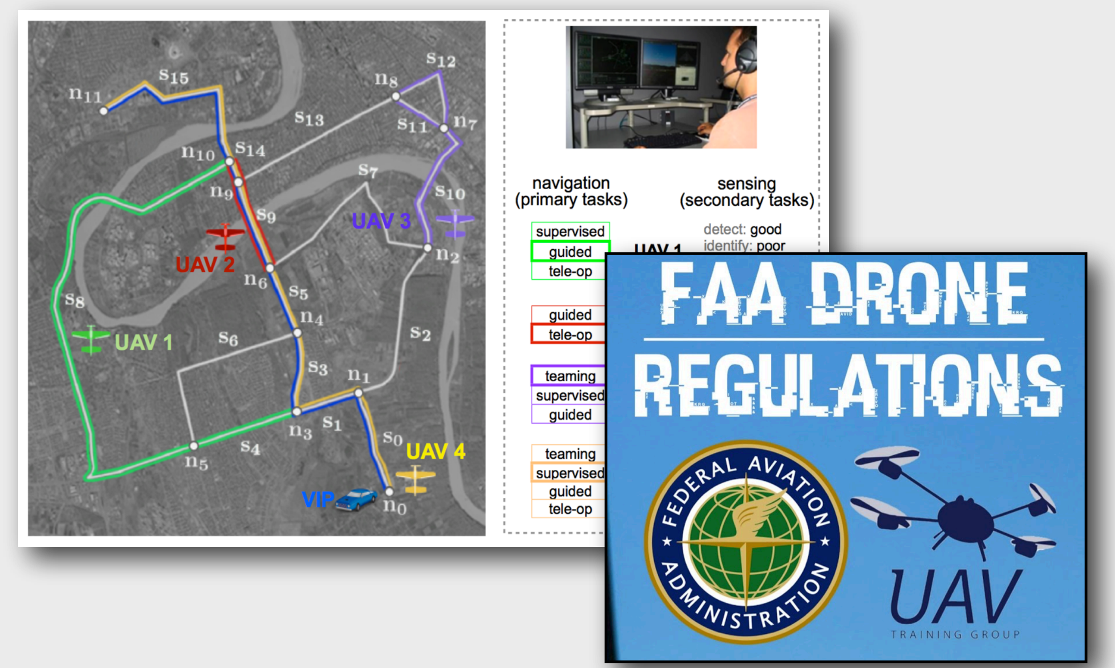
A (sample) synthesis problem

Given:

- ***System model***
  - both continuous & discrete evolution
  - actuation limitations
  - modeling uncertainties & disturbances

- ***Specifications in "temporal logic"***
  - high-level requirements & goals
  - assumptions about the a priori unknown, dynamic environment





Automatically synthesize a control protocol that

- manages the system behavior;
- reacts to changes in allowable external environment; and
- is **provably correct** with respect to the specifications.

De-tour: Specifying behavior with temporal logic

Temporal Logic = **Propositional Logic** + **Temporal Operators**

$\wedge$ (and)
$\vee$ (or)
$\rightarrow$ (implies)
$\neg$ (not)

$\Diamond$ (eventually)
$\Box$ (always)
$\mathcal{U}$ (until)

- Reason about infinite sequences $\sigma = s_0 s_1 s_2 \ldots$ of states
- Many different dialects of temporal logic (with probabilistic and epistemic modalities)
- Specify safe, allowable, required, or desired behavior of system and/or environment.

Coverage:

$$\Box\Diamond uav_1 = w_3 \wedge \Box\Diamond uav_1 = w_4 \wedge \Box\Diamond uav_1 = w_5 \wedge \Diamond uav_2 = w_1 \wedge \Diamond uav_2 = w_2 \wedge \Diamond uav_2 = w_6$$

Sequencing:

$$\Diamond\,(uav = w_1 \wedge \Diamond\,(uav = w_2 \wedge \Diamond uav = w_3))$$

Coordination with mutual exclusion:

$$\neg(uav_1 = uav_2 \vee uav_1 = uav_3 \vee uav_1 = uav_4 \vee uav_2 = uav_3 \vee uav_2 = uav_4 \vee uav_3 = uav_4) \,\mathsf{U}\, (uav_1 = w_1 \wedge uav_2 = w_1 \wedge uav_3 = w_1 \wedge uav_4 = w_1 \wedge \bigcirc^{-1} uav_1 = a_{1,1} \wedge \bigcirc^{-1} uav_2 = a_{1,3} \wedge \bigcirc^{-1} uav_3 = a_{1,5} \wedge \bigcirc^{-1} uav_4 = a_{1,7})$$

Sequencing with avoidance:

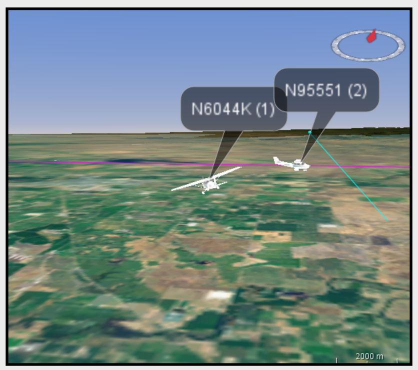
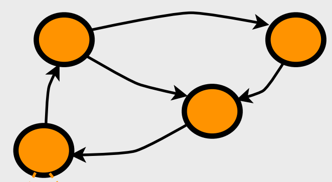
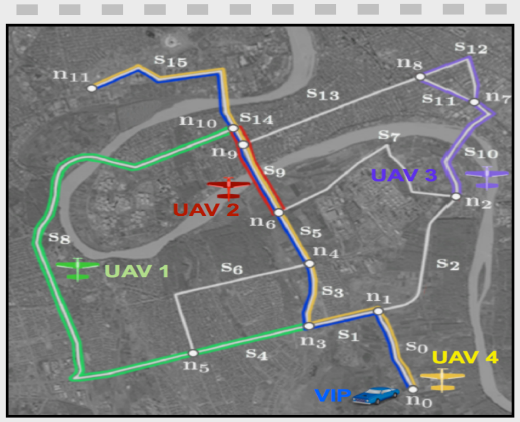
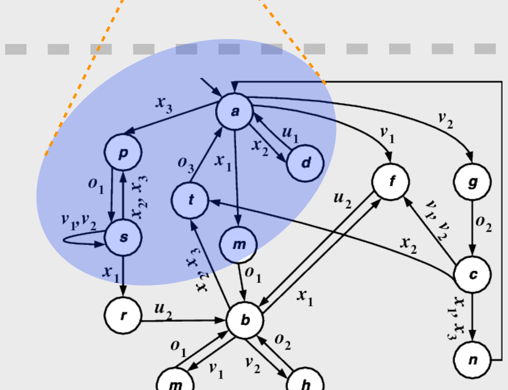
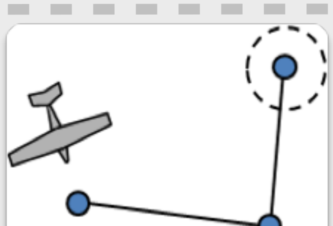
$$\Diamond uav = w_1 \wedge \Diamond uav = w_2 \wedge \Diamond uav = w_3 \wedge \neg w_2 \,\mathsf{U}\, w_1 \wedge \neg w_3 \,\mathsf{U}\, w_2$$

Never after:

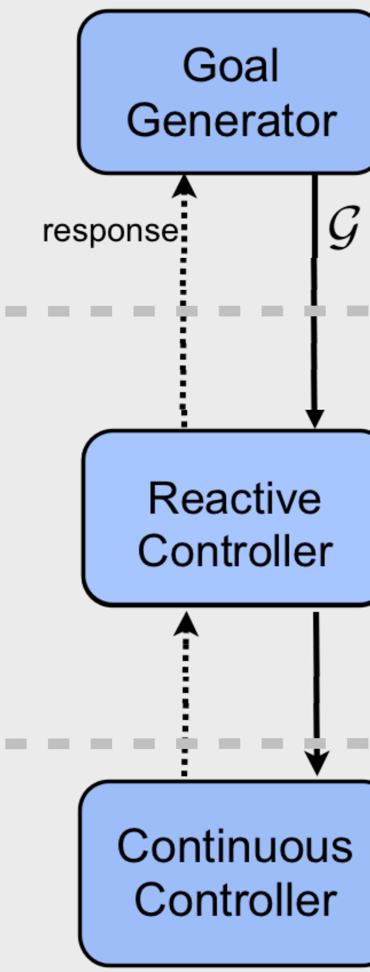
$$\Box(uav = w_1 \rightarrow \bigcirc\Box\neg uav = w_1)$$

A Solution: Hierarchical Control Structure

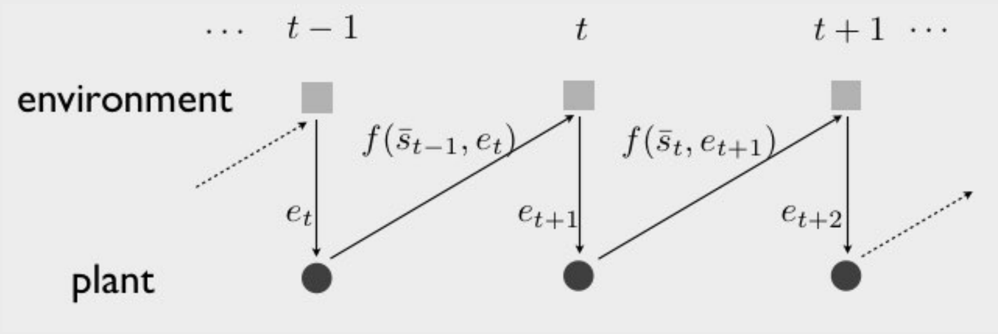
Theorem: *Correctness provably guaranteed* by the construction of the abstractions and splitting of the specifications

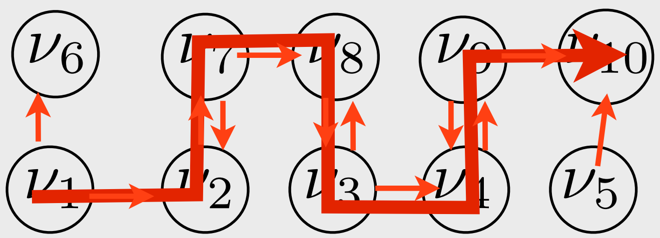
Different views		Multi-scale models	Synthesis method
	long-horizon specifications		Iterative graph search
	short-horizon specifications		Two-player, turn-based graph game
	constraints on continuous state + input	$x_{t+1} = f(x_t, w_t, u_t)$ $x \in \mathcal{X}, u \in \mathcal{U}, w \in \mathcal{W}$	Constrained, finite-horizon optimal control

Control protocol



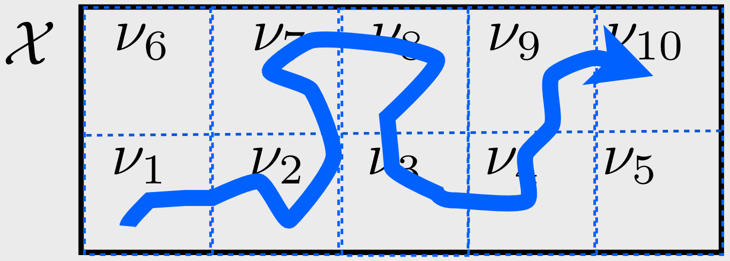
Zoom in a bit…





$$x_{t+1} = f(x_t, w_t, u_t)$$
$$x \in \mathcal{X}, u \in \mathcal{U}, w \in \mathcal{W}$$

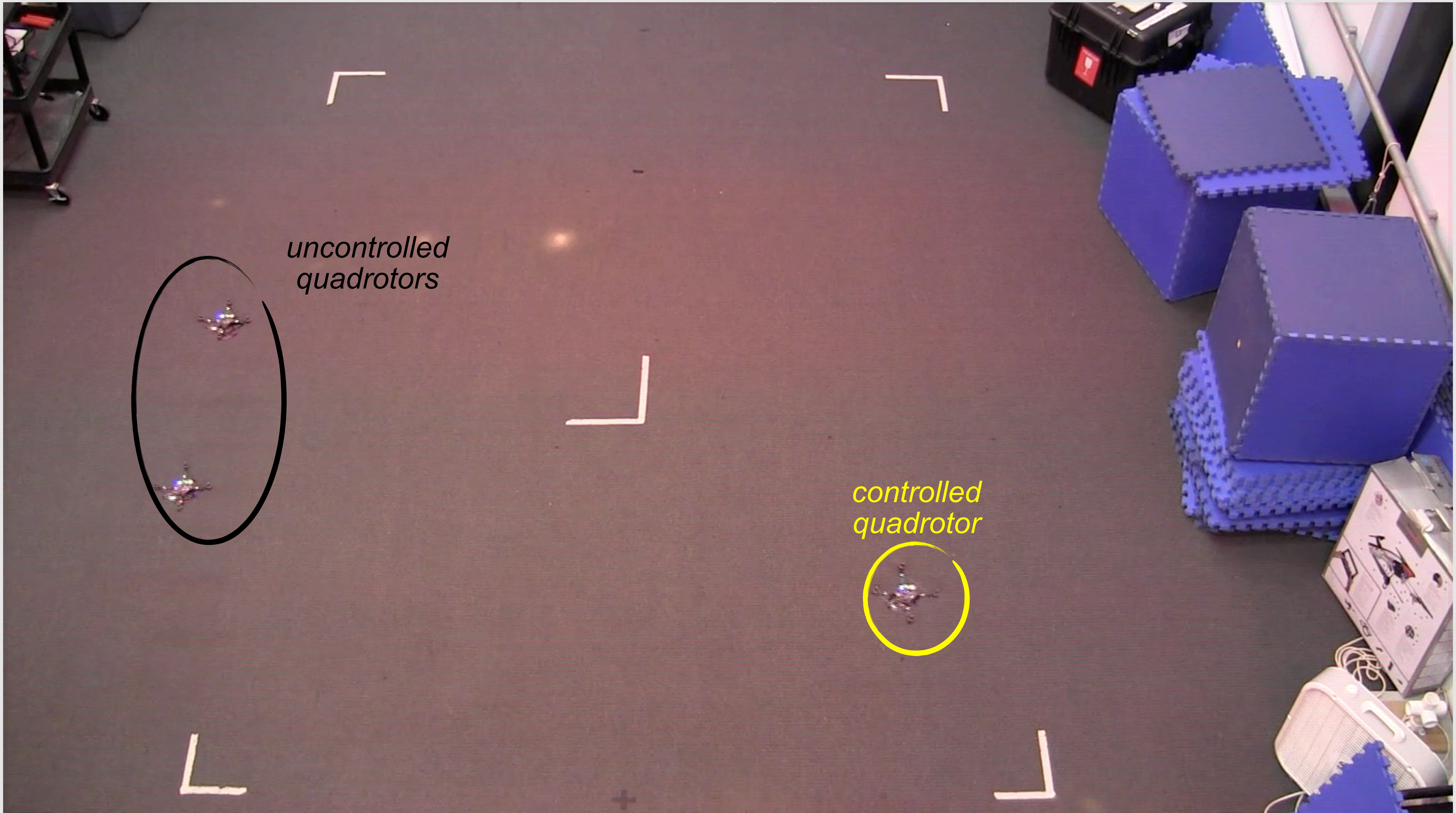
$\exists u \in \mathcal{U}$ and finite $T > 0$ s.t.
$x_t \in \mathcal{X}_{\text{initial}} \cup \mathcal{X}_{\text{target}}$ $\forall t = 0, \ldots, T,$
$x_T \in \mathcal{X}_{\text{target}}$
for all $x_0 \in \mathcal{X}_{\text{initial}}$ and $w \in \mathcal{W}$?



Correctness: For any discrete run satisfying the specification, there exists an admissible control signal leading to a continuous trajectory satisfying the specification.

Proof — “correct by construction”:
Constructive $\longrightarrow$ Finite-state model + Continuous control signals.

Specify + Synthesize



Outline

Correct-by-construction synthesis of hierarchical control protocols

- formal methods ↔ controls

What would it take to construct the control software of an autonomous system **in an hour—as opposed to days or even weeks**—to deliver a mission?

Verifiable reinforcement learning

- formal methods ↔ learning

How can an autonomous system **learn** how to execute a new mission in an a priori unknown environment **efficiently** and **safely?**

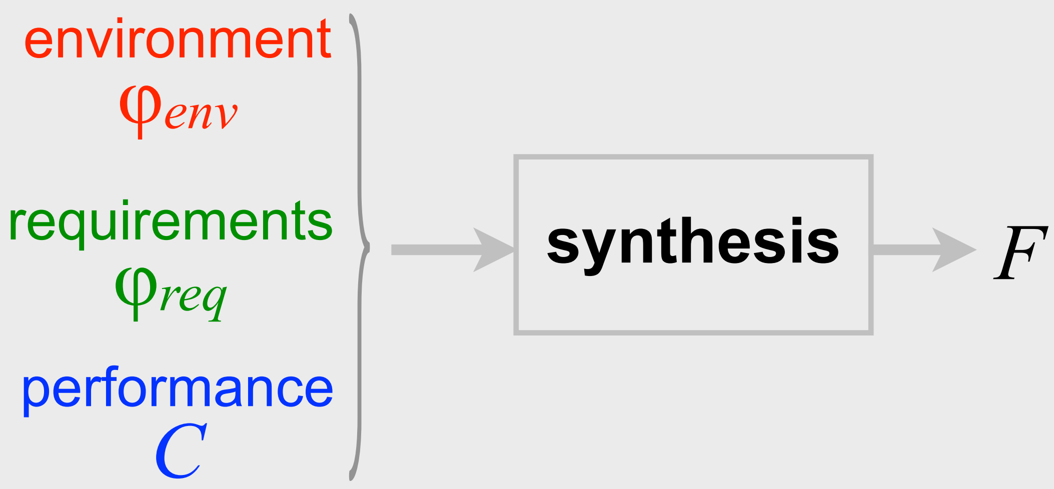
Planning in POMDPs

- formal methods ↔ convex optimization
- formal methods ↔ learning

How can we cope with imperfection and/or limitations in run-time information?

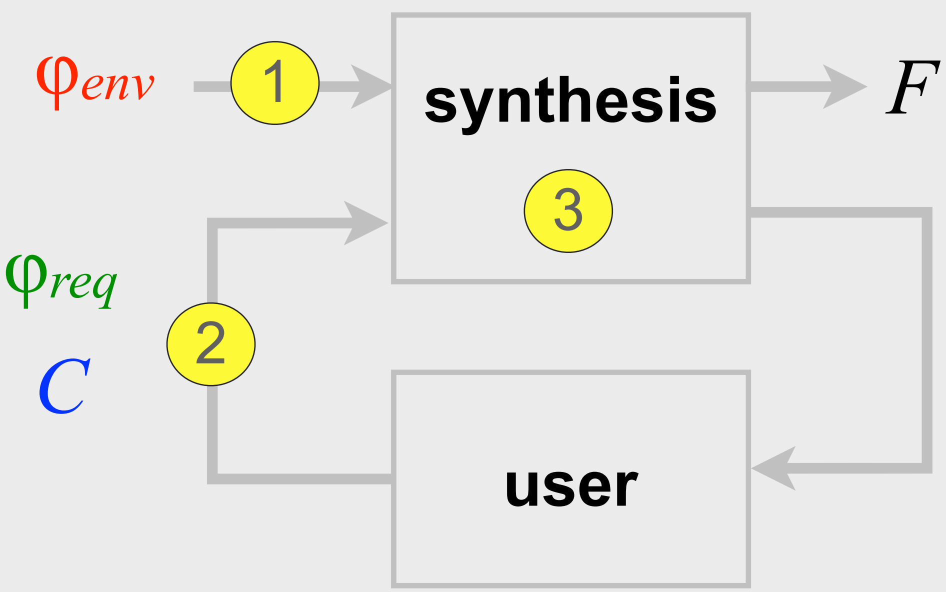
Need for learning

Conventional Synthesis



Known/fixed
temporal logic constraints
+ reward structure

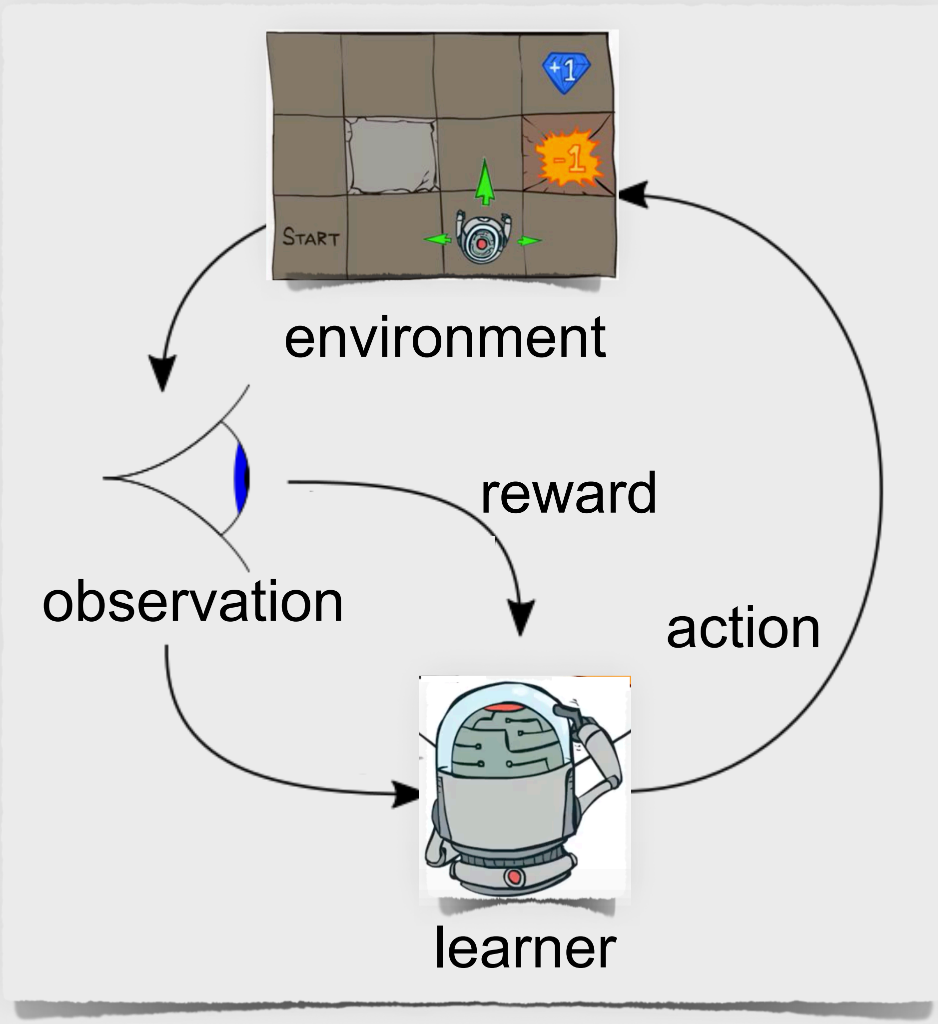
Synthesis + Learning



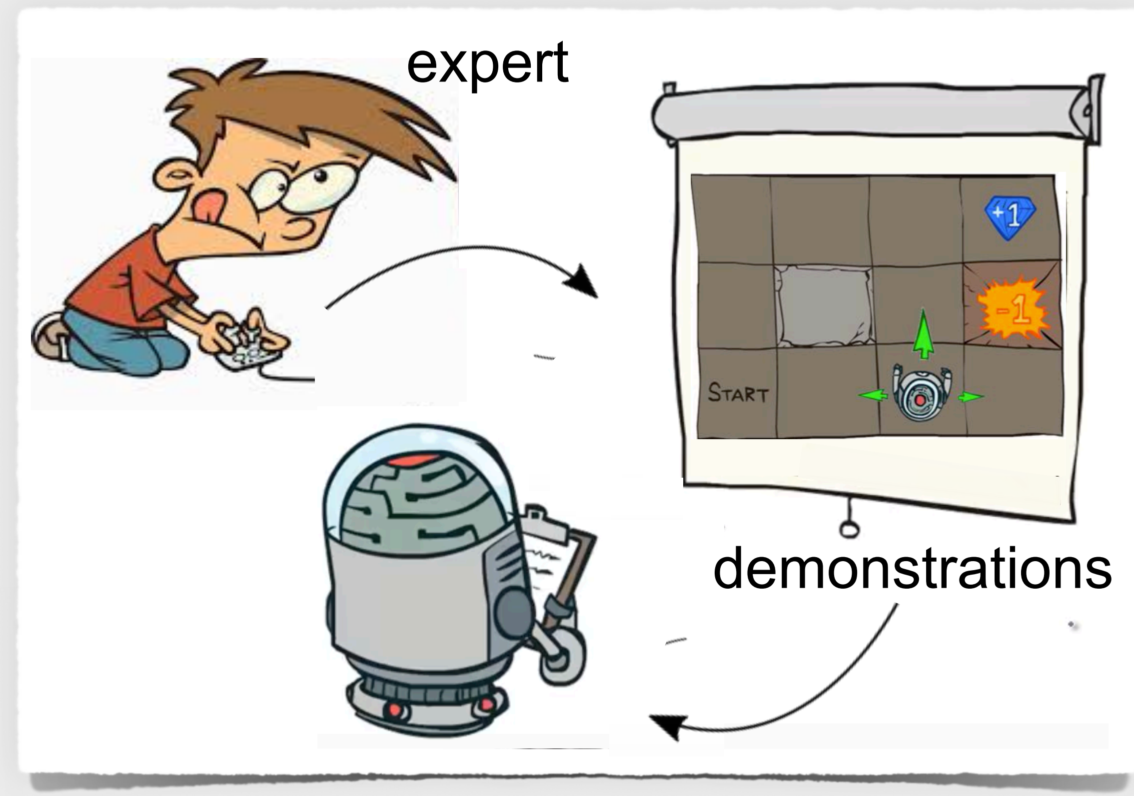
1. No a priori knowledge about environment. But, online (and offline) data available.
2. Operators do not “speak” temporal logic or cannot express complex reward structures.
3. Controller itself may be learned from data or examples.

Learning subject to temporal logic specifications

Reinforcement learning



Inverse reinforcement learning

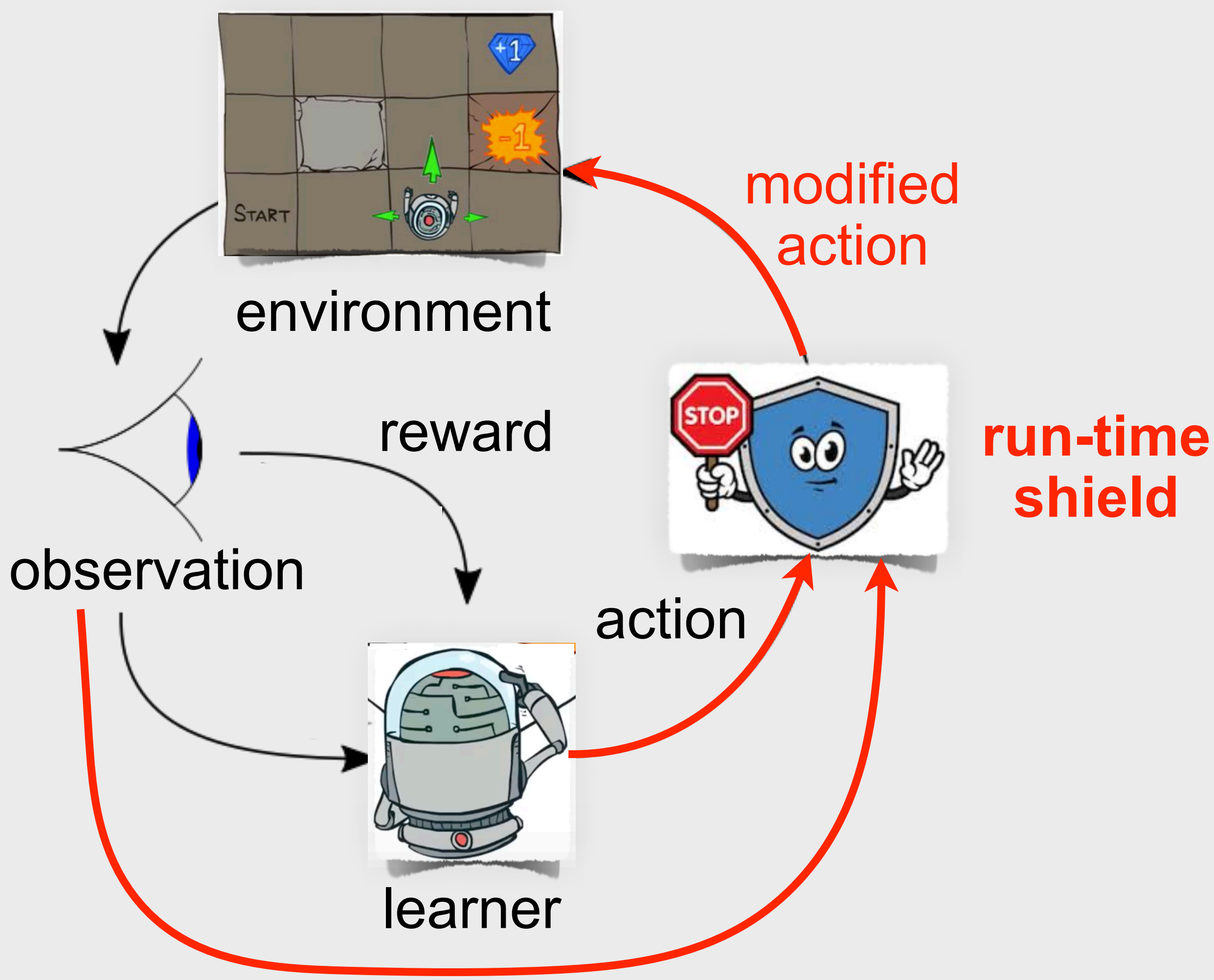


new piece of problem data: temporal logic **specification φ**

Does the learned strategy satisfy φ?

Is φ violated during learning?

Safe reinforcement learning via shielding



Run-time shielding…

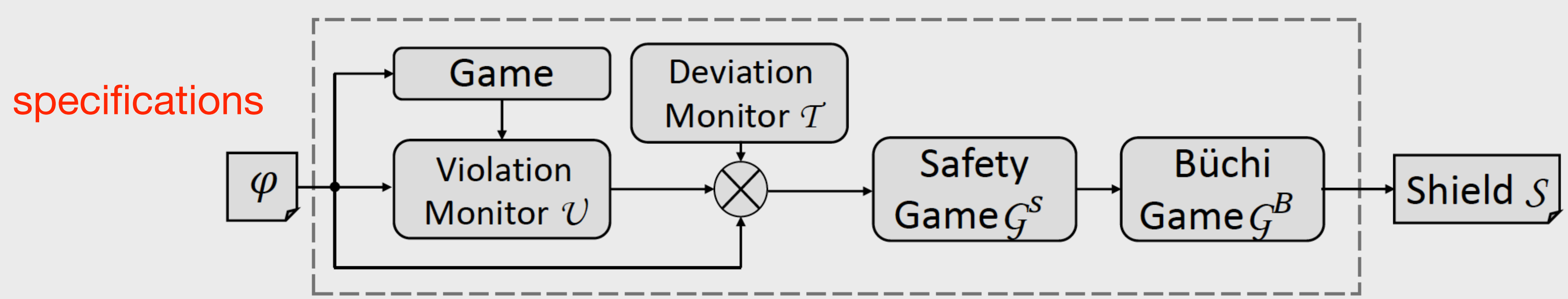
- Corrective w.r.t. specifications in the safety fragment of linear temporal logic
- Minimally interfering
- Agnostic to the learner
- Can be used with function-approximation-based methods
- Preserves (constrained) convergence in some cases

Provable properties

- Improves the data-efficiency in learning

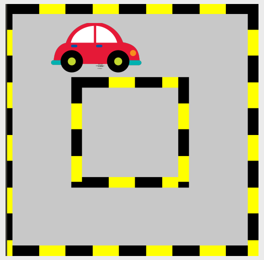
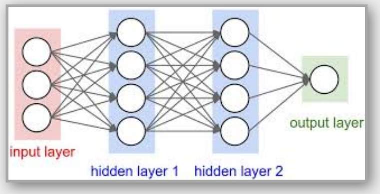
Empirically observed

Shield synthesis



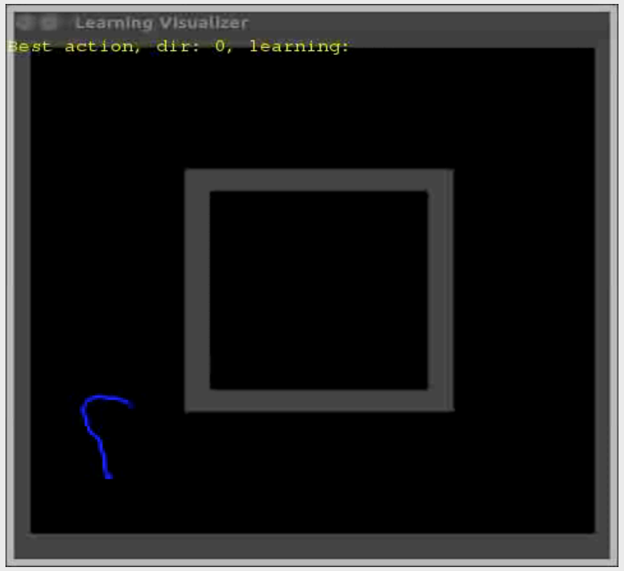
Examples of shielded reinforcement learning

Deep reinforcement learning with a neural network of three layers and Boltzmann exploration

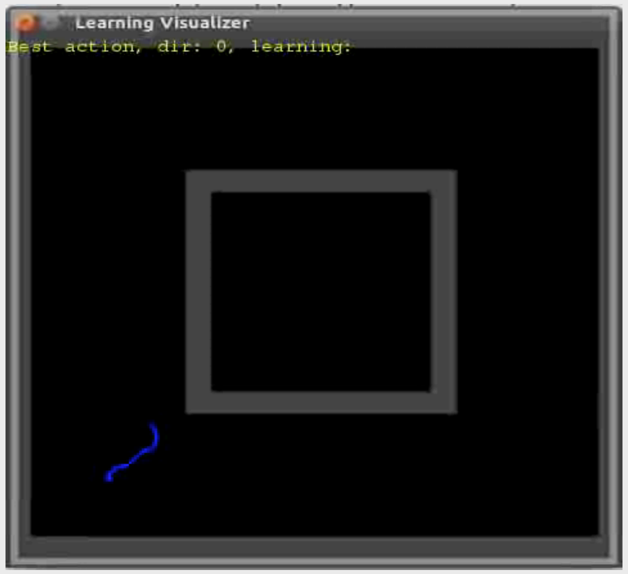
without shielding

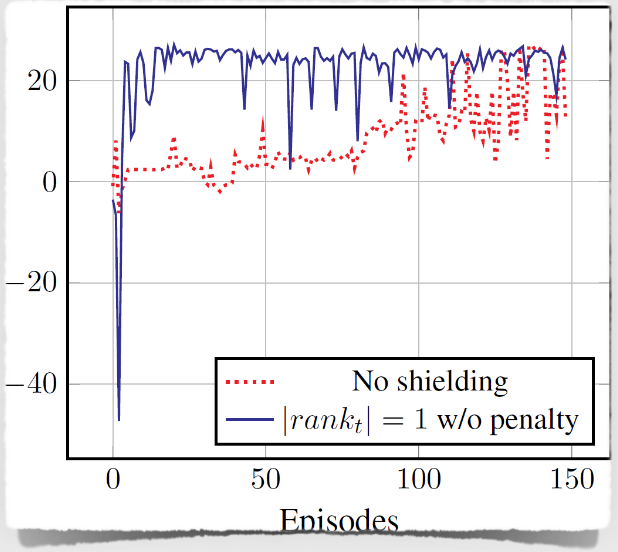
(40 min training + 1600 crashes)



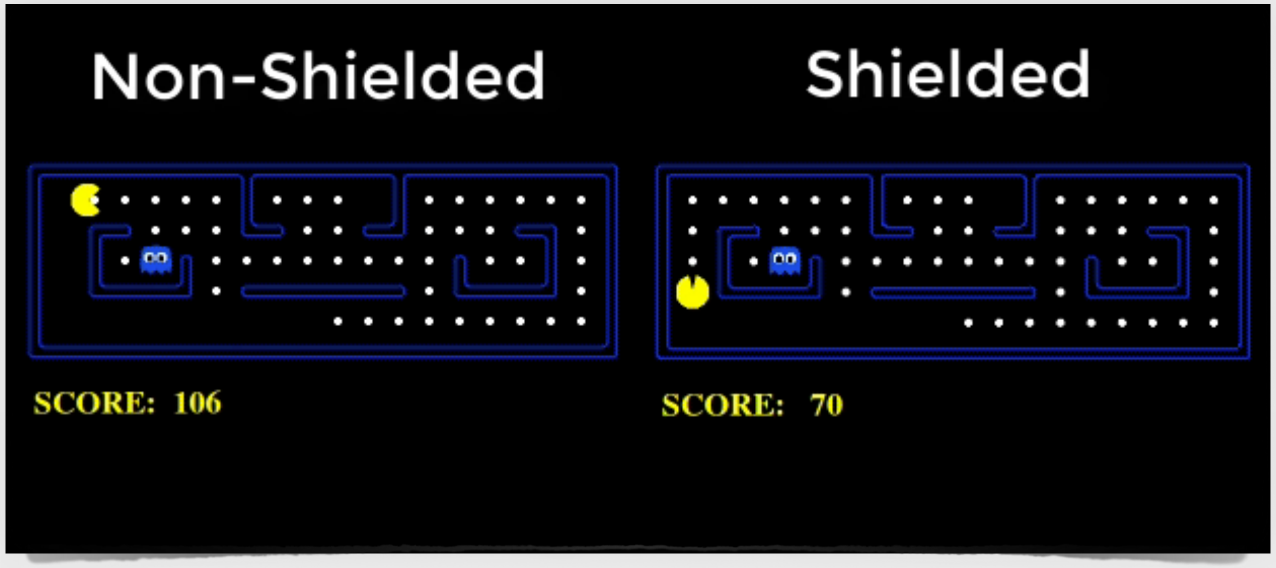
with shielding

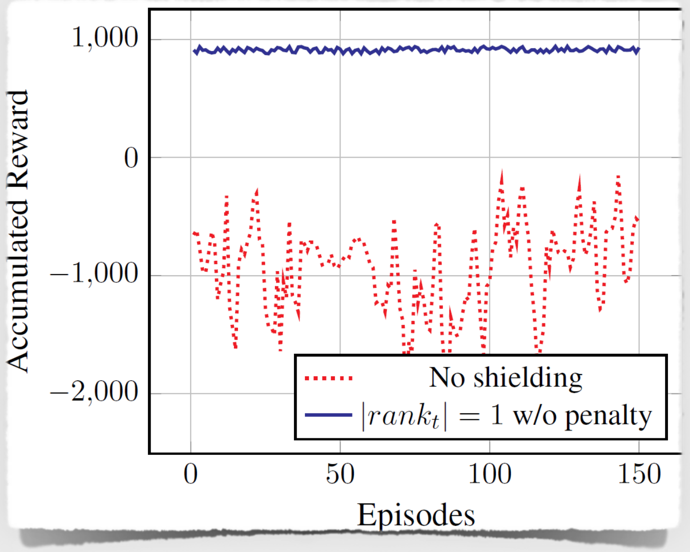
(1 min training + no crashes)





Reinforcement learning for PACMAN

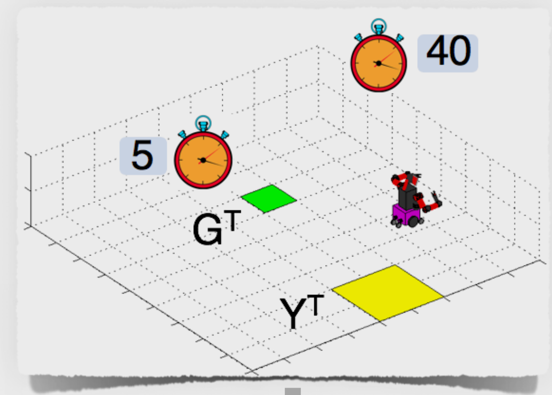




Can task similarities help learn faster?

Source task

Target task



Data

Data

Logical similarity?

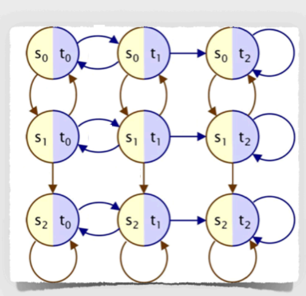
Inferred formulas

$\phi^S = \Diamond_{[1,15]}\Box_{[0,3]}G^S \wedge \Diamond_{[21,39]}Y^S$

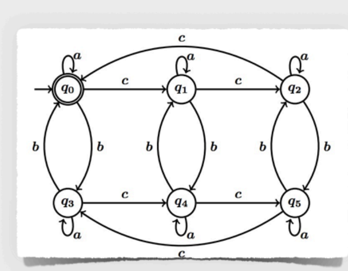
$\phi^T = \Diamond_{[5,18]}\Box_{[0,5]}G^T \wedge \Diamond_{[24,39]}\bar{Y}^T$

Construct timed automaton

Construct timed automaton



x



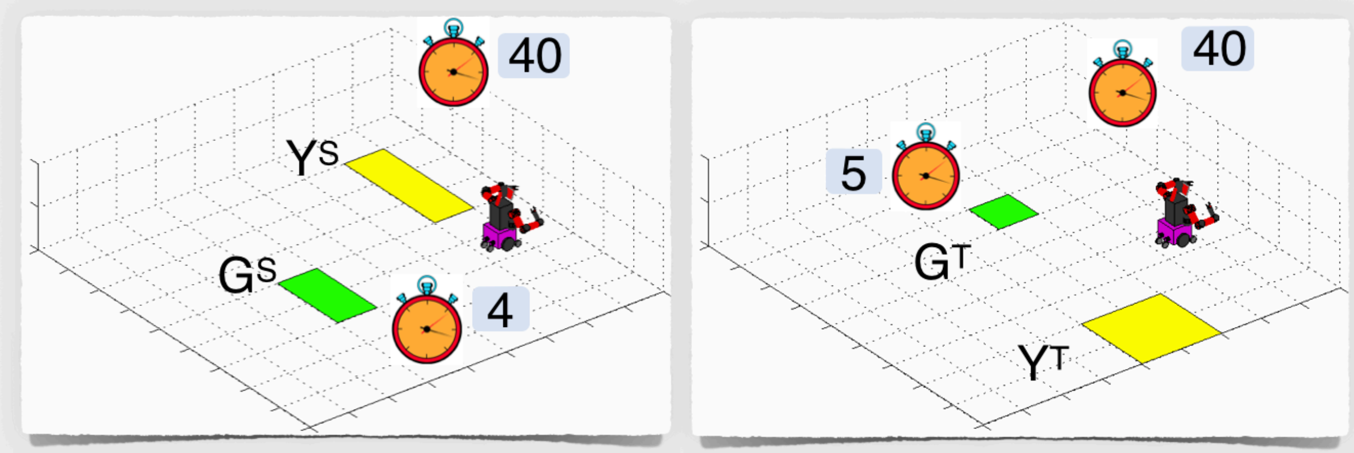
RL in extended state space

Extended Q-functions

RL in extended state space

Transfer of temporal logical similarities improves data efficiency by orders of magnitude.

Source task

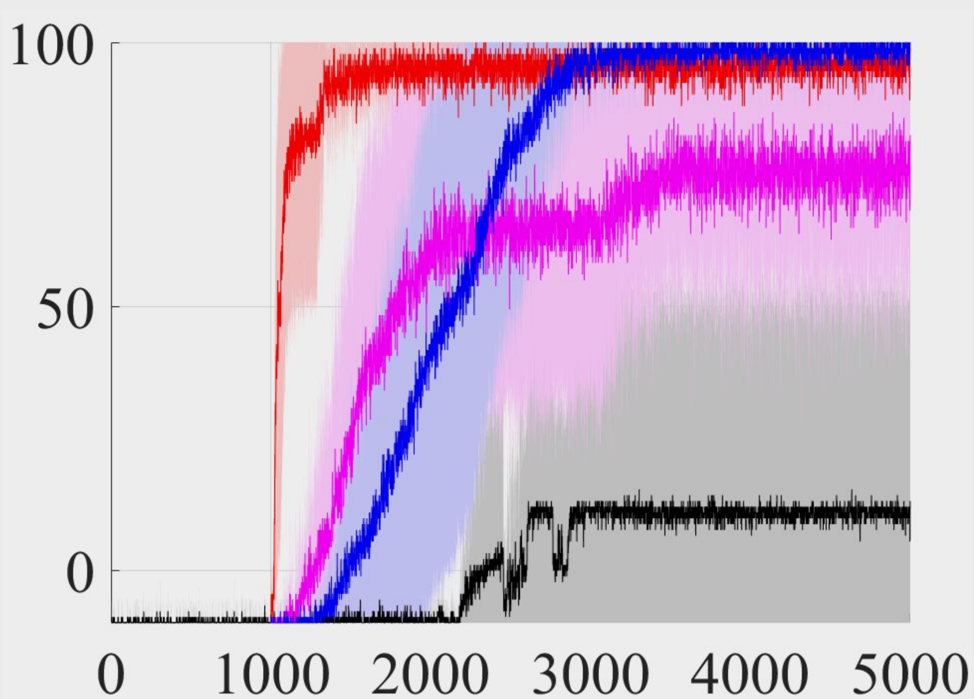


Target task

$\Diamond_{[1,15]}\Box_{[0,3]}G^S \wedge \Diamond_{[21,39]}Y^S$

$\Diamond_{[5,18]}\Box_{[0,5]}G^T \wedge \Diamond_{[24,39]}\bar{Y}^T$

Cumulative reward



Episodes x10

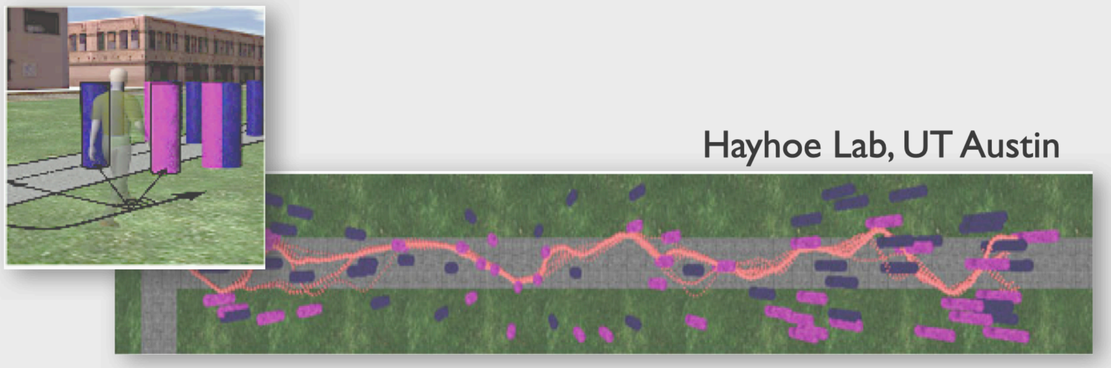
+ transfer of extended Q-functions

+ “informative” MITL formulas

+ extended state space
+ maximize classification rate in MITL inference

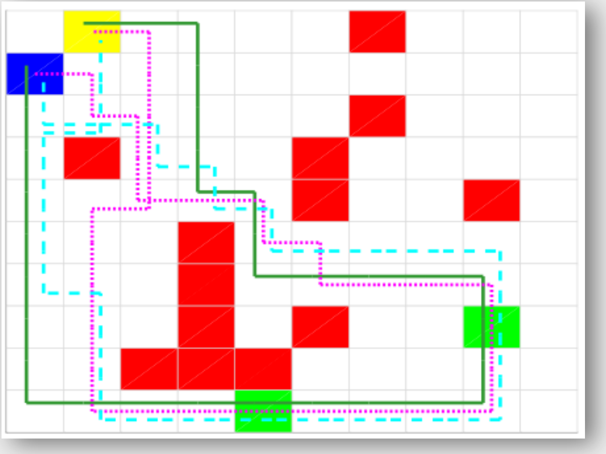
Q-learning

Inverse reinforcement learning



Reward-free MDP
$M = (S, A, T, \gamma)$

inverse reinforcement learning



Expert demonstrations D

Reward function R

Policy that mimics the demonstrations

Inverse reinforcement learning

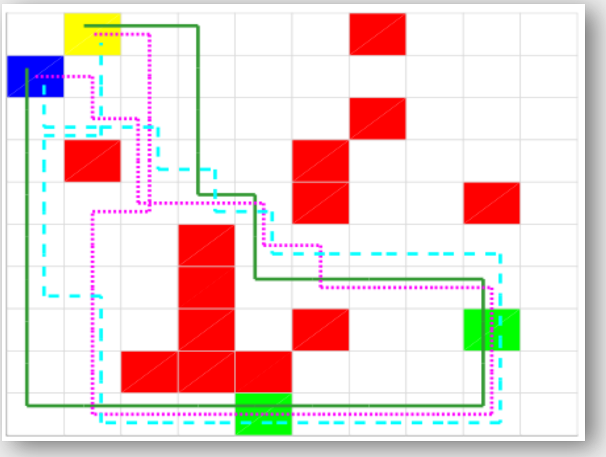
with high-level task as side information

Reward-free MDP
$M = (S, A, T, \gamma)$

modified inverse reinforcement learning

Reward function R

Policy that balances between
mimicking the demonstrations
and
probability of satisfying φ



Expert demonstrations D

$$\varphi_{\mathrm{cs}} = \varphi_{\mathrm{init}} \rightarrow (\varphi_{\mathrm{safe}} \wedge \varphi_{\mathrm{goal}})$$
$$\varphi_{\mathrm{init}} = \neg r \wedge \neg y$$
$$\varphi_{\mathrm{safe}} = \neg r \;\mathcal{U}\; y$$
$$\varphi_{\mathrm{goal}} = \big((\neg y)\;\mathcal{U}\;(\Diamond g_1 \wedge \Diamond g_2)\big) \wedge (\Diamond y)$$

Task knowledge as side information
(encoded as a temporal logic formula φ)

Inverse reinforcement learning with high-level task as side information

$R = [f_1 \cdots f_k]\,\theta$ reward parameterized in θ given the features $f_1, \ldots, f_k$

$Q_\pi(s,a) = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R(s_k, a_k) | s_0, a_0\right]$ expected discounted reward with policy π from state s and action a

$\pi_Q(s,a) = \frac{\exp(Q(s,a))}{\sum_{a'} \exp(Q(s,a'))}$ softmax policy (randomized)

Assumptions:

- Demonstrations are samples of a softmax, randomized policy.
- This unknown policy satisfies φ with probability higher than a threshold.

negative log-likelihood

$$\begin{aligned} \min_\theta \quad & J^{\text{mle}}(\pi_\theta | M, D) \\ \text{s.t.} \quad & \text{Bellman equation}(\theta, M, D) \end{aligned}$$

Inverse reinforcement learning with high-level task as side information

$R = [f_1 \cdots f_k]\,\theta$ reward parameterized in θ given the features $f_1, \ldots, f_k$

$Q_\pi(s,a) = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R(s_k,a_k) | s_0, a_0\right]$ expected discounted reward with policy π from state s and action a

$\pi_Q(s,a) = \frac{\exp(Q(s,a))}{\sum_{a'} \exp(Q(s,a'))}$ softmax policy (randomized)

Assumptions:

- Demonstrations are samples of a softmax, randomized policy.
- This unknown policy satisfies φ with probability higher than a threshold.

negative log-likelihood

all states are in the product automaton

$$\min_{\theta} \quad J^{\text{mle}}(\pi_\theta | M \times \mathcal{A}_\varphi, D) - \mu g(\bar{y})$$

$$\text{s.t.} \quad \text{Bellman equation}(\theta, M \times \mathcal{A}_\varphi, D)$$

penalty for violating the specification

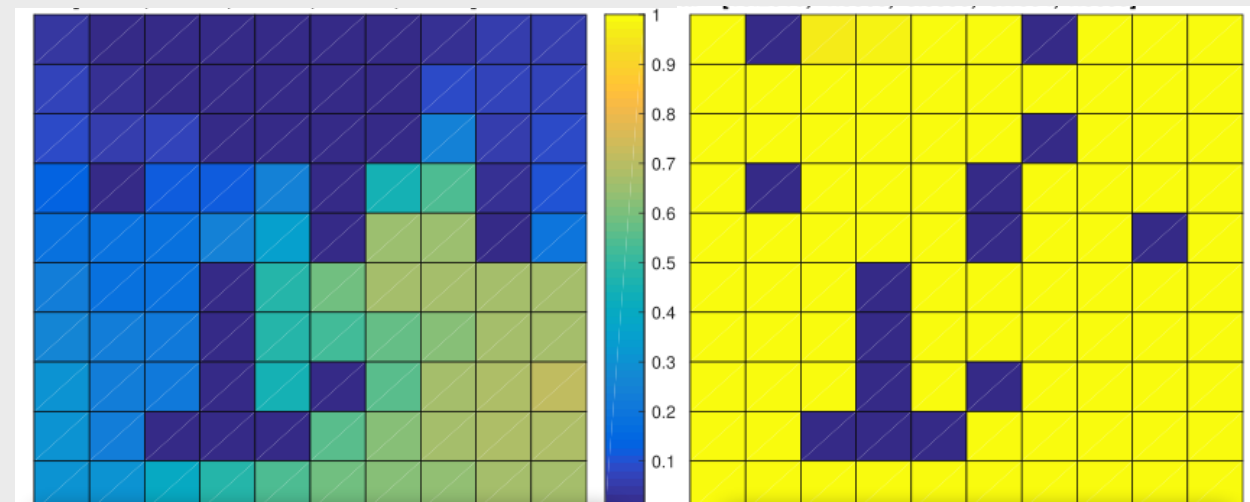
$\bar{y}(s_0, \pi_\theta)$ probability of satisfying the specification by taking policy π_θ from initial state s_0

affine constraints on θ for the fragment of temporal logic considered

strategy

Example

Probability of satisfying the specification for the given initial state:
Poor generalization without side information

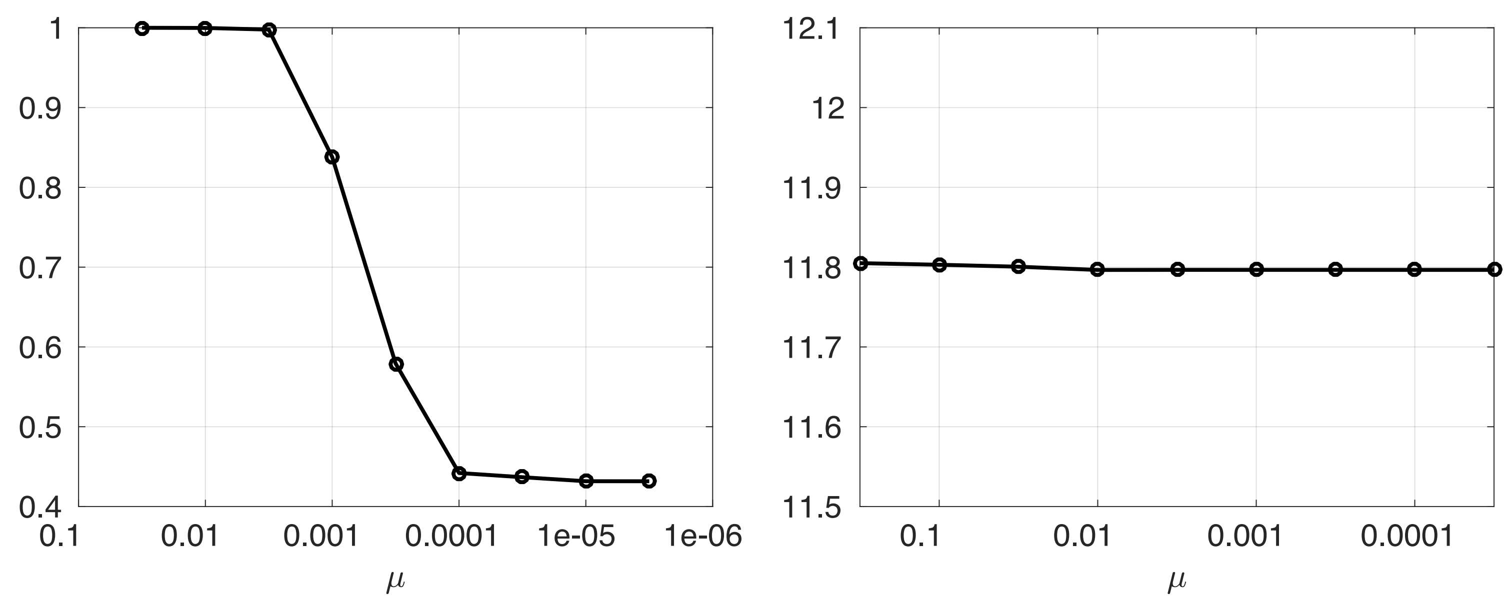


without side information

with side information

Major increase in probability of satisfaction at negligible reduction in likelihood

Minimum probability of satisfying the specification from the initial state



Likelihood of demonstrations

←higher weight for satisfying specification

Outline

Correct-by-construction synthesis of hierarchical control protocols

- formal methods ↔ controls

What would it take to construct the control software of an autonomous system **in an hour—as opposed to days or even weeks**—to deliver a mission?

Verifiable reinforcement learning

- formal methods ↔ learning

How can an autonomous system **learn** how to execute a new mission in an a priori unknown environment **efficiently** and **safely?**

Planning in POMDPs

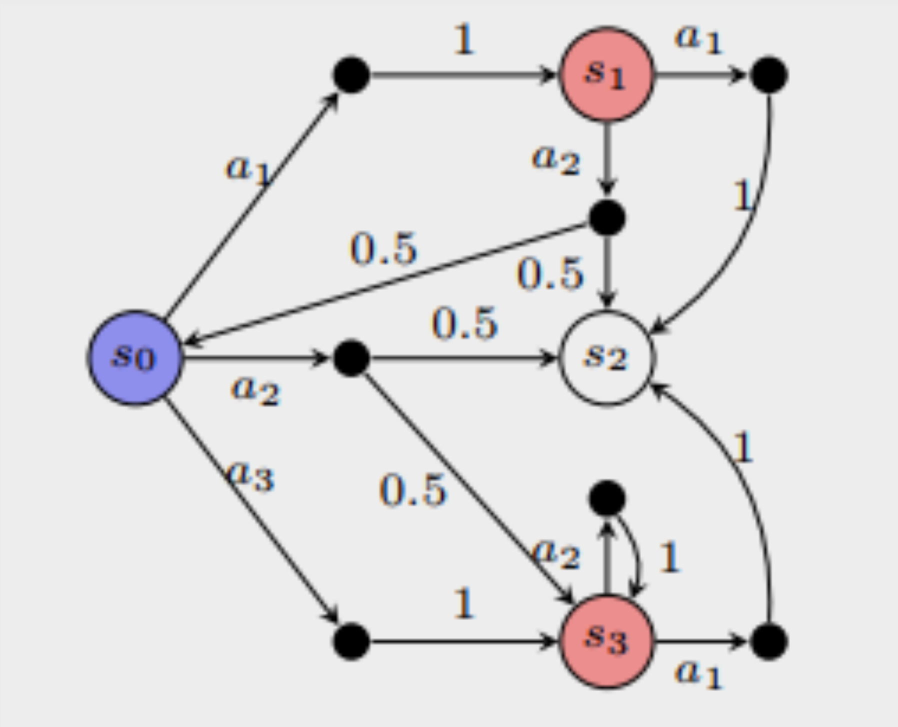
- formal methods ↔ convex optimization
- formal methods ↔ learning

How can we cope with imperfection and/or limitations in run-time information?

A few words on POMDPs

POMDPs — partially observable MDPs: Like MDPs but with limited information

$$\mathcal{M} = (\overset{\text{set of states}}{S},\ \underset{\text{set of actions}}{A},\ \overset{\text{probabilistic transition function}}{P},\ \underset{\text{set of observations}}{Z},\ \overset{\text{observation function}}{O},\ \underset{\text{cost function}}{C})$$



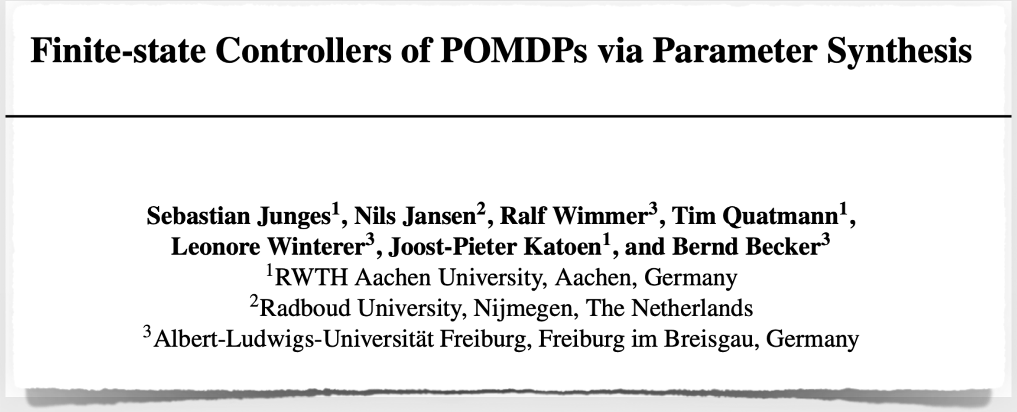
Can distinguish only between the colors of the states

Synthesis in POMDPs is hard.

- Undecidable as infinite memory may be necessary.
- Restriction to finite-memory strategies yields decidable yet still hard problems (and “suboptimality”).
- Finite-memory strategies: Randomized may be better than deterministic.

Computing finite-state controllers for POMDPs by parameter synthesis

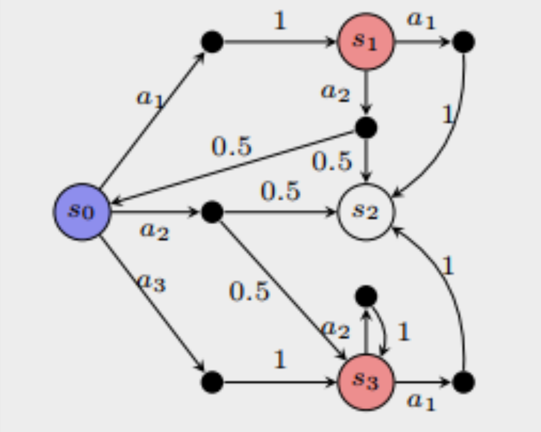
- The set of all finite-state controllers for a POMDP and a fixed memory bound can be represented by a parametric Markov chain (pMC).
- If POMDP states share an observation, the corresponding pMC states will share parameters at their transitions.
- A strategy in the POMDP, corresponds to a parameter instantiation in the pMC.


Finite-state Controllers of POMDPs via Parameter Synthesis

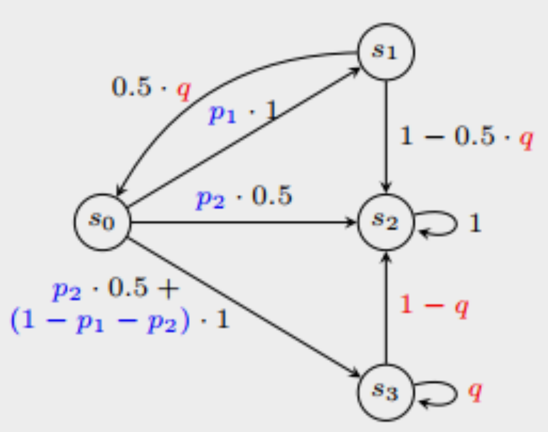
Sebastian Junges[1], Nils Jansen[2], Ralf Wimmer[3], Tim Quatmann[1], Leonore Winterer[3], Joost-Pieter Katoen[1], and Bernd Becker[3]
[1]RWTH Aachen University, Aachen, Germany
[2]Radboud University, Nijmegen, The Netherlands
[3]Albert-Ludwigs-Universität Freiburg, Freiburg im Breisgau, Germany

POMDP under FSC	pMC
states × memory	states
same observation	same parameter
strategy	parameter instantiation
permissive strategy	region of instantiations

Correspondence table between POMDP and pMCs

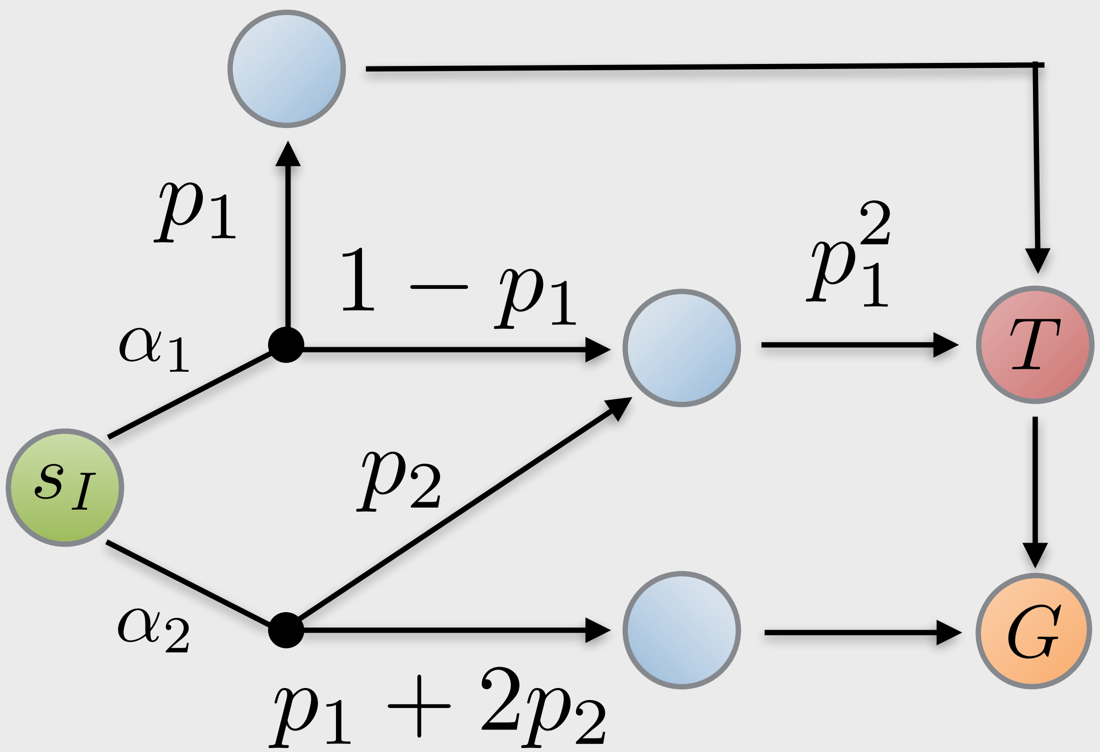


POMDP



Corresponding pMC with Parameters p1,p2 and q

Synthesis in parametric MDPs (pMDPs)



Safety specification
$\varphi = \mathbb{P}_{\leq\lambda}(\Diamond T), \; T \subseteq S$

Performance specification
$\psi = \mathbb{E}_{\leq\kappa}(\Diamond G), \; G \subseteq S$

Objective function $f : V \to \mathbb{R}$

Parameters $p_1, p_2, \ldots, p_n \in V$

Given pMDP $\mathcal{M}$, find a well-defined valuation of parameters and a scheduler $\sigma \in Sched^{\mathcal{M}}$ such that

$$\mathcal{M}^\sigma \models \varphi \wedge \psi$$

and value for objective function $f : V \to \mathbb{R}$ is minimal.

Solution as a "nonlinear program"

Problem variables:

Randomized scheduler:

$\{\sigma^{s,\alpha} \mid s \in S, \alpha \in Act(s)\}$

Probability of reaching T:

$\{p_s \mid s \in S\}$

Expected cost of reaching G:

$\{c_s \mid s \in S\}$

$$\text{minimize} \quad f$$

objective function over parameters

$$\text{subject to}$$

$$p_{s_I} \leq \lambda$$
$$c_{s_I} \leq \kappa$$

safety and performance specifications

$$\forall s \in S. \quad \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} = 1$$

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

well-defined schedulers and parameter instantiations

$$\forall s \in T. \quad p_s = 1$$

$$\forall s \in S \setminus T. \quad p_s = \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \cdot \sum_{s' \in S} \mathcal{P}(s, \alpha, s') \cdot p_{s'}$$

safety probability computation

$$\forall s \in G. \quad c_s = 0$$

expected performance computation

$$\forall s \in S \setminus G. \quad c_s = \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \cdot \left(c(s, \alpha) + \sum_{s' \in S} \mathcal{P}(s, \alpha, s') \cdot c_{s'} \right)$$

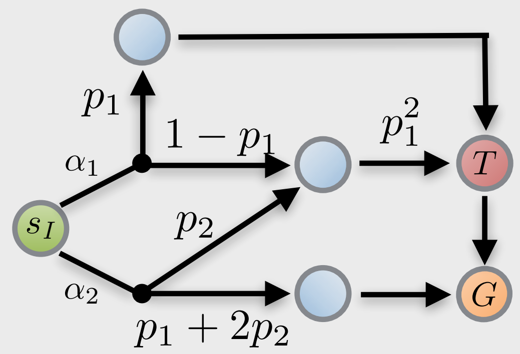
A useful observation

non-negative-valued variable

signomials

non-negative-valued variable

$$p_s = \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \cdot \sum_{s' \in S} \mathcal{P}(s, \alpha, s') \cdot p_{s'}$$



strictly positive

real

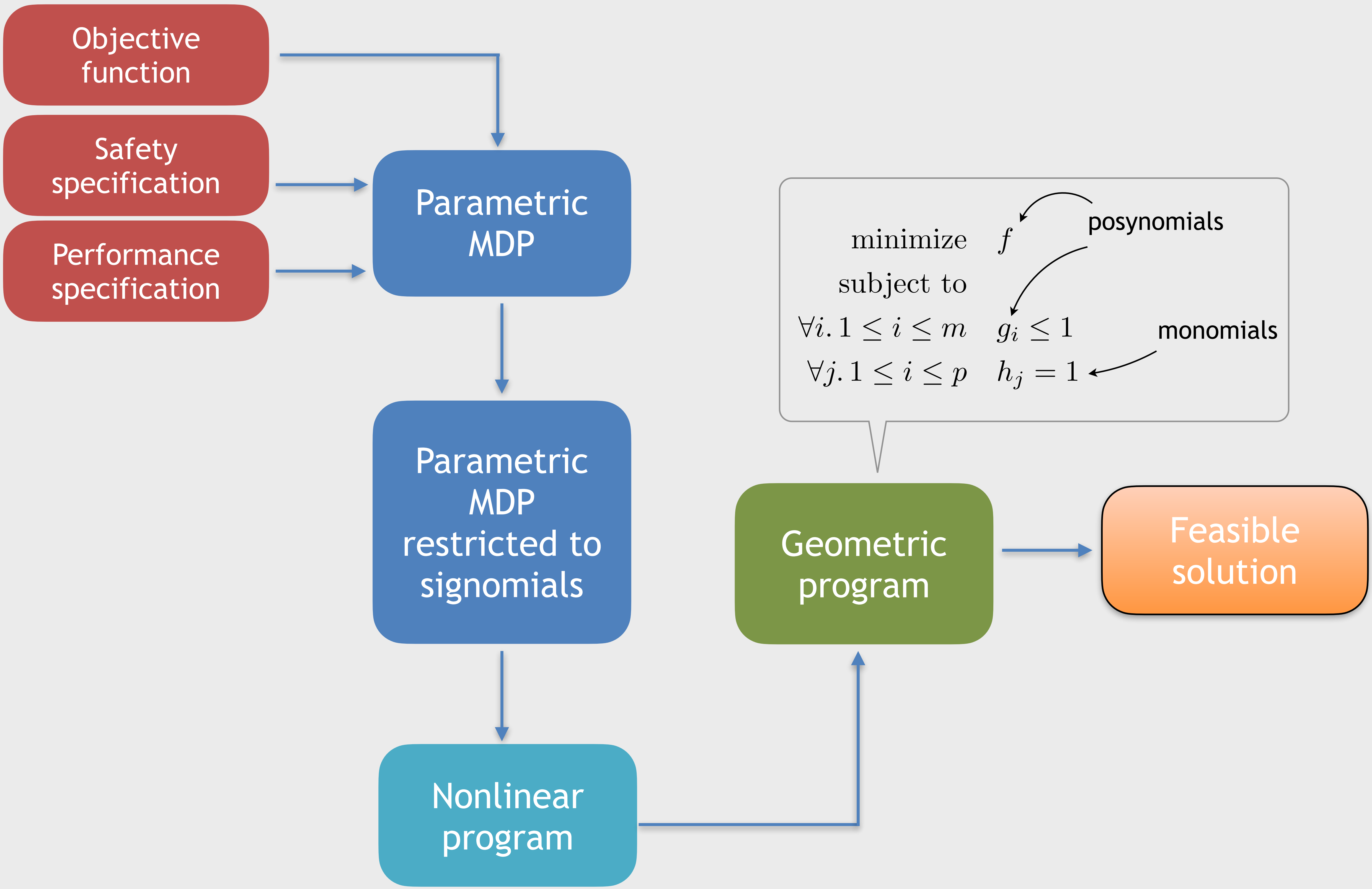
$$f = \sum_{k=1}^{K} c_k \cdot \underbrace{x_1^{a_{1k}} \cdots x_n^{a_{nk}}}_{\text{monomial}}$$

> 0: **posynomial**

no restriction: **signomial**

Question: Can we somehow exploit this structure and solve the parameter synthesis problem as a convex optimization problem (maybe bunch of them)?

Workflow


Objective function
Safety specification
Performance specification
Parametric MDP
Parametric MDP restricted to signomials
Nonlinear program
Geometric program
Feasible solution
minimize f
subject to
$\forall i.\, 1 \le i \le m \quad g_i \le 1$
$\forall j.\, 1 \le i \le p \quad h_j = 1$
posynomials
monomials

Convexification: relaxation + lifting

$$p_s = \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \cdot \sum_{s' \in S} \mathcal{P}(s,\alpha,s') \cdot p_{s'}$$

still signomials
(not geometric
program yet)

$$\frac{\sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \cdot \sum_{s' \in S} \mathcal{P}(s,\alpha,s') \cdot p_{s'}}{p_s} \leq 1$$

(assuming $p_s \neq 0$)

upper bound
on actual probability

lifting: introduce
new variables

$$\bar{p}_{s,\alpha,\bar{s}}$$

$$\mathcal{P}(s,\alpha,\bar{s}) = 1 - \sum_{s' \in S \setminus \{\bar{s}\}} \mathcal{P}(s,\alpha,s')$$

signomial

posynomials

Turns out that all but one transition probability can be written as a posynomial.

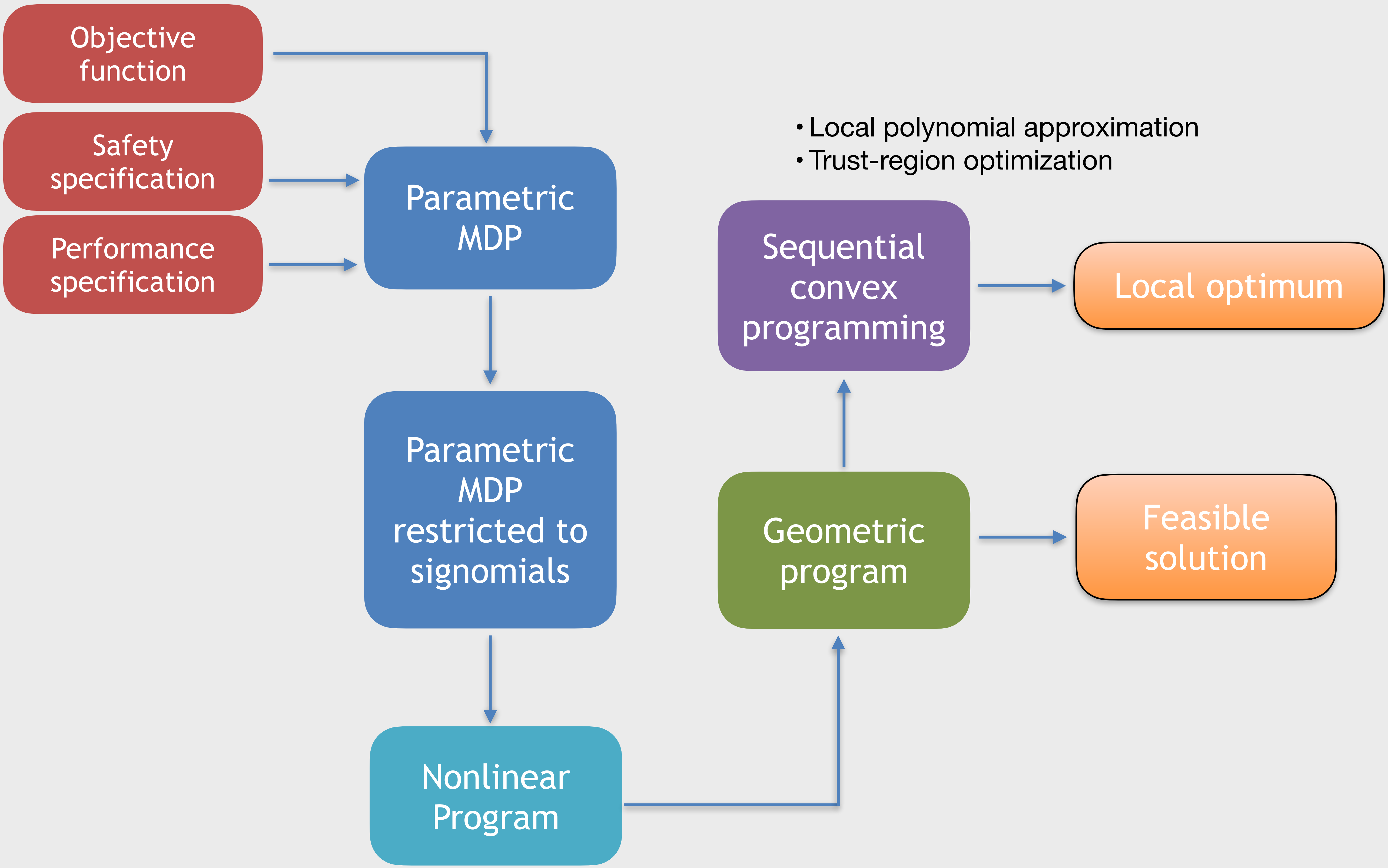
Geometric program (with relaxation tightening)

$$
\begin{aligned}
\text{minimize} \quad & \sum_{p \in V} \frac{1}{p} + \sum_{\bar{p} \in L} \frac{1}{\bar{p}} + \sum_{s \in S, \alpha \in Act(s)} \frac{1}{\sigma_{s,\alpha}} \\
\text{subject to} \quad & \\
& \frac{p_{s_I}}{\lambda} \leq 1 \\
& \frac{c_{s_I}}{\kappa} \leq 1 \\
\forall s \in S. \quad & \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \leq 1 \\
\forall s \in S \, \forall \alpha \in Act(s). \quad & \sum_{s' \in S} \mathcal{P}(s, \alpha, s') \leq 1 \\
\forall s \in S \setminus T. \quad & \frac{\sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \cdot \sum_{s' \in S} \mathcal{P}(s, \alpha, s') \cdot p_{s'}}{p_s} \leq 1 \\
\forall s \in S \setminus G. \quad & \frac{\sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \cdot \Big(c(s, \alpha) + \sum_{s' \in S} \mathcal{P}(s, \alpha, s') \cdot c_{s'} \Big)}{c_s} \leq 1
\end{aligned}
$$

regularization

Theorem: The solution to the geometric program gives a well-defined scheduler and parameter instantiation. But it may be sub-optimal.

Workflow



Objective function
Safety specification
Performance specification
Parametric MDP
• Local polynomial approximation
• Trust-region optimization
Sequential convex programming
Local optimum
Parametric MDP restricted to signomials
Geometric program
Feasible solution
Nonlinear Program

Compare against benchmarks

Alternative tools for optimization TO even in the smallest instances

proposed method

only feasibility

Benchmark	#states	#par	specs	MOSEK (s)		Z3
BRP (pMC)	5382	2	EC, $\mathbb{P}$, $*$	23.17	(6.48)	–
	112646	2	EC, $\mathbb{P}$, $*$	3541.59	(463.74)	–
	112646	4	EC, $\mathbb{P}$, $*$	4173.33	(568.79)	–
	5382	2	EC, $\mathbb{P}$	3.61		904.11
	112646	2	EC, $\mathbb{P}$	479.08		*TO*
NAND (pMC)	4122	2	EC, $\mathbb{P}$, $*$	14.67	(2.51)	–
	35122	2	EC, $\mathbb{P}$, $*$	1182.41	(95.19)	–
	4122	2	EC, $\mathbb{P}$	1.25		1.14
	35122	2	EC, $\mathbb{P}$	106.40		11.49
BRP (pMDP)	5466	2	EC, $\mathbb{P}$, $*$	31.04	(8.11)	–
	112846	2	EC, $\mathbb{P}$, $*$	4319.16	(512.20)	–
	5466	2	EC, $\mathbb{P}$	4.93		1174.20
	112846	2	EC, $\mathbb{P}$	711.50		*TO*
CONS (pMDP)	4112	2	EC, $\mathbb{P}$, $*$	102.93	(1.14)	–
	65552	2	EC, $\mathbb{P}$, $*$	*TO*		–
	4112	2	EC, $\mathbb{P}$	6.13		*TO*
	65552	2	EC, $\mathbb{P}$	1361.96		*TO*

More benchmarks

(using a related but different method)

parametric MCs

POMDPs

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

Synthesis in pMDPs: A Tale of 1001 Parameters

Murat Cubuktepe[1], Nils Jansen[2], Sebastian Junges[3], Joost-Pieter Katoen[3], Ufuk Topcu[1]

[1] The University of Texas at Austin, Austin, TX, USA*
[2] Radboud University, Nijmegen, The Netherlands
[3] RWTH Aachen University, Aachen, Germany**

Abstract. This paper considers parametric Markov decision processes (pMDPs) whose transitions are equipped with affine functions over a finite set of parameters. The synthesis problem is to find a parameter valuation such that the instantiated pMDP satisfies a (temporal logic) specification under all strategies. We show that this problem can be formulated as a quadratically-constrained quadratic program (QCQP) and is non-convex in general. To deal with the NP-hardness of such problems, we exploit a convex-concave procedure (CCP) to iteratively obtain local optima. An appropriate interplay between CCP solvers and probabilistic model checkers creates a procedure — realized in the tool PROPheSY — that solves the synthesis problem for models with thousands of parameters.

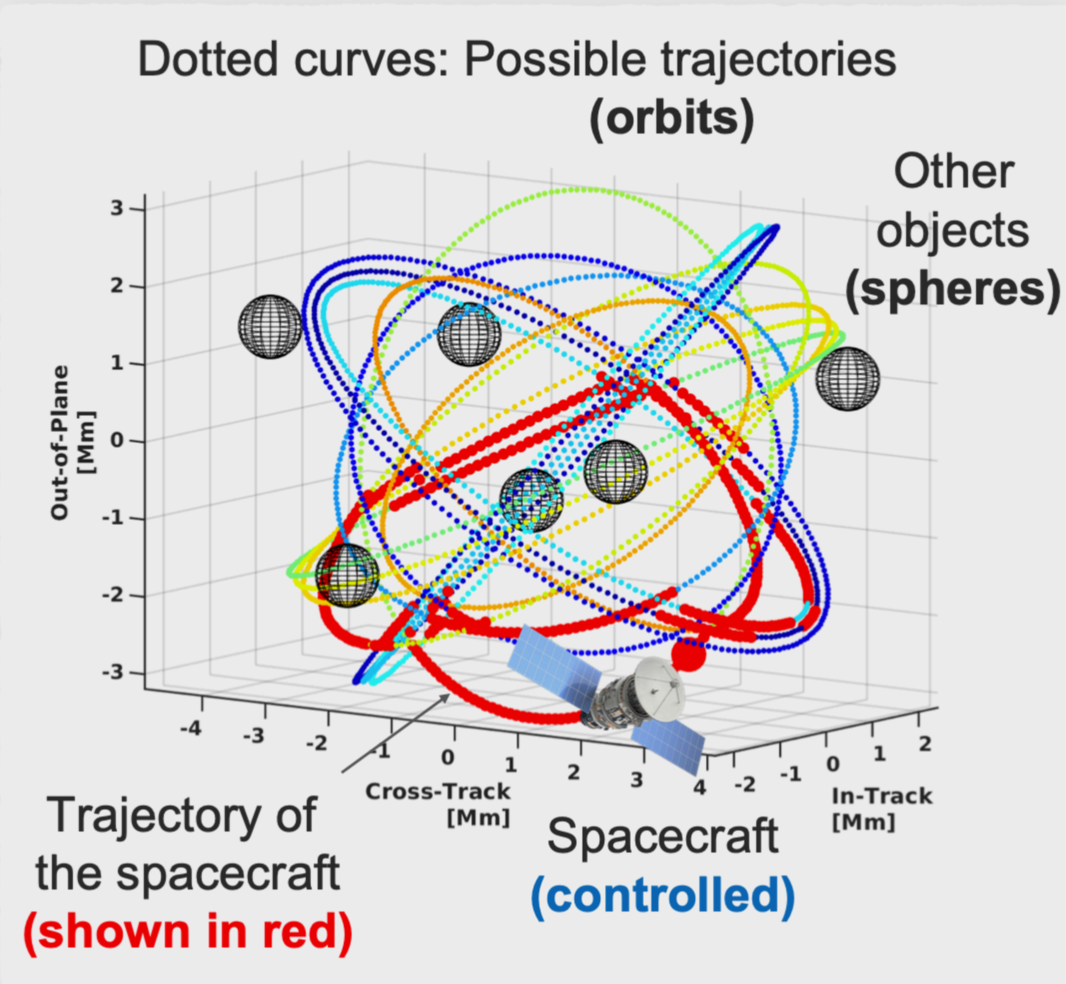
1 Introduction

The parameter synthesis problem. Probabilistic model checking concerns the automatic verification of models such as Markov decision processes (MDPs). Unremitting improvements in algorithms and efficient tool implementations [14,22,26] have opened up a wide variety of applications, most notably in dependability, security, and performance analysis as well as systems biology. However, at early development stages, certain system quantities such as fault or reaction rates are often not fully known. This lack of information gives rise to parametric models where transitions are functions over real-valued parameters [12,21,27], forming symbolic descriptions of (uncountable) families of concrete MDPs. The *parameter synthesis problem* is: Given a finite-state parametric MDP, find a parameter

parametric MDPs

Problem			Info				PSO			SMT	CCP		
Set	Inst	Spec	States	Act	Trans.	Par.	tmin	tmax	**tavg**	**t**	**t**	solv	iter
BRP	4,128	$\mathbb{P}_{\leq 0.1}$	17131	17396	23094	**2**	45	47	46	TO	**39**	33%	4
Coin	32	$\mathbb{E}_{\leq 500}$	4112	6160	7692	**2**	117	119	**118**	TO	TO	-	-
CoinX	32	$\mathbb{E}_{\leq 210}$	16448	24640	30768	**2**	1196	1222	1208	TO	**32**	78%	3
Zeroconf	1	$\mathbb{P}_{\geq 0.99}$	31402	55678	70643	**3**	18	19	**19**	TO	79	82%	2
CSMA	2,4	$\mathbb{E}_{\leq 69.3}$	7958	7988	10594	**26**	n.s.	n.s.	n.s.	TO	**79**	86%	10
Virus	-	$\mathbb{E}_{\leq 10}$	809	3371	6741	**18**	113	113	113	TO	**13**	76%	4
Wlan	0	$\mathbb{E}_{\leq 580}$	2954	3972	5202	**15**	n.s.	n.s.	n.s.	TO	**7**	72%	2

Uncertain POMDPs, through a more visual example



Spacecraft motion planning:

Switching between orbits is possible if the orbits are close to each other, but costs fuel.

Uncertain POMDP: Partial observability over the current position of spacecraft, uncertainty on the location of other objects and operator

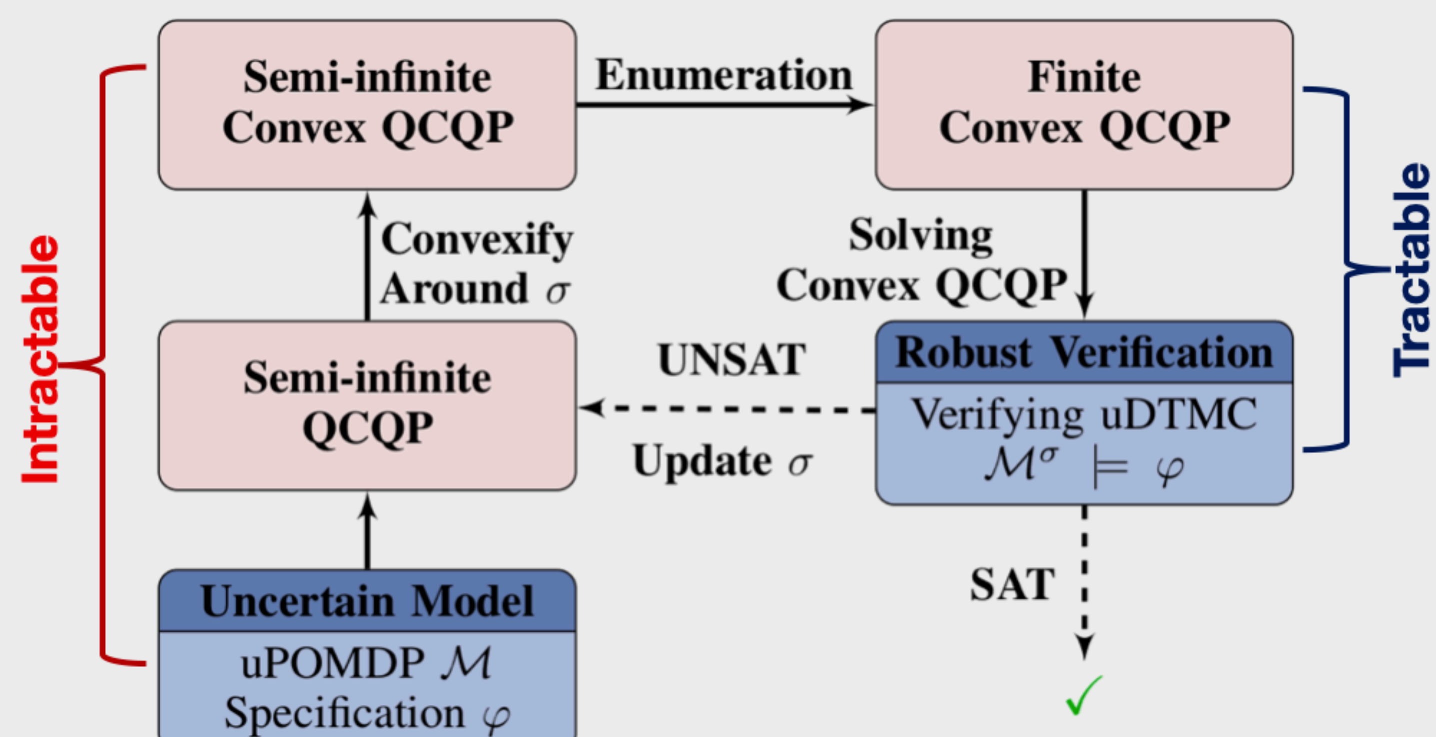
induced uncertain Markov chain

$$\mathcal{M}_\sigma^{\mathcal{P}} \models \varphi \quad \text{for all } P \in \mathcal{P}$$

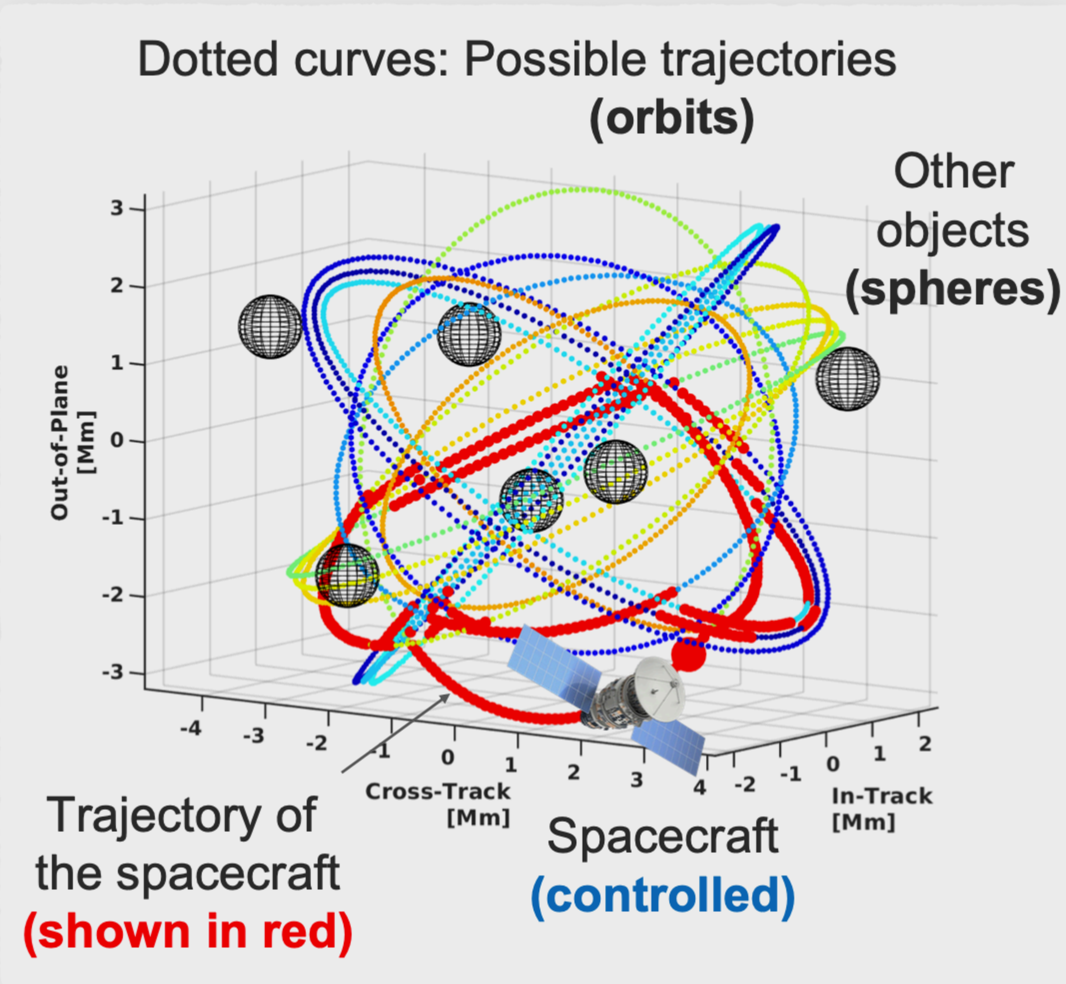
satisfies the specification

transition function

uncertainty set



Uncertain POMDPs, through a more visual example



Spacecraft motion planning:

Switching between orbits is possible if the orbits are close to each other, but costs fuel.

Uncertain POMDP: Partial observability over the current position of spacecraft, uncertainty on the location of other objects and operator

induced uncertain Markov chain

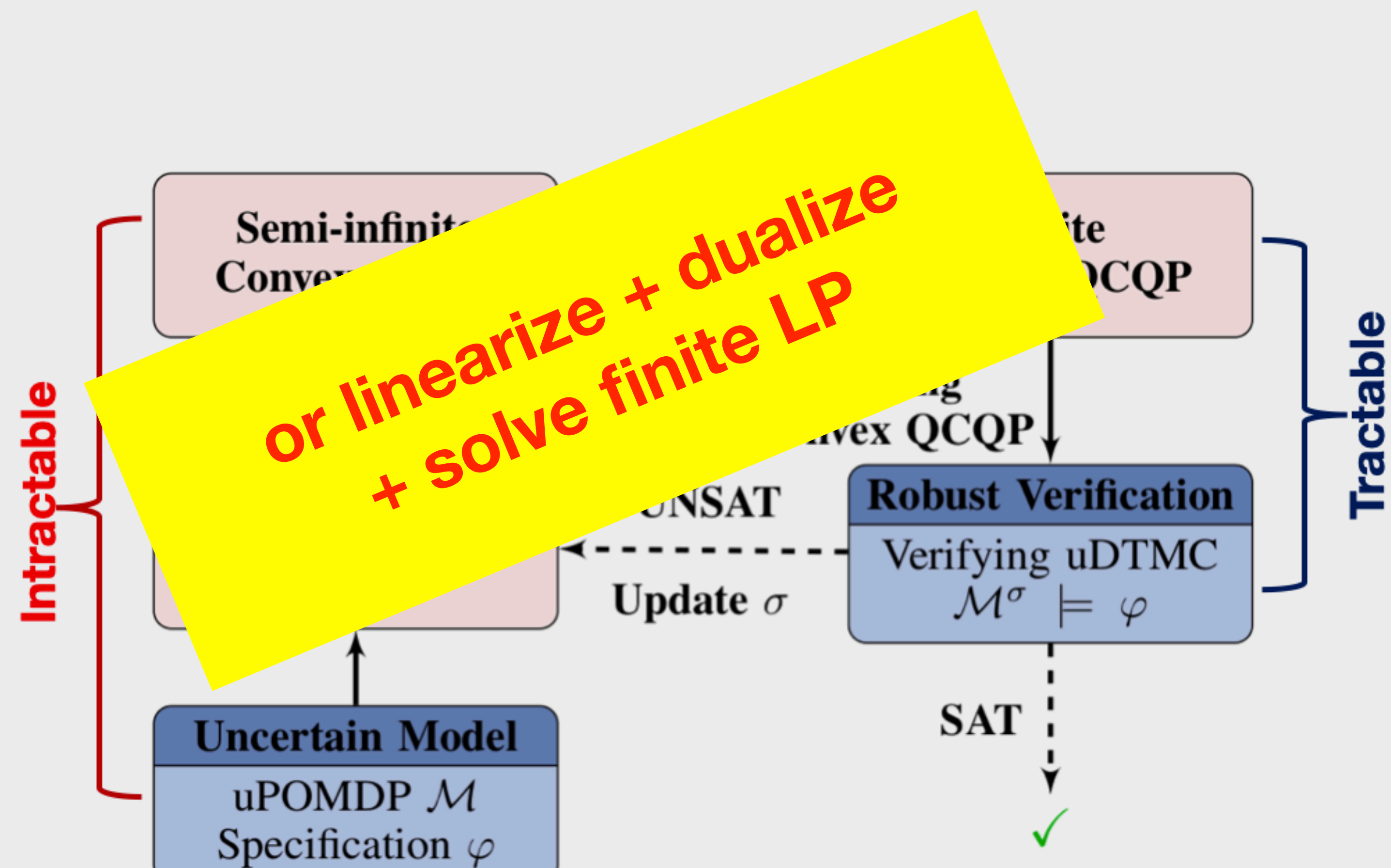
$$\mathcal{M}_\sigma^{\mathcal{P}} \models \varphi$$

satisfies the specification

transition function

for all $P \in \mathcal{P}$

uncertainty set



Outline

Correct-by-construction synthesis of hierarchical control protocols

- **formal methods ↔ controls**

What would it take to construct the control software of an autonomous system **in an hour—as opposed to days or even weeks**—to deliver a mission?

Verifiable reinforcement learning

- **formal methods ↔ learning**

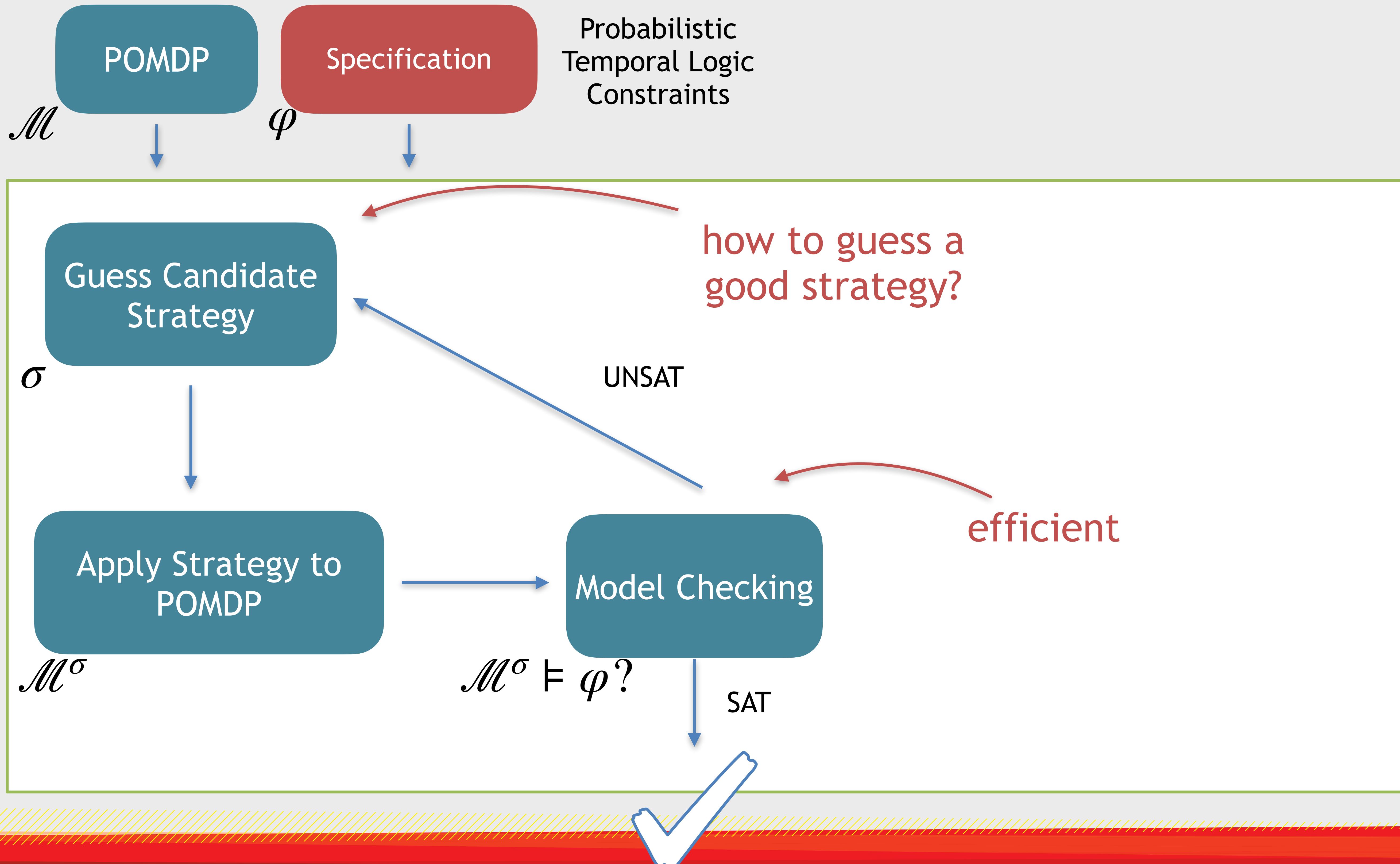
How can an autonomous system **learn** how to execute a new mission in an a priori unknown environment **efficiently** and **safely?**

Planning in POMDPs

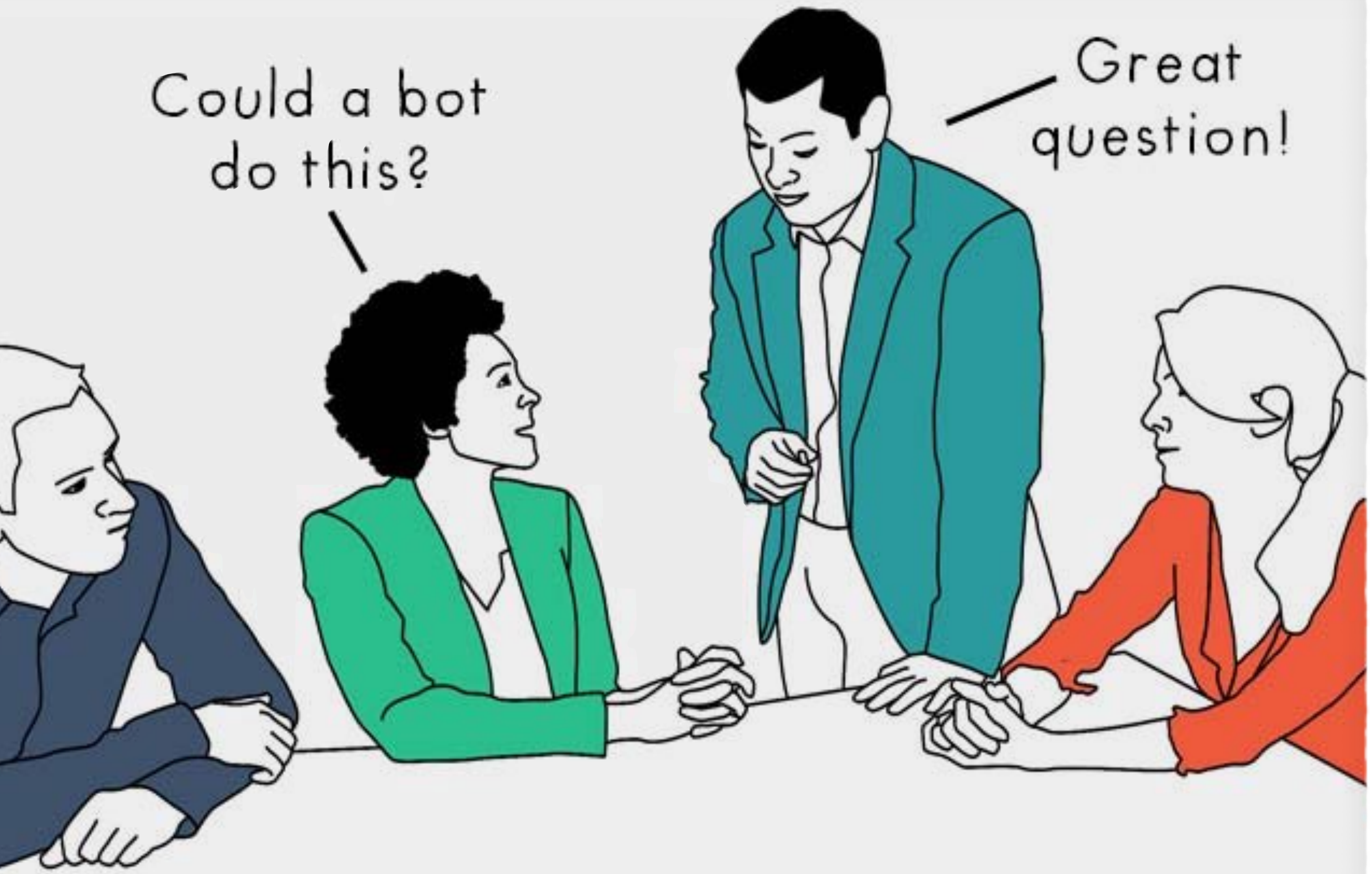
- **formal methods ↔ convex optimization**
- **formal methods ↔ learning**

How can we cope with imperfection and/or limitations in run-time information?

Synthesis is hard. Guess a strategy and verify!

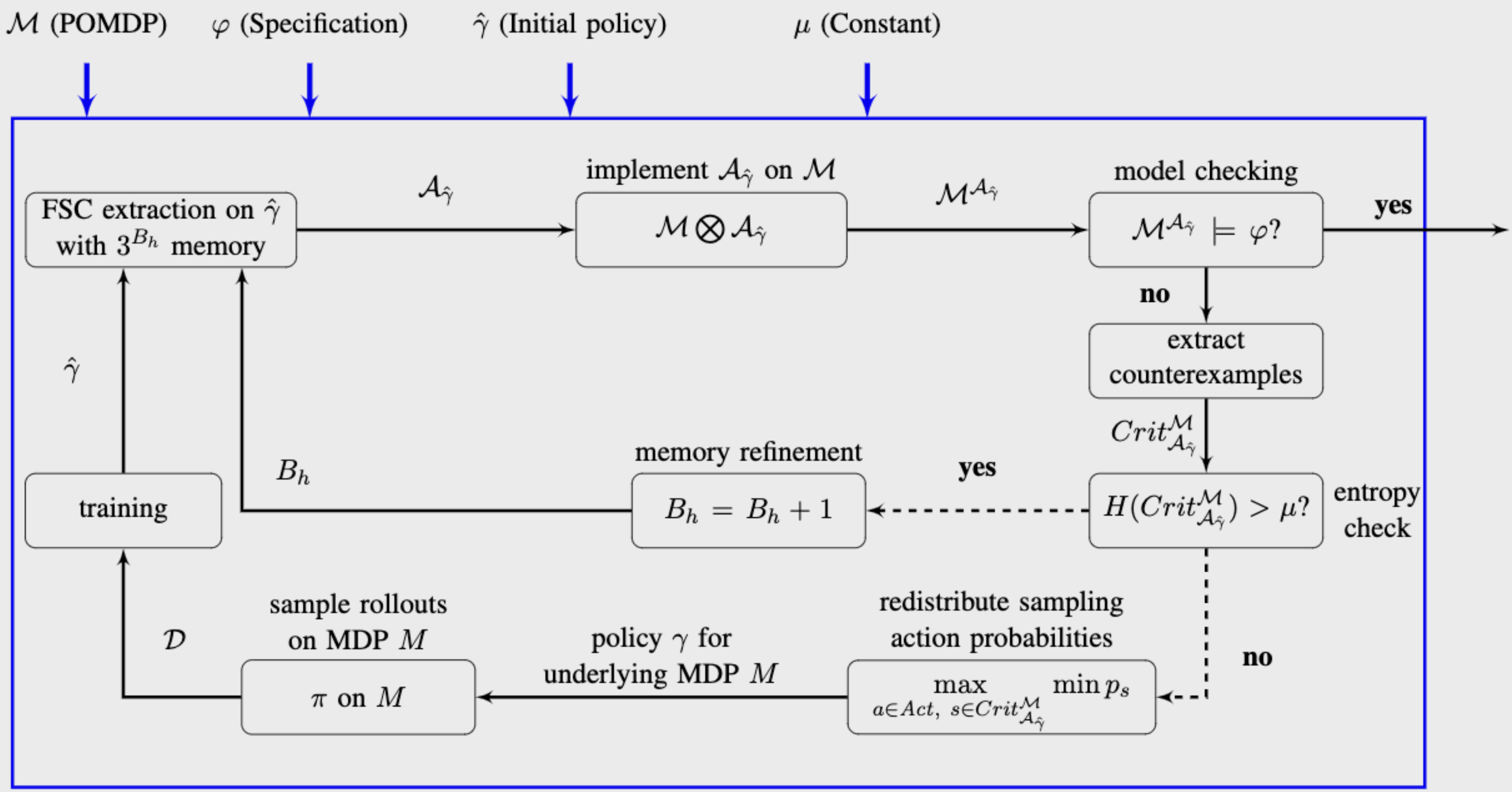


How to guess a "good" strategy?

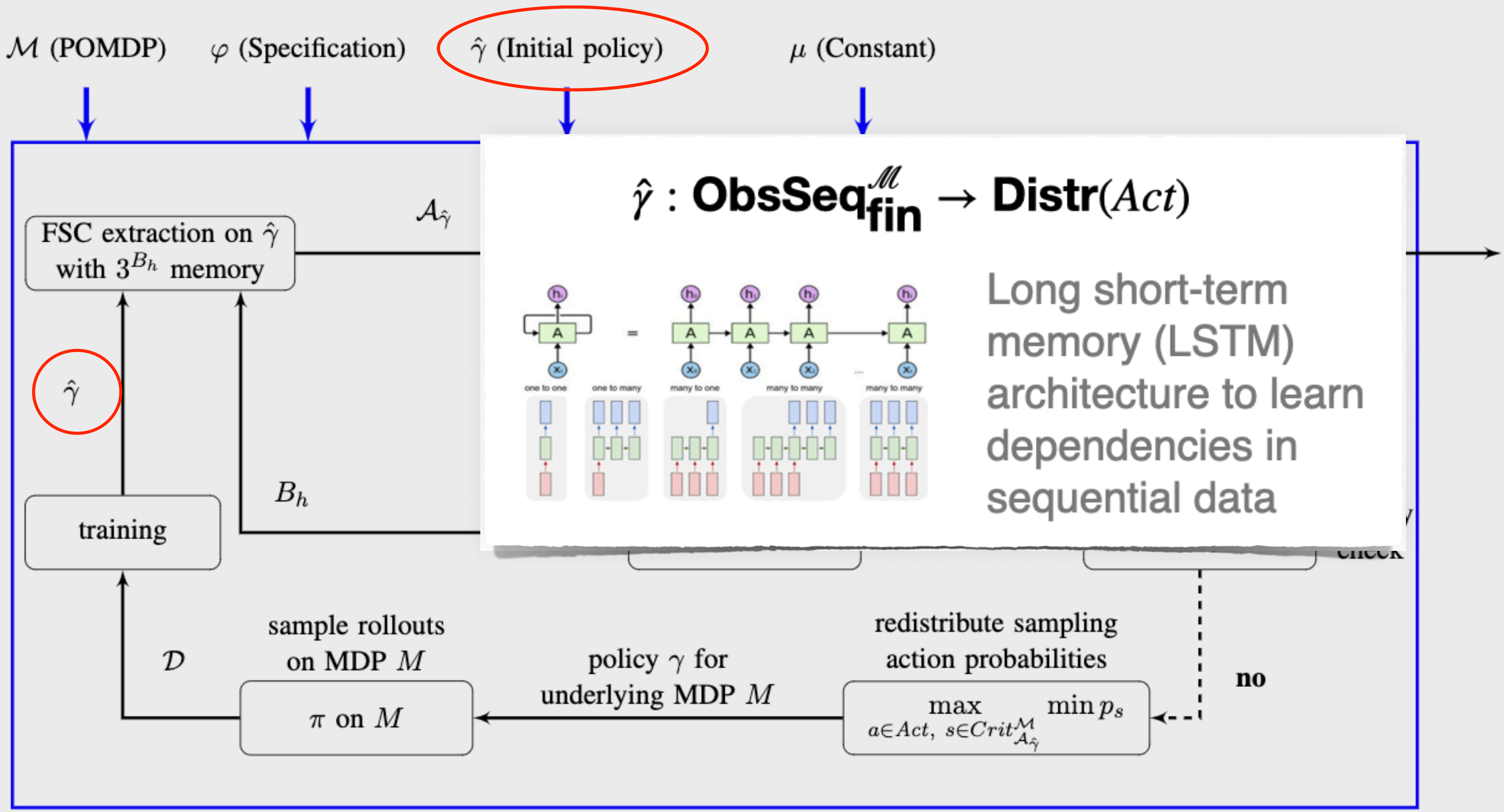


FTFY

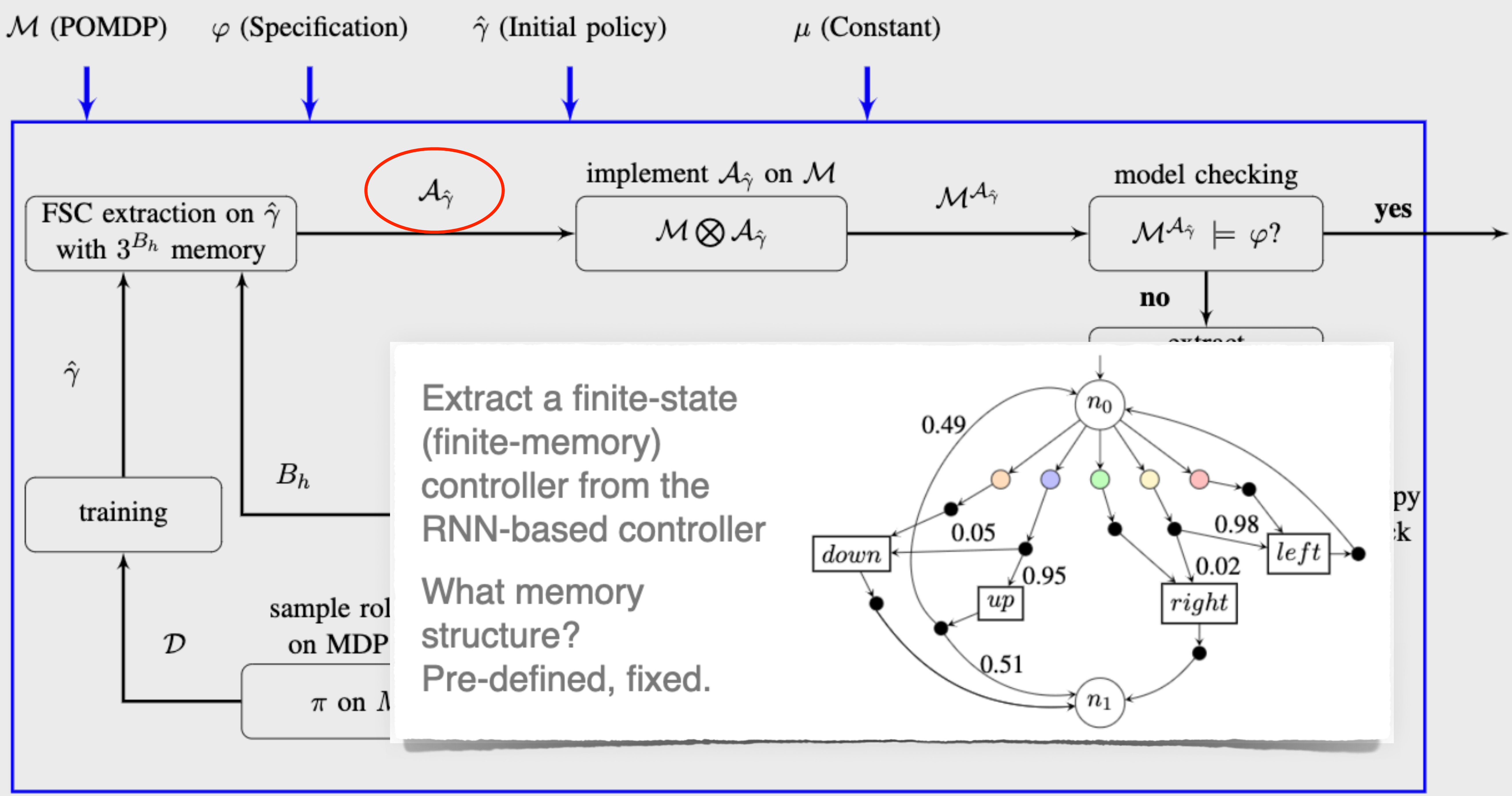
Learn a guess and improve (with help from verification).



Learn a guess and improve (with help from verification).



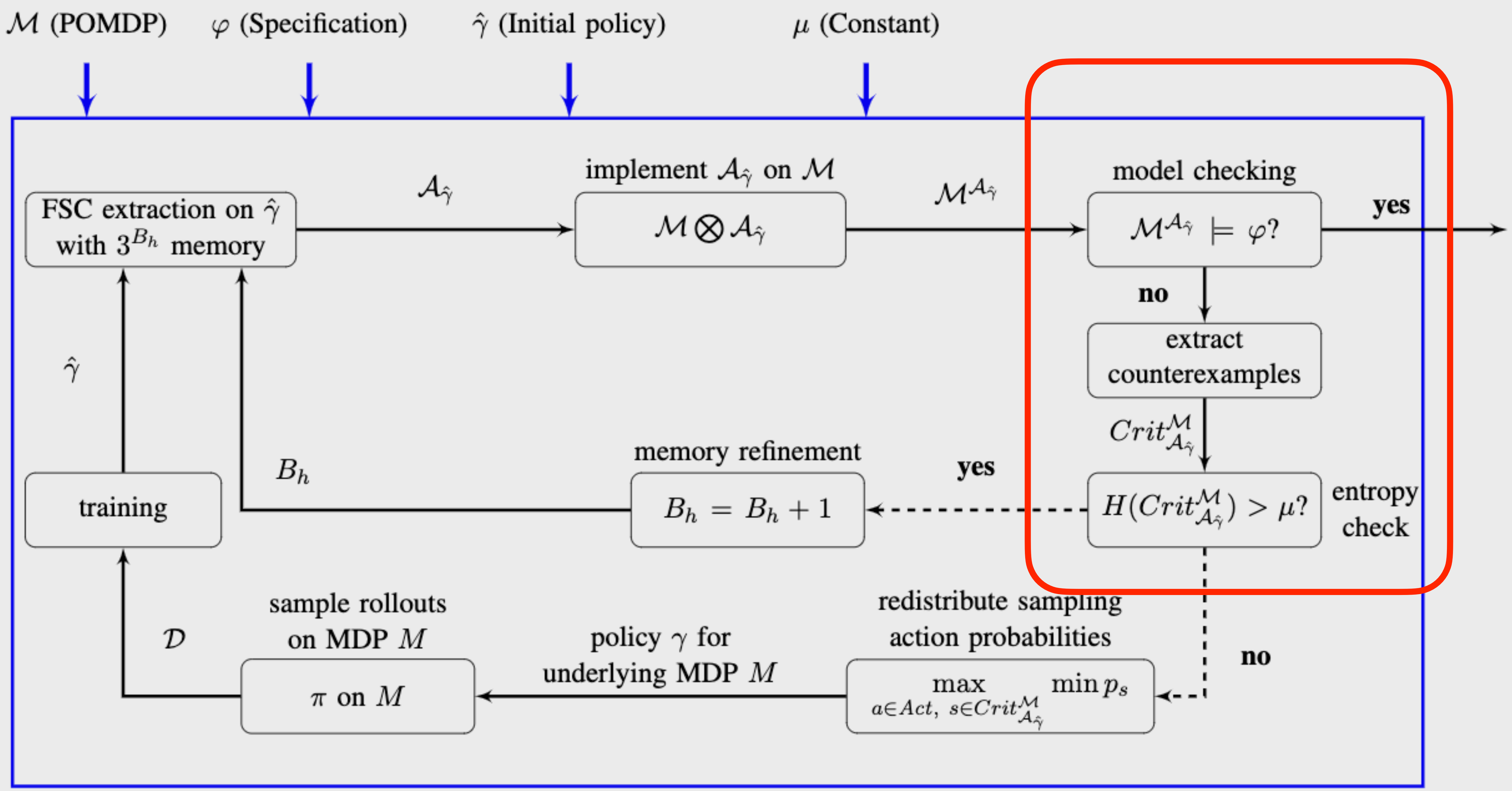
Learn a guess and improve (with help from verification).



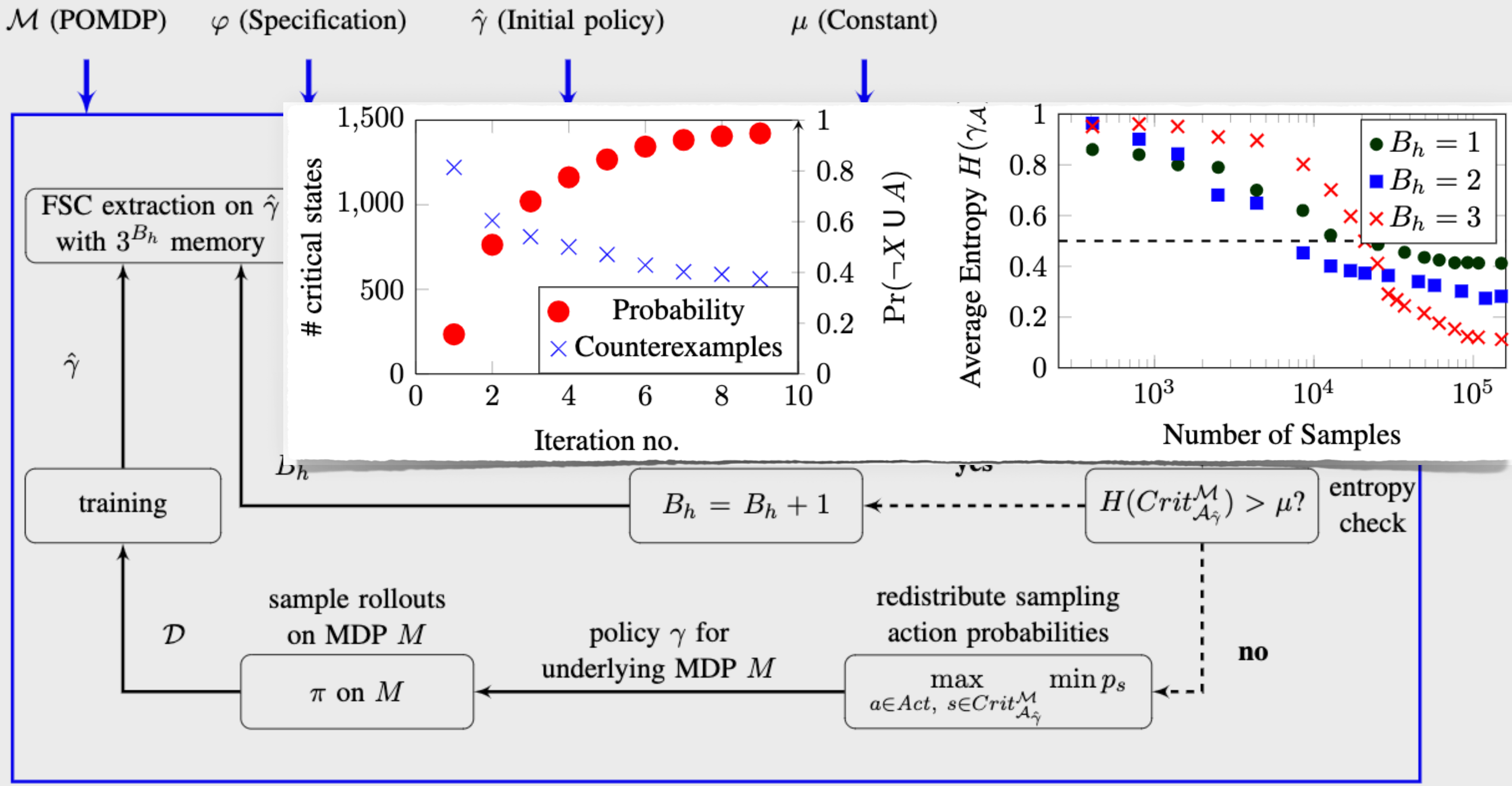
Extract a finite-state (finite-memory) controller from the RNN-based controller

What memory structure?
Pre-defined, fixed.

Learn a guess and improve (with help from verification).



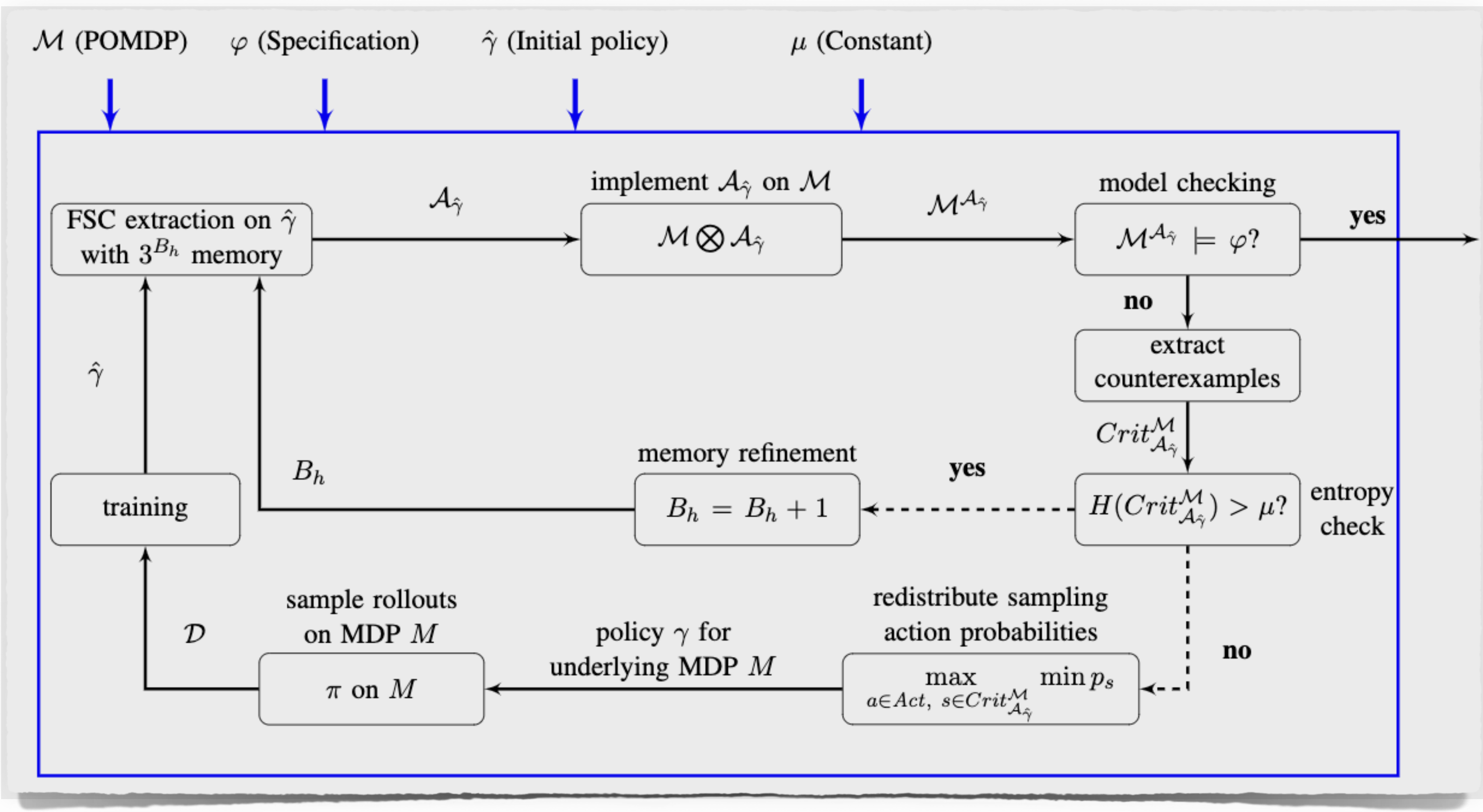
Learn a guess and improve (with help from verification).



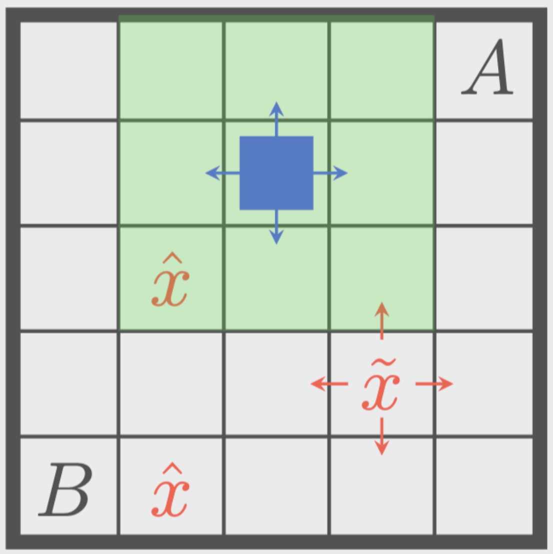
What do we get at the end?

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

Not complete, as we may never find a feasible strategy.
Obviously, expected!



Numerical examples with LTL constraints

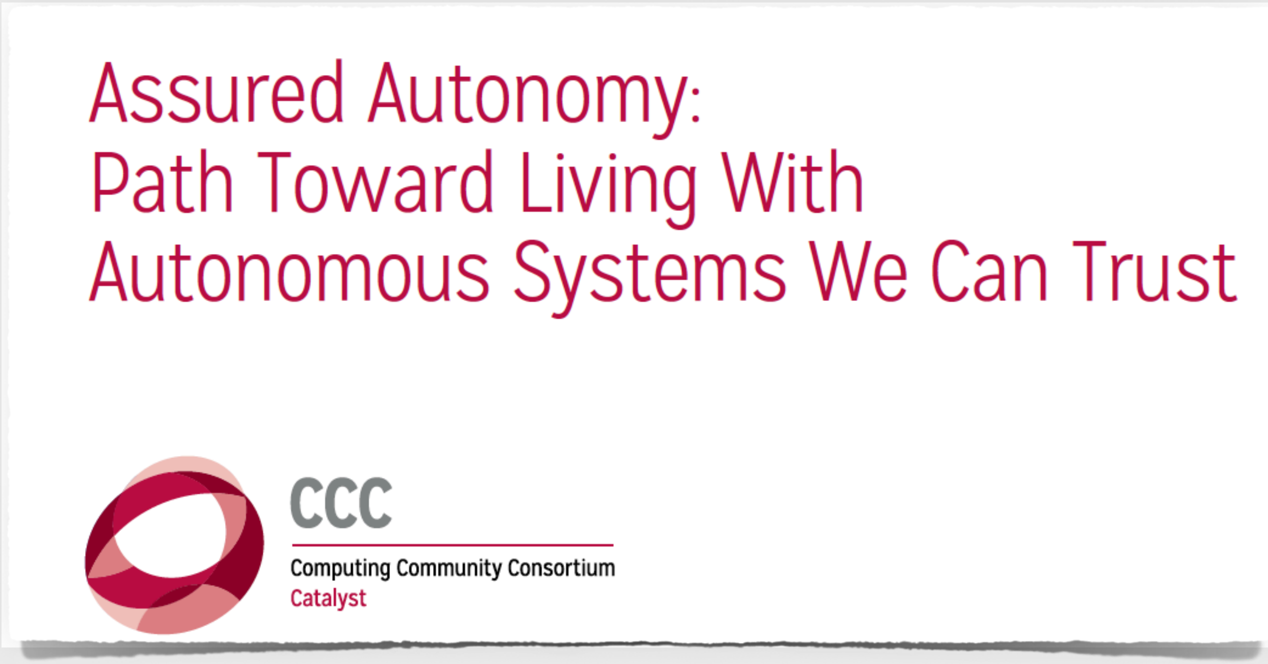


Problem	$\lvert S \rvert$	$\lvert Act \rvert$	$\lvert Z \rvert$
Navigation (c)	c^4	4	256
Delivery (c)	c^2	4	256
Slippery (c)	c^2	4	256
Maze(c)	$3c+8$	4	7
Grid(c)	c^2	4	2
RockSample$[4,4]$	257	9	2
RockSample$[5,5]$	801	10	2
RockSample$[7,8]$	12545	13	2

			RNN-based Synthesis		PRISM-POMDP	
Problem	States	Type, φ	Res.	Time (s)	Res.	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

Numerical examples on standard POMDP benchmarks

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



Assured Autonomy:
Path Toward Living With
Autonomous Systems We Can Trust

CCC
Computing Community Consortium
Catalyst

Some common comments:

- No assurance = no useful autonomy
- Diverse set of vulnerabilities
- Open world
- Interdisciplinary approaches needed, **not** as an afterthought

1. Safety and verification
2. Security
3. Certification and regulation
4. Human-system integration and trust
5. Privacy
6. Ethics
7. Societal impacts
8. Governance and policy
9. Education