Offline Reinforcement Learning in Regular Decision Processes

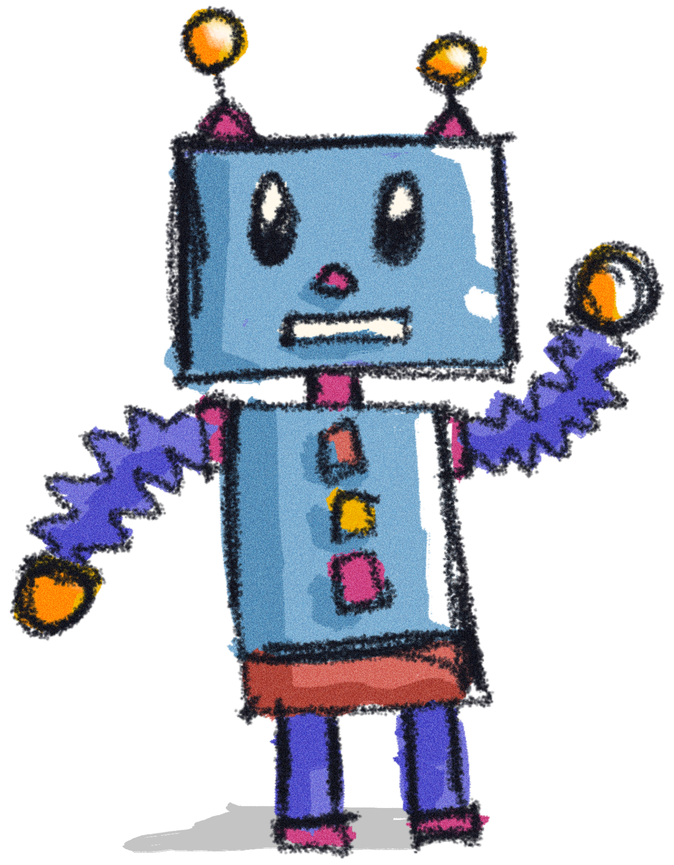
Ahana Deb[1], Roberto Cipollone[2], Anders Jonsson[1], Alessandro Ronca[3], Mohammad Sadegh Talebi[4]

Universitat Pompeu Fabra Barcelona [1]
Sapienza Università di Roma [2]
University of Oxford [3]
University of Copenhagen [4]

Contents

Reinforcement Learning

Agent



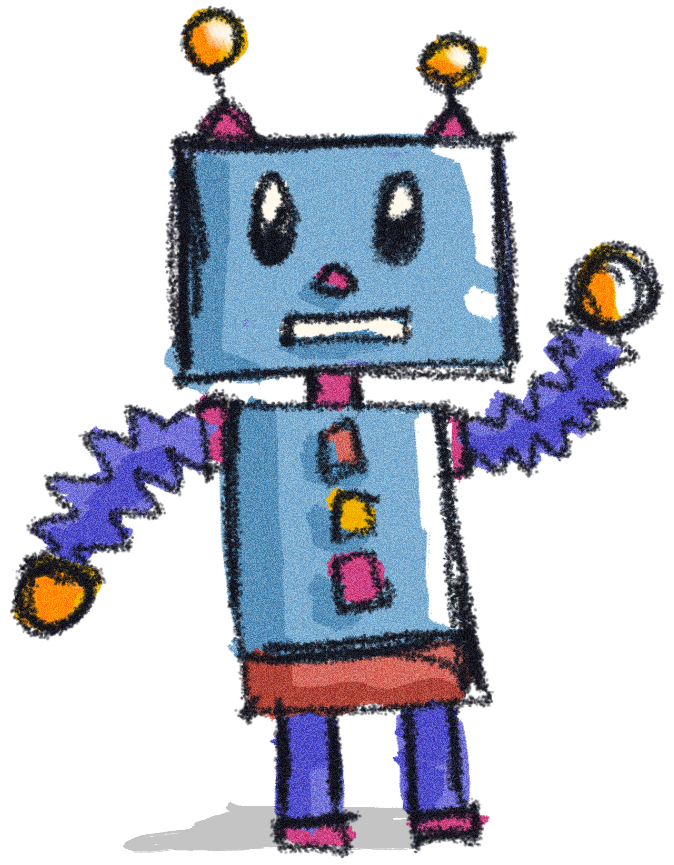
Action

Environment

Perception +
Reward

Reinforcement Learning

Agent



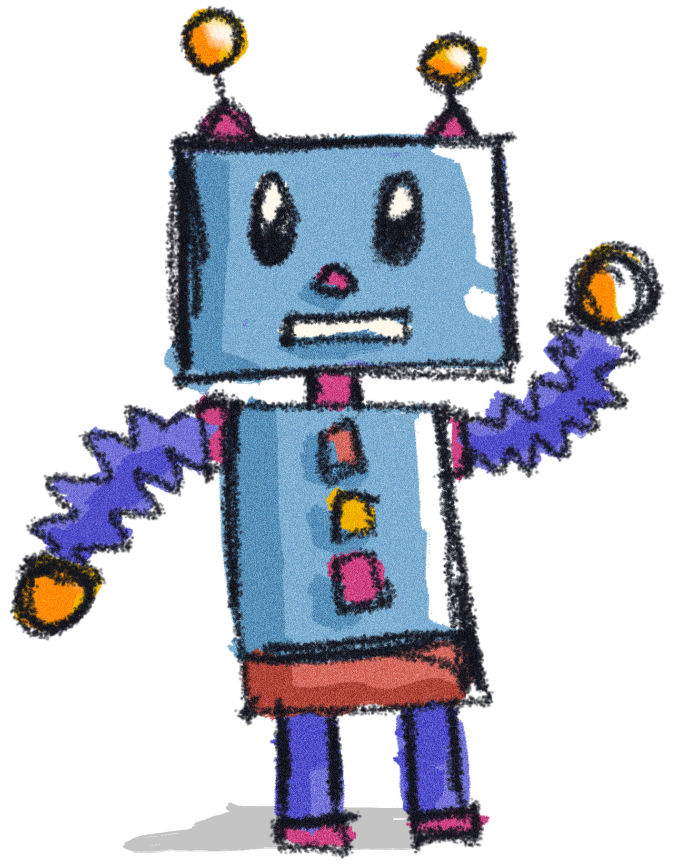
Current state s_t,
Action taken a_t

Environment

Next state s_{t+1},
Reward r_{t+1}

Reinforcement Learning

Agent



Current state s_t,
Action taken a_t

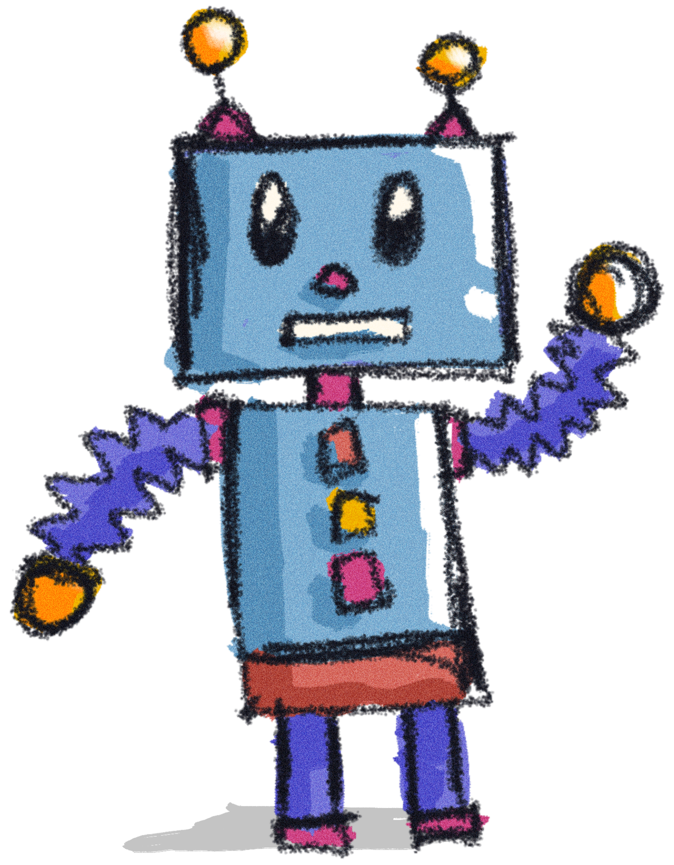
Environment

Next state s_{t+1},
Reward r_{t+1}

Reward: Scalar signal. The agent's goal is to maximise this reward

Reinforcement Learning

Agent



Current state s_t,
Action taken a_t

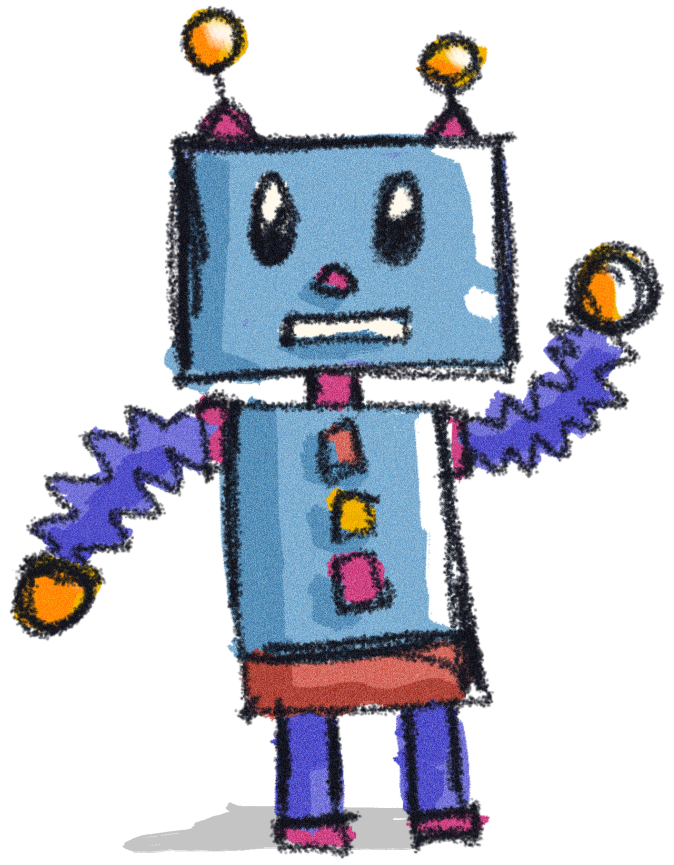
Policy: Map from state to action,
Can be *deterministic* $a = \pi(s)$
stochastic
$\pi(a \mid s) = \mathbb{P}[A_t = a \mid S_t = s]$

Next state s_{t+1},
Reward r_{t+1}

Reward: Scalar signal.
The agent's goal is to maximise this reward

Reinforcement Learning

Agent



Current state s_t,
Action taken a_t

Policy: Map from state to action,
Can be *deterministic* $a = \pi(s)$
stochastic
$\pi(a \mid s) = \mathbb{P}[A_t = a \mid S_t = s]$

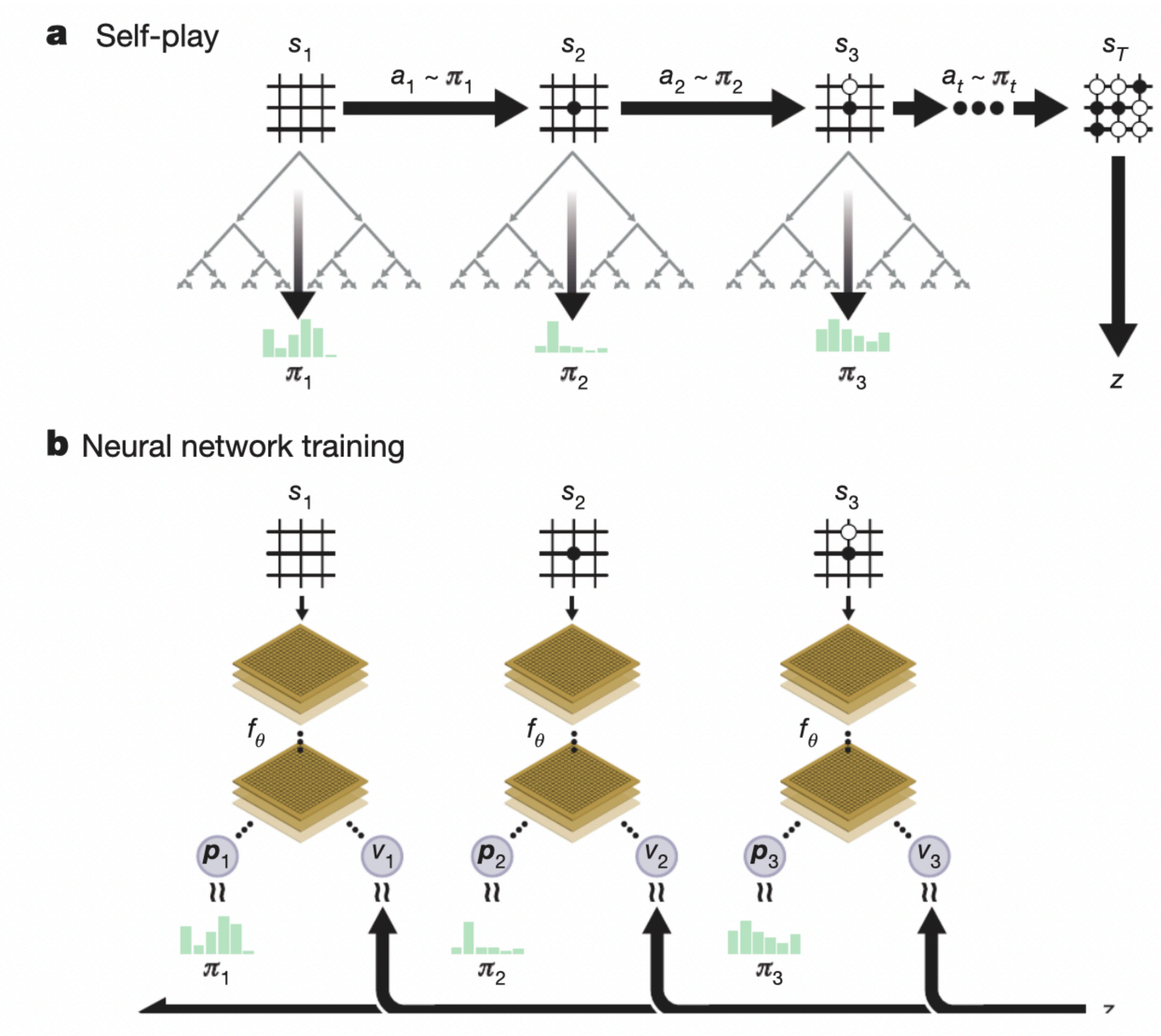
Next state s_{t+1},
Reward r_{t+1}

Reward: Scalar signal.
The agent's goal is to maximise this reward

Value Function: Tells the agent how good or bad a particular state is.

$$v_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots \mid S_t = s]$$

AlphaGo / AlphaZero



> What the agent "sees": States of the board

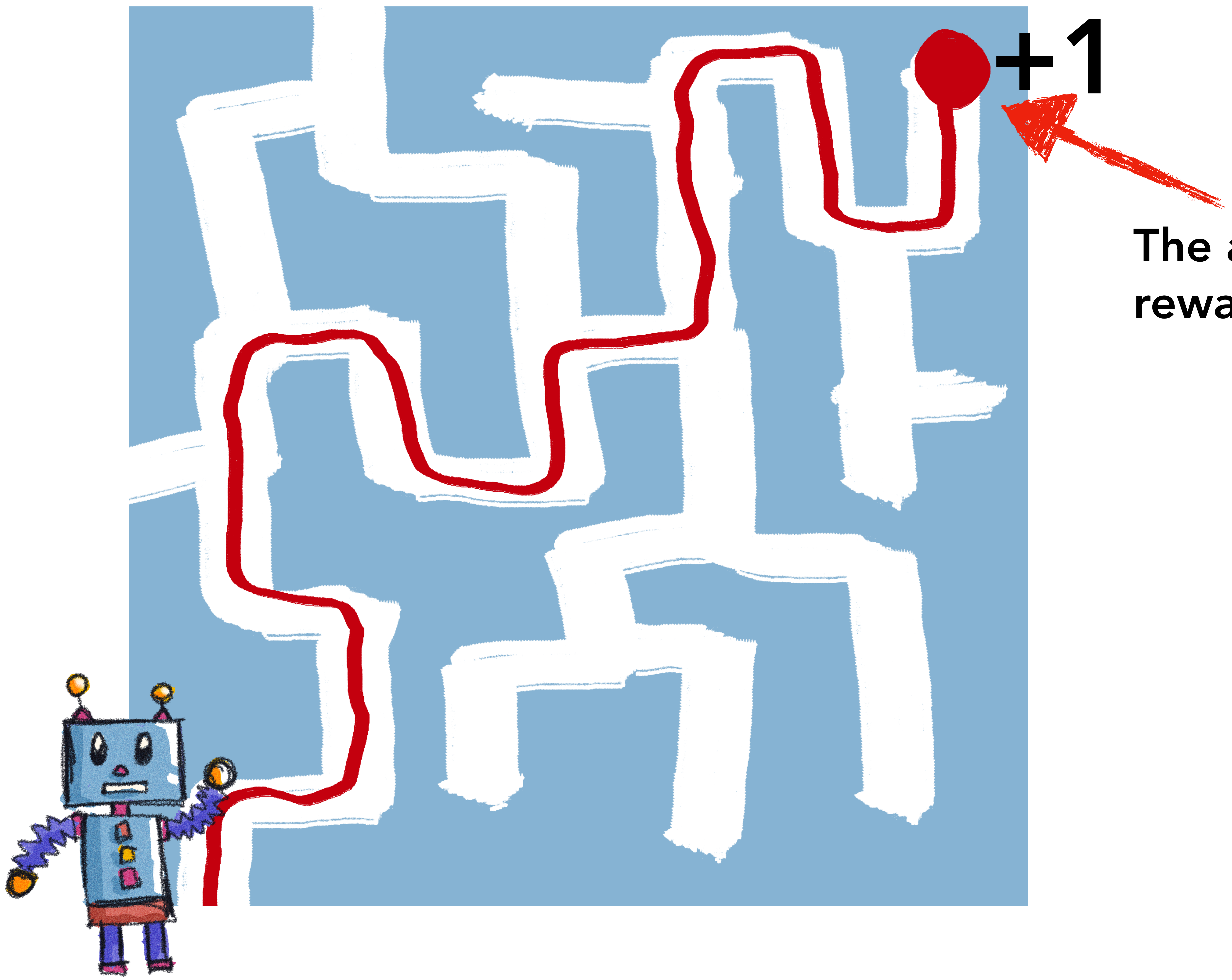
> Reward: At the end of the game, +1 or -1

> Actions: Valid actions

>Training: Trains by self-play

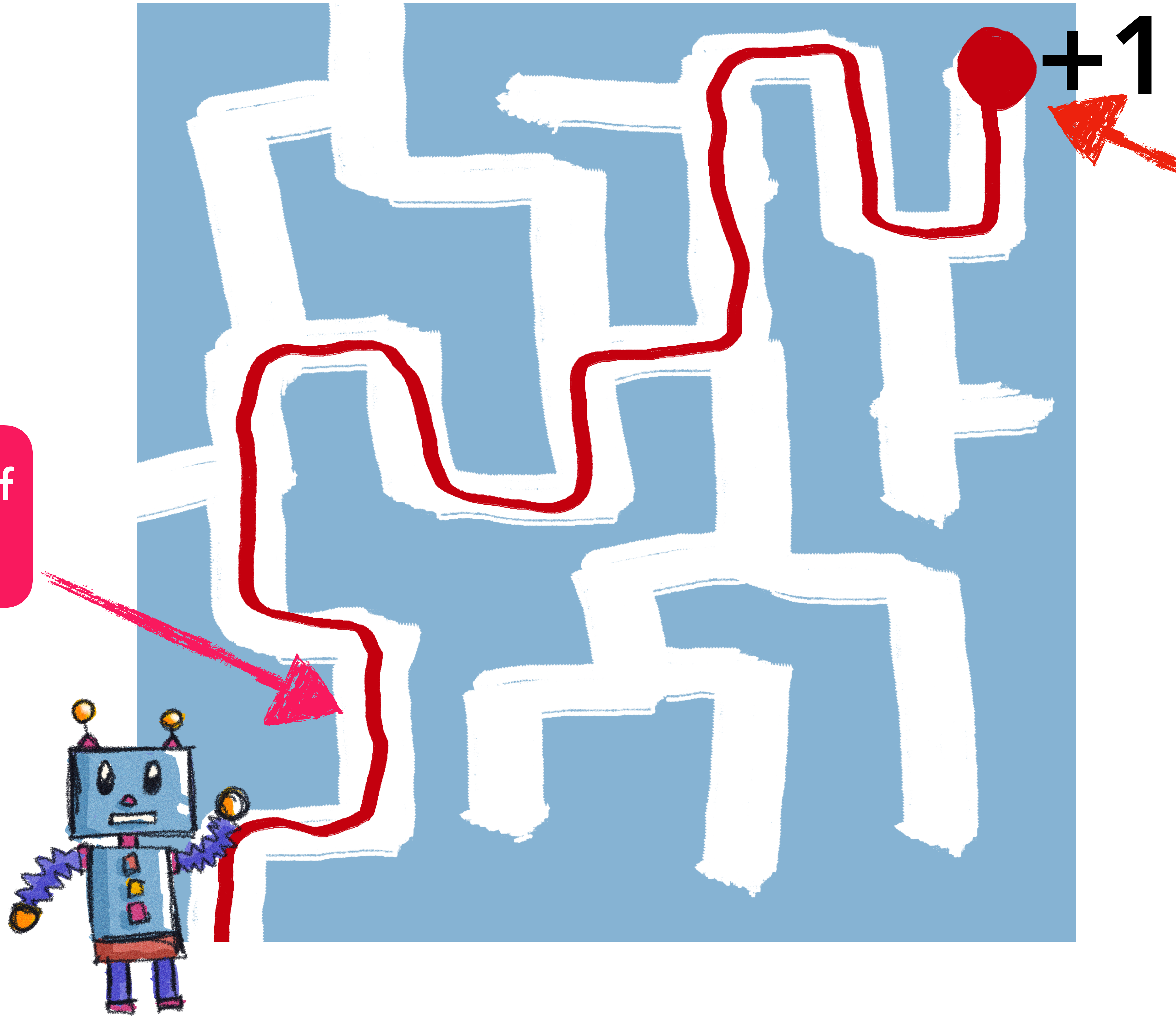
Silver, D., Schrittwieser, J., Simonyan, K. et al. Mastering the game of Go without human knowledge

Reinforcement Learning



The agent only gets a
reward at the very end!

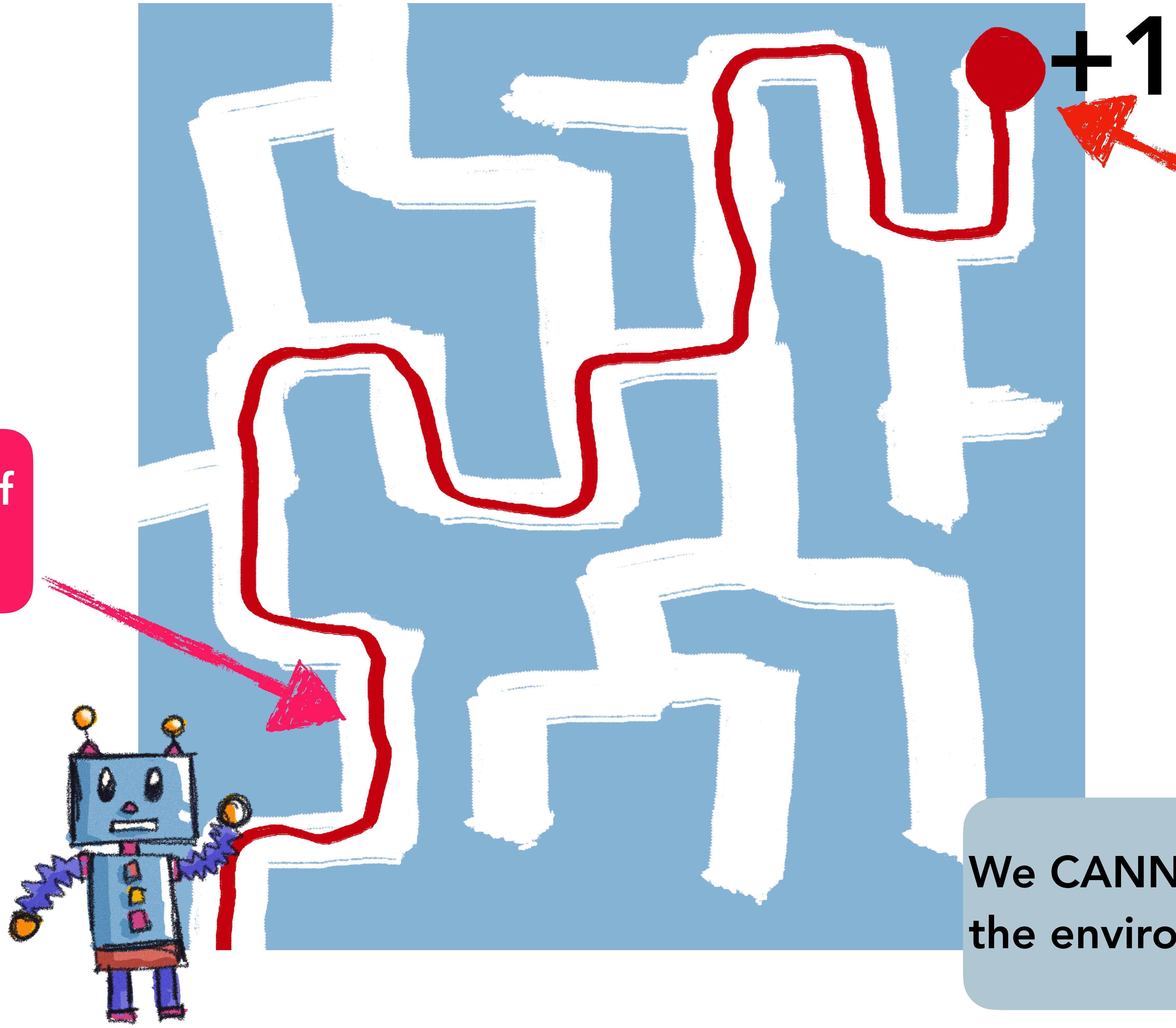
Reinforcement Learning



The agent only gets a reward at the very end!

We do not know the effect of an action taken here

Reinforcement Learning

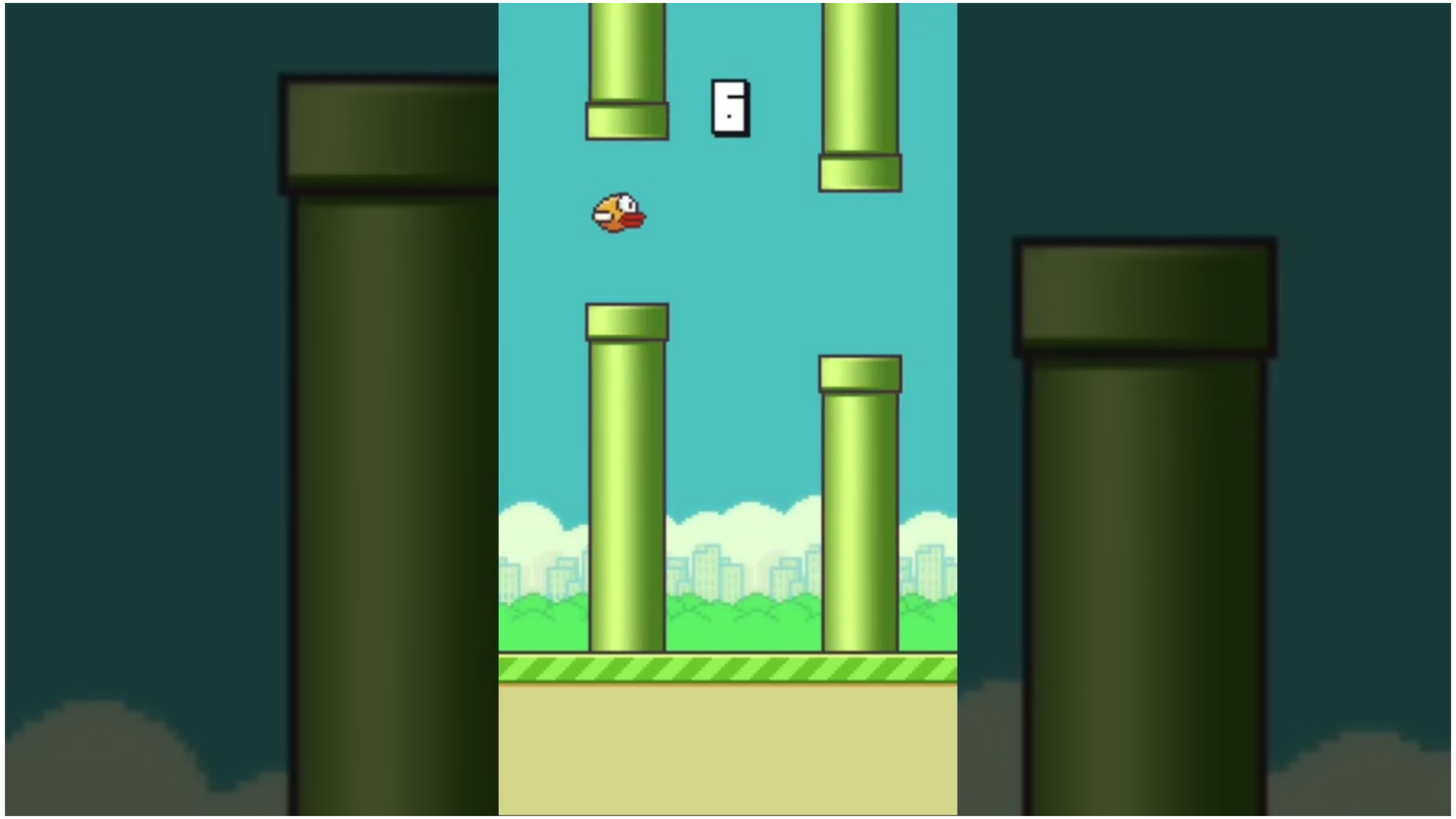


The agent only gets a reward at the very end!

We do not know the effect of an action taken here

We CANNOT get i.i.d data from the environment!

Reinforcement Learning



Code at https://github.com/ahanadeb/Flappy_birds

Challenges of RL

- A lot of the RL approaches which work in real life, are often backed by incomplete theoretical understanding
- A lot of exploration is required
- The agent needs to make a LOT of mistakes to learn

Challenges of RL

- A lot of the RL approaches which work in real life, are often backed by incomplete theoretical understanding
- A lot of exploration is required
- The agent needs to make a LOT of mistakes to learn

Solution?

> Design sample efficient algorithms with strong theoretical guarantees!

> Offline or off-policy learning!

Offline learning

Constraints: During the learning process, the agent CANNOT interact with the environment

Offline learning

Constraints: During the learning process, the agent CANNOT interact with the environment

Goal: Learning an optimal policy!

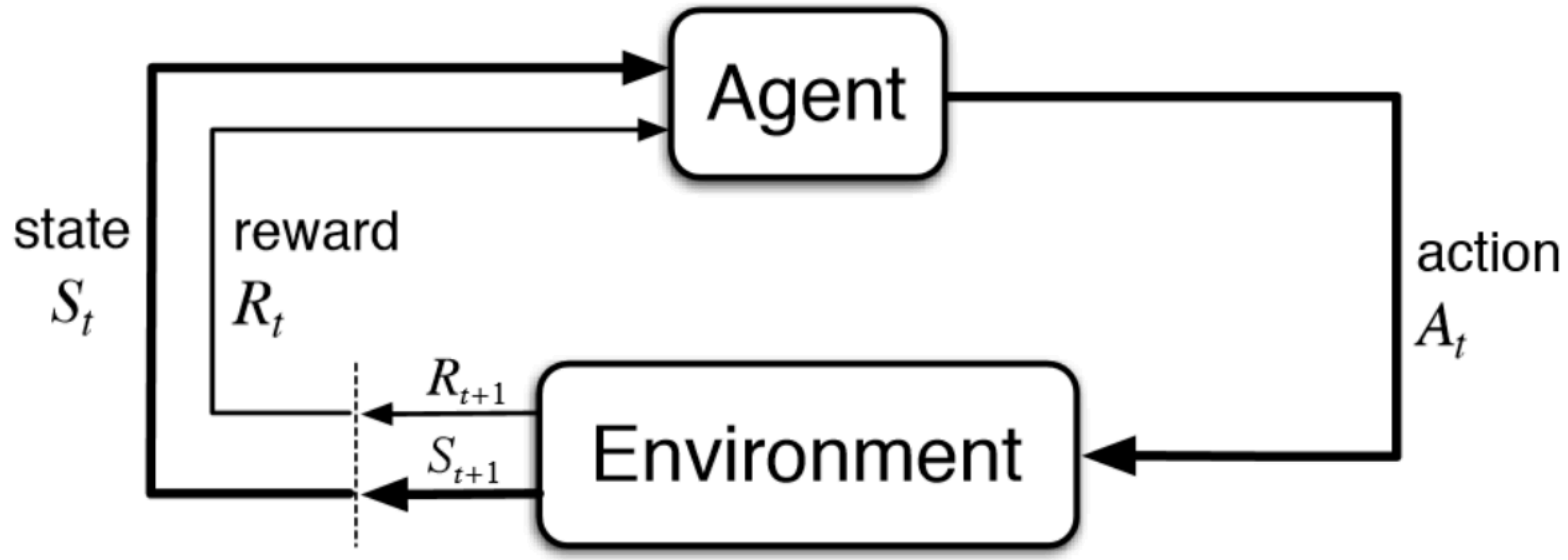
Offline learning

Constraints: During the learning process, the agent CANNOT interact with the environment

Goal: Learning an optimal policy!

Advantages:
> Efficient use of already available data!
> Allows efficient learning in environments where interaction with the environment is risky or costly or simply not possible

Markov Decision Process



> Agent interacts with environment, with interaction sequence $S_0, A_0, R_0, S_1, A_1, R_1 \dots$ (S_t, A_t and R_t are the state, action and reward at time-step t)

> In MDPS, the distributions on R_{t+1} and S_{t+1} are only dependent on S_t and A_t

Figure from "Reinforcement Learning: An introduction", Sutton and Barto[1]

Markov Decision Process

Value of state $V(s)$

Under policy π $\quad V^{\pi}(s) = \mathbb{E}\{R_t \,|\, s_t = s\} = \mathbb{E}_{\pi}\{\Sigma_{k=0}^{\infty} \gamma^k r_{t+k+1} \,|\, s_t = s\}$

"Reinforcement Learning: An introduction", Sutton and Barto

Markov Decision Process

Value of state $V(s)$

Under policy π

$$V^{\pi}(s) = \mathbb{E}\{R_t \,|\, s_t = s\} = \mathbb{E}_{\pi}\{\Sigma_{k=0}^{\infty}\gamma^k r_{t+k+1} \,|\, s_t = s\}$$

$$= \mathbb{E}_{\pi}\{r_{t+1} + \Sigma_{k=0}^{\infty}\gamma^k r_{t+k+2} \,|\, s_t = s\}$$

$$= \Sigma_a \pi(s,a) \Sigma_{s'} \mathbb{P}^a_{ss'} \left[R^a_{ss'} + \gamma \mathbb{E}_{\pi}\{\Sigma_{k=0}^{\infty}\gamma^k r_{t+k+2} \,|\, s_t = s\} \right]$$

$$= \Sigma_a \pi(s,a) \Sigma_{s'} \mathbb{P}^a_{ss'} \left[R^a_{ss'} + \gamma V^{\pi}(s') \right]$$

"Reinforcement Learning: An introduction", Sutton and Barto

Markov Decision Process

	Value of state $V(s)$	Value of a state-action pair $Q(s,a)$
Under policy π	$V^{\pi}(s) = \Sigma_a \pi(s,a) \Sigma_{s'} \mathbb{P}^a_{ss'} \left[R^a_{ss'} + \gamma V^{\pi}(s') \right]$	$Q^{\pi}(s,a) = \Sigma_{s'} \mathbb{P}^a_{ss'} \left[R^a_{ss'} + \gamma V^{\pi}(s') \right]$

"Reinforcement Learning: An introduction", Sutton and Barto

Markov Decision Process

	Value of state $V(s)$	Value of a state-action pair $Q(s,a)$
Under policy π	$V^{\pi}(s) = \Sigma_a \pi(s,a) \Sigma_{s'} \mathbb{P}^a_{ss'} \left[R^a_{ss'} + \gamma V^{\pi}(s') \right]$	$Q^{\pi}(s,a) = \Sigma_{s'} \mathbb{P}^a_{ss'} \left[R^a_{ss'} + \gamma V^{\pi}(s') \right]$
Under optimal policy $\pi*$	$V*(s) = max_{a \in A(s)} \Sigma_{s'} \mathbb{P}^a_{ss'} \left[R^a_{ss'} + \gamma V*(s') \right]$	$Q*(s,a) = \Sigma_{s'} \mathbb{P}^a_{ss'} \left[R^a_{ss'} + \gamma max_{a'} Q*(s',a') \right]$

"Reinforcement Learning: An introduction", Sutton and Barto

Non Markov Decision Process

> Non-Markovian Decision Processes exhibit explicit dependency on past events.

> Agent interacts with environment to create traces of Actions, Observations and Rewards:

$$a_0 o_0 r_0 a_1 o_1 r_1 \ldots a_H o_H r_H = \mathscr{E}_H$$

> Non-Markovian if doesn't satisfy

$o_{t+1} \perp a_0 o_0 \ldots o_{t-1} a_t \,|\, o_t, a_{t+1}$ or $r_{t+1} \perp a_0 o_0 \ldots o_{t-1} a_t \,|\, o_t, a_{t+1}$.

> History: Defined as a sequence, $h_t = a_0 o_0 \ldots a_t o_t \in (AO)^{t+1} = \mathscr{H}^t$

NMDPs

> The **unrestricted dynamics** of NMDPs make them intractactable to learn

> This has steered research effort towards **tractable subclasses**-

POMDPs: Agent operates under partial observability: the state space is hidden and mapped to an observation space.

More POMDP domains http://www.pomdp.org/

NMDPs

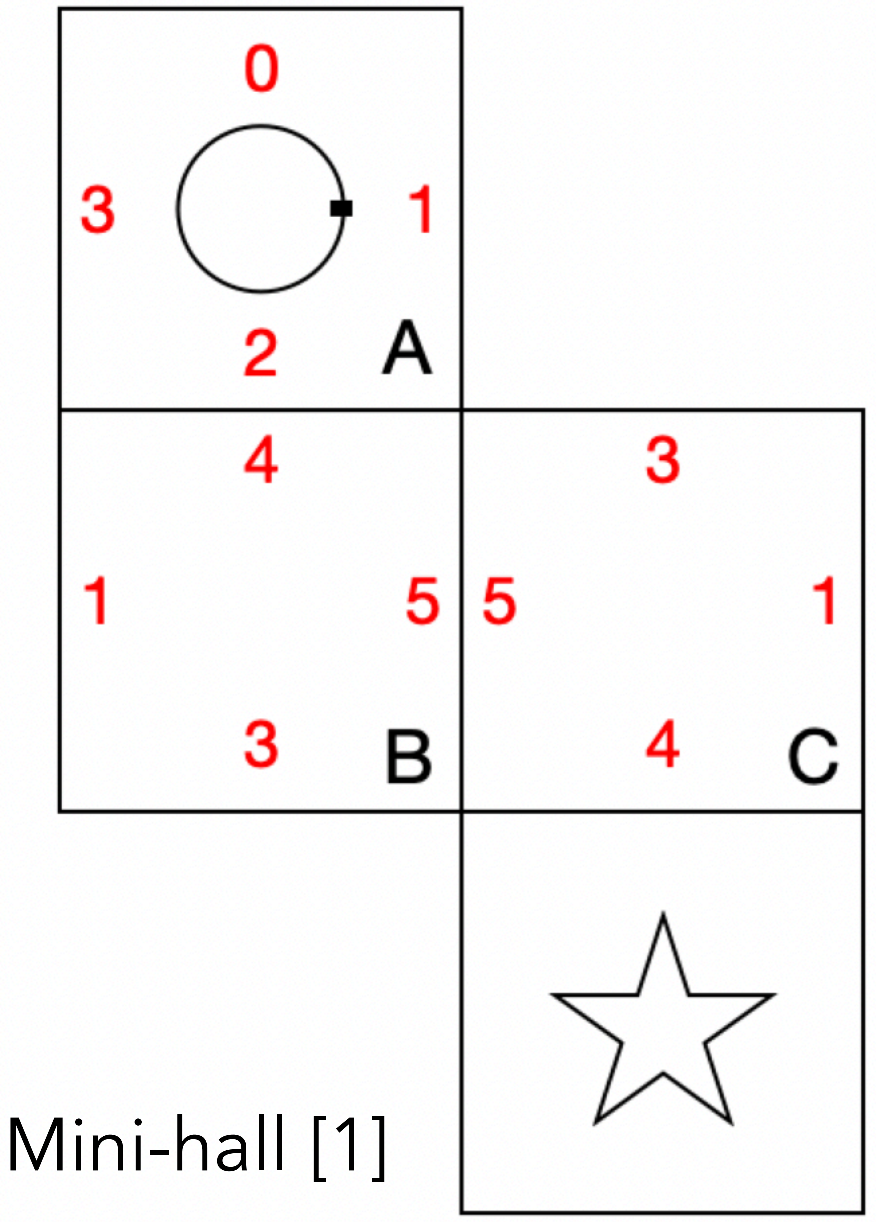
> The **unrestricted dynamics** of NMDPs make them intractactable to learn

> This has steered research effort towards **tractable subclasses**-

POMDPs: Agent operates under partial observability: the state space is hidden and mapped to an observation space.

Example:

- A robot with a malfunctioning sensor
- Partially occluded vision, ex: Mini-hall domain[1]



Mini-hall [1]

[1] Littman et al., 1997, Learning Policies for Partially Observable Environments: Scaling Up

NMDPs

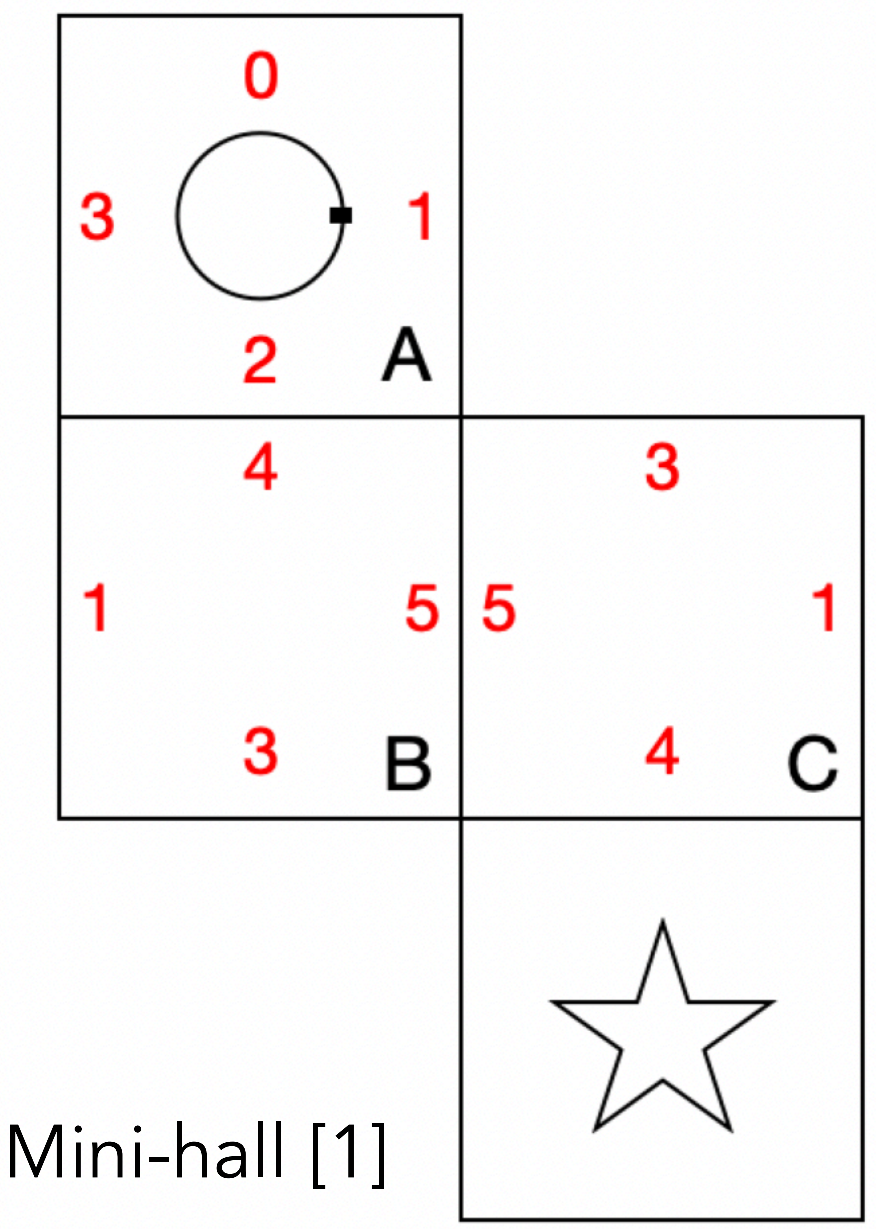
> The **unrestricted dynamics** of NMDPs make them intractactable to learn

> This has steered research effort towards **tractable subclasses**-

POMDPs: Agent operates under partial observability: the state space is hidden and mapped to an observation space.

Example:

- A robot with a malfunctioning sensor
- Partially occluded vision, ex: Mini-hall domain[1]



Mini-hall [1]

However, POMDPs are often intractable[1]!

Sub-problems of POMDPs-

>undercomplete POMDPs

>few-step reachability

>ergodicity

>few-step decodability

> weakly-revealing, etc

[1] Papadimitriou and Tsitsiklis, 1987, The complexity of Markov decision Processes

NMDPs

> The **unrestricted dynamics** of NMDPs make them intractactable to learn

> This has steered research effort towards **tractable subclasses**-

RDPs or Regular Decision Processes:

> NMDP where the underlying dynamics can be represented by a Probabilistic Deterministic Finite Automaton

> can capture complex temporal dependencies

[1] Kaelbling et al., 1998, Planning and acting in partially observable stochastic domain

NMDPs

> The **unrestricted dynamics** of NMDPs make them intractactable to learn

> This has steered research effort towards **tractable subclasses**-

RDPs or Regular Decision Processes:

> NMDP where the underlying dynamics can be represented by a Probabilistic Deterministic Finite Automaton

> can capture complex temporal dependencies

RDPs are essentially a subclass of POMDPs[1] where hidden states are determined by the history of observation!

[1] Kaelbling et al., 1998, Planning and acting in partially observable stochastic domain

NMDPs

> The **unrestricted dynamics** of NMDPs make them intractactable to learn

> This has steered research effort towards **tractable subclasses**-

RDPs or Regular Decision Processes:

> NMDP where the underlying dynamics can be rep

Deterministic Finite Automaton

> can capture complex temporal dependencies

RDPs are essentially a subclass of POMDPs[1] wh

determined by the history of observation!

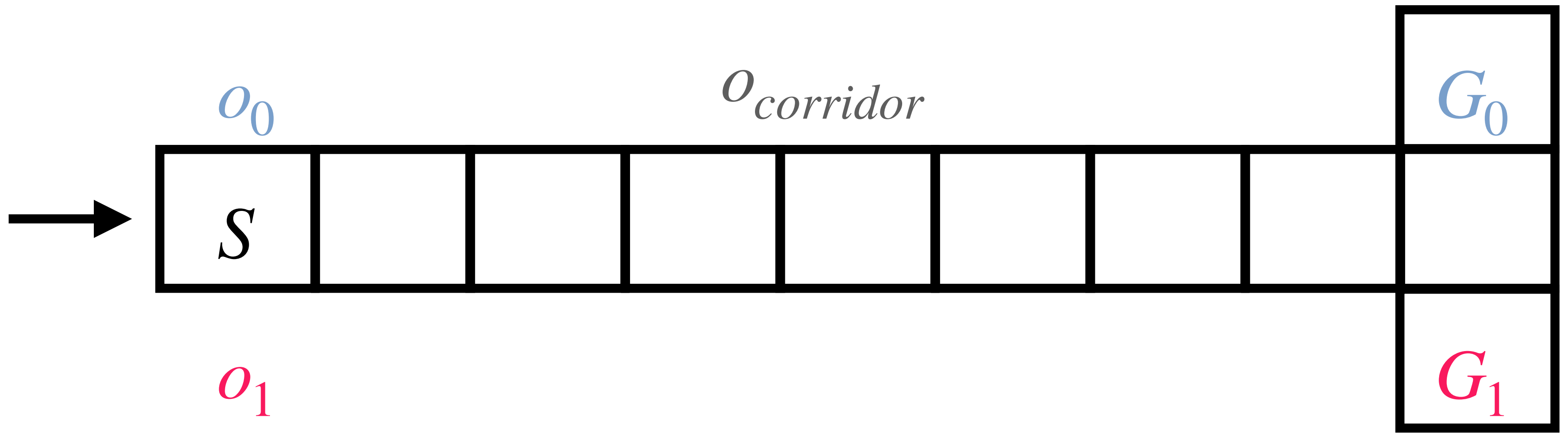
> Learning the automaton underlying an RDP amounts to learning a representation of the histories!

> Now we can work with the associated MDP!

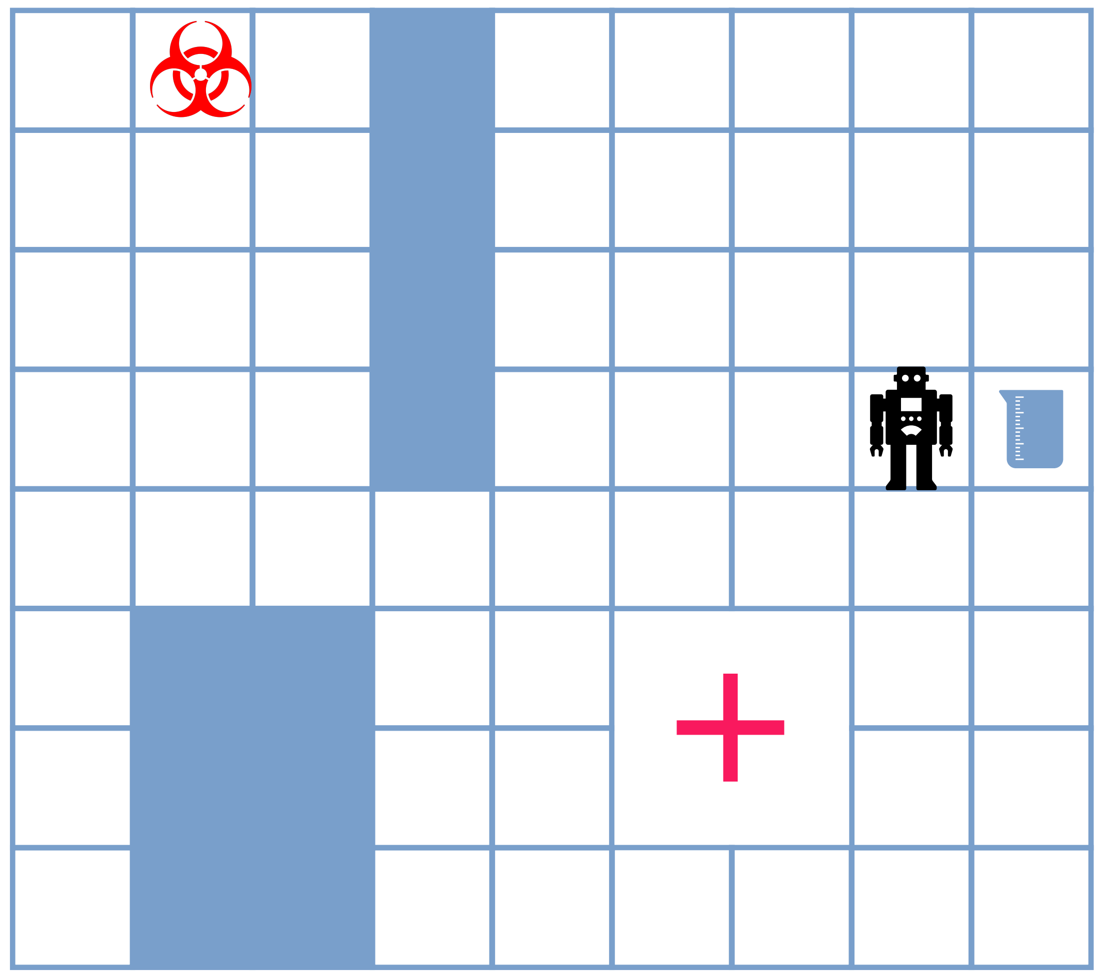
[1] Kaelbling et al., 1998, Planning and acting in partially observable stochastic domain

Regular Decision Process

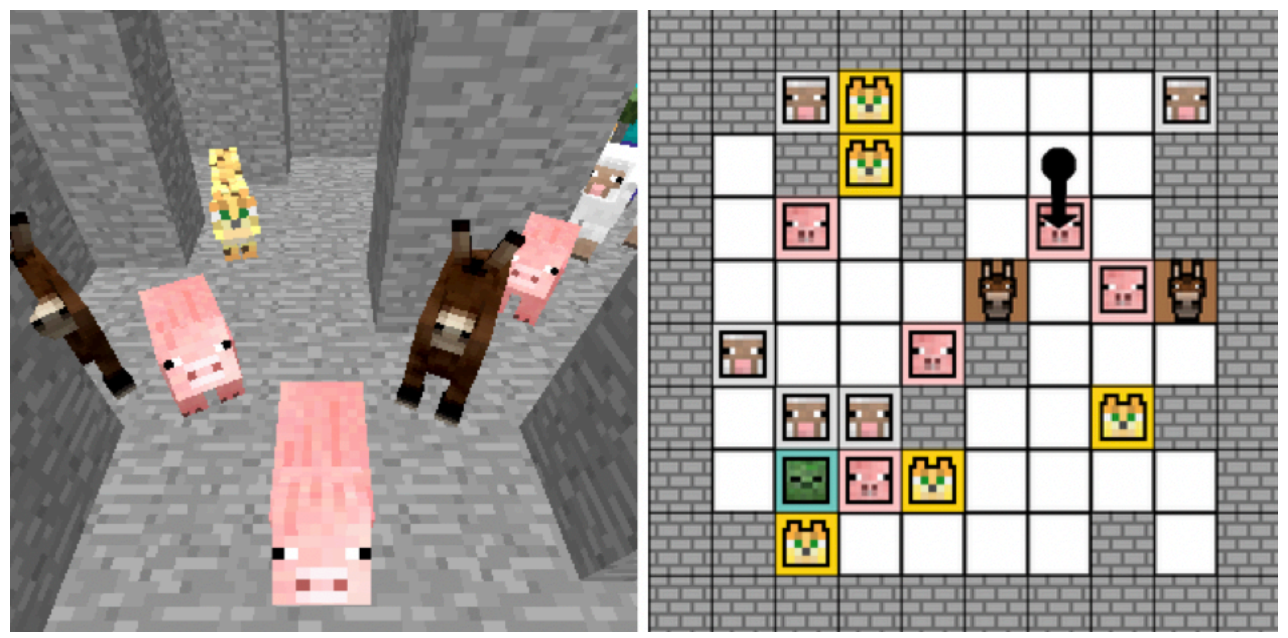
> In Regular Decision Processes, the next observation o_{t+1} and next reward r_{t+1} depend regularly on the history of the interaction (and not just the last observation and action).



T-maze[1]



Grid [2]



Minecraft multi-task environment [3]

[1] Bram Bakker, 2001 Reinforcement Learning with long short-term memory
[2] Alfredo Gabaldon, 2011, Non-markovian control in the situation calculus
[3] Oh et al. Zero-Shot Task Generalization with Multi-Task Deep Reinforcement Learning

Regular Decision Process

> **A general non-Markov episodic decision process** can be given as $\langle O, A, R, \bar{T}, \bar{R}, H \rangle$, with horizon $H \in \mathbb{N}_+$ where $\bar{T} : \mathscr{H} \times A \rightarrow \Delta(O)$ and $\bar{R} : \mathscr{H} \times A \rightarrow \Delta(R)$.

> We assume the tabular case.

> A policy here is a mapping from history to action, $\pi : \mathscr{H} \rightarrow \Delta(A)$ and we can define the values as

$$V_t^\pi(h_t) = \mathbb{E}[\sum_{t+1}^{H} r_i \,|\, h_t, \pi]$$

Regular Decision Process

> **A general non-Markov episodic decision process** can be given as $\langle O, A, R, \bar{T}, \bar{R}, H \rangle$, with horizon $H \in \mathbb{N}_+$ where $\bar{T} : \mathscr{H} \times A \to \Delta(O)$ and $\bar{R} : \mathscr{H} \times A \to \Delta(R)$.

> We assume the tabular case.

> A policy here is a mapping from history to action, $\pi : \mathscr{H} \to \Delta(A)$ and we can define the

val

> **RDPs : In a regular decision process, $\bar{T}$ and $\bar{R}$ are regular functions.**

> **Regularity: A function $f : \Sigma^* \to \Omega$ is considered regular for finite Ω if, for each $\omega \in \Omega$ the set $f^{-1}(\omega)$ is a regular language].**

Regular Decision Process

Formal Definition : RDP $\langle Q, \Sigma, \Omega, \tau, \theta_o, \theta_r, q_0 \rangle$ where,

> Q is the set of automaton states

> $\Sigma = AO$ is the set of input symbol

> $\tau : Q \times AO \rightarrow Q$ is the transition function

> $\theta_o : Q \times A \rightarrow \Delta(O)$ and $\theta_r : Q \times A \rightarrow \Delta(R)$ are the two output functions.

Regular Decision Process

Formal Definition : RDP $\langle Q, \Sigma, \Omega, \tau, \theta_o, \theta_r, q_0 \rangle$ where,

> Q is the set of automaton states

> $\Sigma = AO$ is the set of input symbol

> $\tau : Q \times AO \to Q$ is the transition function

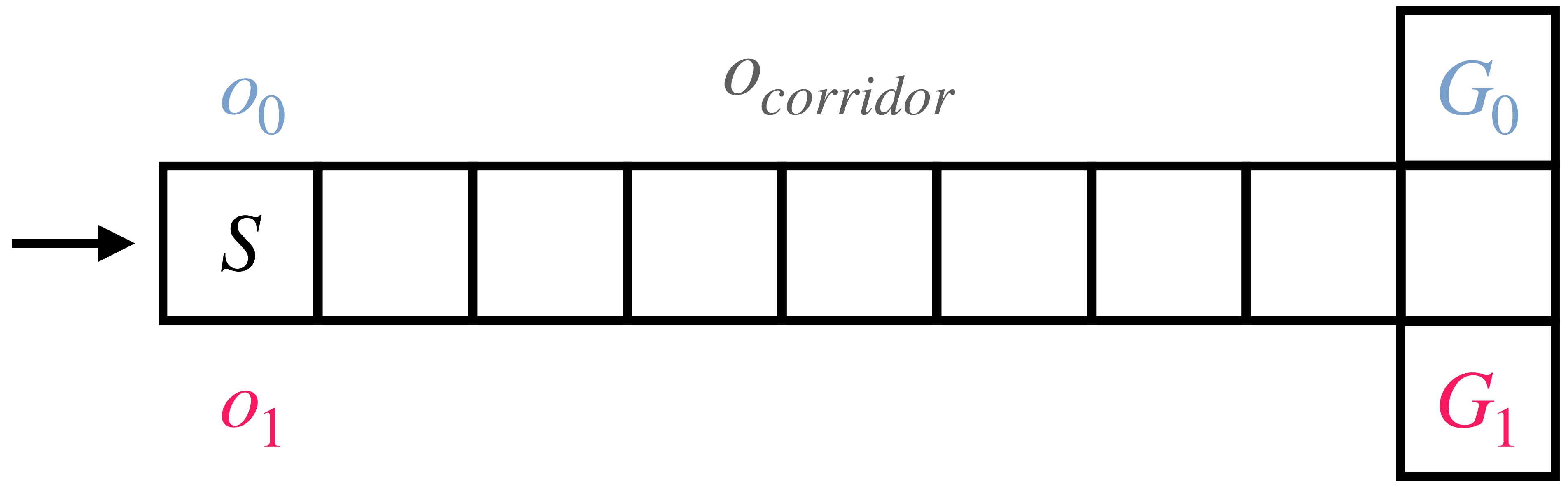
> $\theta_o : Q \times A \to \Delta(O)$ and $\theta_r : Q \times A \to \Delta(R)$ are the two output functions.

Cannot be observed by the agent!

Objective

> To learn minimal automata from data (episodes) from an RDP, since the number of histories is exponential in the horizon.

> For any RDP, the distribution over episodes can be modeled as a PDFA.

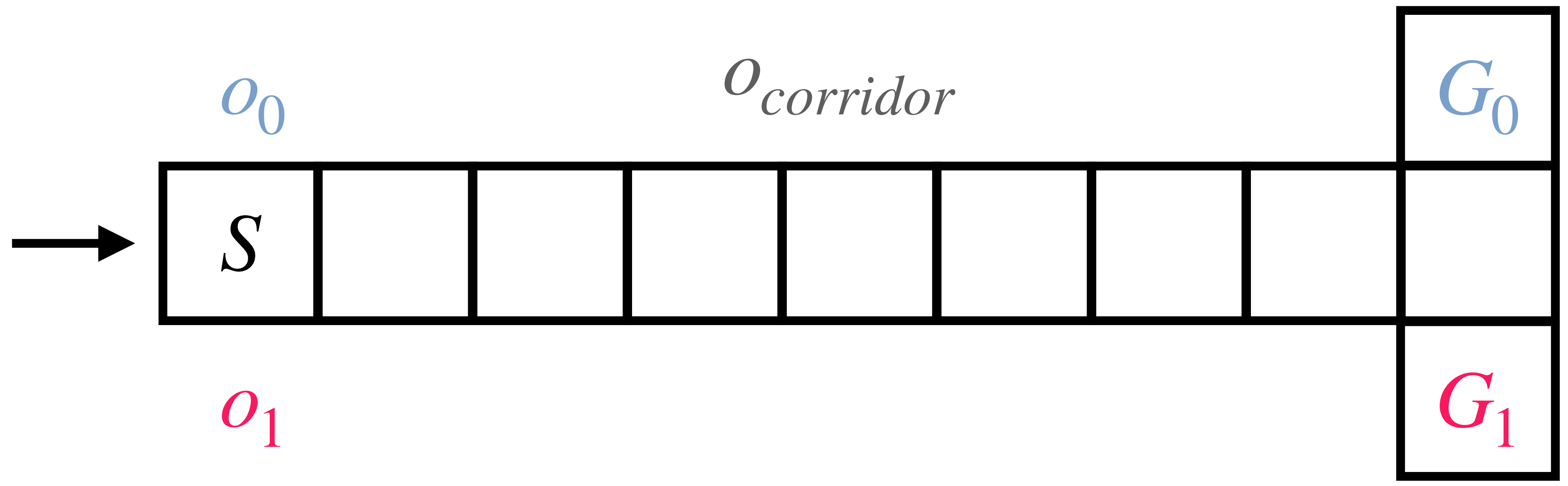


T-maze

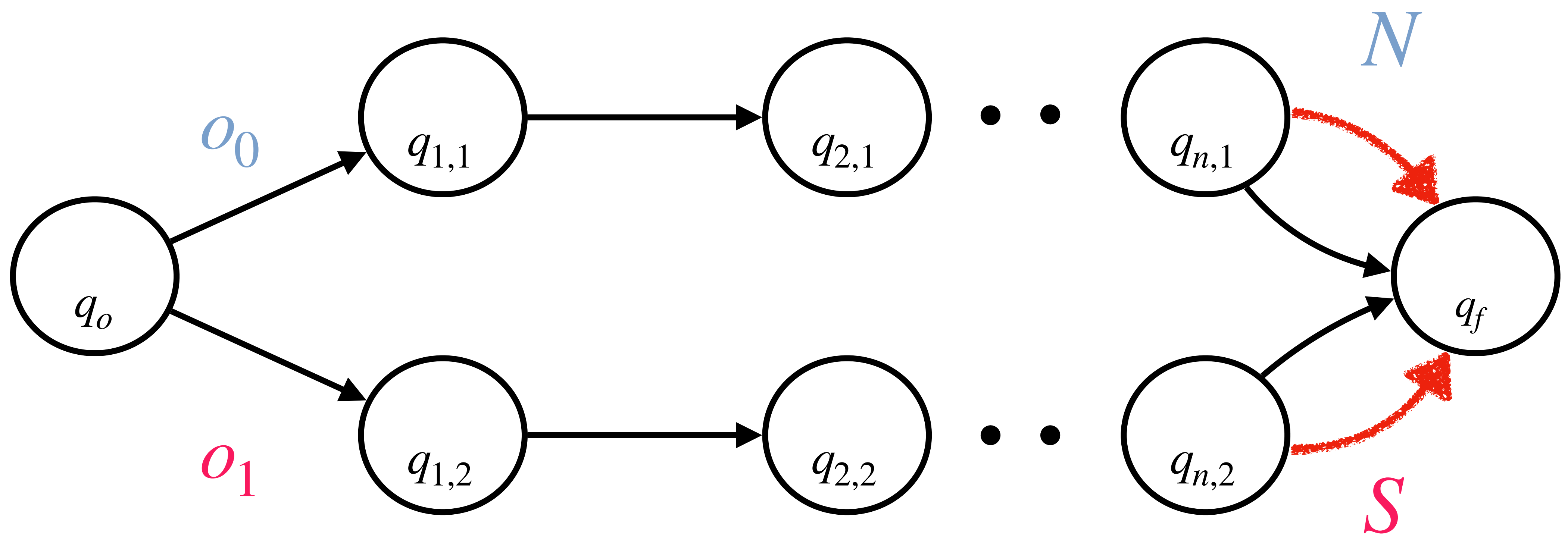
Objective

> To learn minimal automata from data (episodes) from an RDP, since the number of histories is exponential in the horizon.

> For any RDP, the distribution over episodes can be modeled as a PDFA.



T-maze



Corresponding abstraction

Regular Policy

> A policy π is a **regular policy** if, for an RDP R, for two histories $h, h' \in \mathscr{H}$, for which $\bar{\tau}(h) = \bar{\tau}(h')$, the policy over the histories are also the same, i.e., $\pi(h) = \pi(h')$.

> Every RDP has a optimal policy that is also **regular**.

> We also have

$$\mathbb{P}(e_{t+1:H} \,|\, h_t, \pi) = \mathbb{P}(e_{t+1:H} \,|\, h'_t, \pi)$$

where $e_{i:j} = a_i o_i r_i \ldots a_j o_j r_j$.

Offline RL in RDPs

> Problem Statement: Given a dataset D of episodes collected from unknown RDP R with behavioral policy π^b, our goal is to compute near-optimal policy for R, using the smallest D, and find PAC sample complexity guarantees on the bound on required number of episodes, $|D|$.

Offline RL in RDPs

> We propose a modification on Cipollone 2023[1], using Count-Min-Sketch to improve space complexity

Cipollone et al. Provably efficient offline reinforcement learning in regular decision processes, NEURIPS 2023

Offline RL in RDPs

> We propose a modification on Cipollone 2023[1], using Count-Min-Sketch to improve space complexity
> A language based approach to remove the dependency on some complicated distinguishability parameters

Cipollone et al. Provably efficient offline reinforcement learning in regular decision processes, NEURIPS 2023

Offline RL in RDPs

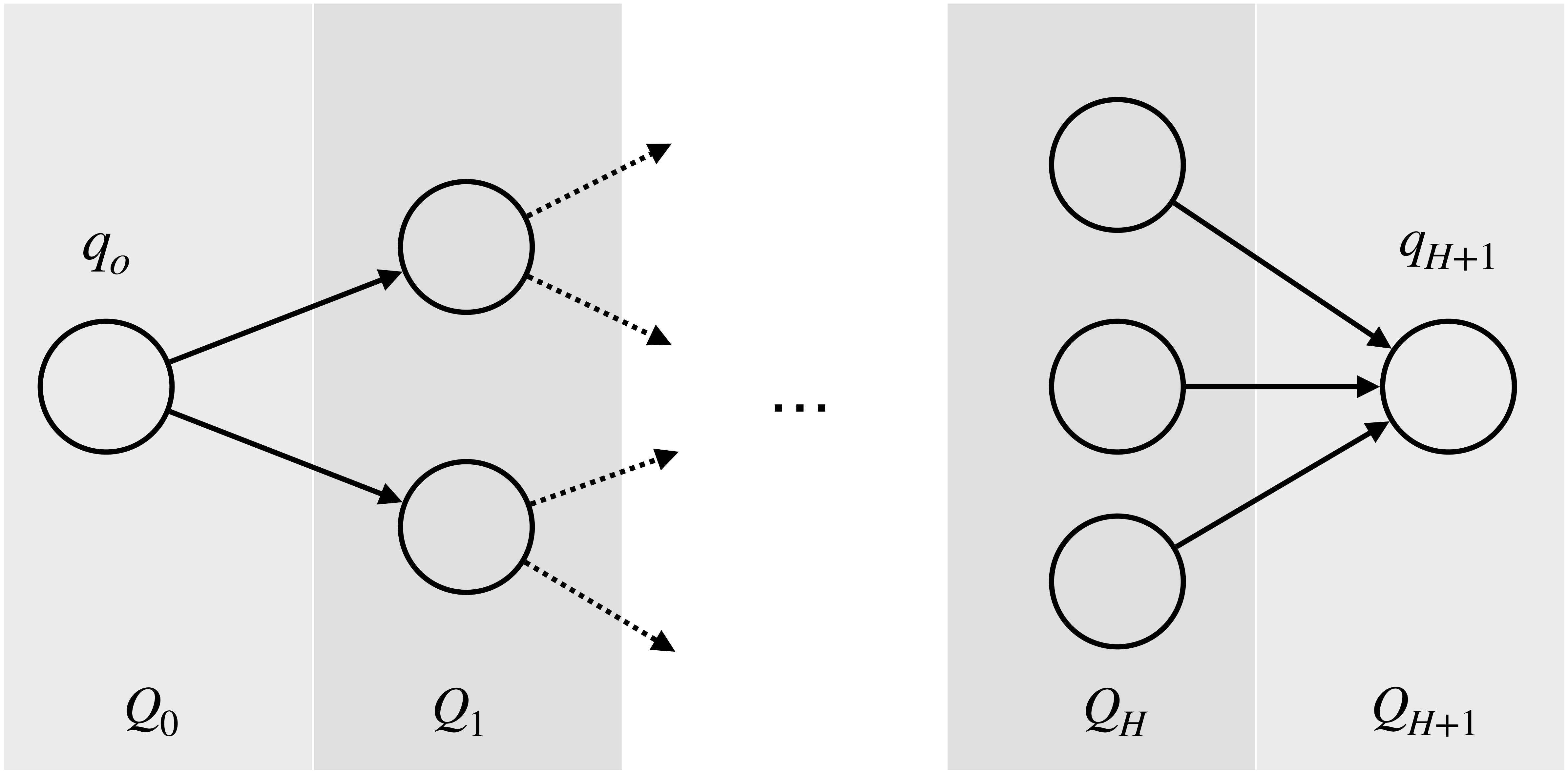
> We propose a modification on Cipollone 2023[1], using Count-Min-Sketch to improve space complexity
> A language based approach to remove the dependency on some complicated distinguishability parameters
> Respective bounds on sample complexity for offline RL in RDPs.

Cipollone et al. Provably efficient offline reinforcement learning in regular decision processes, NEURIPS 2023

ADACT-H

> We assume a fixed horizon setting, and an end symbol after H transitions.

> Goal is to build a set of states $Q = Q_o \sqcup Q_1 \sqcup \ldots Q_{H+1}$ and $\tau(q_t) \in Q_{t+1}$ for any $q \in Q_t$.

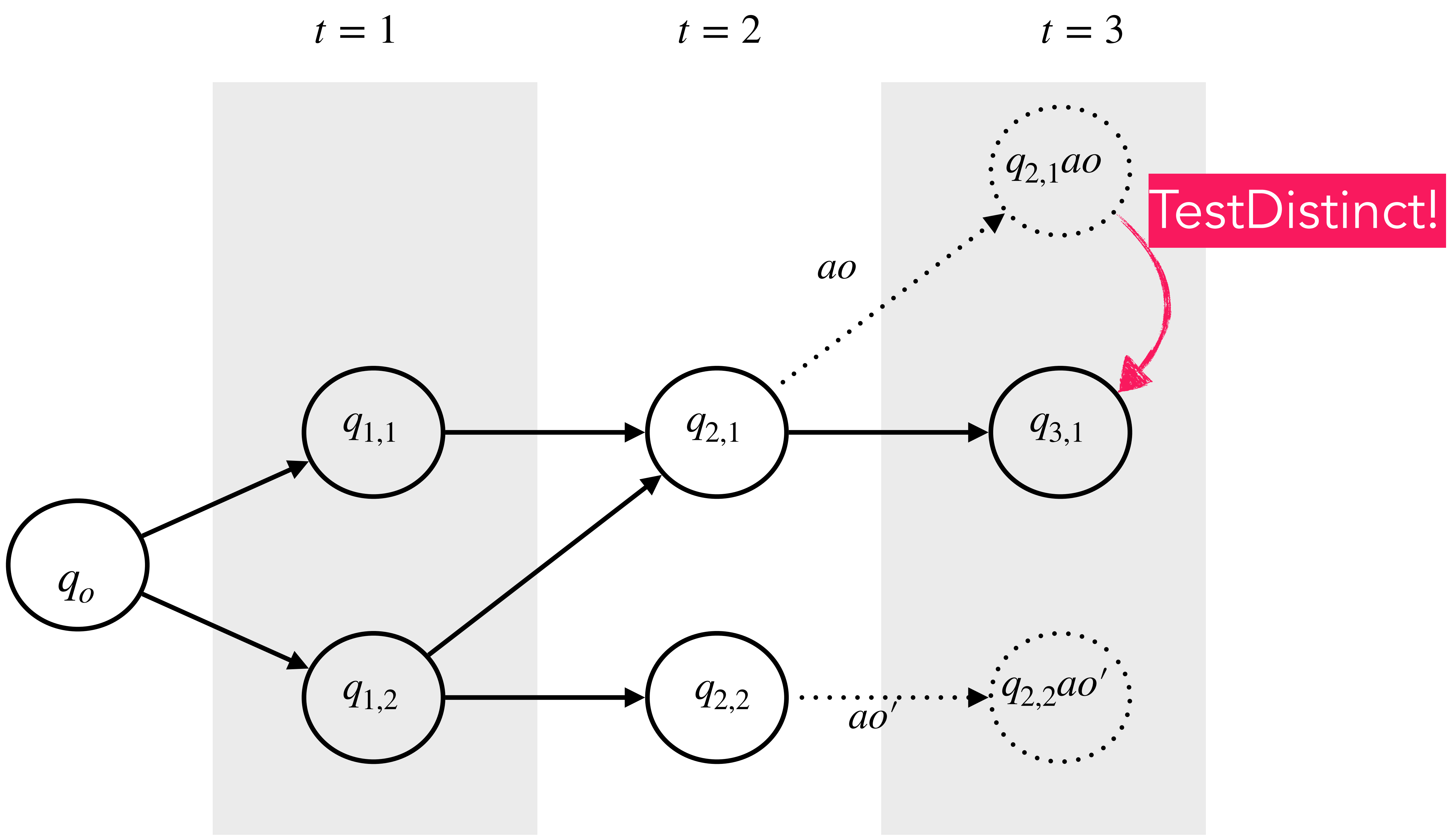


ADACT-H

> The set of states is built by time-steps.

> At iteration 3, we have **safe states** $Q_3 = \{q_{3,1}\}$, and **candidate states** $Q_{c,3} = \{q_{2,1}ao, q_{2,2}ao'\}$.

> **TestDistinct compares estimates of the suffixes**, i.e. $\hat{\mathbb{P}}(e_{4:H} \,|\, q_{3,1}, \pi^b)$ and $\hat{\mathbb{P}}(e_{4:H} \,|\, q_{2,1}ao, \pi^b)$.

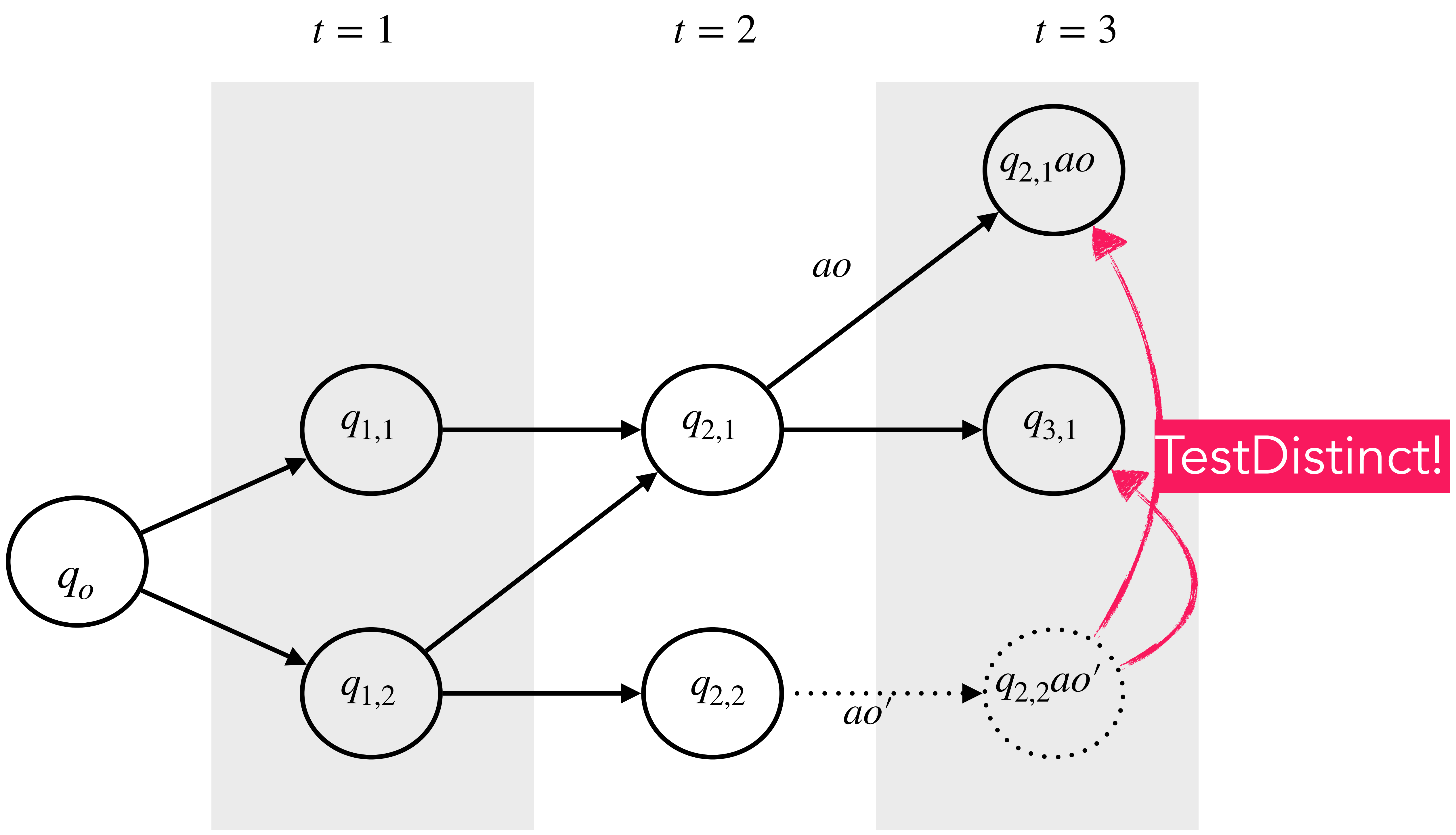


ADACT-H

> If **TestDistinct** returns **TRUE,** $q_{2,1}ao$ is **promoted** as a new state.

> Candidate state $q_{2,2}ao'$ is tested against the safe states at $t = 3$.

> Now we have to compare $\hat{\mathbb{P}}(e_{4:H} | q_{2,2}ao', \pi^b)$ **with** $\hat{\mathbb{P}}(e_{4:H} | q_{3,1}, \pi^b)$ and $\hat{\mathbb{P}}(e_{4:H} | q_{2,1}ao, \pi^b)$, and so on



ADACT-H

Input: Dataset $\mathcal{D}$ containing N traces in $\mathcal{E}_H$, failure probability $0 < \delta < 1$
Output: Set $\mathcal{Q}$ of RDP states, transition function $\tau : \mathcal{Q} \times \mathcal{AO} \to \mathcal{Q}$

$\mathcal{Q}_0 \leftarrow \{q_0\}, \mathcal{X}(q_0) \leftarrow \mathcal{D}$ `// initial state`
for $t = 0, \ldots, H$ **do**
  $\mathcal{Q}_{\mathsf{c},t+1} \leftarrow \{qao \mid q \in \mathcal{Q}_t, ao \in \mathcal{AO}\}$ `// get candidate states`
  foreach $qao \in \mathcal{Q}_{\mathsf{c},t+1}$ **do** $\mathcal{X}(qao) \leftarrow \{e_{t+1:H} \mid aroe_{t+1:H} \in \mathcal{X}(q)\}$ `// compute suffixes`
  $q_{\mathsf{m}}a_{\mathsf{m}}o_{\mathsf{m}} \leftarrow \arg\max_{qao \in \mathcal{Q}_{\mathsf{c},t+1}} |\mathcal{X}(qao)|$ `// most common candidate`
  $\mathcal{Q}_{t+1} \leftarrow \{q_{\mathsf{m}}a_{\mathsf{m}}o_{\mathsf{m}}\}, \tau(q_{\mathsf{m}}, a_{\mathsf{m}}o_{\mathsf{m}}) = q_{\mathsf{m}}a_{\mathsf{m}}o_{\mathsf{m}}$ `// promote candidate`
  $\mathcal{Q}_{\mathsf{c},t+1} \leftarrow \mathcal{Q}_{\mathsf{c},t+1} \setminus \{q_{\mathsf{m}}a_{\mathsf{m}}o_{\mathsf{m}}\}$ `// remove from candidate states`
  for $qao \in \mathcal{Q}_{\mathsf{c},t+1}$ **do**
    $\mathit{Similar} \leftarrow \{q' \in \mathcal{Q}_{t+1} \mid \text{not } \textsc{TestDistinct}(t, \mathcal{X}(qao), \mathcal{X}(q'), \delta)\}$ `// confidence test`
    if $\mathit{Similar} = \emptyset$ **then** $\mathcal{Q}_{t+1} \leftarrow \mathcal{Q}_{t+1} \cup \{qao\}, \tau(q, ao) = qao$ `// promote candidate`
    else $q' \leftarrow$ element in $\mathit{Similar}$, $\tau(q, ao) = q', \mathcal{X}(q') \leftarrow \mathcal{X}(q') \cup \mathcal{X}(qao)$ `// merge states`
  end
end
return $\mathcal{Q}_0 \cup \cdots \cup \mathcal{Q}_{H+1}, \tau$

Function $\textsc{TestDistinct}(t, \mathcal{X}_1, \mathcal{X}_2, \delta)$
  return $L^{\mathsf{p}}_{\infty}(\mathcal{X}_1, \mathcal{X}_2) \geq \sqrt{2 \log(8(ARO)^{H-t}/\delta) / \min(|\mathcal{X}_1|, |\mathcal{X}_2|)}$

State Similarity

FUNCTION(TESTDISTINCT)

Input: Time t, multisets $\mathcal{X}_1$ and $\mathcal{X}_2$ of traces, failure probability $0 < \delta < 1$

Output: True if $\mathcal{X}_1$ and $\mathcal{X}_2$ are regarded as distinct, False otherwise

return $L_\infty^{\mathsf{p}}(\mathcal{X}_1, \mathcal{X}_2) \geq \sqrt{2 \log(8(ARO)^{H-t}/\delta) / \max(|\mathcal{X}_1|, |\mathcal{X}_2|)}$

> Two dissimilar states q and q' must satisfy $p_q \neq p_{q'}$.

State Similarity

FUNCTION(TESTDISTINCT)

Input: Time t, multisets $\mathcal{X}_1$ and $\mathcal{X}_2$ of traces, failure probability $0 < \delta < 1$

Output: True if $\mathcal{X}_1$ and $\mathcal{X}_2$ are regarded as distinct, False otherwise

return $L_\infty^{\mathsf{p}}(\mathcal{X}_1, \mathcal{X}_2) \geq \sqrt{2 \log(8(ARO)^{H-t}/\delta) / \max(|\mathcal{X}_1|, |\mathcal{X}_2|)}$

> Two dissimilar states q and q' must satisfy $p_q \neq p_{q'}$.

> In ADACT-H, the metric used is the L_∞^p metric, which is defined as,

$L_\infty^p(p_1, p_2) = max\,|\,p_1(e*) - p_2(e*)\,|$ or more accurately $L_\infty^p(\hat{p}_1, \hat{p}_2) = max\,|\,\hat{p}_1(e*) - \hat{p}_2(e*)\,|$.

where $\hat{p}_i(e) = \sum_{x \in \mathcal{X}_q} \mathbb{I}(x = e)/|\mathcal{X}_q|$ and $\mathcal{X}_{q_t} = \{e_{t:H} \,|\, e_{0:t-1}e_{t:H} \in D$ and $\bar{\tau}(h_{t-1}) = q_t\}$.

State Similarity

> In ADACT-H, the metric used is the L_∞^p metric, which is defined as,

$L_\infty^p(p_1, p_2) = max|p_1(e*) - p_2(e*)|$ or more accurately $L_\infty^p(\hat{p}_1, \hat{p}_2) = max|\hat{p}_1(e*) - \hat{p}_2(e*)|$.

where $\hat{p}_i(e) = \sum_{x \in \mathcal{X}_q} \mathbb{I}(x = e)/|\mathcal{X}_q|$ and $\mathcal{X}_{q_t} = \{e_{t:H} | e_{0:t-1}e_{t:H} \in D$ and $\bar{\tau}(h_{t-1}) = q_t\}$.

Distinguishability Assumption

For any two distinct states $q, q' \in Q_t$ and suffix $e_{t:H}$, for an RDP R, the L_∞^p-distinguishability must be greater than $\mu_0 \geq 0$, i.e.,

$$L_\infty^p(p_q(e_{t:H}), p_{q'}(e_{t:H})) \geq \mu_0$$

State Similarity

> **Occupancy:** Occupancy of state-action-observation pair $q_t, a_t o_t$ under policy π

$$d_t^\pi(q_t, a_t o_t) = \Sigma_{(q,ao)\in\tau^{-1}(q_t)} d_{t-1}^\pi(q, ao) \cdot \pi(a_t \mid q_t) \cdot \theta_0(o_t \mid q_t, a_t),\ t > 0$$

State Similarity

> **Occupancy:** Occupancy of state-action-observation pair $q_t, a_t o_t$ under policy π

$$d_t^{\pi}(q_t, a_t o_t) = \Sigma_{(q,ao)\in\tau^{-1}(q_t)} d_{t-1}^{\pi}(q, ao) \cdot \pi(a_t | q_t) \cdot \theta_0(o_t | q_t, a_t),\ t > 0$$

> **Single Policy RDP concentrability coefficient**

> We define the **single-policy RDP concentrability coefficient** of RDP **R** with behavorial policy π^b

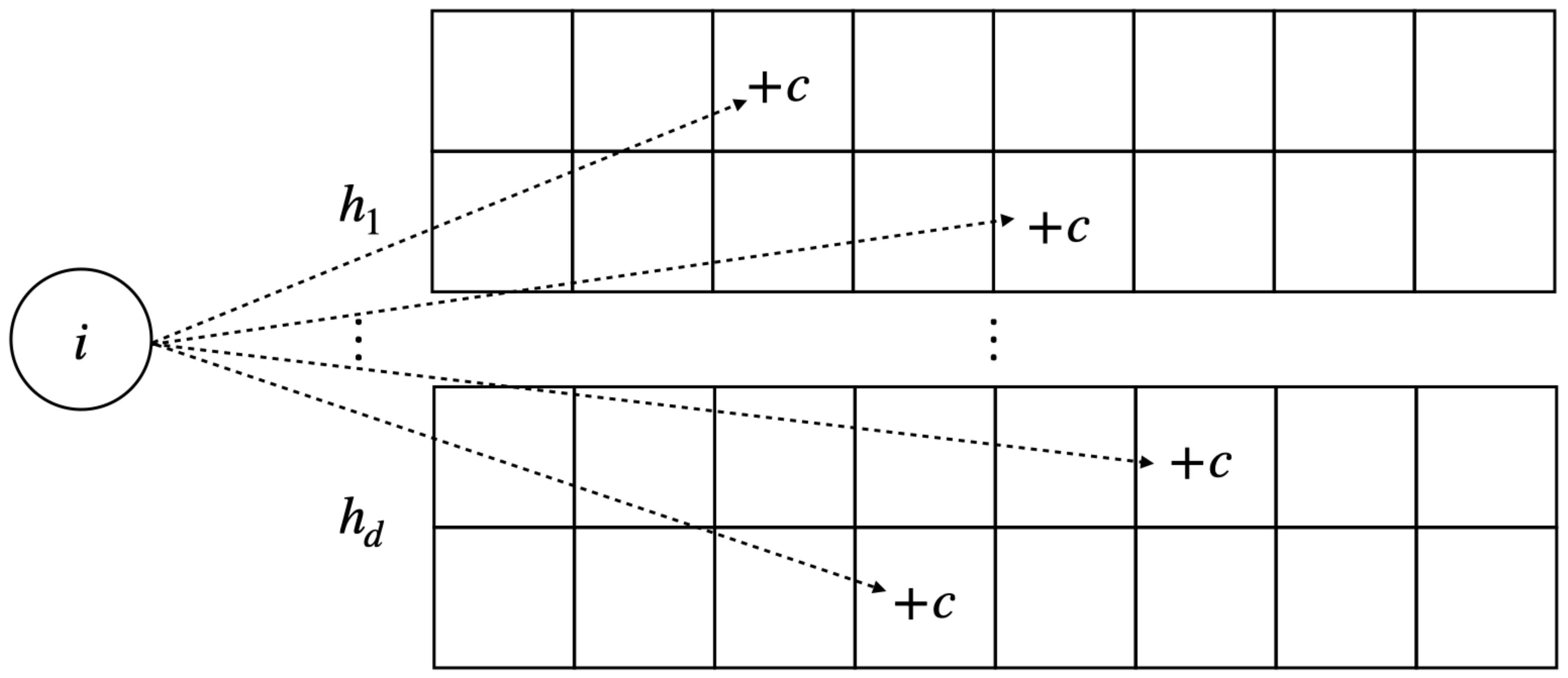
as $$C_R^* = \max_{t \in H, q \in Q_t, ao \in AO} \frac{d_t^*(q, ao)}{d_t^b(q, ao)}$$

Property: let C_R* be the RDP coefficient for **R** and D_2 and $C*$ be the concentrability coefficient for M_{R} and D_2', then $C_R* = C*$.

Count-Min-Sketch

> Count-Min-Sketch can store $v = [v_1, \ldots, v_m]$ and allows point queries, which returns an estimate v_i.

> For δ, ϵ, Cormode and Muthukrishnan (2005) shows that with probability at least $1 - \delta$, $\tilde{v}_i \leq v_i + \epsilon || v ||$..



Updating the CMS[1] with $d = log[1/\delta]$ rows and $w = [e/\epsilon]$ columns.

[1] Cormode and Muthukrishnan (2005) , An improved data stream summary: the count-min sketch and its applications

Languages

>So far **we do not take advantage of the internal structure of the suffix distributions.**

> We define some basic patterns -

$$\begin{aligned}
\mathcal{G}_1 &= \{a\mathcal{O}/\mathcal{R} \mid a \in \mathcal{A}\} \cup \{\mathcal{A}\mathcal{O}/r \mid r \in \mathcal{R}\} \cup \{\mathcal{A}o/\mathcal{R} \mid o \in \mathcal{O}\},\\
\mathcal{G}_2 &= \mathcal{G}_1 \cup \{ao/\mathcal{R} \mid a \in \mathcal{A}, o \in \mathcal{O}\} \cup \{a\mathcal{O}/r \mid a \in \mathcal{A}, r \in \mathcal{R}\} \cup \{\mathcal{A}o/r \mid a \in \mathcal{A}, r \in \mathcal{R}\},\\
\mathcal{G}_3 &= \mathcal{G}_2 \cup \{ao/r \mid a \in \mathcal{A}, o \in \mathcal{O}, r \in \mathcal{R}\}.
\end{aligned}$$

Languages

>So far **we do not take advantage of the internal structure of the suffix distributions.**

> We define some basic patterns -

$$
\begin{aligned}
\mathcal{G}_1 &= \{a\mathcal{O}/\mathcal{R} \mid a \in \mathcal{A}\} \cup \{\mathcal{A}\mathcal{O}/r \mid r \in \mathcal{R}\} \cup \{\mathcal{A}o/\mathcal{R} \mid o \in \mathcal{O}\},\\
\mathcal{G}_2 &= \mathcal{G}_1 \cup \{ao/\mathcal{R} \mid a \in \mathcal{A}, o \in \mathcal{O}\} \cup \{a\mathcal{O}/r \mid a \in \mathcal{A}, r \in \mathcal{R}\} \cup \{\mathcal{A}o/r \mid a \in \mathcal{A}, r \in \mathcal{R}\},\\
\mathcal{G}_3 &= \mathcal{G}_2 \cup \{ao/r \mid a \in \mathcal{A}, o \in \mathcal{O}, r \in \mathcal{R}\}.
\end{aligned}
$$

> For $l \in \mathbb{N}$ and $k \in [l]$, the operator C_k^ℓ maps any set of languages to a new set of languages as follows:

$$
\mathrm{C}_k^\ell(\mathcal{G}) = \{\{x_0 G_1 \cdots x_{k-1} G_k x_k \mid x_0, \ldots, x_k \in \Gamma^*, \ |x_0 \cdots x_k| = (\ell - k)\} \mid G_1, \ldots, G_k \in \mathcal{G}\}
$$

Languages

> We can define a 2-dimensional hierarchy of sets of languages,

$$\mathcal{X}_{i,j} = \bigcup_{k \in [\![j]\!]} \mathrm{C}_k^{\ell}(\mathcal{G}_i), \qquad \forall i \in [\![3]\!],\ \forall j \in [\![\ell]\!].$$

> We define the respective **language metric** as

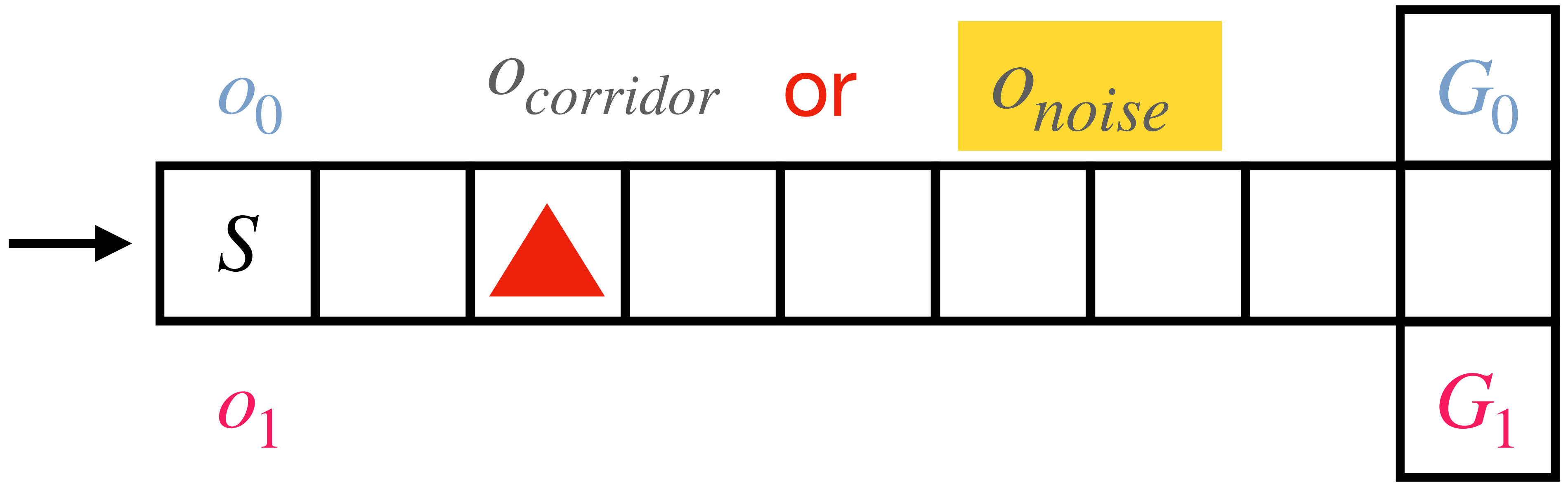
$$L_{\mathcal{X}}(p, p') := \max_{X \in \mathcal{X}} |p(X) - p'(X)|$$

where the probability of a language is

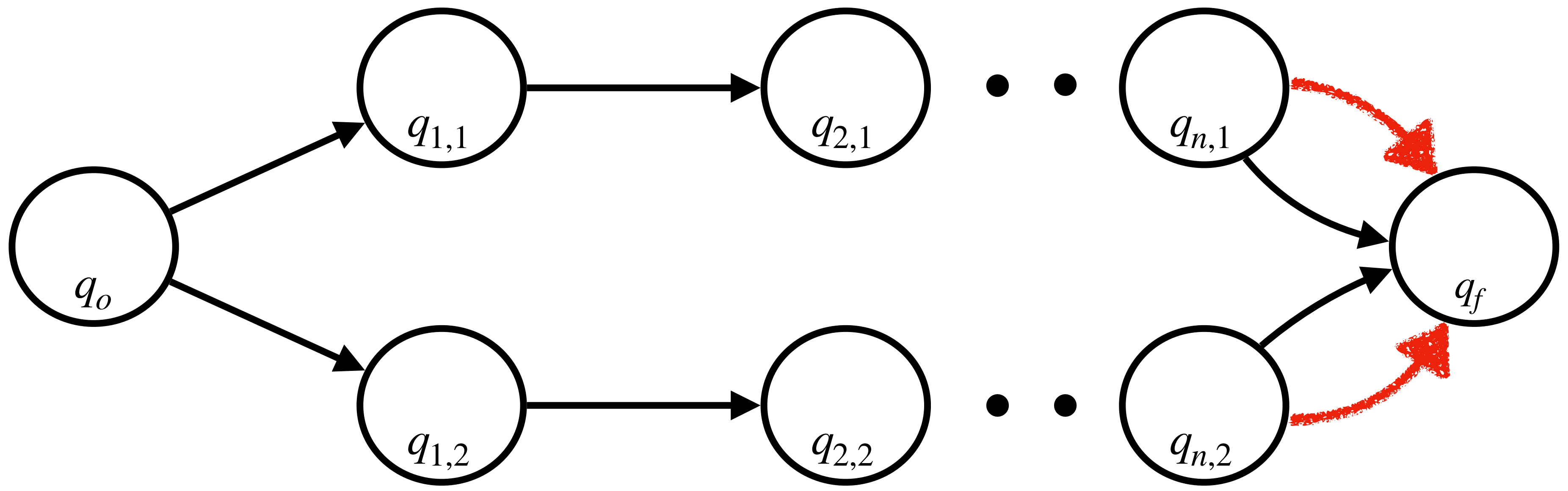
$$p(X) := \sum_{x \in X} p(x)$$

Language in noisy T-maze

> π^b chooses *East* in the corridor and *North* and *South* at the junction with equal probability.



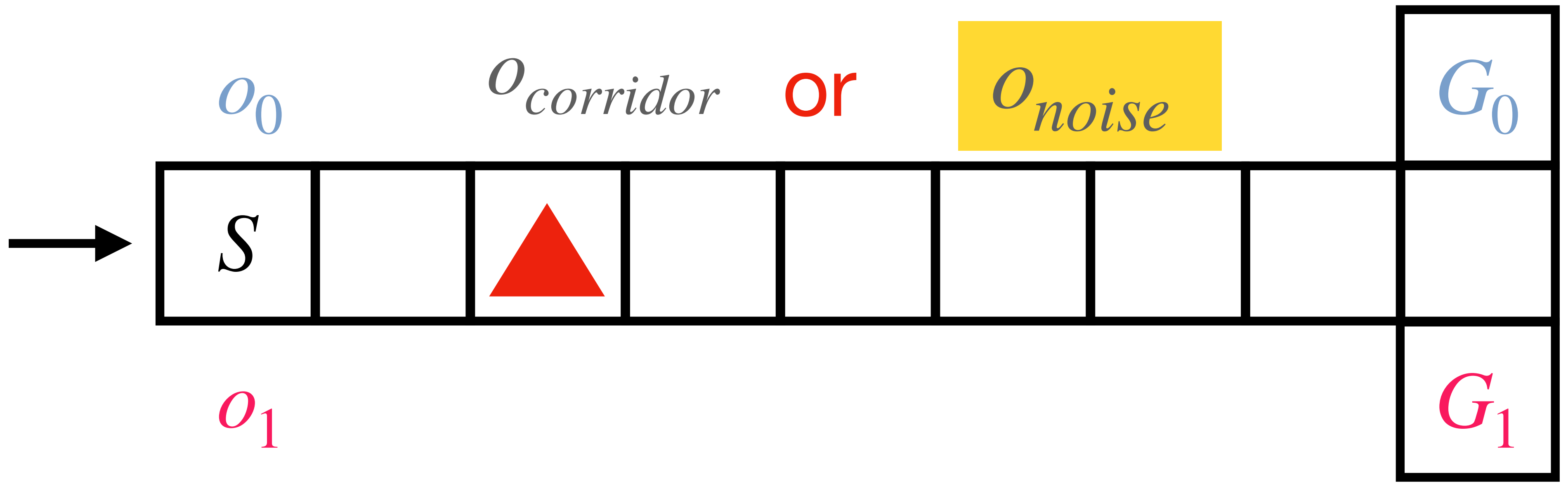
Noisy T-maze, with equal probability of observing $o_{corridor}$ or o_{noise} in the corridor



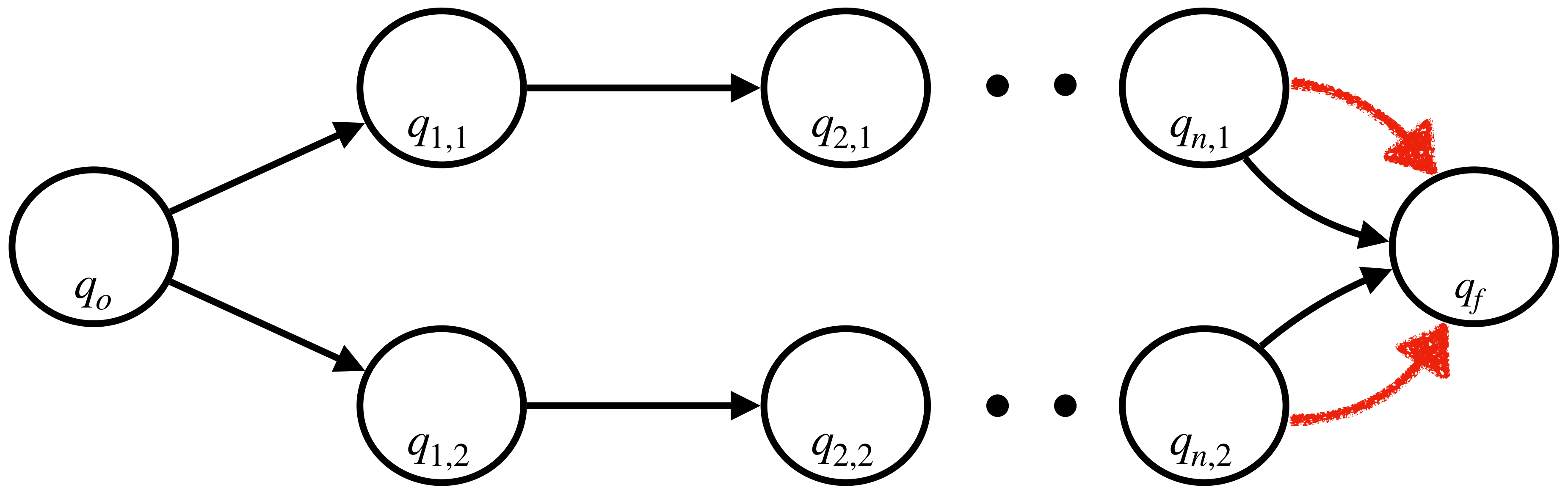
Language in noisy T-maze

> π^b chooses *East* in the corridor and *North* and *South* at the junction with equal probability.

> **Since the distance between states is determined by the probability of single episode suffixes, L_∞^p-distinguishability decreases exponentially with the corridor length *N*.**



Noisy T-maze, with equal probability of observing $o_{corridor}$ or o_{noise} in the corridor

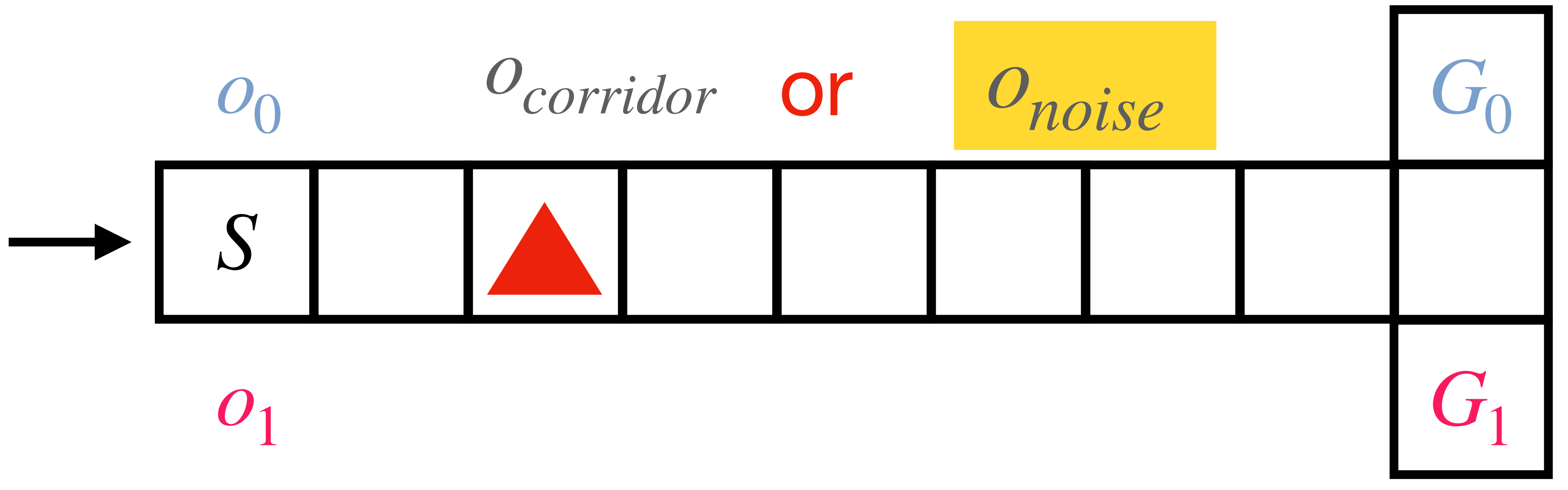


Language in noisy T-maze

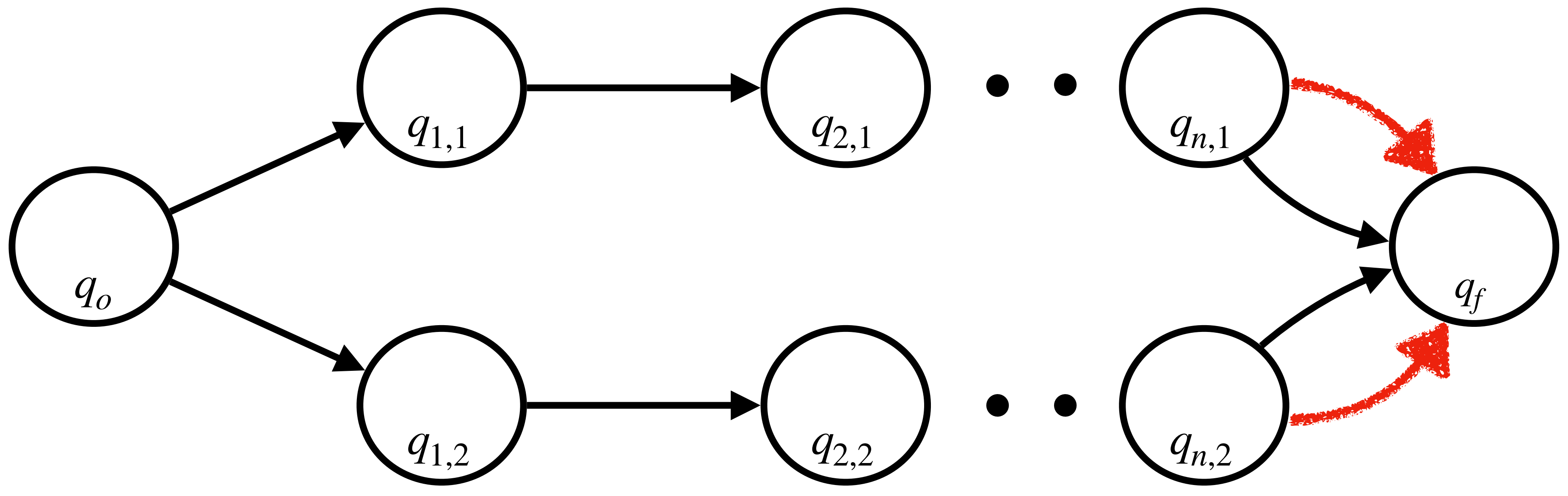
> π^b chooses *East* in the corridor and *North* and *South* at the junction with equal probability.

> Since the distance between states is determined by the probability of single episode suffixes, L_∞^p**-distinguishability decreases exponentially with the corridor length *N*.**

> However, $L_{\mathcal{X}_{2,1}}$-distinguishability will be constant and independent of *N*.



Noisy T-maze, with equal probability of observing $o_{corridor}$ or o_{noise} in the corridor



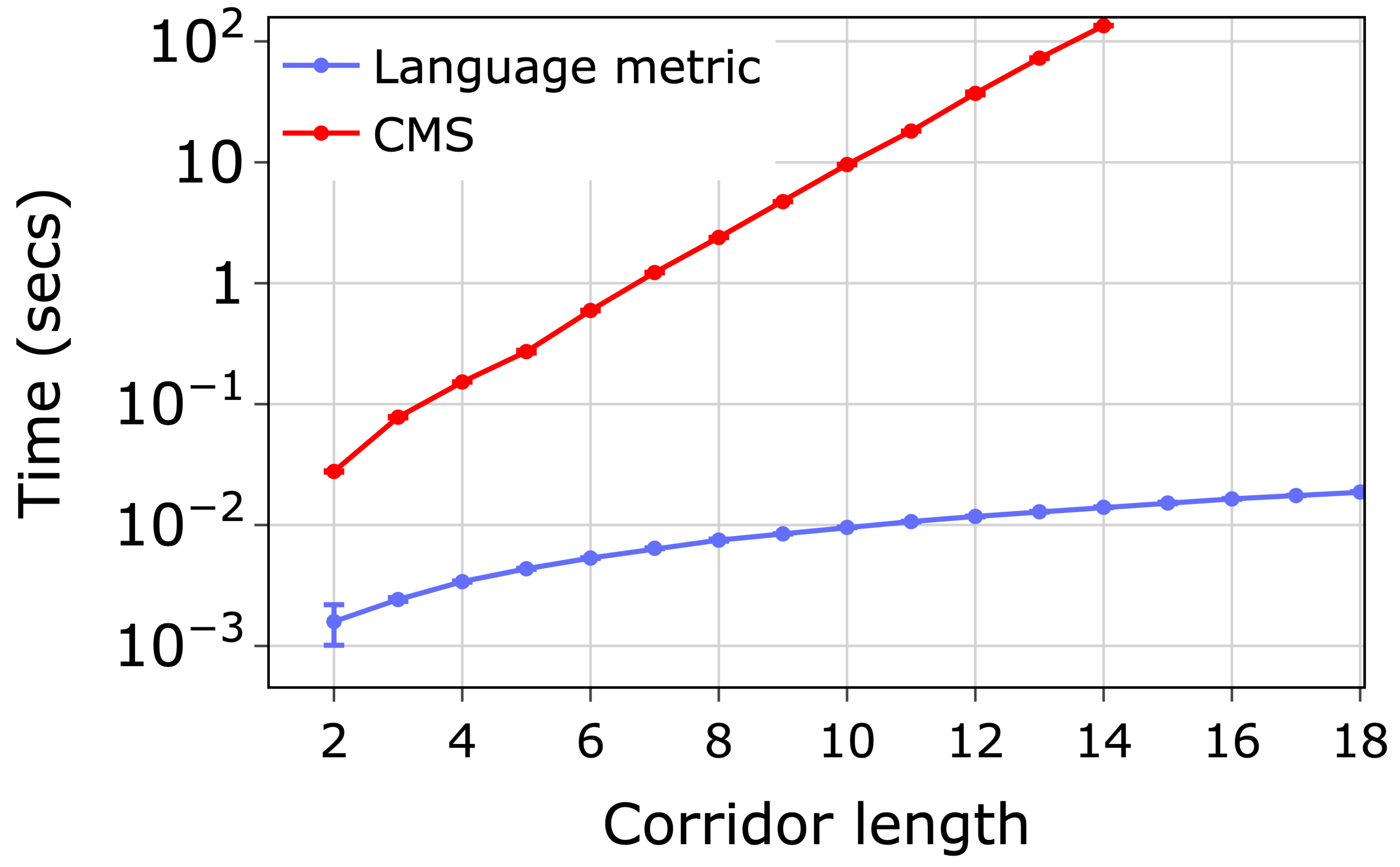
Experimental Results

>For each domain, H is the horizon, and for each algorithm, Q is the number of states of the learnt automaton, r is the reward of the derived policy, averaged over 100 episodes, and time is the running time in seconds of automaton learning. The maximum reward for all the domains is 1, except for T-maze where the reward upon reaching the goal is 4.

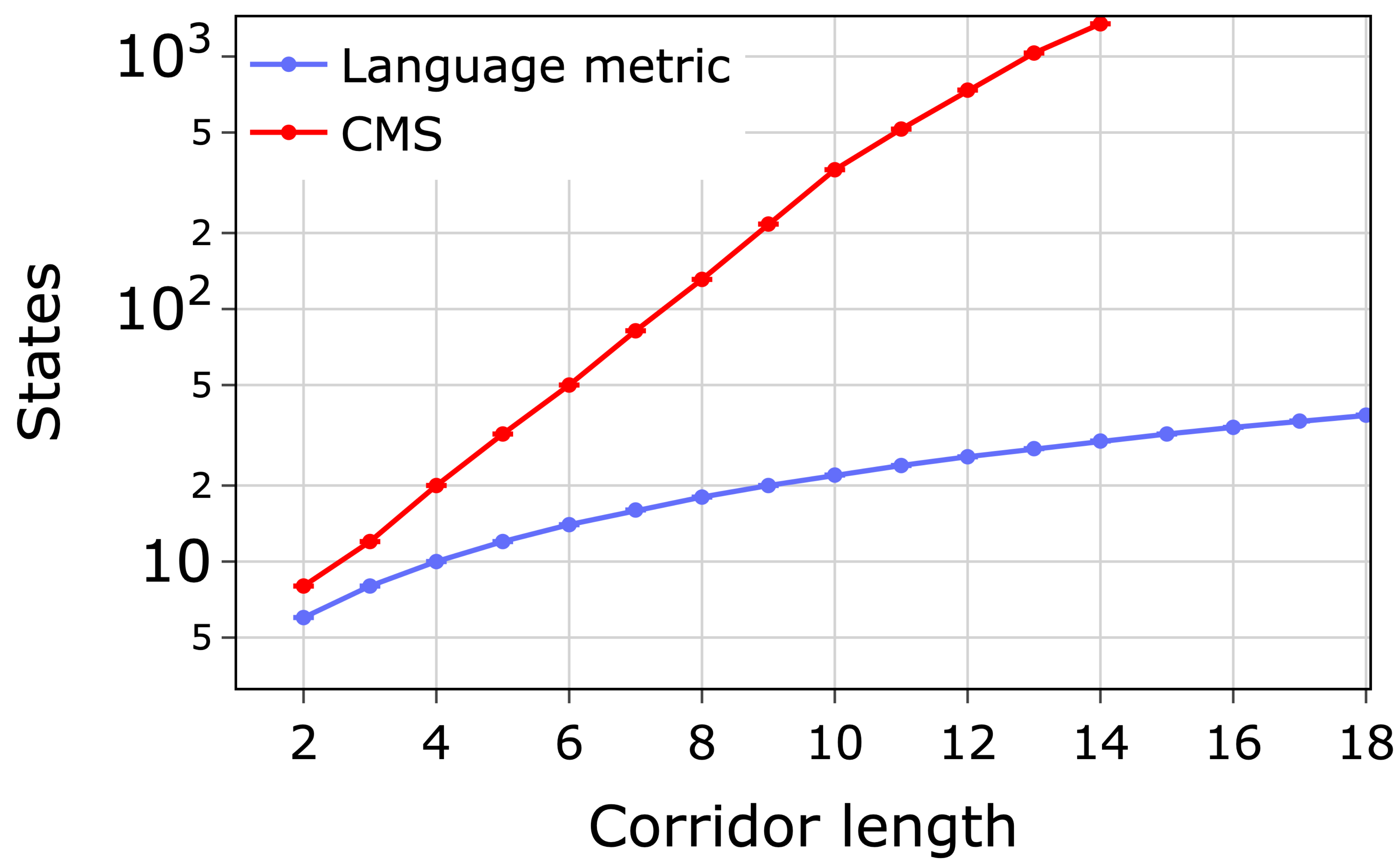
		FlexFringe			CMS			Language metric		
Name	H	Q	r	time	Q	r	time	Q	r	time
Corridor	5	11	**1.0**	0.03	11	**1.0**	0.3	11	**1.0**	**0.01**
T-maze(c)	5	29	0.0	0.11	104	**4.0**	10.1	18	**4.0**	**0.26**
Cookie	9	220	**1.0**	0.36	116	**1.0**	6.05	91	**1.0**	**0.08**
Cheese	6	669	$0.69 \pm .04$	19.28	1158	$0.4 \pm .05$	207.4	326	$\mathbf{0.81 \pm .04}$	**2.23**
Mini-hall	15	897	$0.33 \pm .04$	25.79	-	-	-	5134	$\mathbf{0.91 \pm .03}$	**23.9**

Experimental Results

> Comparing for noisy T-maze where the observations in the corridor can be $o_{corridor}$ or o_{noise} with equal probability



a) Time taken (secs) vs length of corridor



b) Number of RDP states vs length of corridor

Sample Complexity Upper Bounds

> **Theorem 2**: ADACT-H(D, δ) returns the minimal RDP **R** with probability at least $1 - 3AOQ\delta$ when CMS is used to store empirical probability estimates, with the statistical test:

$$L^p_\infty(Z_1, Z_2) \geq \sqrt{8log(4(ARO)^{H-t}/\delta)/min(|Z_1|), |Z_2|)}$$

And the size of the dataset D is at least $|D| \geq \tilde{O}\ (\frac{HC^*_R log(1/\delta)}{d^*_m \mu_0^2})$, where

$d^*_{min} := min_{t,q_t,ao}\{d^b_t(q, ao) \,|\, d^b_t(q, ao) > 0\}$, and μ_0 is the L^p_∞-distinguishability.

Sample Complexity Upper Bounds

> **Theorem 2**: ADACT-H(D, δ) returns the minimal RDP **R** with probability at least $1 - 3AOQ\delta$ when CMS is used to store empirical probability estimates, with the statistical test:

$$L^p_\infty(Z_1, Z_2) \geq \sqrt{8log(4(ARO)^{H-t}/\delta)/min(|Z_1|), |Z_2|)}$$

And the size of the dataset D is at least $|D| \geq \tilde{O}\left(\frac{HC^*_R log(1/\delta)}{d^*_m \mu_0^2}\right)$, where

L^p_∞-distinguishability

$d^*_{min} := min_{t,q_t,ao}\{d^b_t(q, ao) \mid d^b_t(q, ao) > 0\}$, and μ_0 is the L^p_∞-distinguishability.

Min occupancy

Sample Complexity Upper Bounds

> **Theorem 3**: ADACT-H(D, δ) returns the minimal RDP **R** with probability at least $1 - 2AOQ\delta$ when using language metric $L_{\mathcal{X}}$ to define a statistical test:

$$L_{\mathcal{X}}(Z_1, Z_2) \geq \sqrt{2log(4|\mathcal{X}|/\delta)/min(|Z_1|), |Z_2|)}$$

And the size of the dataset D is at least $|D| \geq \tilde{O}(\frac{C_R^* log(1/\delta) log|\mathcal{X}|}{d_m^* \mu_0^2})$,

where $d_{min} := min_{t,q_t,ao}\{d_t^b(q, ao) | d_t^b(q, ao) > 0\}$, and μ_0 is the $L_{\mathcal{X}}$-distinguishability.

Sample Complexity Upper Bounds

> **Theorem 3**: ADACT-H(D, δ) returns the minimal RDP **R** with probability at least $1 - 2AOQ\delta$ when using language metric $L_{\mathcal{X}}$ to define a statistical test:

$$L_{\mathcal{X}}(Z_1, Z_2) \geq \sqrt{2log(4|\mathcal{X}|/\delta)/min(|Z_1|), |Z_2|)}$$

And the size of the dataset D is at least $|D| \geq \tilde{O}(\frac{C_R^* log(1/\delta) log|\mathcal{X}|}{d_m^* \mu_0^2})$,

$L_{\mathcal{X}}$**-distinguishability**

where $d_{min} := min_{t,q_t,ao}\{d_t^b(q, ao) | d_t^b(q, ao) > 0\}$, and μ_0 is the $L_{\mathcal{X}}$-distinguishability.

Proof Structure

Function $\textsc{TestDistinct}(t, \mathcal{X}_1, \mathcal{X}_2, \delta)$

  return $L^{\mathsf{p}}_{\infty}(\mathcal{X}_1, \mathcal{X}_2) \geq \sqrt{2\log(8(ARO)^{H-t}/\delta)/\min(|\mathcal{X}_1|, |\mathcal{X}_2|)}$

Proof Structure

Function $\textsc{TestDistinct}(t, \mathcal{X}_1, \mathcal{X}_2, \delta)$

  return $L^{\mathsf{p}}_{\infty}(\mathcal{X}_1, \mathcal{X}_2) \geq \sqrt{2\log(8(ARO)^{H-t}/\delta)/\min(|\mathcal{X}_1|, |\mathcal{X}_2|)}$

Lemma 15. *For $t \in [\![0, H]\!]$, let $\mathcal{Z}_1$ and $\mathcal{Z}_2$ be multisets sampled from distributions p_1 and p_2 on $\Delta(\Gamma^{H-t})$, and let $\widehat{\mathrm{p}}_1$ and $\widehat{\mathrm{p}}_2$ be empirical estimates of p_1 and p_2 due to $\mathcal{Z}_1$ and $\mathcal{Z}_2$, respectively. Under event $\mathcal{E}_{\mathcal{X}}$, if $\mathrm{p}_1 = \mathrm{p}_2$ then $\textsc{TestDistinct}_{\mathcal{X}}(t, \mathcal{Z}_1, \mathcal{Z}_2, \delta)$ answers false.*

Proof Structure

Function $\textsc{TestDistinct}(t, \mathcal{X}_1, \mathcal{X}_2, \delta)$

  return $L^{\mathsf{p}}_{\infty}(\mathcal{X}_1, \mathcal{X}_2) \geq \sqrt{2\log(8(ARO)^{H-t}/\delta)/\min(|\mathcal{X}_1|, |\mathcal{X}_2|)}$

Lemma 15. *For $t \in [\![0, H]\!]$, let $\mathcal{Z}_1$ and $\mathcal{Z}_2$ be multisets sampled from distributions p_1 and p_2 on $\Delta(\Gamma^{H-t})$, and let $\widehat{\mathrm{p}}_1$ and $\widehat{\mathrm{p}}_2$ be empirical estimates of p_1 and p_2 due to $\mathcal{Z}_1$ and $\mathcal{Z}_2$, respectively. Under event $\mathcal{E}_{\mathcal{X}}$, if $\mathrm{p}_1 = \mathrm{p}_2$ then $\textsc{TestDistinct}_{\mathcal{X}}(t, \mathcal{Z}_1, \mathcal{Z}_2, \delta)$ answers false.*

Lemma 16. *For $t \in [\![0, H]\!]$, let $\mathcal{Z}_1$ and $\mathcal{Z}_2$ be multisets sampled from distributions p_1 and p_2 on $\Delta(\Gamma^{H-t})$, and let $\widehat{\mathrm{p}}_1$ and $\widehat{\mathrm{p}}_2$ be empirical estimates of p_1 and p_2 due to $\mathcal{Z}_1$ and $\mathcal{Z}_2$, respectively. Under event $\mathcal{E}_{\mathcal{X}}$, if $\mathrm{p}_1 \neq \mathrm{p}_2$ then $\textsc{TestDistinct}_{\mathcal{X}}(t, \mathcal{Z}_1, \mathcal{Z}_2, \delta)$ answers true if $\mathcal{Z}_1$ and $\mathcal{Z}_2$ satisfy $\min(|\mathcal{Z}_1|, |\mathcal{Z}_2|) \geq 8\log(2|\mathcal{X}|/\delta)/\mu_0^2$.*

Minimum cardinality

Related works

> Abadi and Brafman (2020)

> Ronca and De Giacomo (2021), Ronca, Licks et al. (2022)

> ADACT (Balle et al), online algorithm, if applied directly gives bound

$$\widetilde{\mathcal{O}}\left(\frac{Q^4 A^2 O^2 H^5 \log(1/\delta)}{\varepsilon^2} \max\left\{\frac{1}{\mu_0^2}, \frac{H^4 O^2 A^2}{\varepsilon^4}\right\}\right)$$

> Toto Icarte et al. (2019)

> Mahmud (2010)

> POMDP algorithms

> Predicitive State Representations (PSRs) and generic NMDP algorithms (Hutter 2009, Lattimore et al. 2013)

Slide from Roberto Cipollone

ADACT-H-A

Theorem 17. ADACT-H-A$(\mathcal{D}, \delta, \varepsilon, \overline{Q}, \overline{C})$ *returns an* $\frac{\varepsilon}{2}$*-approximate RDP* $\mathbf{R}'$ *with probability at least* $1 - 2AOQ\delta$ *when using the language metric* $L_\mathcal{X}$ *to define a statistical test*

$$L_\mathcal{X}(\mathcal{Z}_1, \mathcal{Z}_2) \geq \sqrt{2\log(2|\mathcal{X}|/\delta)/\min(|\mathcal{Z}_1|, |\mathcal{Z}_2|)},$$

and the size of the dataset $\mathcal{D}$ *is at least*

$$|\mathcal{D}| \geq \widetilde{\mathcal{O}}\left(\overline{C}\log(1/\delta)\left(\frac{\overline{Q}AO\log|\mathcal{X}|}{\varepsilon\mu_0^2} + \frac{1}{d_m^*}\right)\right),$$

Proof Sketch for ADACT-H-A

> A state is only learned if it is frequent, i.e. $\frac{|Z(qao)|}{N} \geq \frac{3\epsilon}{10\bar{Q}AOH\bar{C}}$

> For all frequent states, the proof follows as before

>For the other states, we bound $d_t^b(qao)$ with Bernstein's

>With high probability, the maximum loss for infrequent states is $\epsilon/2$.

Conclusions

> We propose two new approaches to offline RL for Regular Decision Processes and provide their respective theoretical analysis.

Conclusions

> We propose two new approaches to offline RL for Regular Decision Processes and provide their respective theoretical analysis.

> We also improve upon existing algorithms for RDP learning, and propose a modified algorithm using Count-Min-Sketch with reduced memory complexity.

Conclusions

> We propose two new approaches to offline RL for Regular Decision Processes and provide their respective theoretical analysis.

> We also improve upon existing algorithms for RDP learning, and propose a modified algorithm using Count-Min-Sketch with reduced memory complexity.

>We define a hierarchy of language families and introduce a language-based approach, removing the dependency on L_∞^p-distinguishability parameters, and compare and evaluate the performance of our algorithms.

Conclusions

> We propose two new approaches to offline RL for Regular Decision Processes and provide their respective theoretical analysis.

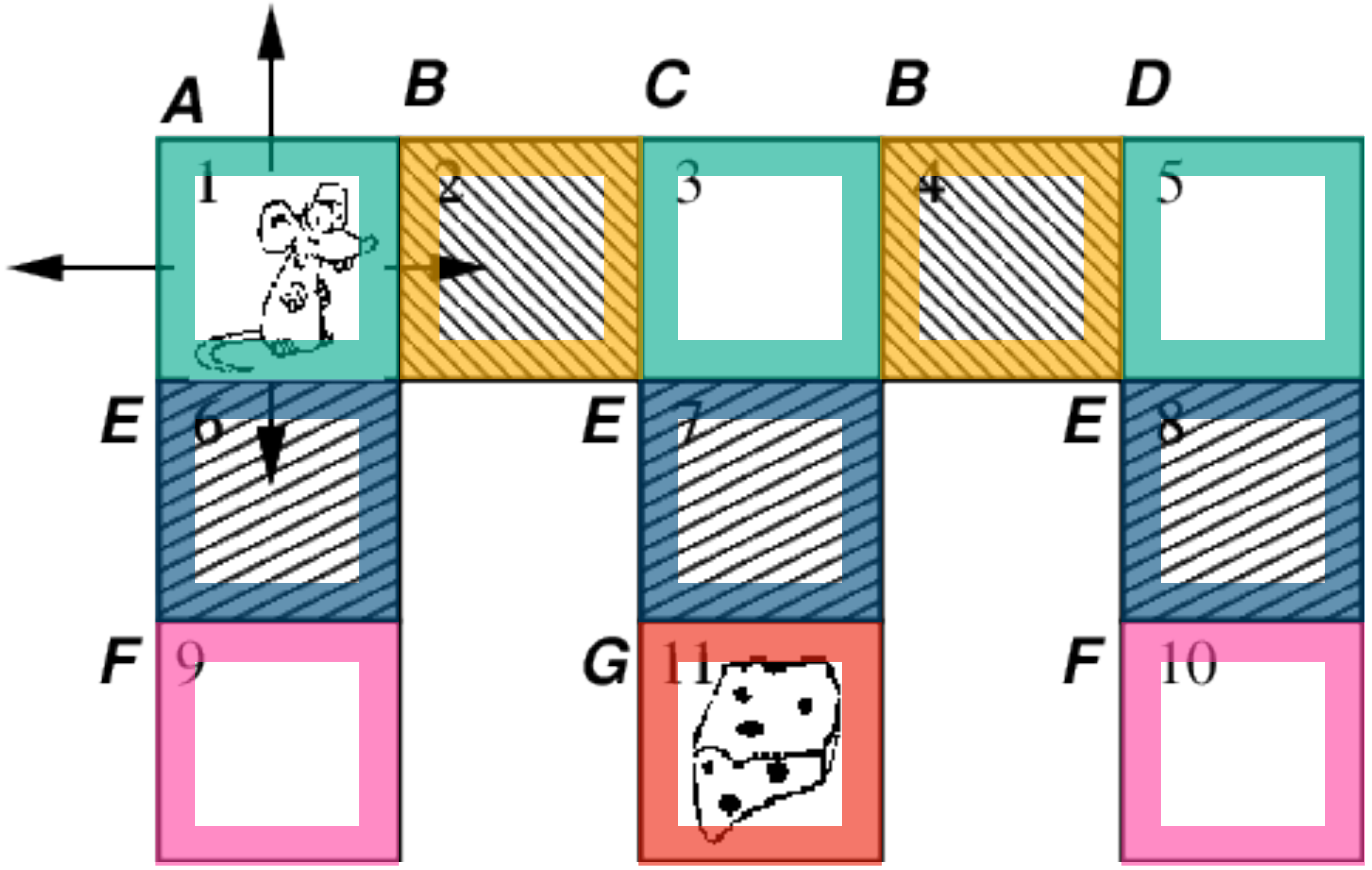
> We also improve upon existing algorithms for RDP learning, and propose a modified algorithm using Count-Min-Sketch with reduced memory complexity.

>We define a hierarchy of language families and introduce a language-based approach, removing the dependency on L^p_∞-distinguishability parameters, and compare and evaluate the performance of our algorithms.

> Finally as a future work, we plan to expand to the online setting!

Limitations and Open Questions

> We learn acyclic abstractions, which is not useful in a lot of cases - example: Cheese-maze[1]



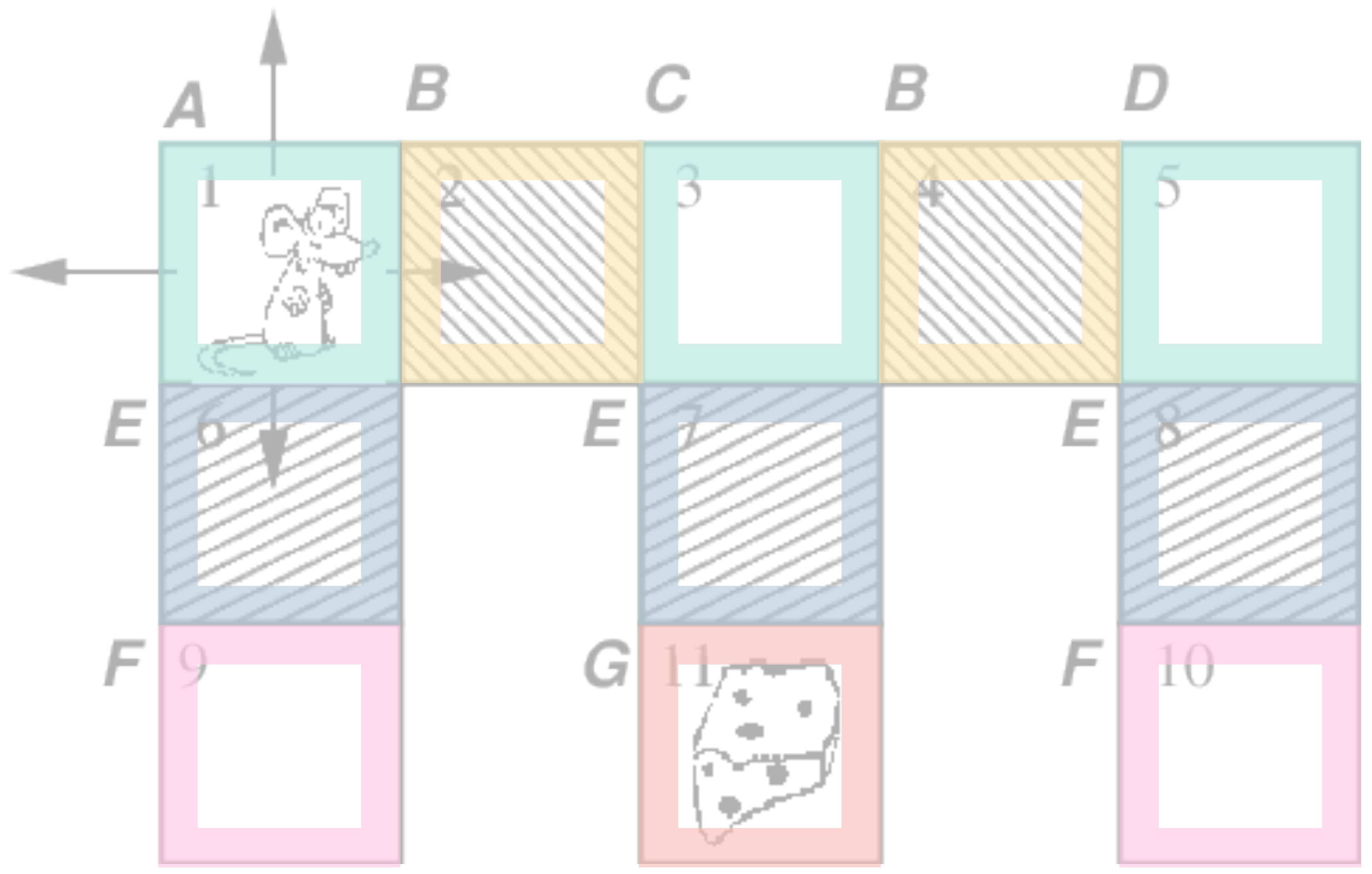
Cheese-maze

[1] R. Andrew McCallum, 1992, First results with utile distinction memory for reinforcement learning

Limitations and Open Questions

> We learn acyclic abstractions, which is not useful in a lot of cases - example: Cheese-maze[1]

> We still use a uniform policy π^b in our data gathering phase.



Cheese-maze

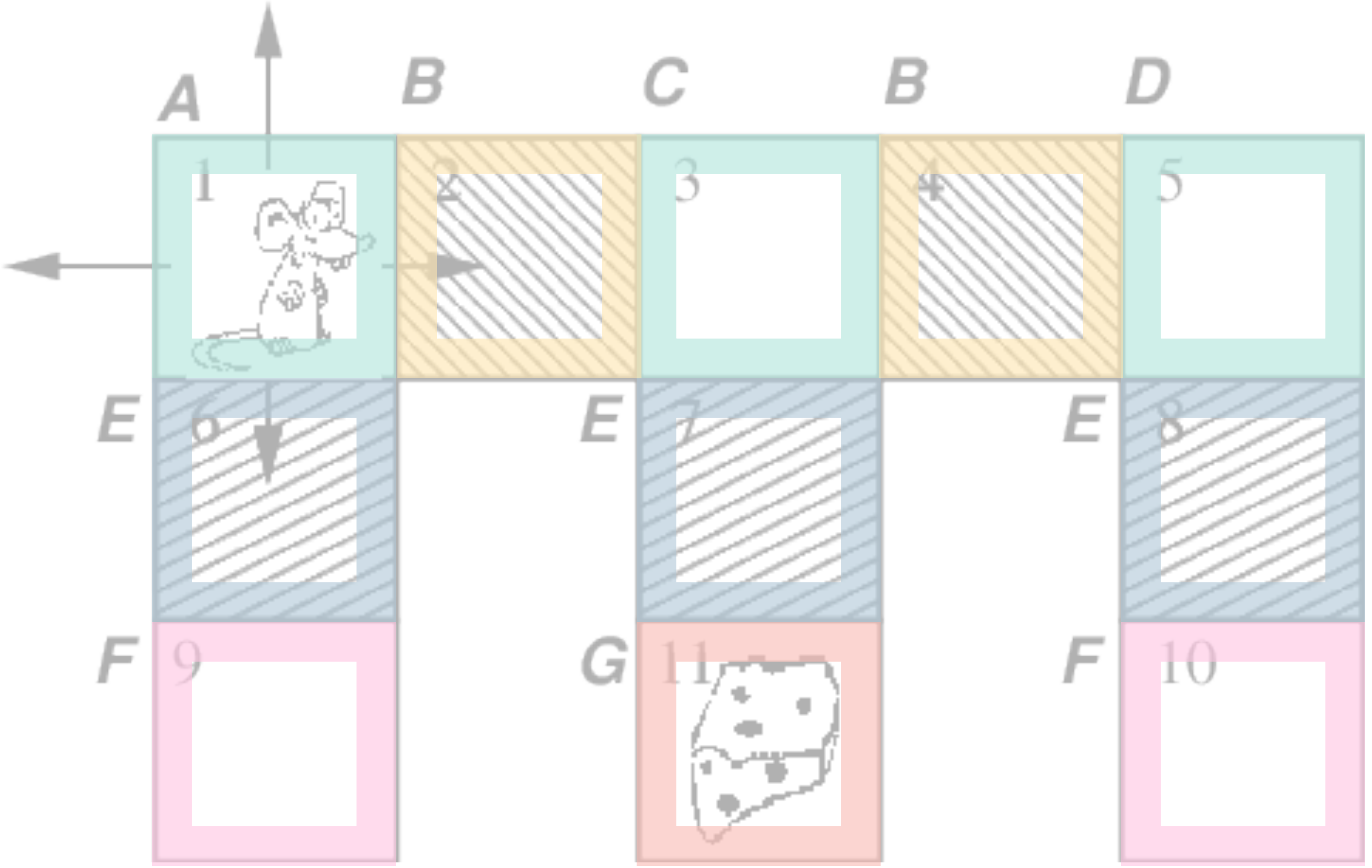
[1] R. Andrew McCallum, 1992, First results with utile distinction memory for reinforcement learning

Limitations and Open Questions

> We learn acyclic abstractions, which is not useful in a lot of cases - example: Cheese-maze[1]

> We still use a uniform policy π^b in our data gathering phase.

> Future direction: Extending to the online setting



Cheese-maze

[1] R. Andrew McCallum, 1992, First results with utile distinction memory for reinforcement learning

Thank you!

Check out our paper here!