MDP Modeling and Interaction with pyRDDLGym Lab

Ayal Taitler

ICAPS Summer School

June 23, 2026

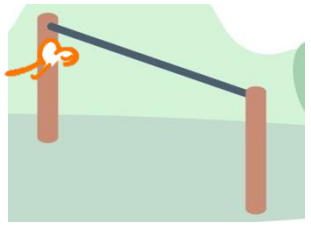
אוניברסיטת בן-גוריון בנגב
Ben-Gurion University of the Negev

About Me

- I’ve completed my **PhD** in 2022 in the Technion (IL) with Prof. Erez Karpas,
  - I did a **postdoc** at UofT (CA) with Prof. Scott Sanner,
    - I am a **faculty** member in BGU (IL) since late 2024, where I direct the Decision-making, Robotics and Intellilgent Vehicles lab (DRIVE)

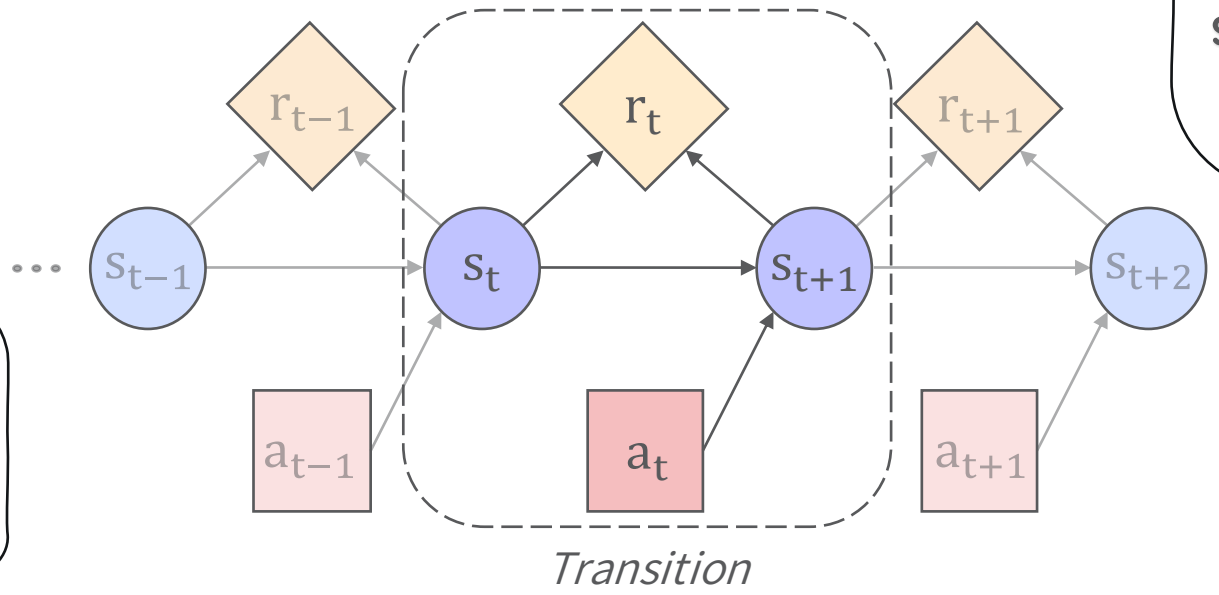
- In a nutshell, I work at the intersection of Control, Planning, and Reinforcement learning
  - Industrial orientation: Continuous and mixed-discrete continuous processes.
  - Time matters: Temporal and Hybrid Planning
  - Bluring the lines between stochastic planning and RL

The Modeling Setting



Formally,

$\langle S, A, R, T, \gamma \rangle$



The Controlled Process

Markov Decision Process (MDP)

Objective:

- Find a *PLAN*
  $a_{0:H} = (a_0, a_1, \ldots a_H)$
- Find a *POLICY*
  $a_t = \pi(s_t)$

Such That

$R = \sum_{t=0}^{H} r_t$ is maximized

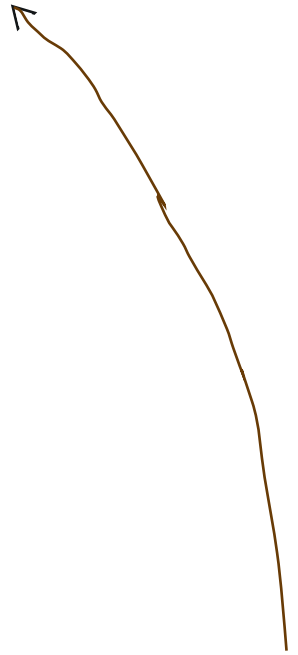
As many bones as possible

Joint work with Scott Sanner and Michael Gimelfarb

**“Language shapes the way we think,
and determines what we can think about.”**

– Benjamin Lee Whorf

- **STRIPS**
  - Propositions, actions, initial state and goal state
  - Everything is grounded
- **PDDL**
  - Similar to STRIPS but lifted
- **FDR\SAS**
  - Grounded, multi-valued variables
- **PPDDL**
  - Like PDDL but with non-deterministic actions effect
- **RDDL**



Why do we need another language?

Joint work with Scott Sanner and Micheal Gimelfarb

RDDL Components

There are several parts to a RDDL problem

- **pvariables** block
  - All the variables of the problem – think about it as where you define S, A and constants.
- **cpfs** block
  - The transition function, this is where you define $s' = f(s, a), s \in S, a \in A$
- **Reward**

ALL are lifted!

Grounding is done in an instance file

We will write RDDL in the lab

And aditional two

- **Constaints** blocks
  - There are two actually, *state-invariants* and *action-preconditions*
- **Terminal** conditions

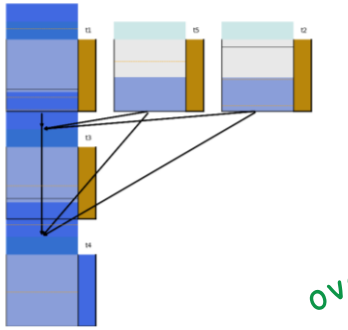
International Planning Competition 2023

With Scott Sanner & Micheal Gimelfarb

ENCOURAGE RESEARCH ON HYBRID METHODS: MODEL-BASED PLANNING & MODEL-FREE LEARNING

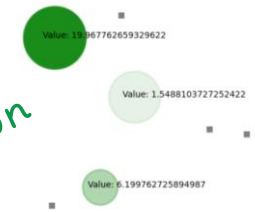
SET A NEW REAL-WORLD PROBLEM SET AS THE DE-FACTO BENCHMARK FOR DECISION-MAKING RESEARCH

steady state



Reservoir Control

over-subscription planning

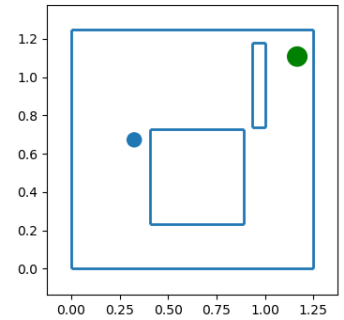


Mars Rover

40K actions!

GOOGLE

Recommender System



Goal oriented

Race Car

OpenAI Gym: A Standard Interface for RL* Environments

What is Gym?

- Standard MDP interaction interface
- *Decouples* agent ⇔ environment
- Enables reproducibility

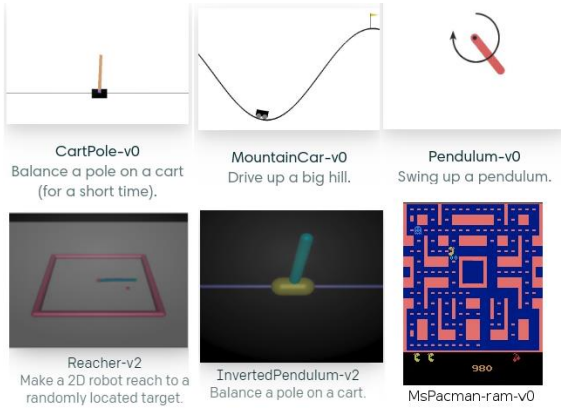
Core:

$$s', r, done = env.step(a)$$

Gym = interface, not algorithm

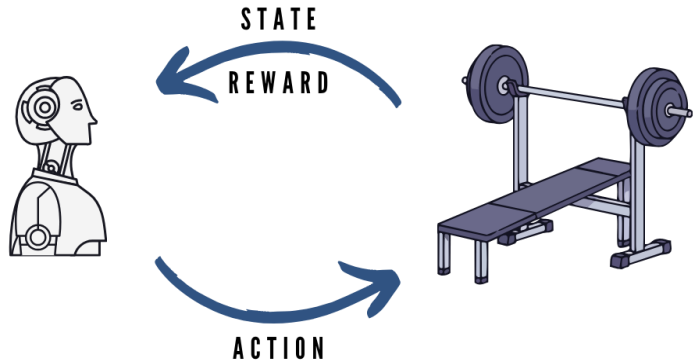
```
import gym
from RLagent import PPOagent

env = gym.make("CartPole-v1")
s = env.reset()
agent = PPOagent()
for _ in range(1000):
   a = agent.sample_action(s)
   s, r, done, _, _ = env.step(a)
   if done:
      s = env.reset()
```



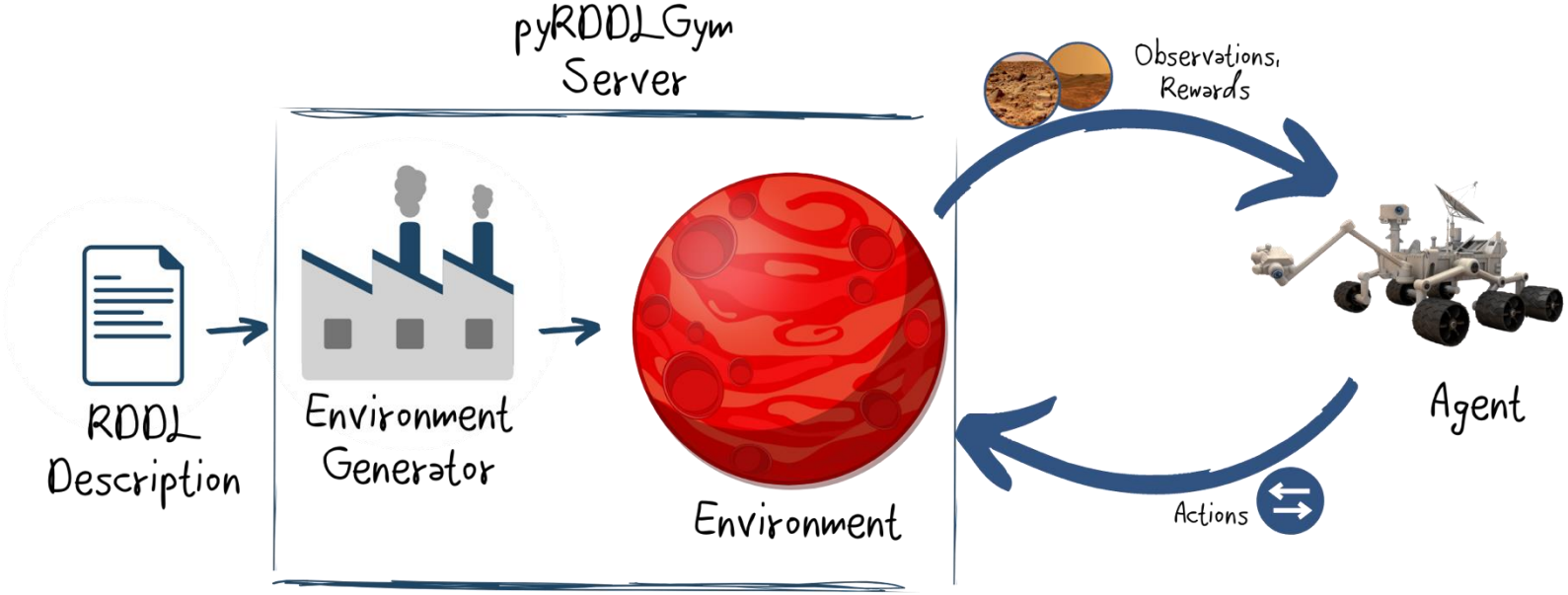
SAMPLING PARADIGM

Your favorite RL/Control/Planning agent



Python* described Dynamic environment

Single Framework for Model-based and Model-free



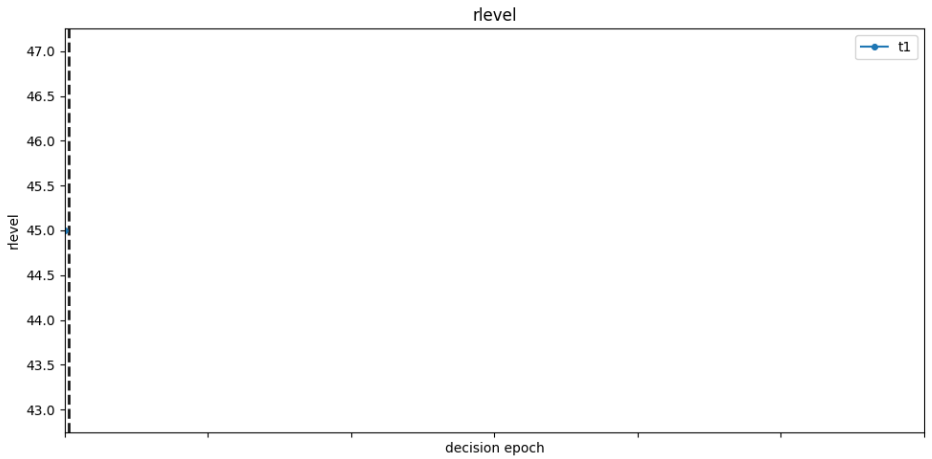
RDDL → compiler → Gym environment

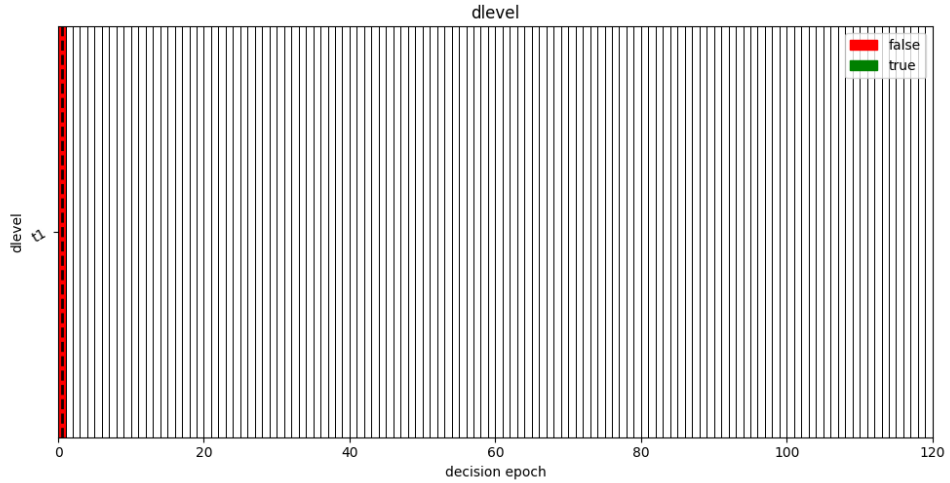
- Standard Gym interface and spaces
- Full access to the underlying model
- Differentiable dynamics*
- Buit-in + custom visualizers
- Symbolic toolkit

A. Taitler, M. Gimelfarb, S. Gopalakrishnan, X. Liu, S. Sanner, “pyRDDLGym: From RDDL to Gym Environments”, ICAPS 2023, Workshop on Bridging the Gap Between AI Planning and Reinforcement Learning (PRL)

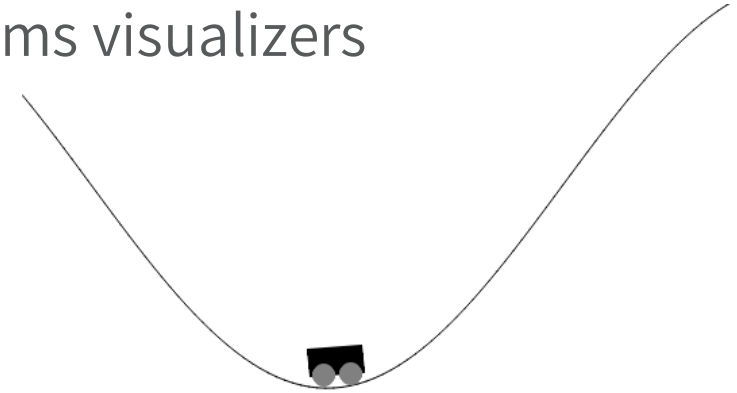
Visualizers

- pyRDDLGym comes with a built-in *ChartVisualizer*, *HeatmapVisualizer* and *TextVisualizer* classes





- It is simple to create customs visualizers

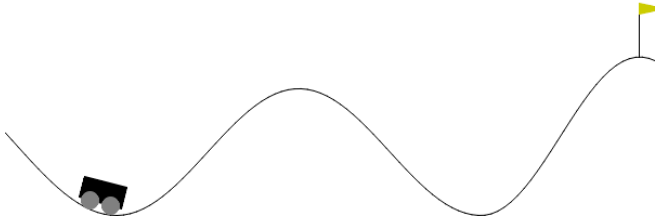


- Inherit base class *pyRDDLGym.Visualizer.StateViz*
- Use your favorite graphical lib, e.g., *matplotlib*, *pygame*, etc.

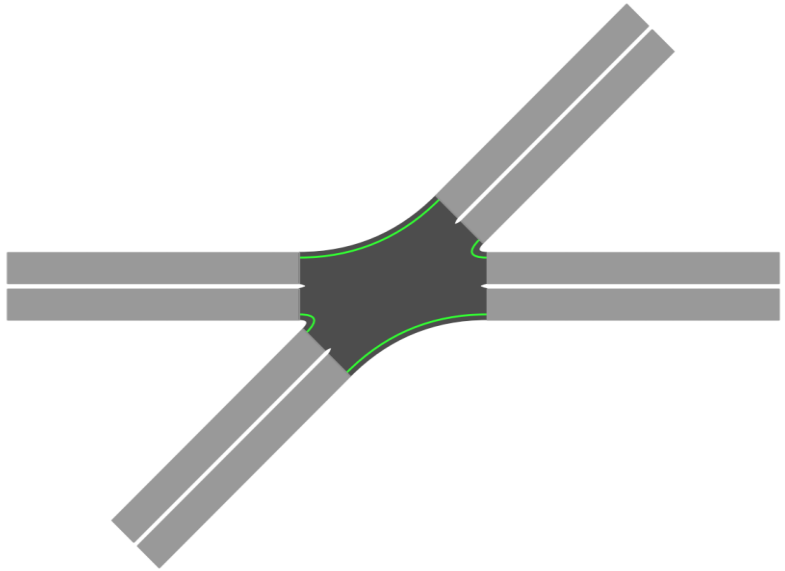
Auxillary Tools

Movie Generator

- Built-in functionality for movie generations of episodes



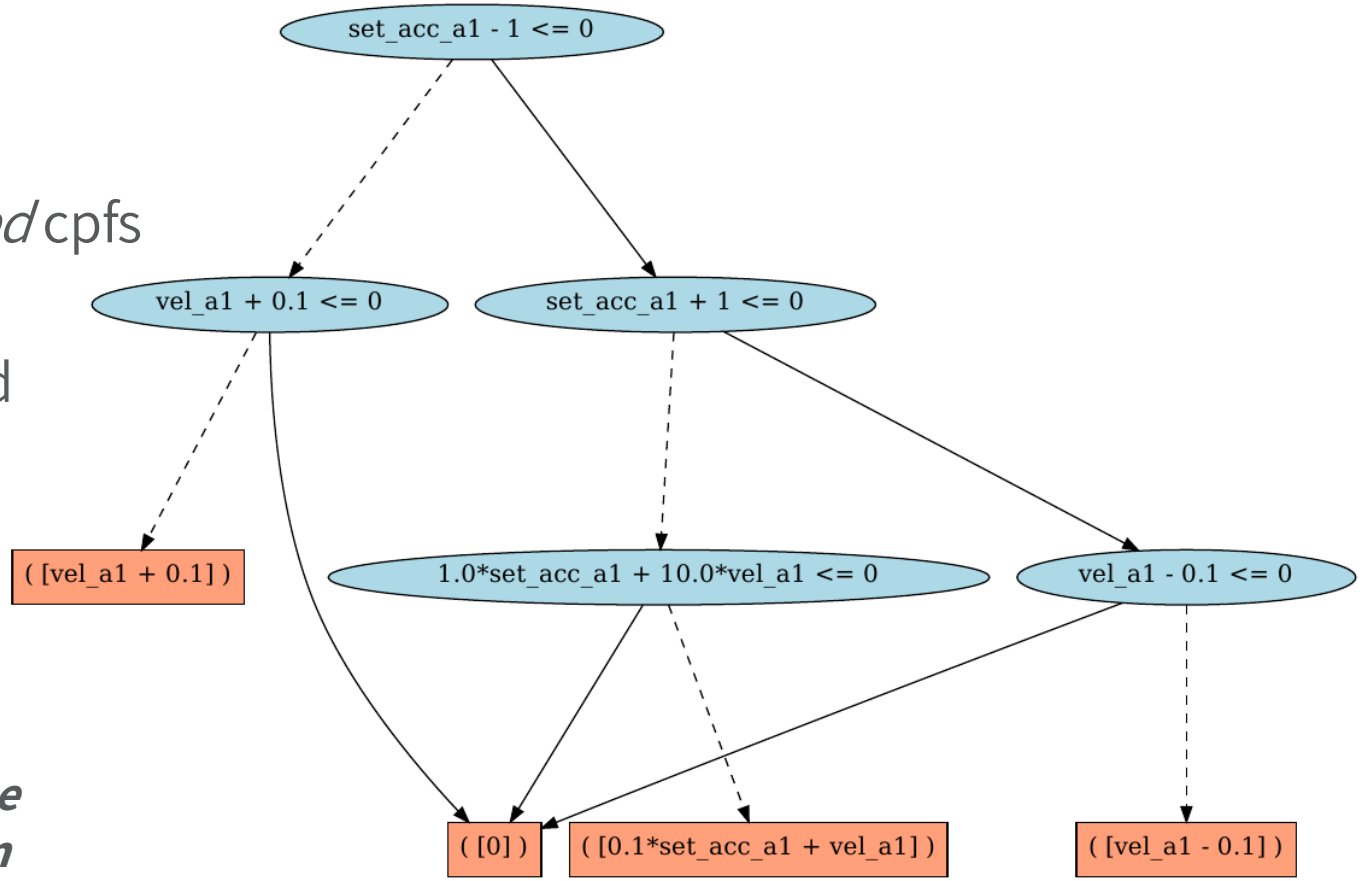
- Supports GIF and MP4



Symbolic Toolkit (I)

Extended Algebric Decision Diagrams (XADDs)

- Symbolic function representation for Piecewise Linear functions
- Compact representation of the *grounded* cpfs
- Symbolic Dynamic Programming (SDP)
- Representation and framework backend



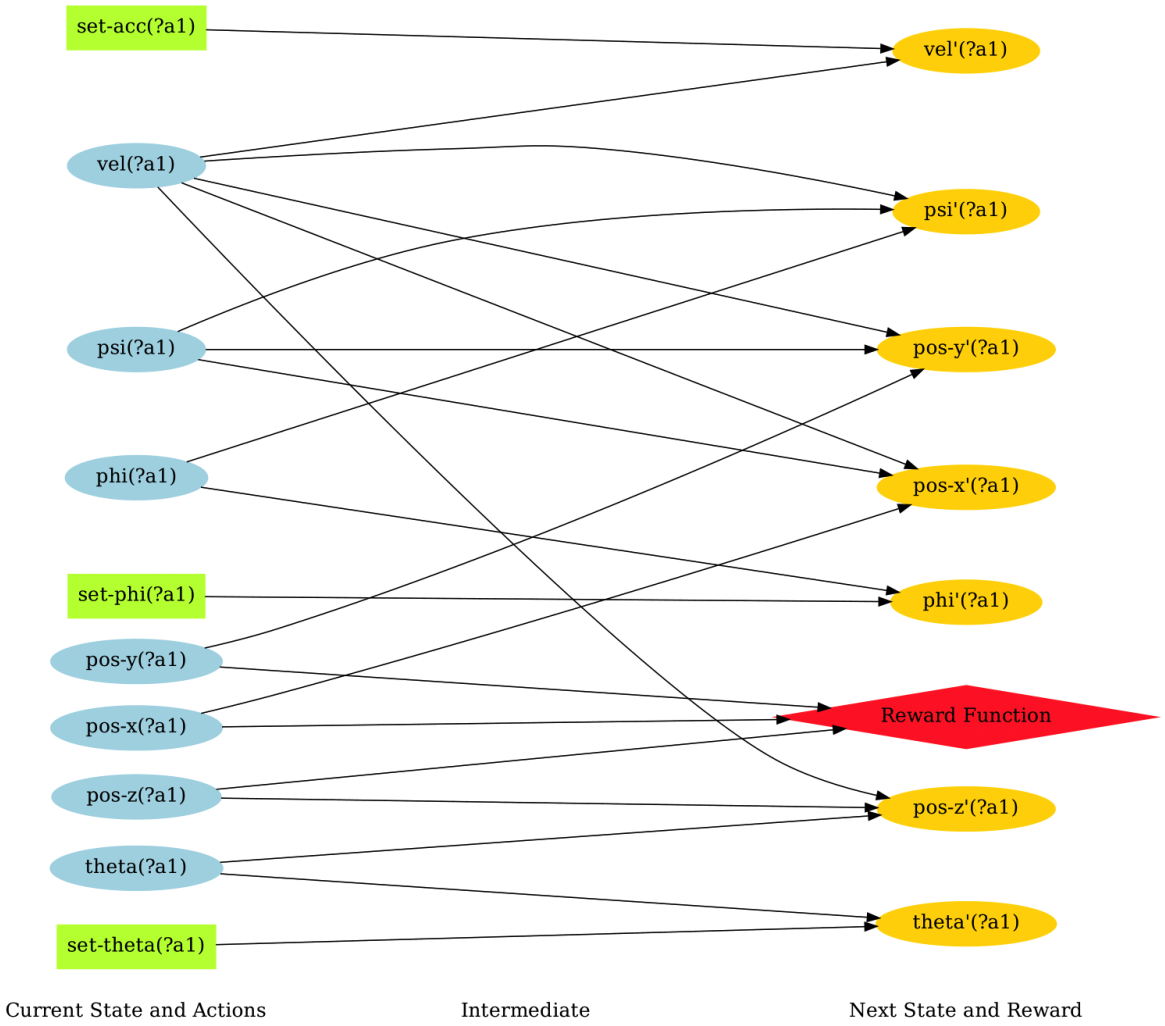
XADD for the UAV domain

https://github.com/ pyrddlgym-project /pyRDDLGym-symbolic

Symbolic Toolkit (II)

Dynamic Bayes Nets (DBNs) visualization

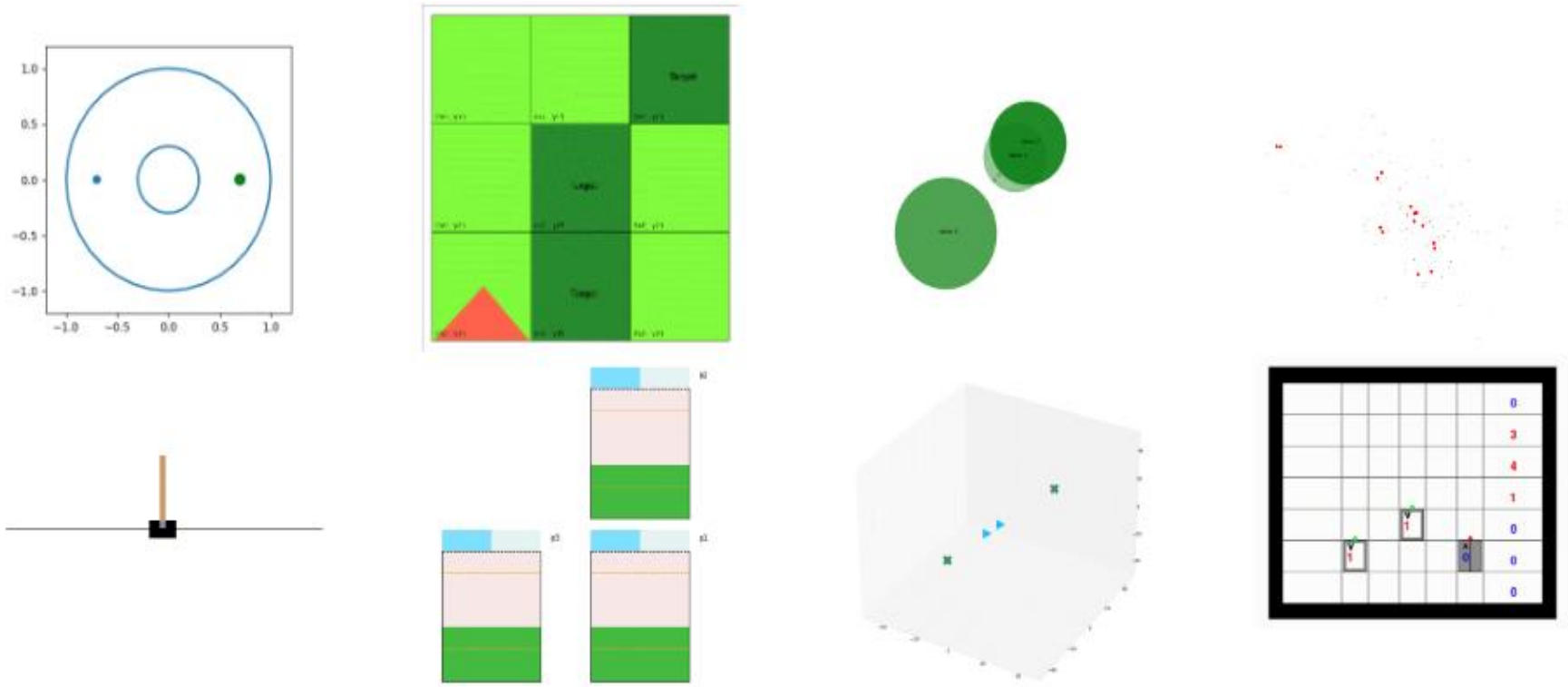
- Visualization of the causal relations
- Causality inference
- Symbolic Dynamic Programming (SDP)
- Direct GCN methods
  e.g., SymNets (symbolic Networks)



DBN visualization

https://github.com/ pyrddlgym-project /pyRDDLGym-symbolic

RDDLRepository*



Gym's Classical control environments

All previous RDDL domains

New exciting environments

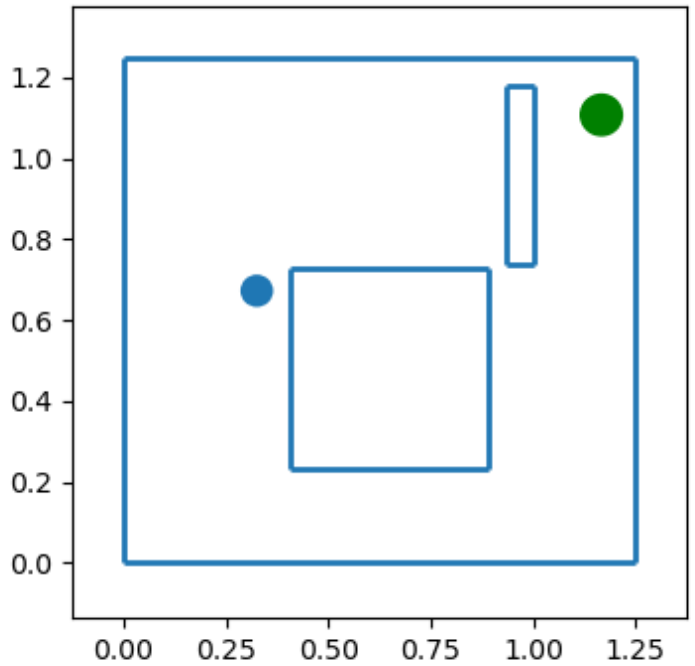
https://github.com/pyrddlgym-project/rddlrepository

RaceCar

- Goal oriented problem
- Plan trajectory for a kinematic agent (2nd order) in presence of obstacles
- **Action:** *continuous* force/acceleration in two axes ($n_a = 2$)
- **Observation:** *continuous* positions and velocities ($n_s = 4$)
- **Reward:**

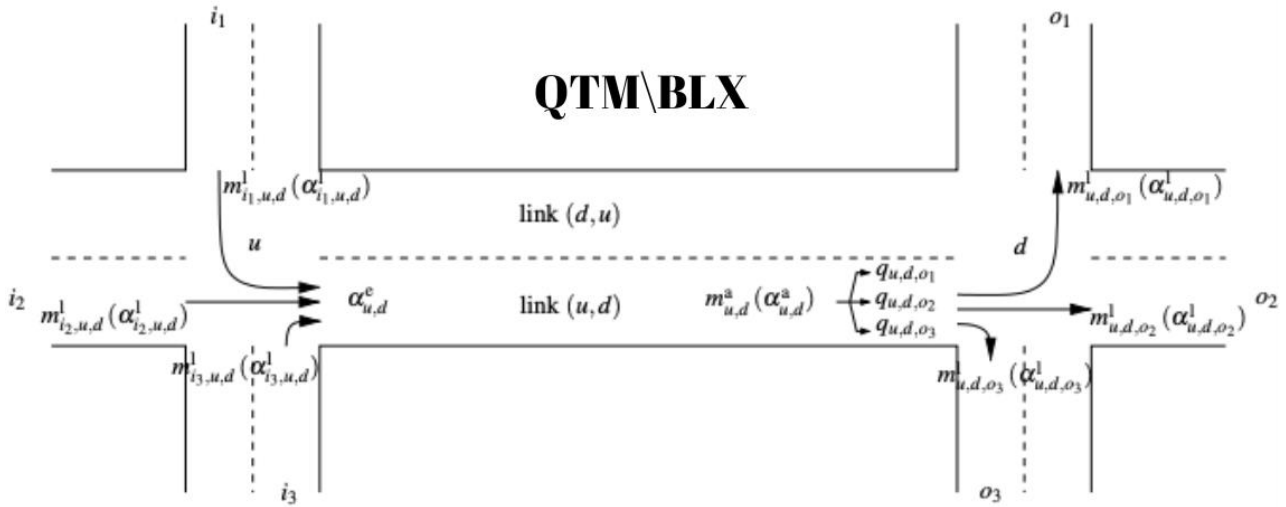
$$R = -\sum_{k=1}^{H} a_x^2[k] + a_y^2[k] + R_G \cdot 1_{\{a_x^2[k]+a_y^2[k]<r_g\}}$$

Termination: $a_x^2[k] + a_y^2[k] < r_g$ **or** hitting a wall.



Traffic

- Traffic network cogestion control

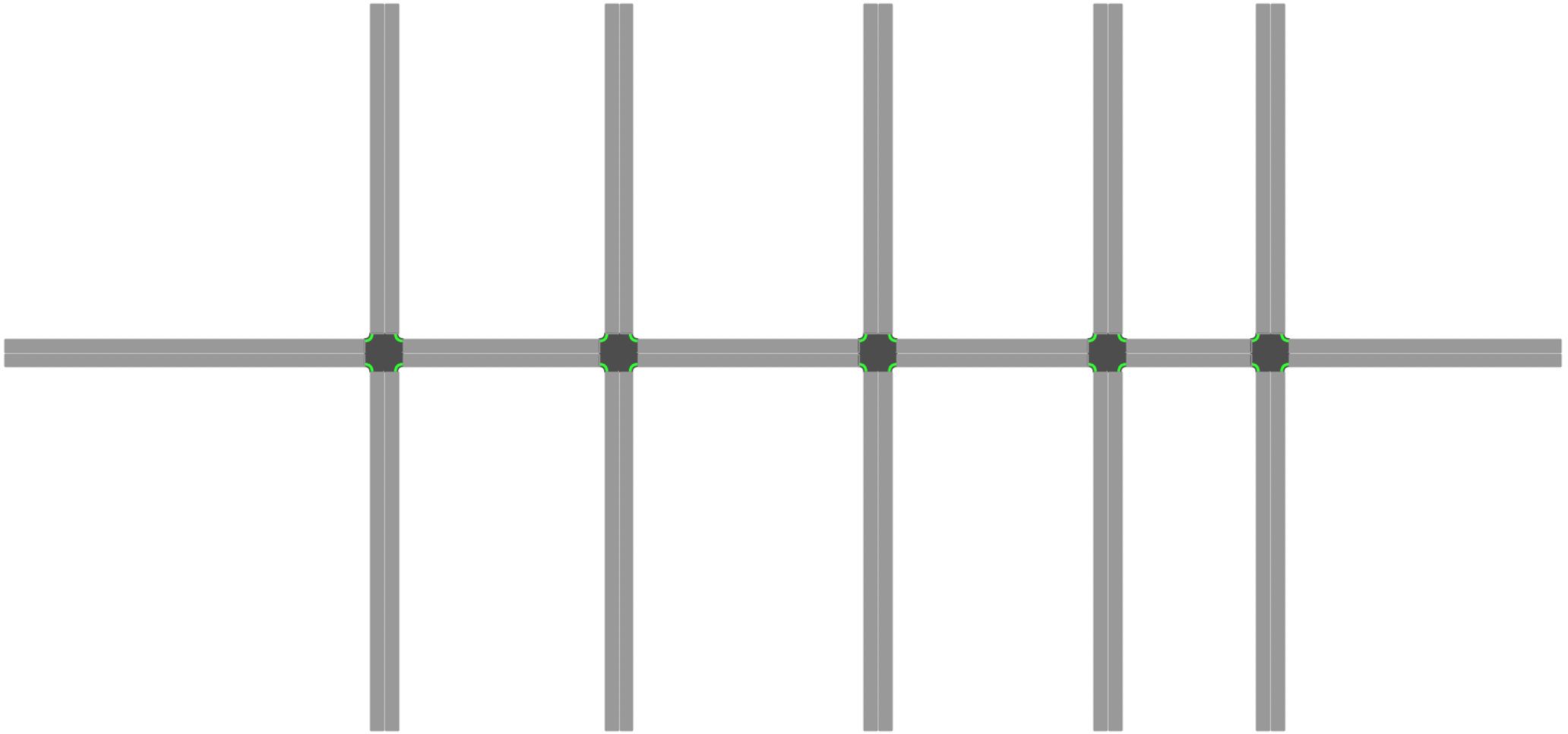


$$q_{u,d,o_m}(k_d+1) = q_{u,d,o_m}(k_d) + \left(\alpha^{a}_{u,d,o_m}(k_d) - \alpha^{l}_{u,d,o_m}(k_d)\right) \cdot c_d$$

$$q_{u,d}(k_d) = \sum_{o_m \in O_{u,d}} q_{u,d,o_m}(k_d)$$

- **Action:** Extend/Change for light phases (each intersection)
- **Observation:** Cars in queues, phase, phase time, etc.
- **Reward:** Total travel time (number of cars in the network)
- **Constraints:** Min/max time in phase

Traffic



1x5 Network

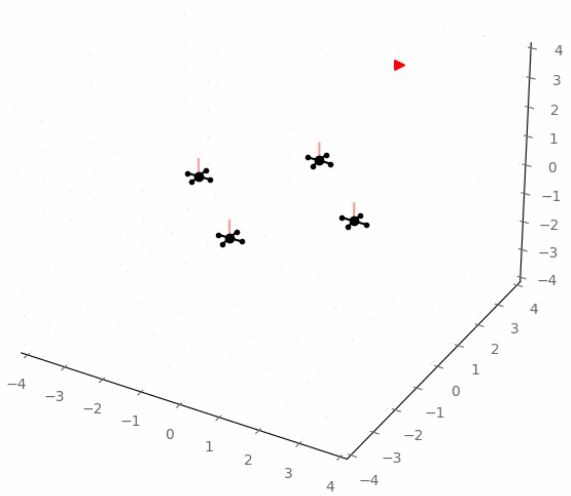
Model-based Planner - JaxPlan

GOAL: Planner that can address the full spectrum of what we can express in RDDL, *with learning flavor*

Concurrent → multiple actions

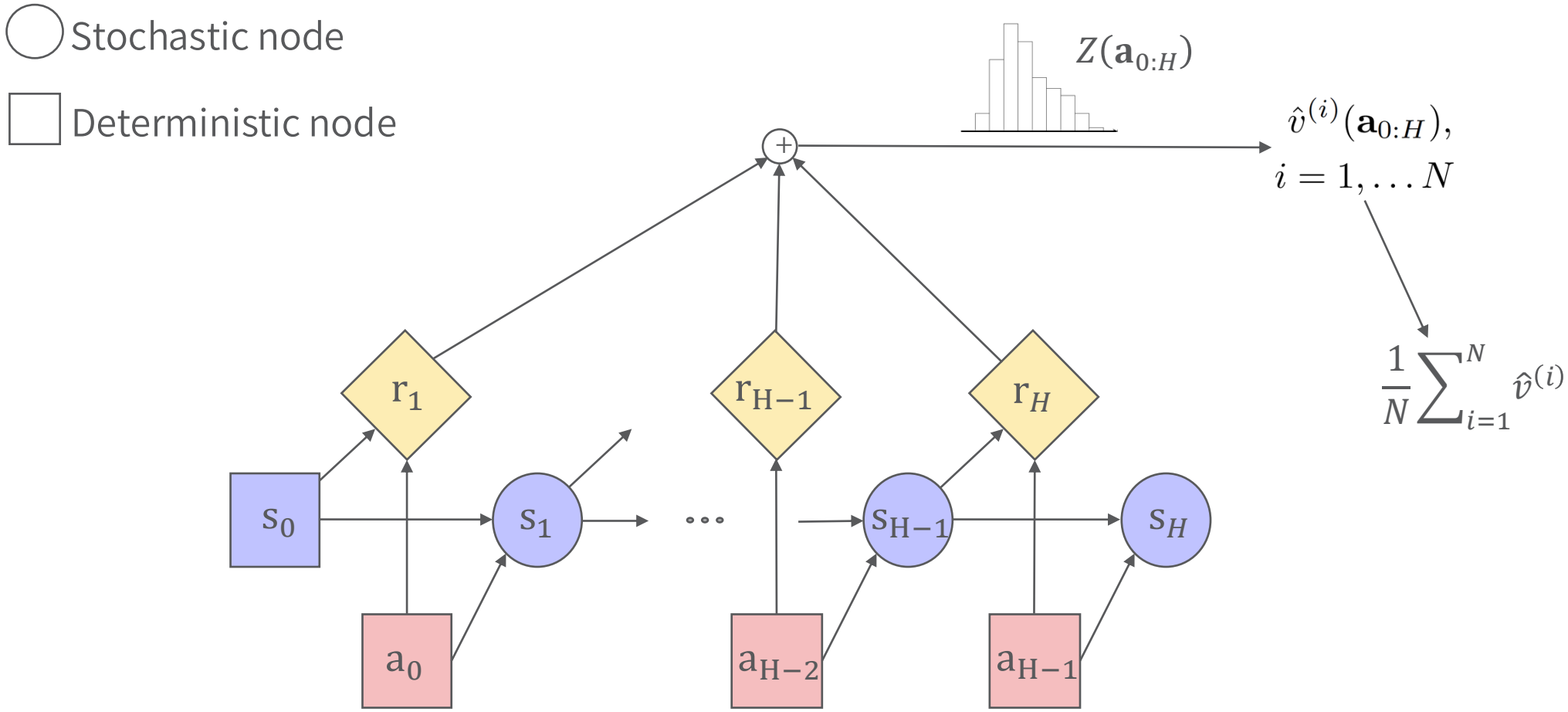
Stochastic

Hybrid → Dynamics, Actions



Anytime algorithm

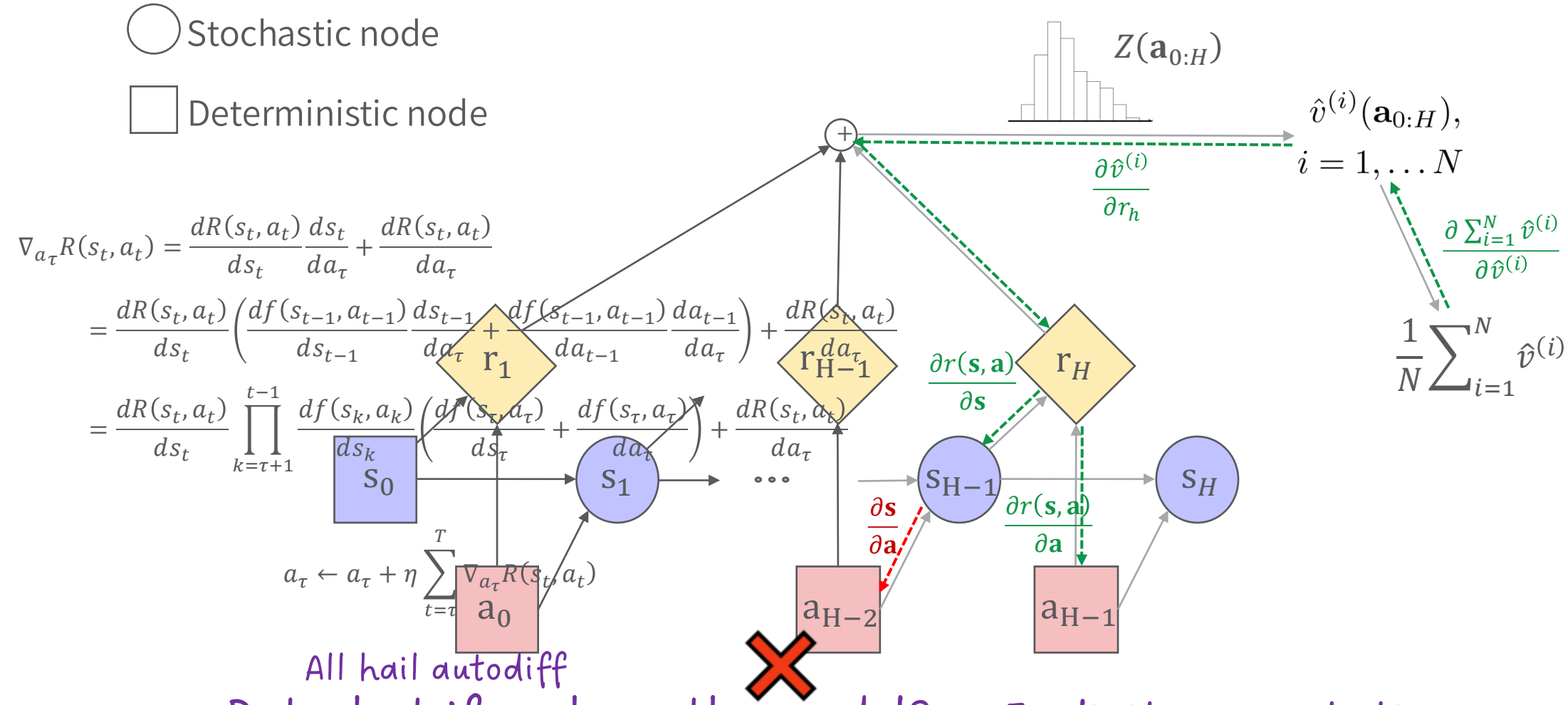
Monte-Carlo Chain – Forward View



But what if we have the model? → Explicit computation graph!

G., Wu, B. Say, and S., Sanner, "Scalable planning with tensorflow for hybrid nonlinear domains," NeurIPS, 2017

Monte-Carlo Chain – Backward View



G., Wu, B. Say, and S., Sanner, "Scalable planning with tensorflow for hybrid nonlinear domains," NeurIPS, 2017

Monte-Carlo Chain – Backward View

Stochastic node

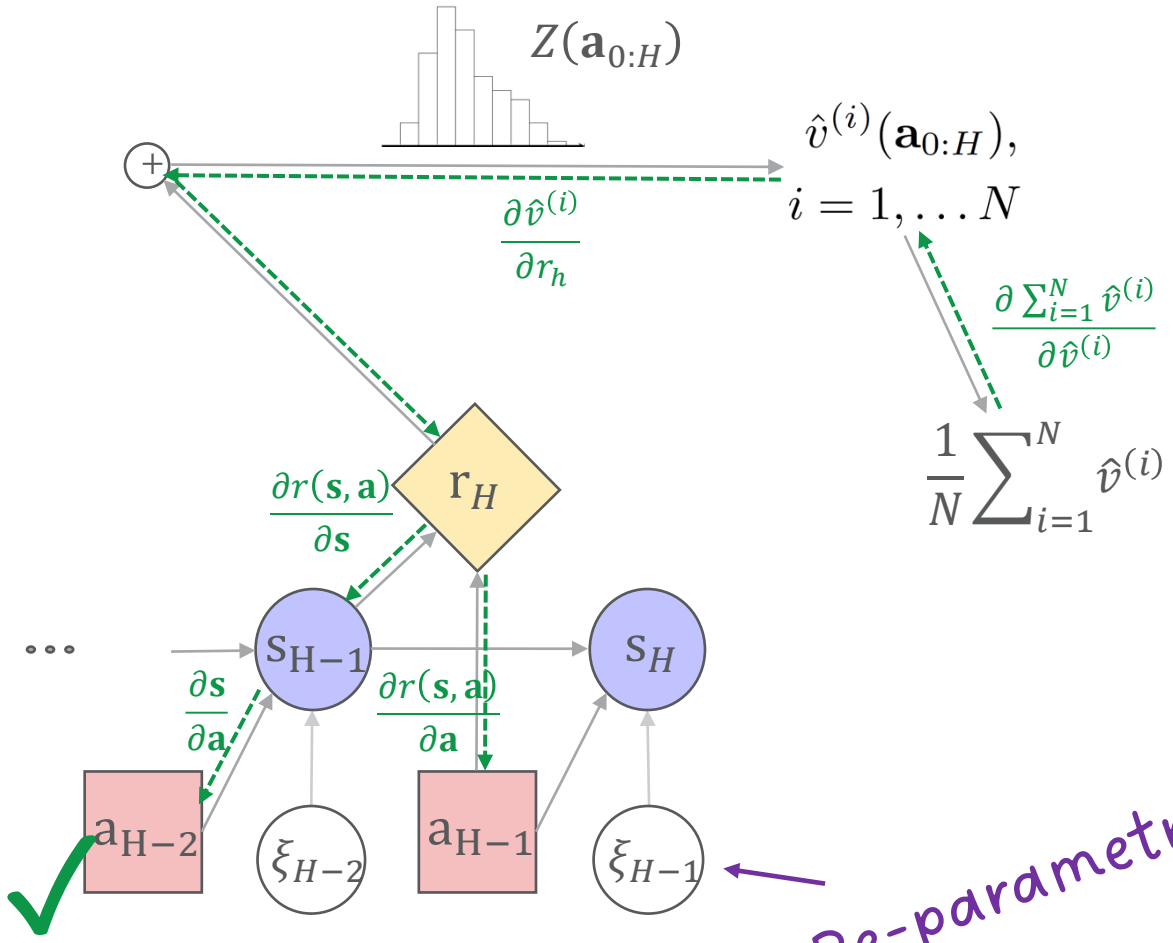
Deterministic node

$$\mathbf{s}_{t+1} \sim p(\cdot\,|\mathbf{s}_t, \mathbf{a}_t)$$

$$\mathbf{s}_{t+1} = \phi(\mathbf{s}_t, \mathbf{a}_t, \xi_t)$$

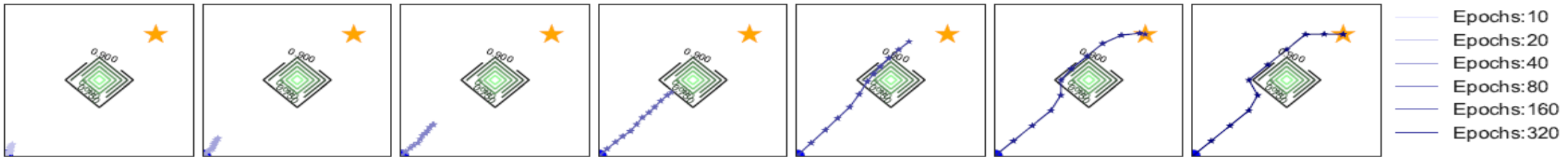
$$\mathrm{s}_{t+1} \sim \mathcal{N}(\mathrm{s}+\mathrm{a}, \sigma^2)$$

$$\Rightarrow \mathrm{s}+\mathrm{a}+\sigma\underbrace{\mathcal{N}(0.\mathrm{I})}_{\xi_t}$$



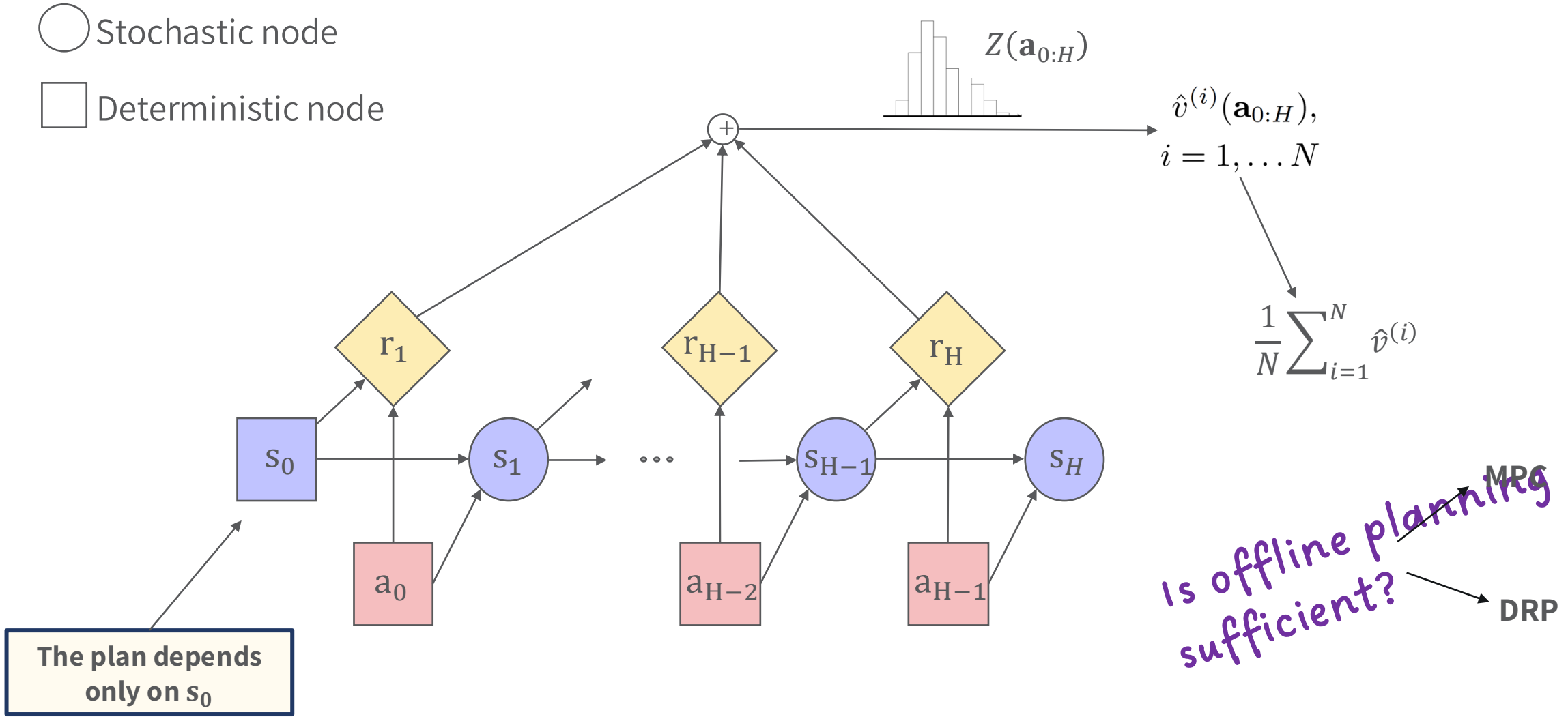
T. P., Bueno, L. N., de Barros, D. D., Maua, S., Sanner. “Deep reactive policies for planning in stochastic nonlinear domains." AAAI, 2019

Empirical Evaluation

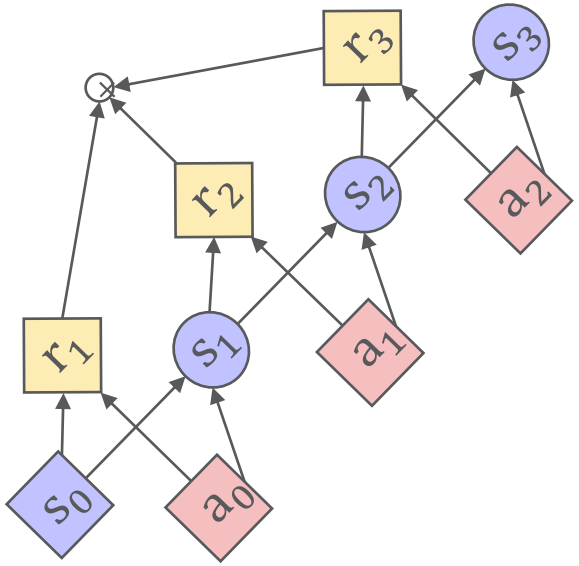


First the planner evolves towards the most direct route to the goal, then continues descent towards a more cost-efficient solution.

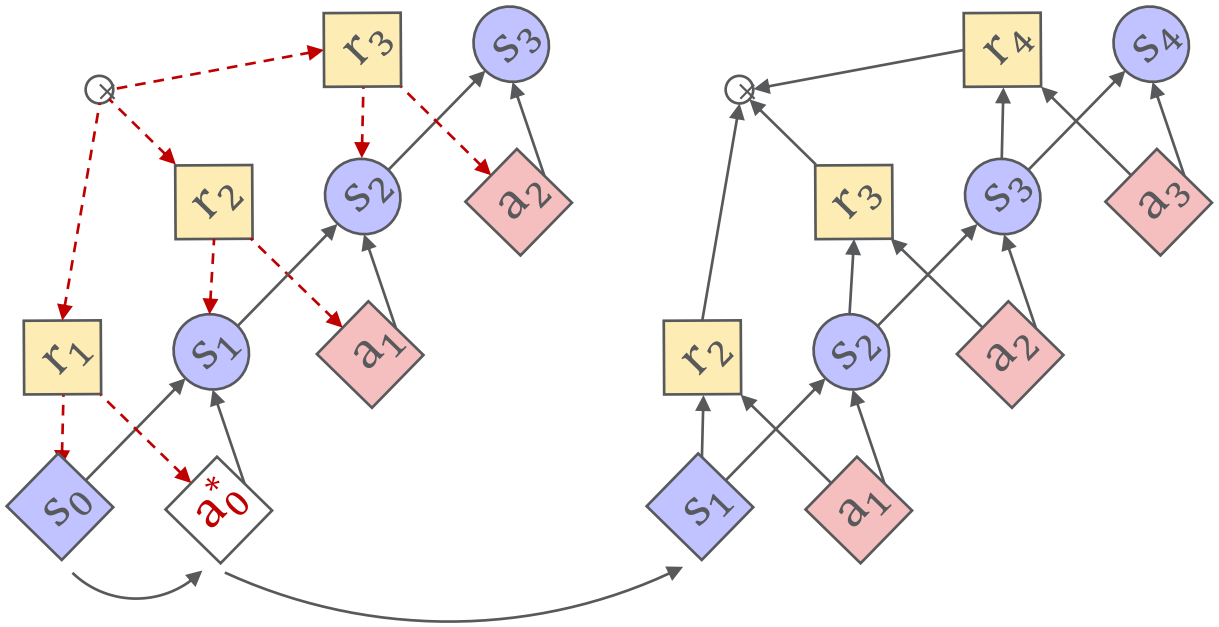
Offline Plan



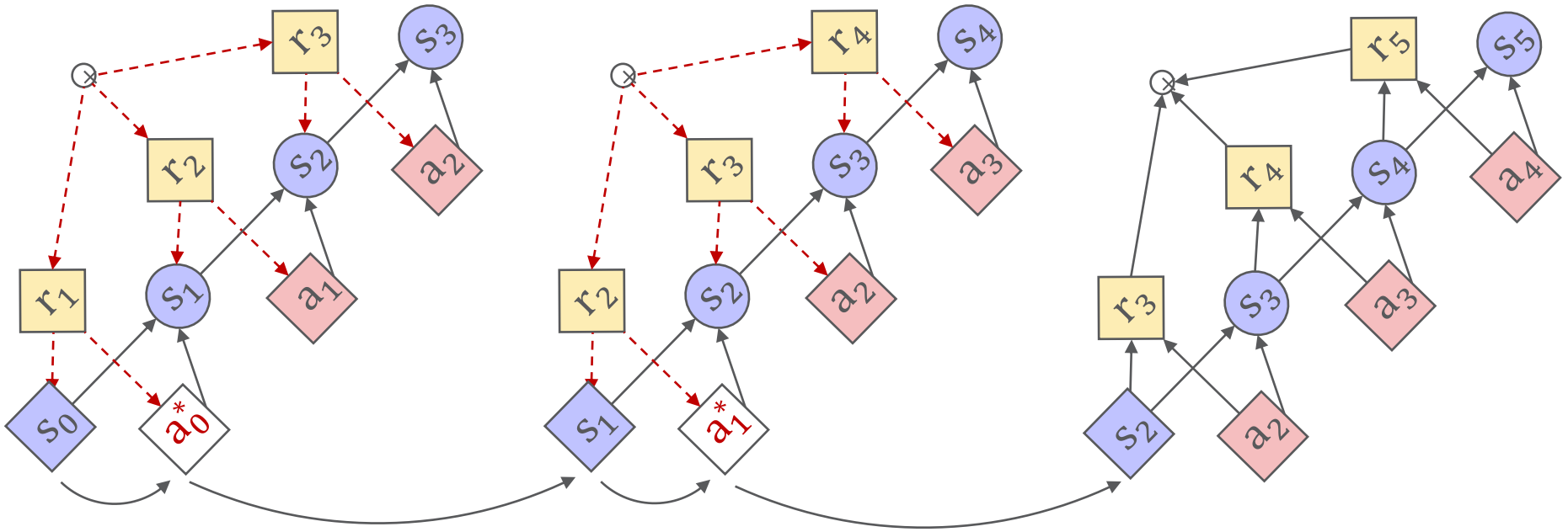
Rolling Horizon Replanning



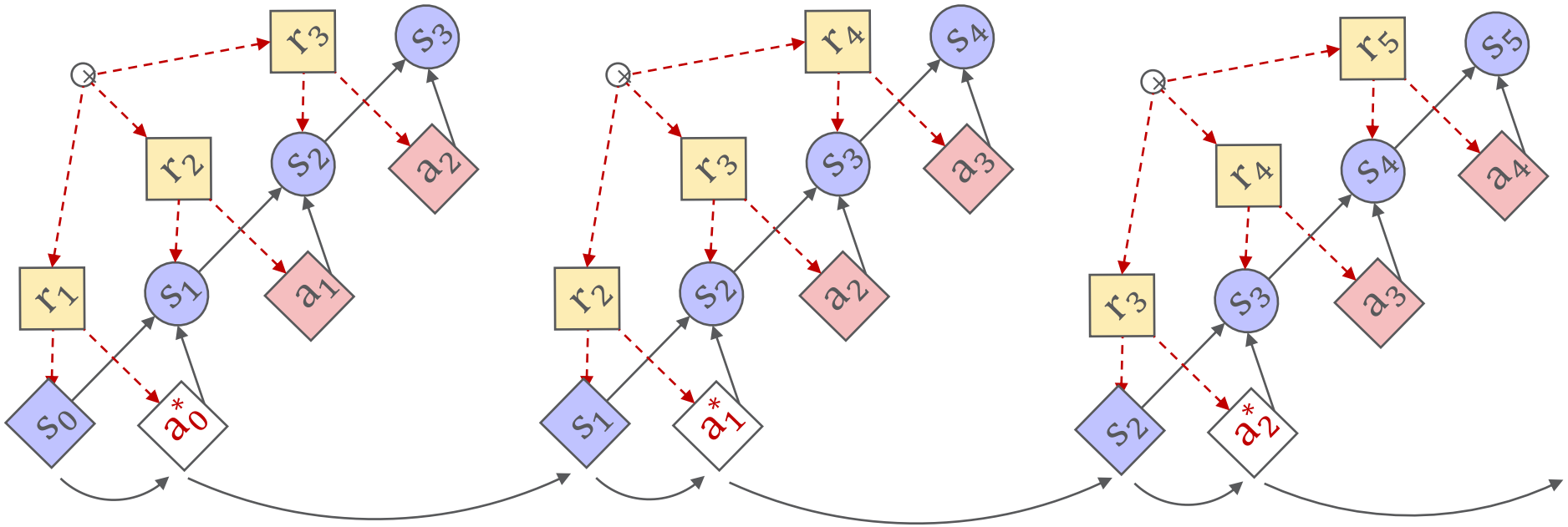
Rolling Horizon Replanning



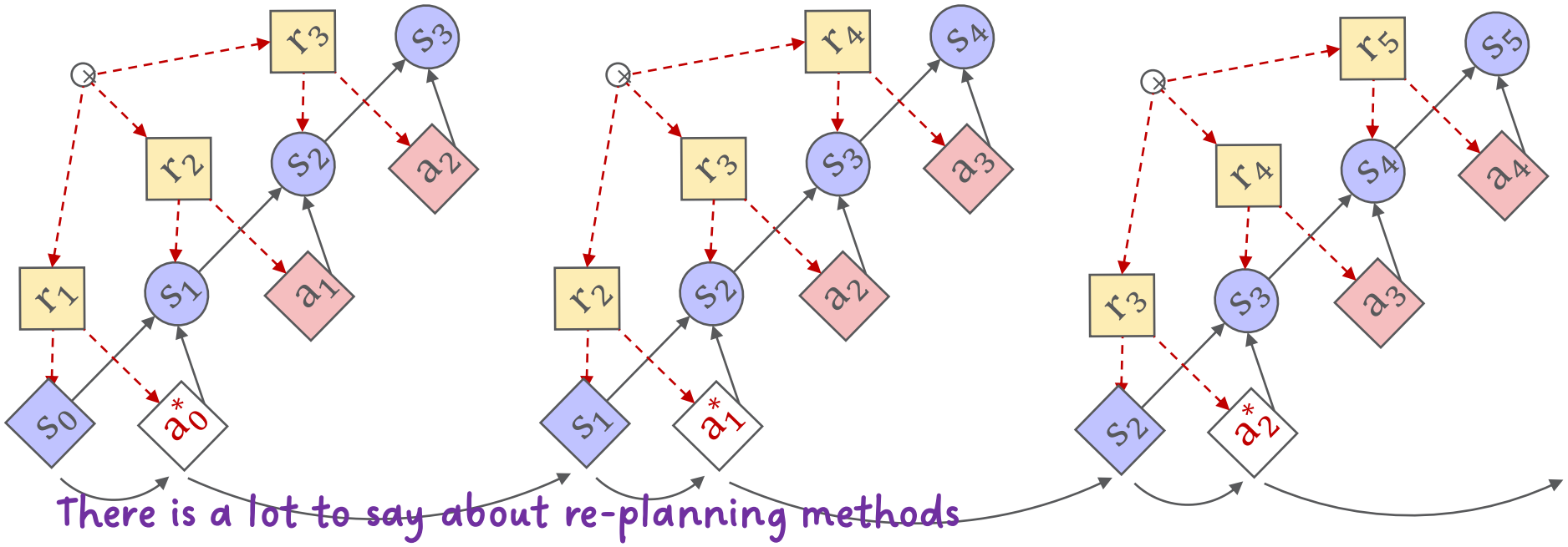
Rolling Horizon Replanning



Rolling Horizon Replanning



Rolling Horizon Replanning



There is a lot to say about re-planning methods

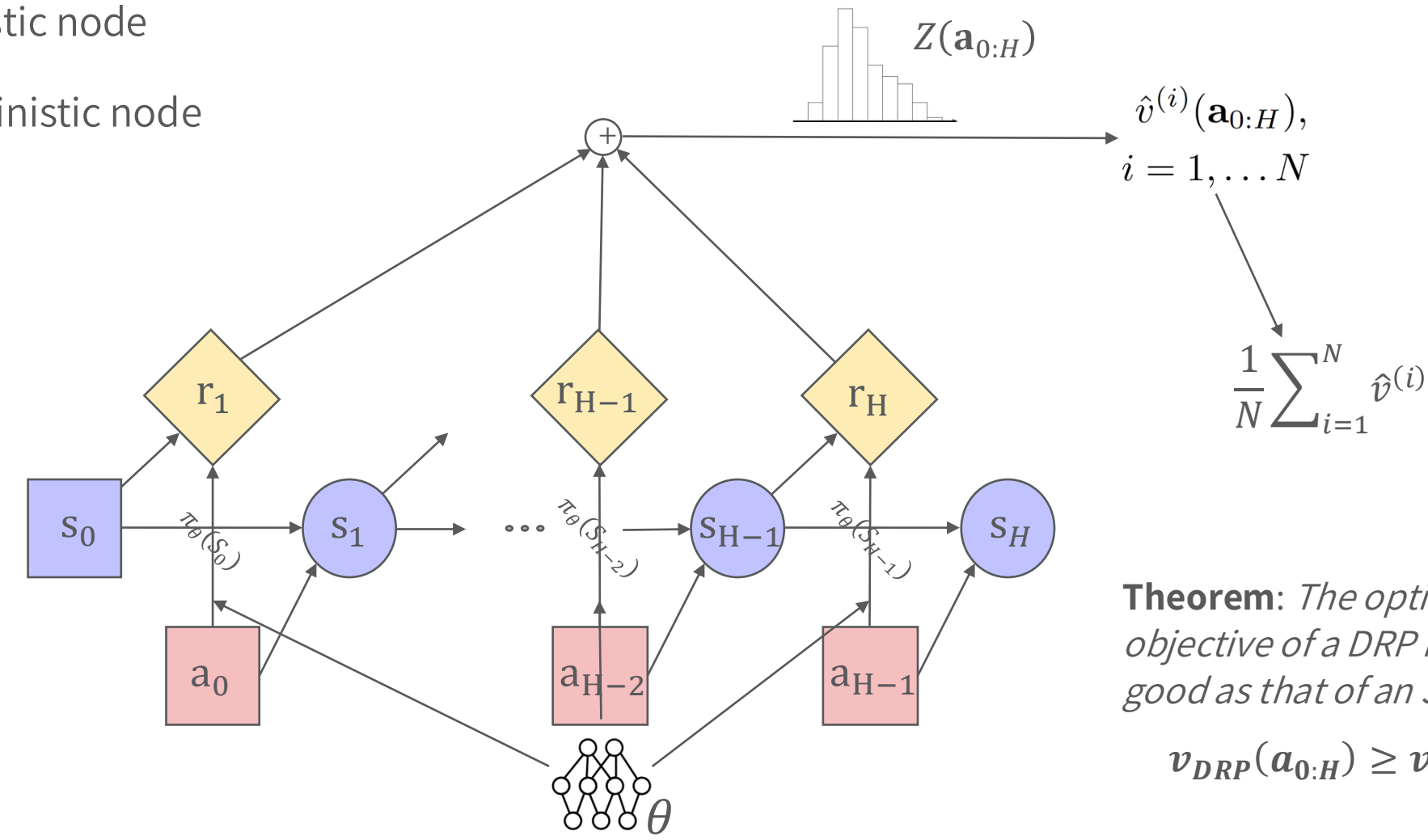
They actually win competitions and applied in reality

But that's for another time...

Deep Reactive Policies

Stochastic node

Deterministic node



Theorem: *The optimal objective of a DRP is at least as good as that of an SLP*

$$v_{DRP}(a_{0:H}) \geq v_{SLP}(a_{0:H})$$

What If We Cannot Compute Gradients?

“Not all domains are born continuous”

– Anonymous

Branches

Logical expressions

```
cpfs {
   burning'(?x, ?y) = if ( put-out(?x, ?y) ) // Intervention to put out fire?
       then false
       else if (~out-of-fuel(?x, ?y) ^ ~burning(?x, ?y)) // Ignition of a new fire? Depends on neighbors.
           then [if (TARGET(?x,?y)^~(exists_{?x2:x_pos,?y2:y_pos}(NEIGHBOR(?x,?y,?x2,?y2)^burning(?x2 ?y2))))
              then false
              else Bernoulli(1.0/(1.0 + exp[4.5 - (sum_{?x2: x_pos,?y2:y_pos} (NEIGHBOR(?x,?y,?x2,?y2)^burning(?x2,?y2))))])]
           else burning(?x, ?y); // State persists

   out-of-fuel'(?x, ?y) = out-of-fuel(?x, ?y) | burning(?x,?y) | (~TARGET(?x, ?y) ^ cut-out(?x, ?y));
};
```

Discrete distributions

Discrete-Continuous Approximations

Logical conjunctions, i.e. a ∧ b, can be replaced by T-norms

Definition: *T-norm Fuzzy logic* $T: \{0,1\}^n \rightarrow [0,1]$

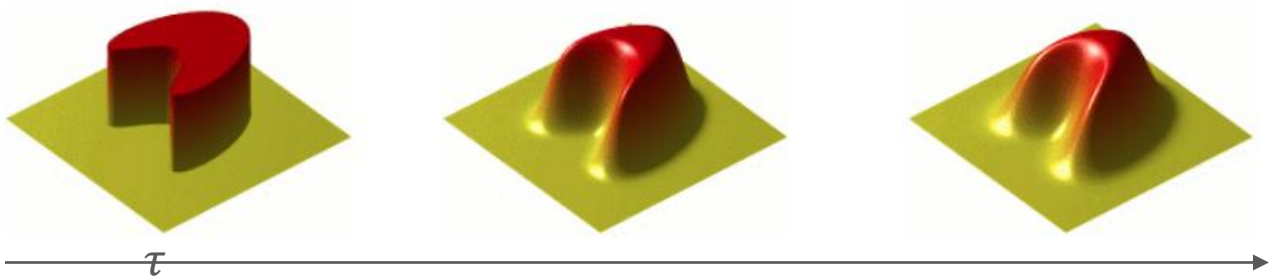
- *Commutativity:* $T(a,b) = T(b,a)$
- *Monotonicity:* $T(a,b) \leq T(c,d) \;\; if \;\; a \leq c \;\&\; b \leq d$
- *Associativity:* $T\big(a, T(b,c)\big) = T(T(a,b),c)$
- 1 *acts as the identity element* $T(a,1) = a$

Some examples

- Product: $T(a,b) = a * b$
- Lukasiewicz: $T(a,b) = \max\{a+b-1, 0\}$
- Godel-Dummett: $T(a,b) = \min\{a,b\}$

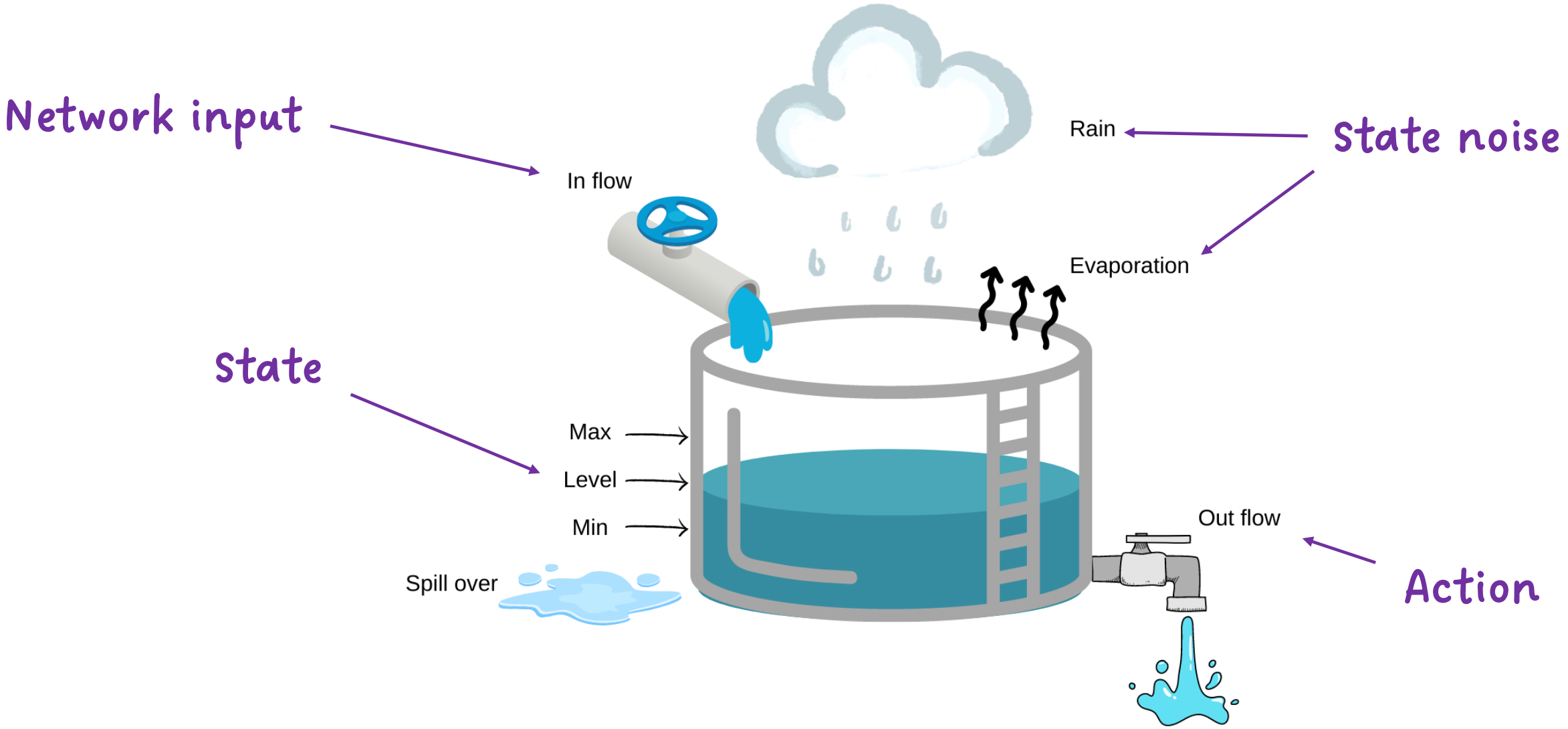
Discrete-Continuous Approximations (T-norms)

Exact RDDL Operation	Continuous Relaxed Expression
$a \wedge b$	$a * b$ → $\boldsymbol{T}: \{\mathbf{0}, \mathbf{1}\}^{\boldsymbol{n}} \rightarrow [\mathbf{0}, \mathbf{1}]$
$\neg a$	$1 - a$
IF c $THEN$ a $ELSE$ b	$c * a + (1 - c) * b$
$forall\_\{?p: type\}x(?p)$	$\prod_{?p} x(?p)$
$a > b$	$sigmoid\left(\frac{1}{\tau} * (a - b)\right)$
$signum(a)$	$\tanh\left(\frac{1}{\tau} * a\right)$
$argmax\_\{?p: type\}x(?p)$	$\sum_{i=1,2,\ldots,\lvert type\rvert} i * softmax\left(\frac{1}{\tau} * x\right)[i]$
$Bernoulli(p)$	Gumbel-softmax trick



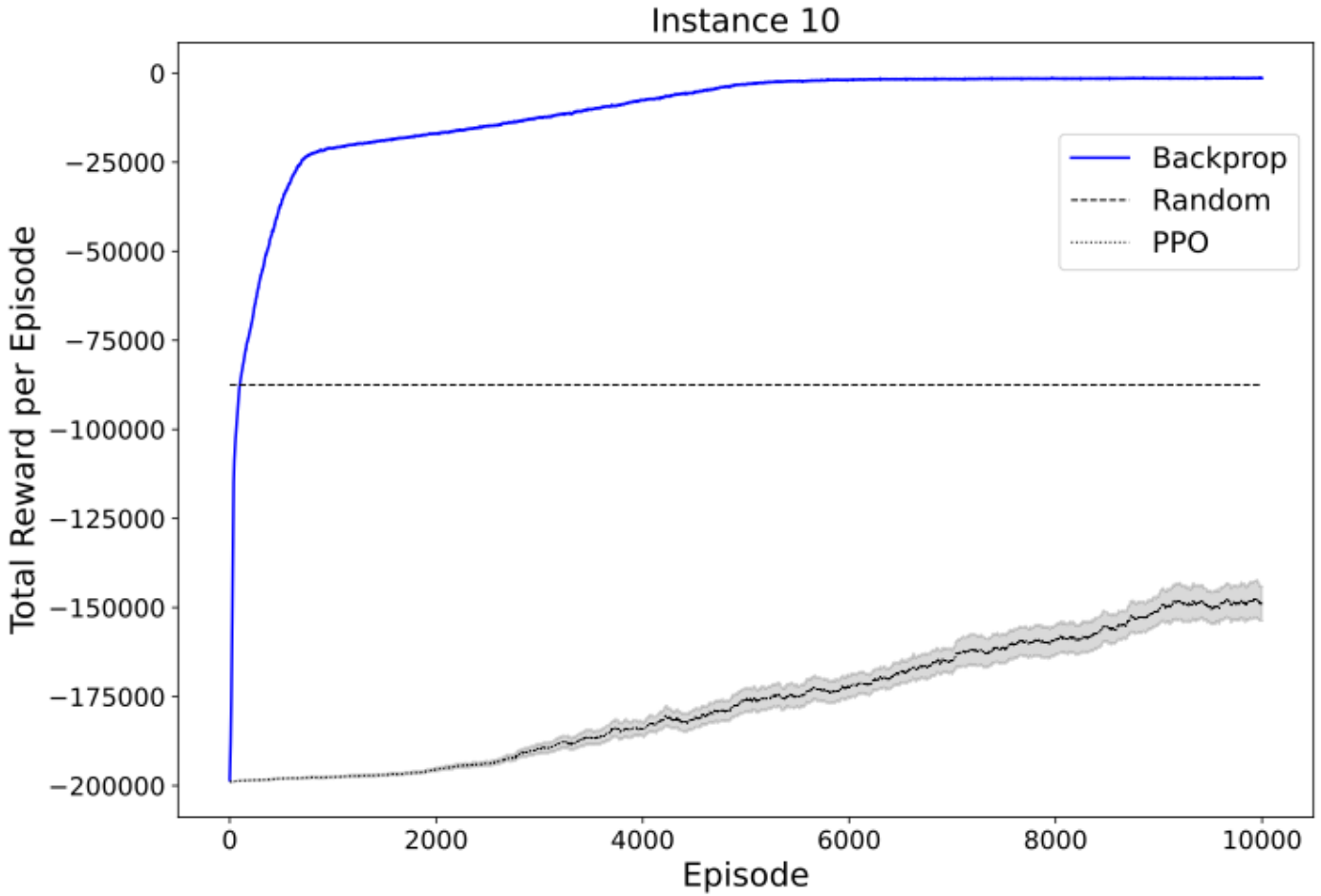
M. Gimelfarb, A. Taitler, S. Sanner, "JaxPlan and GurobiPlan: Optimization Baselines for Replanning in Discrete and Mixed Discrete and Continuous Probabilistic Domains", 2024 International Conference on Automated Planning and Scheduling (ICAPS)

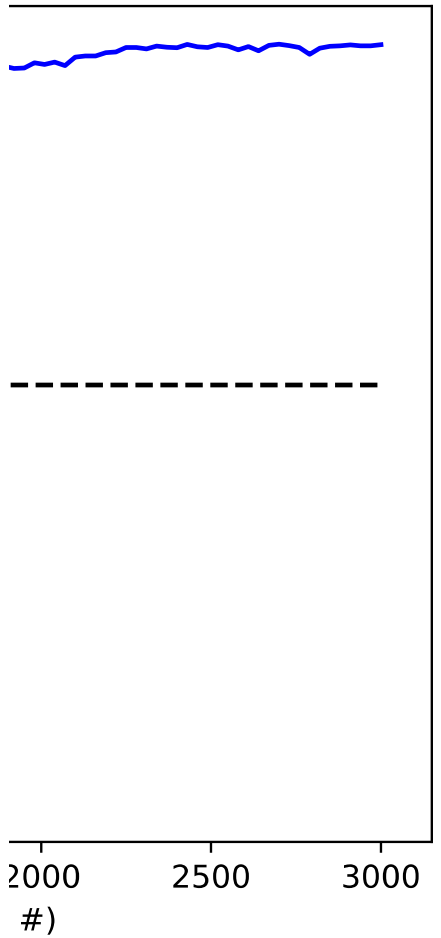
Example


Network input
In flow
Rain
State noise
Evaporation
State
Max
Level
Min
Spill over
Out flow
Action

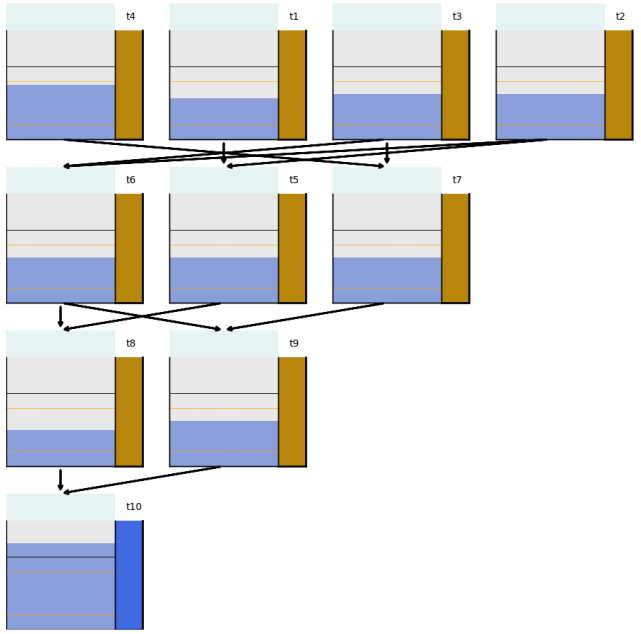
Example

- **Setup**
  - 10 rese
  - 120 de
  - Total of
- **policy opti**
  - 3,000 b
    - Ar

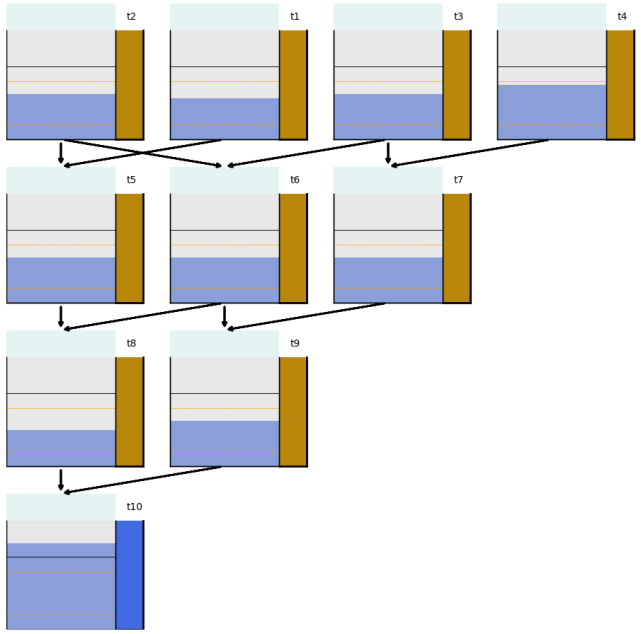




From Model to Plan – Multi Reservoir Control



Backprop planner



Random policy

Hands-on

Colab notebook

- Basic pyRDDLGym usage
- Modeling and execution of environments
- JaxPlan

There is a lot more to the story…

Questions?

Dr. Ayal Taitler

Head of Drive-lab

Department of Industrial Engineering & Management

Ben-Gurion University of the Negev