Optimal Attack and Defense for Reinforcement Learning

Jeremy McMahan, Young Wu, Xiaojin Zhu, Qiaomin Xie

RL Basics

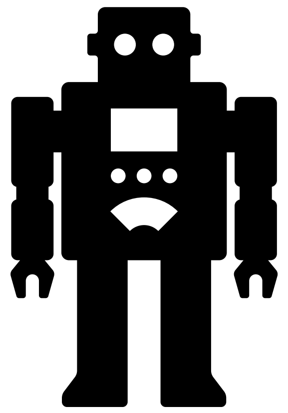
RL Interaction Protocol

RL Interaction Protocol

Models sequential decision making in uncertain environments

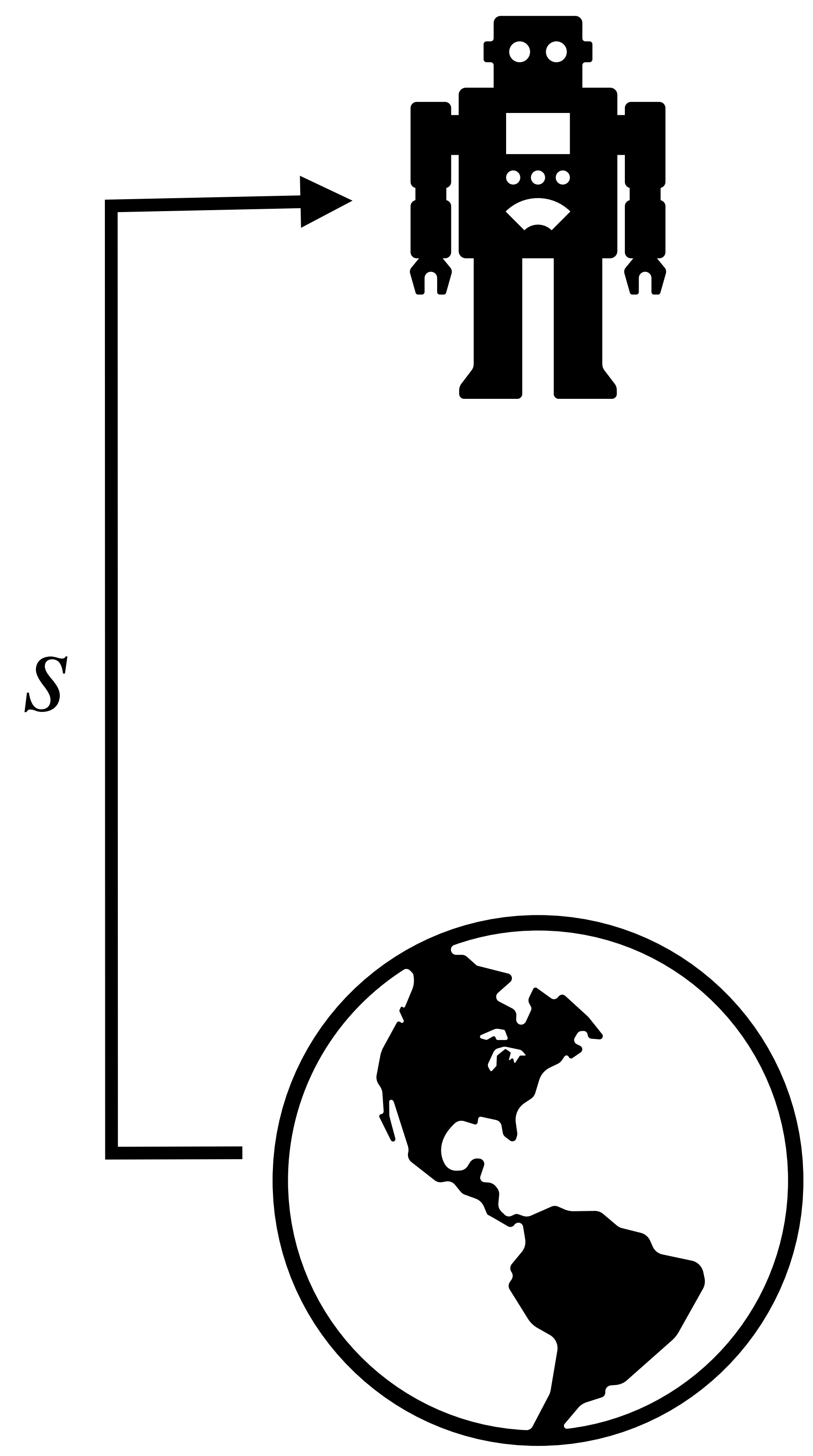
RL Interaction Protocol

Models sequential decision making in uncertain environments



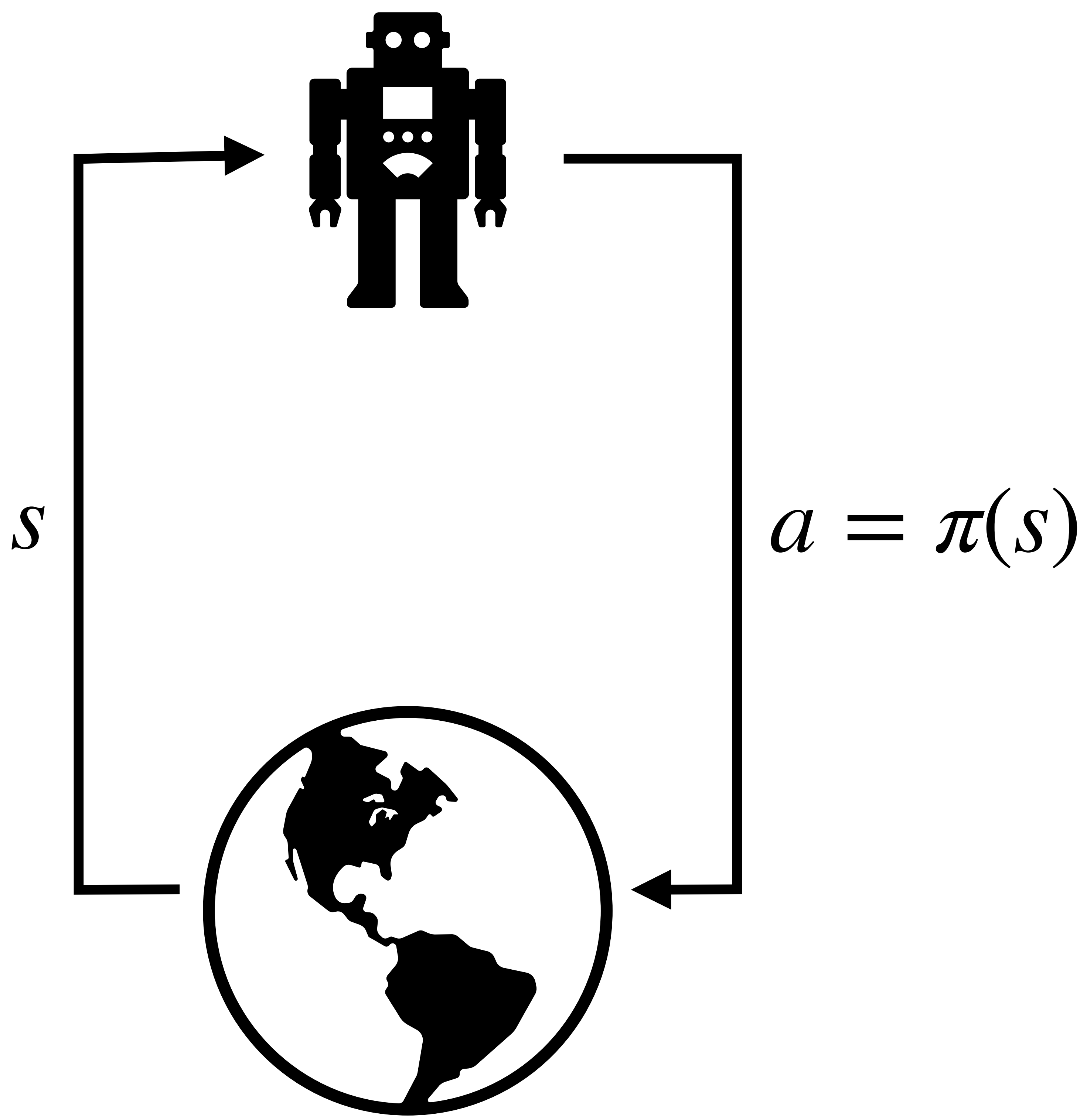
RL Interaction Protocol

Models sequential decision making in uncertain environments



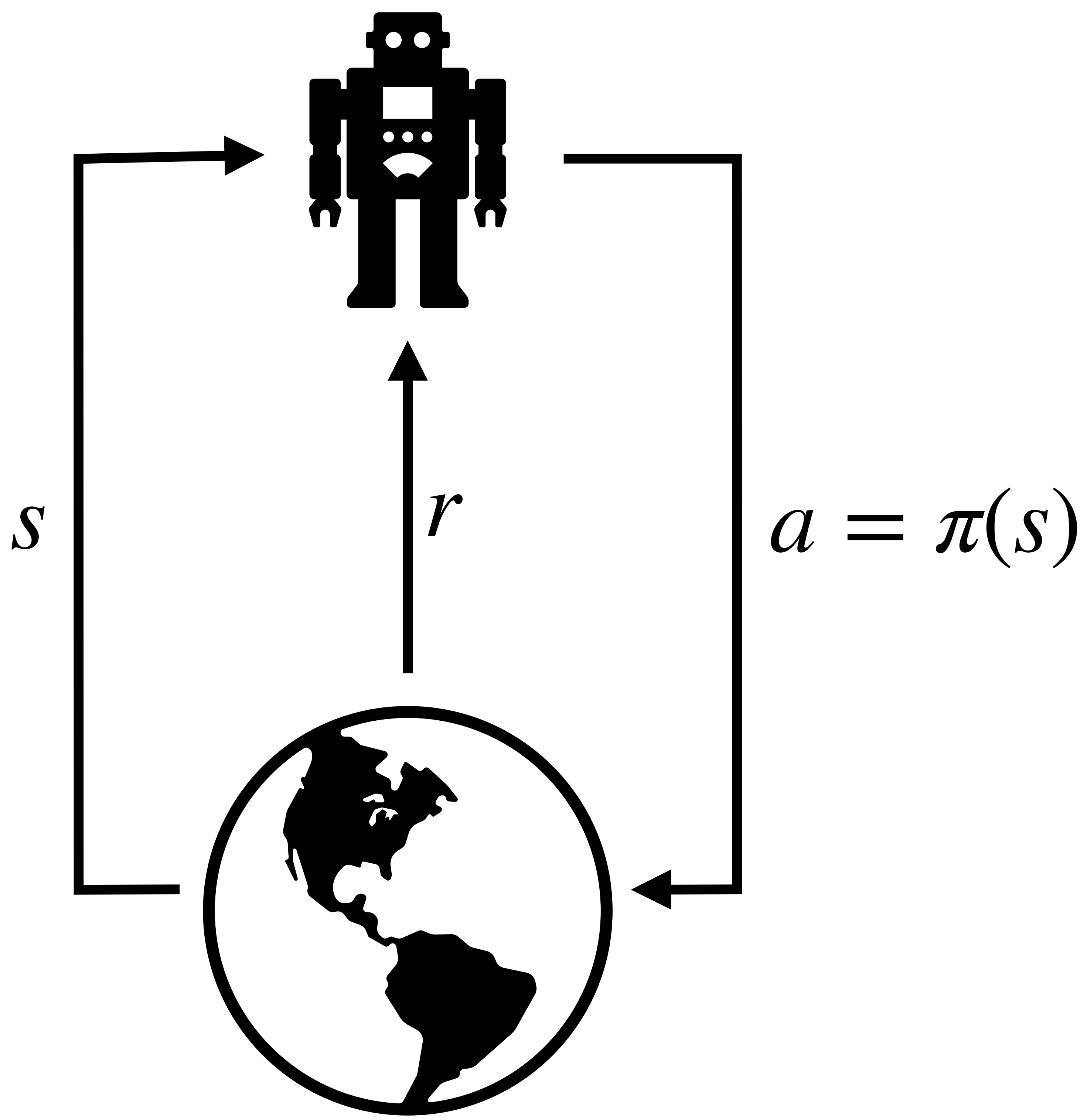
RL Interaction Protocol

Models sequential decision making in uncertain environments



RL Interaction Protocol

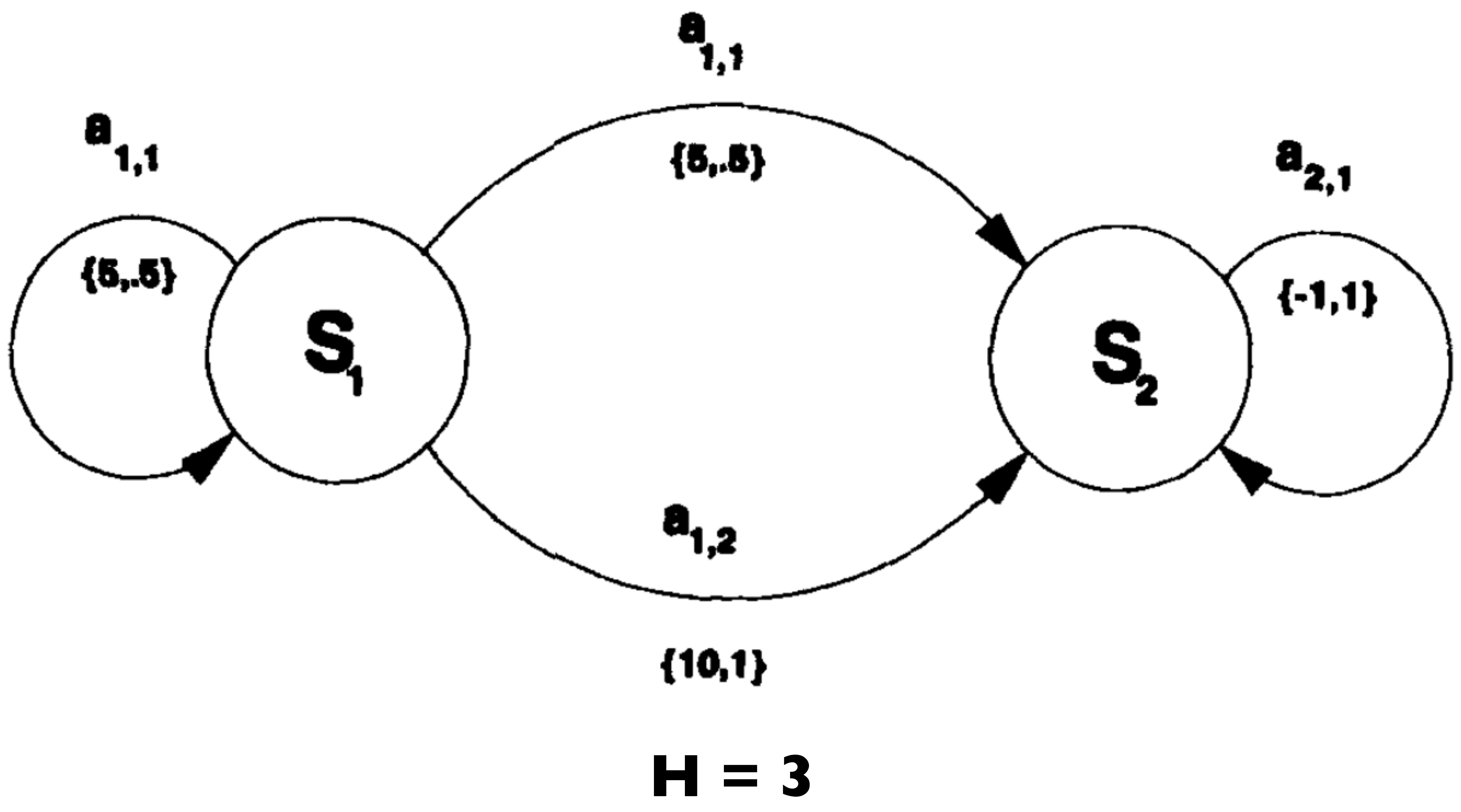
Models sequential decision making in uncertain environments



Model: MDPs

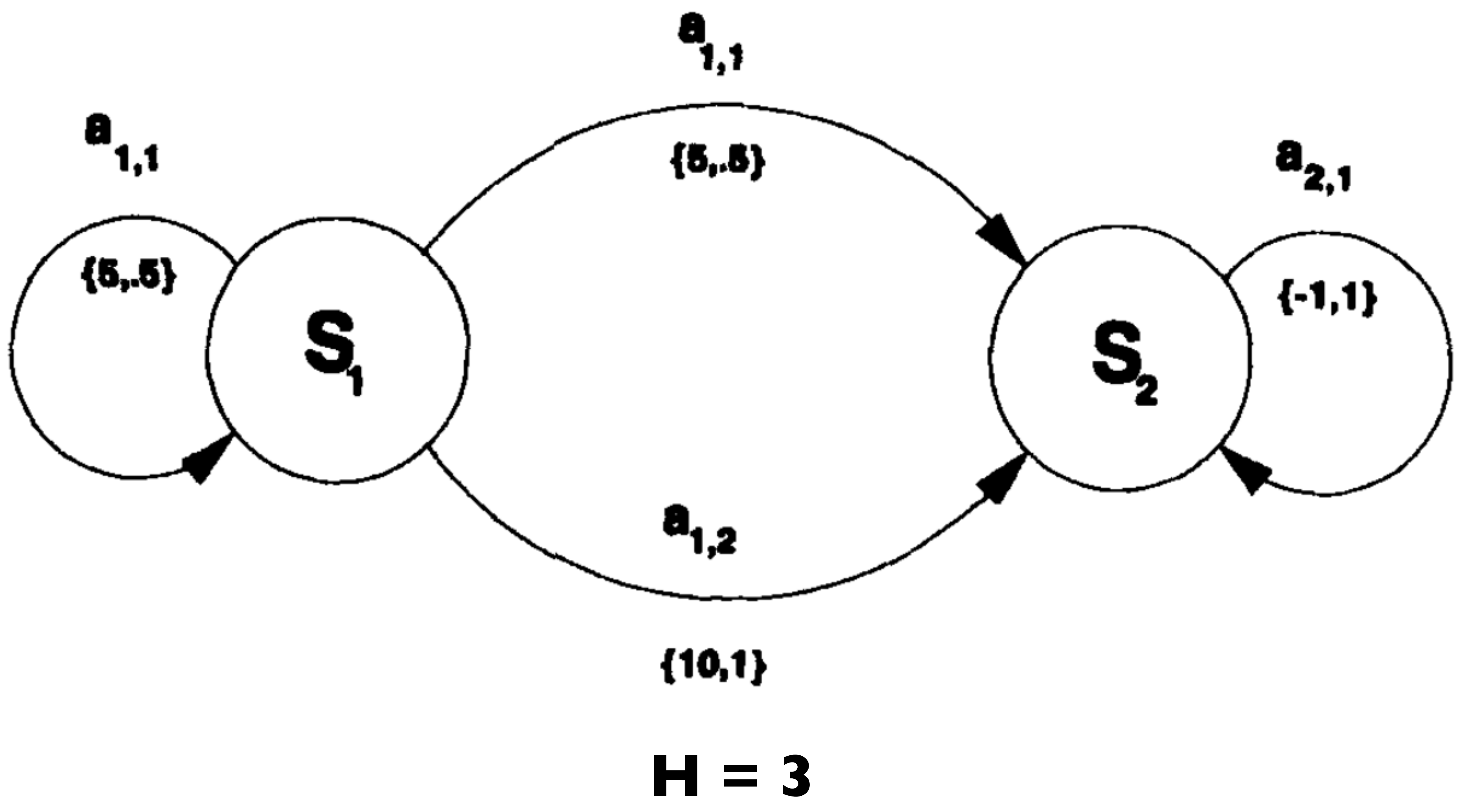
H = 3

Model: MDPs



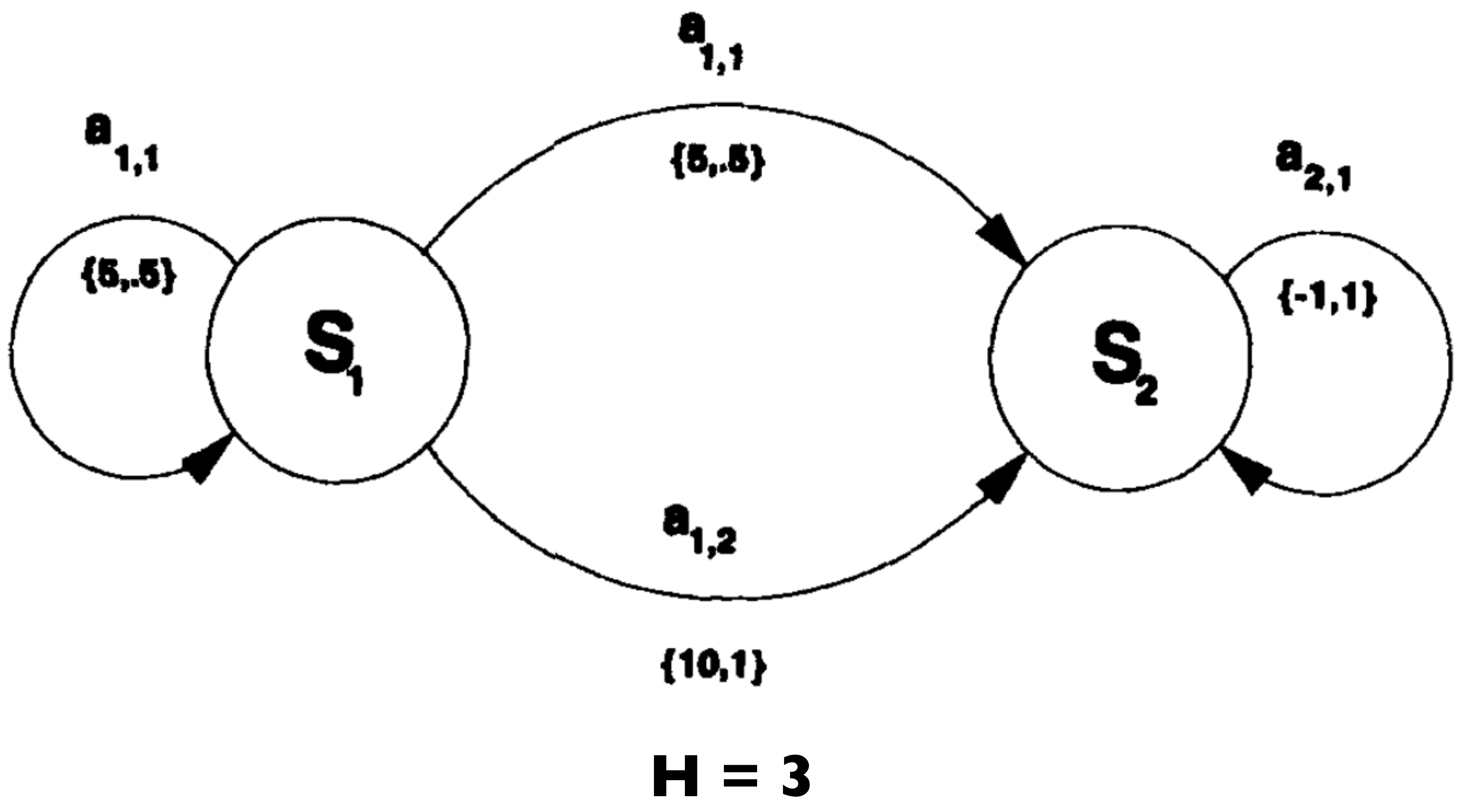
Model: MDPs

- States, S



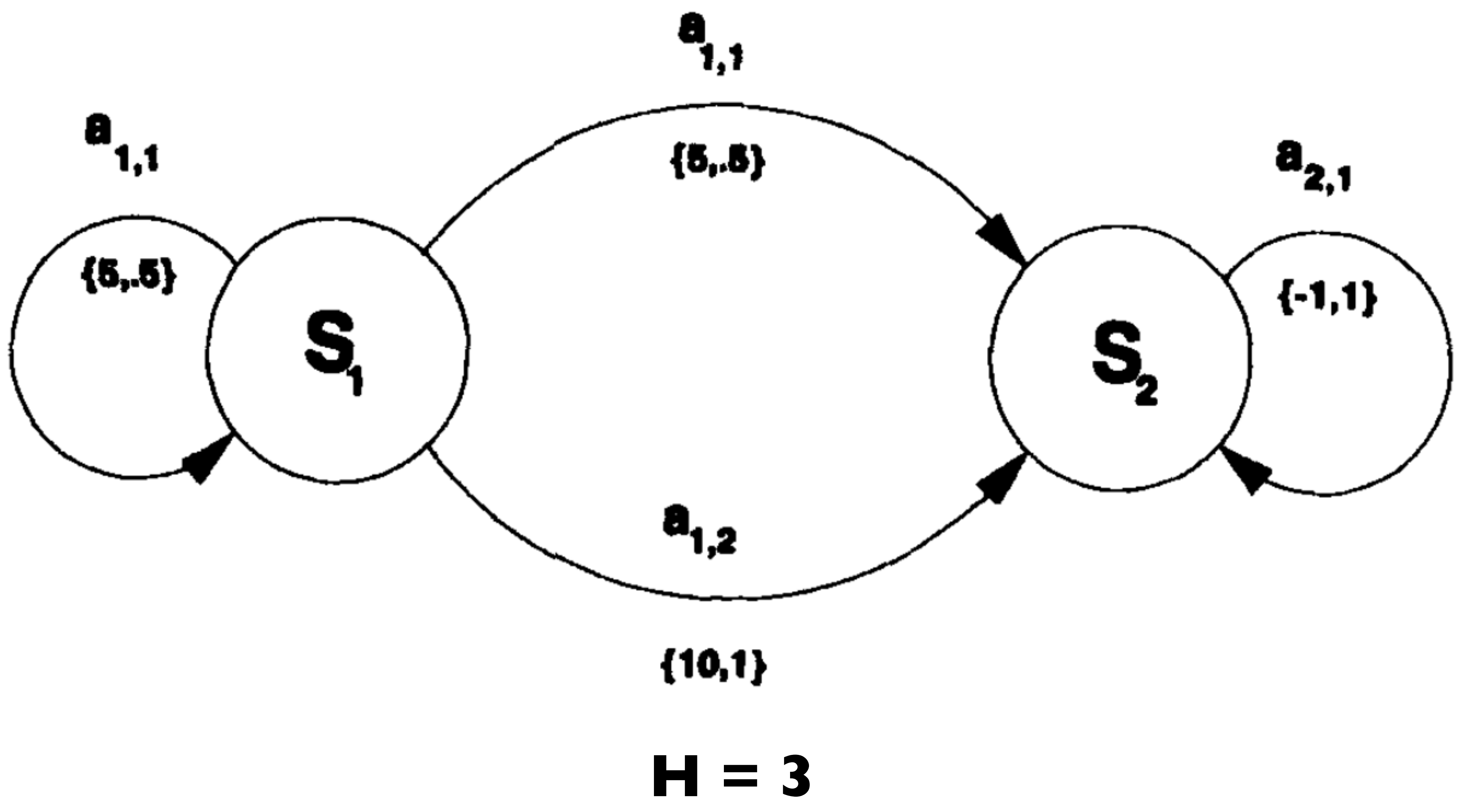
Model: MDPs

- States, S
- Actions, A



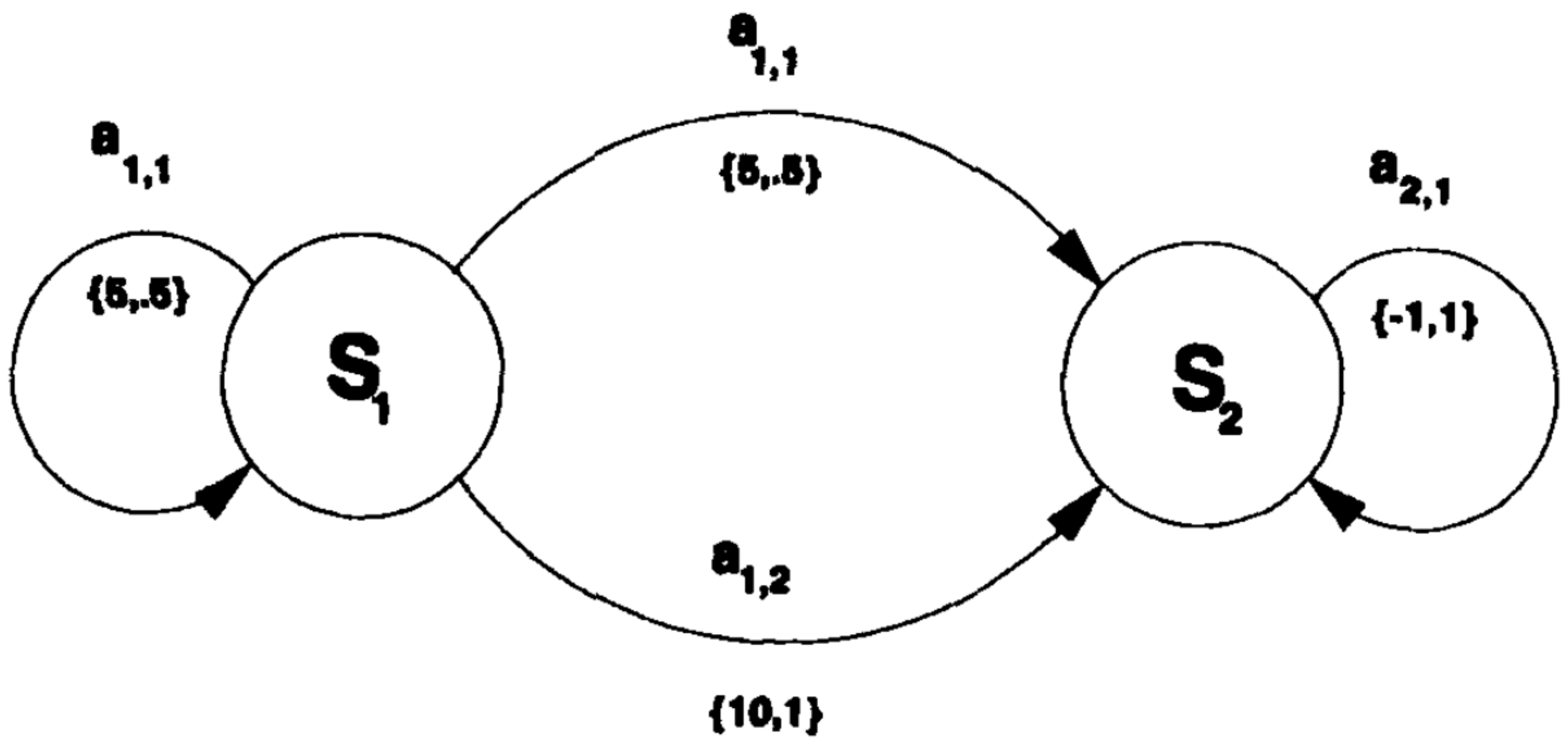
Model: MDPs

- States, S
- Actions, A
- Rewards, $r(s, a)$



Model: MDPs

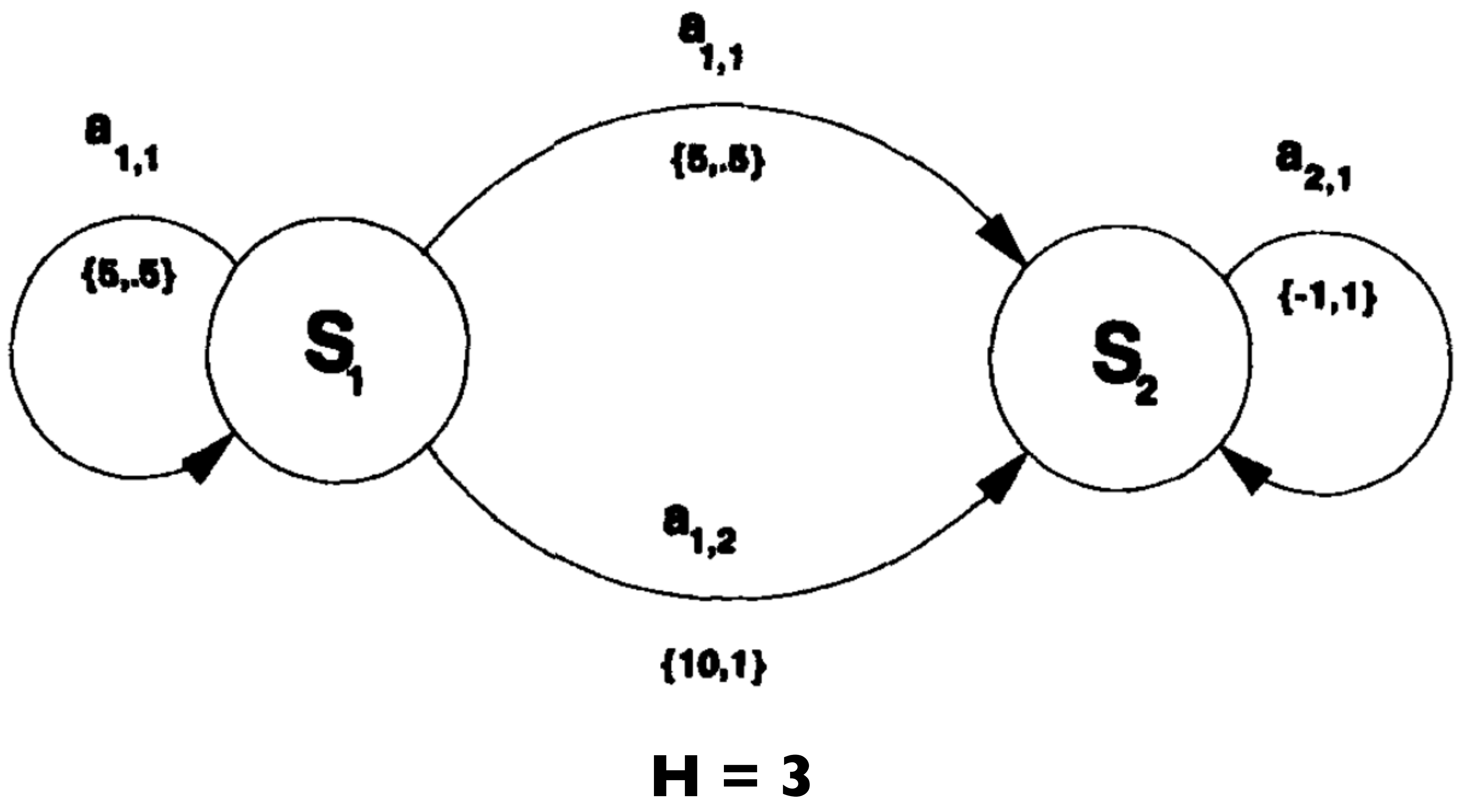
- States, S
- Actions, A
- Rewards, $r(s, a)$
- Transition Probabilities, $P(s' \mid s, a)$



H = 3

Model: MDPs

- States, S
- Actions, A
- Rewards, $r(s, a)$
- Transition Probabilities, $P(s' \mid s, a)$
- Time Horizon, H



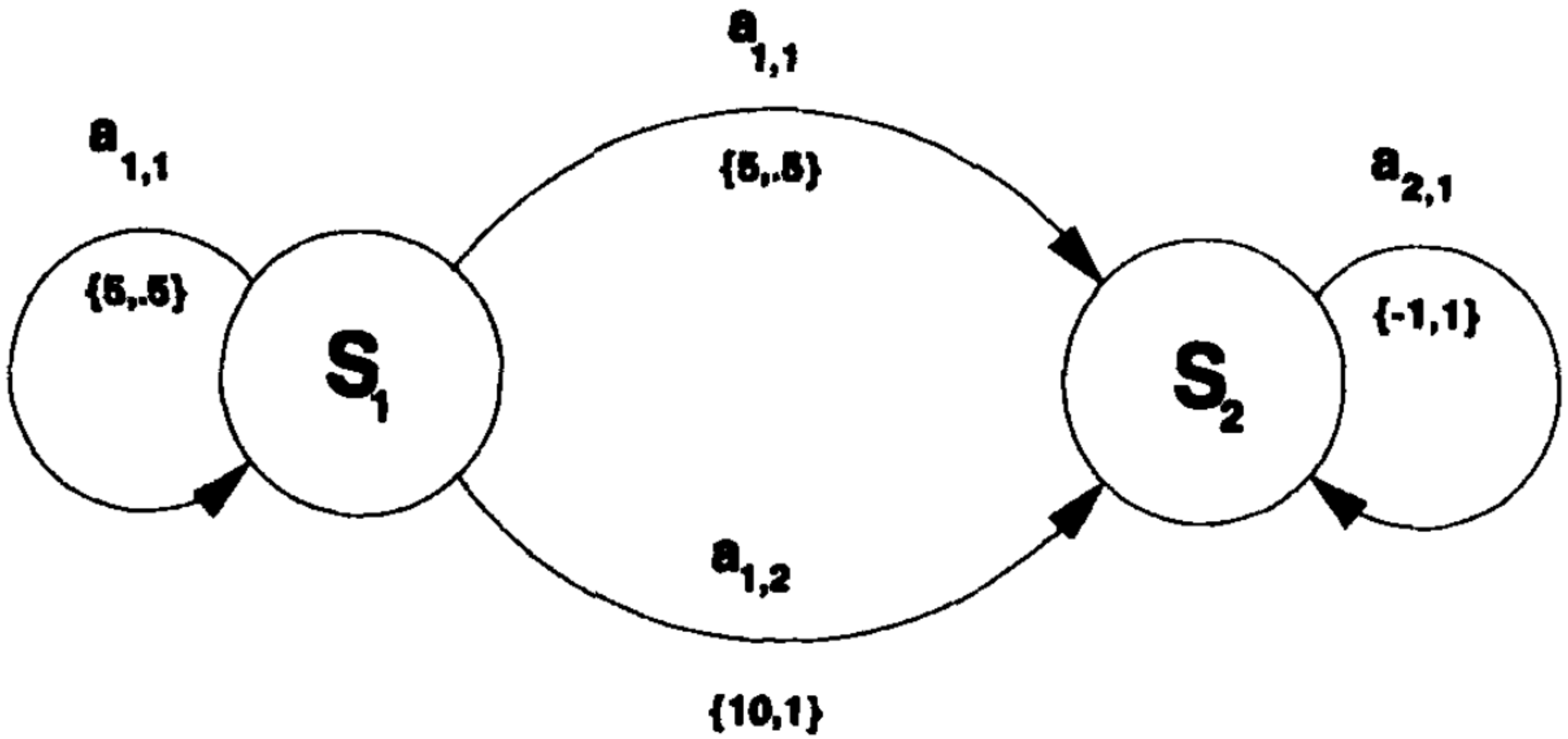
Policies

Policies

A *policy* is a plan of what action to take in each state.

Policies

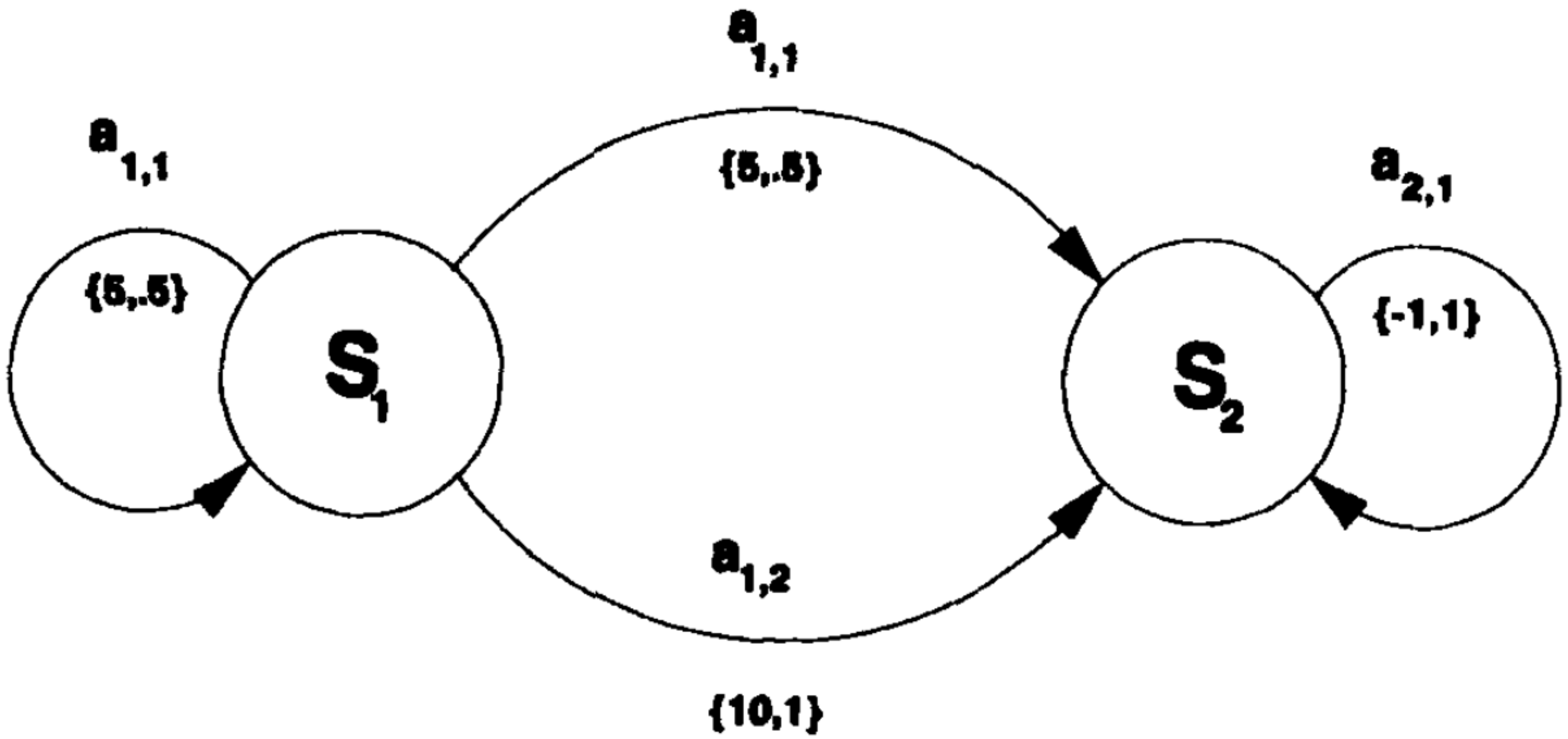
A *policy* is a plan of what action to take in each state.



Policies

A *policy* is a plan of what action to take in each state.

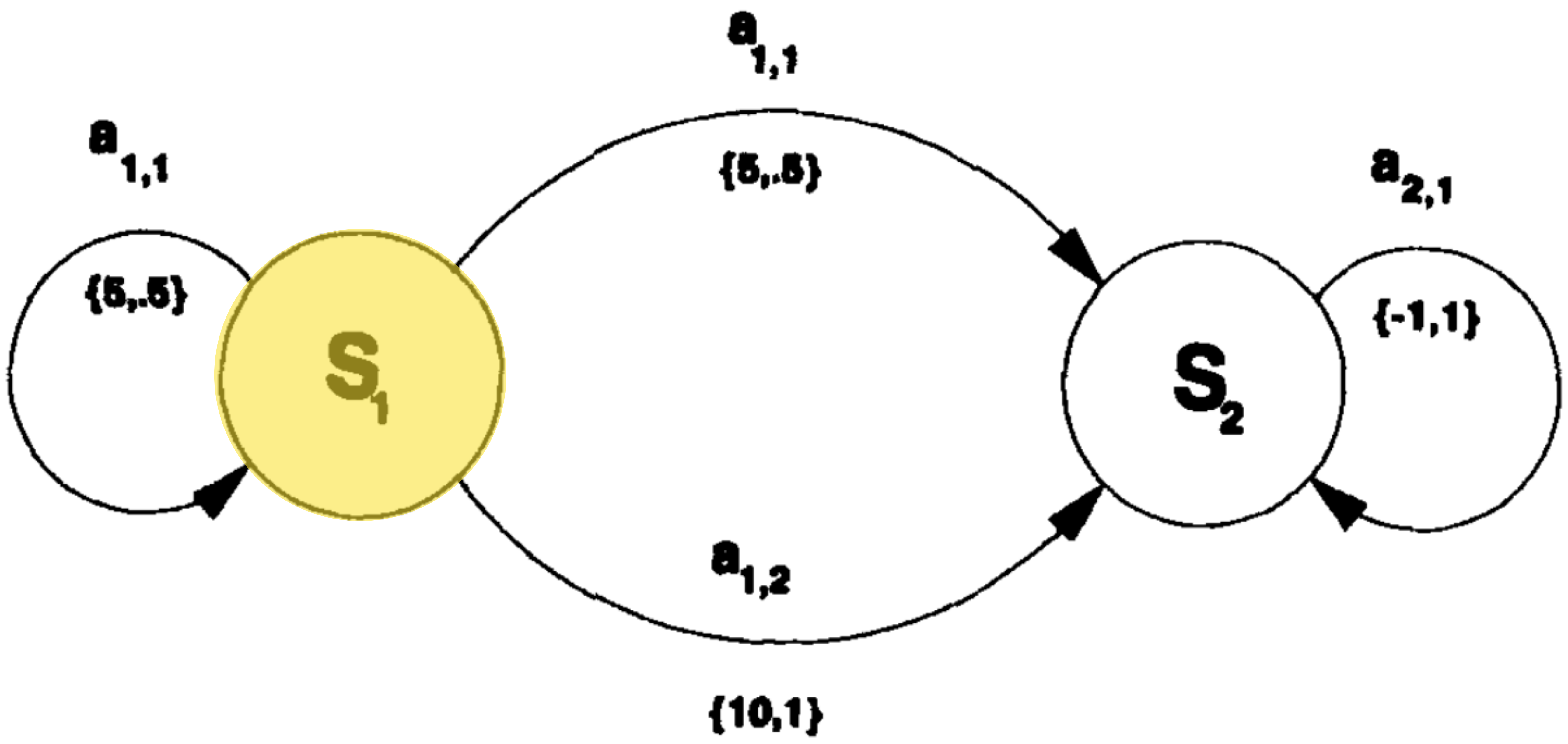
$$\pi(s_1) = a_{1,2} \quad \pi(s_2) = a_{2,1}$$



Value

The *value* of M under π is: $V^{\pi}(s) = E_{\pi}\left[\sum_{h=1}^{H} r_h(s,a) \mid s_0 = s\right]$.

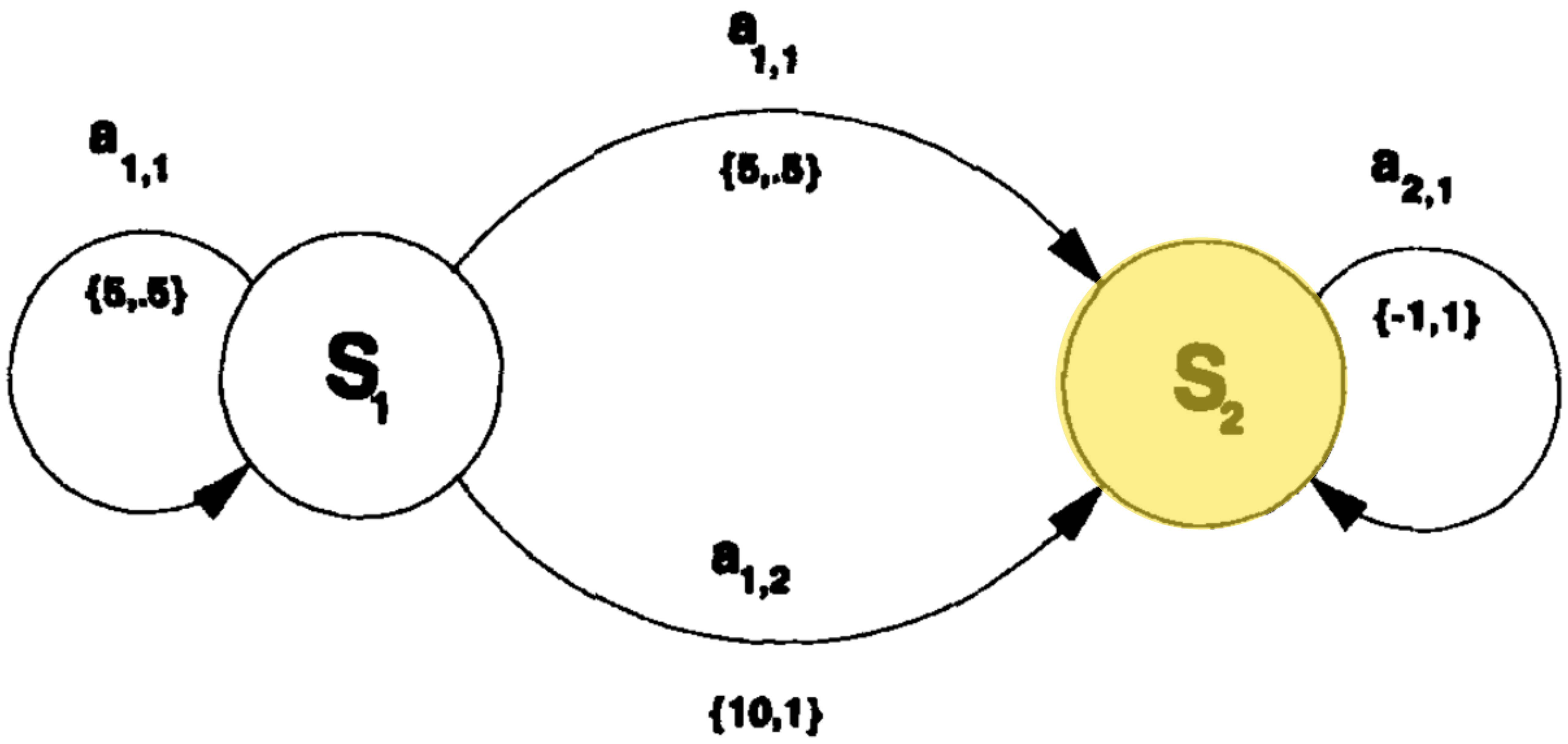
Value



$\pi(s_1) = a_{1,2}$

Reward = 10

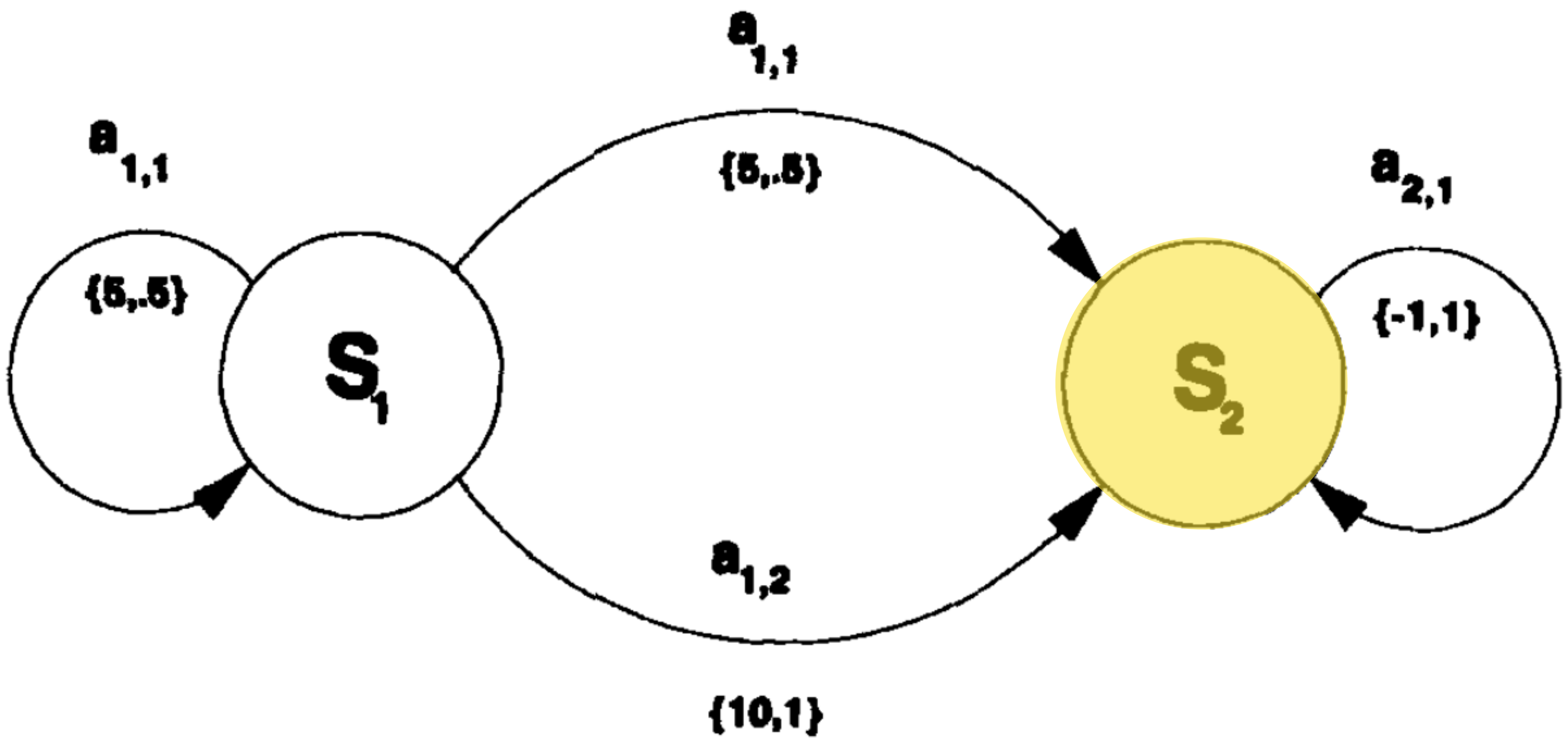
Value



$\pi(s_2) = a_{2,1}$

Reward = -1

Value



$\pi(s_2) = a_{2,1}$

Reward = -1

Value

$$V^{\pi}(s_1) = 10 - 1 - 1 = 8$$

Optimal Policies

Optimal Policies

$$\pi^* = \sup_{\pi} V^{\pi}(s_0)$$

Example MDP

Example MDP

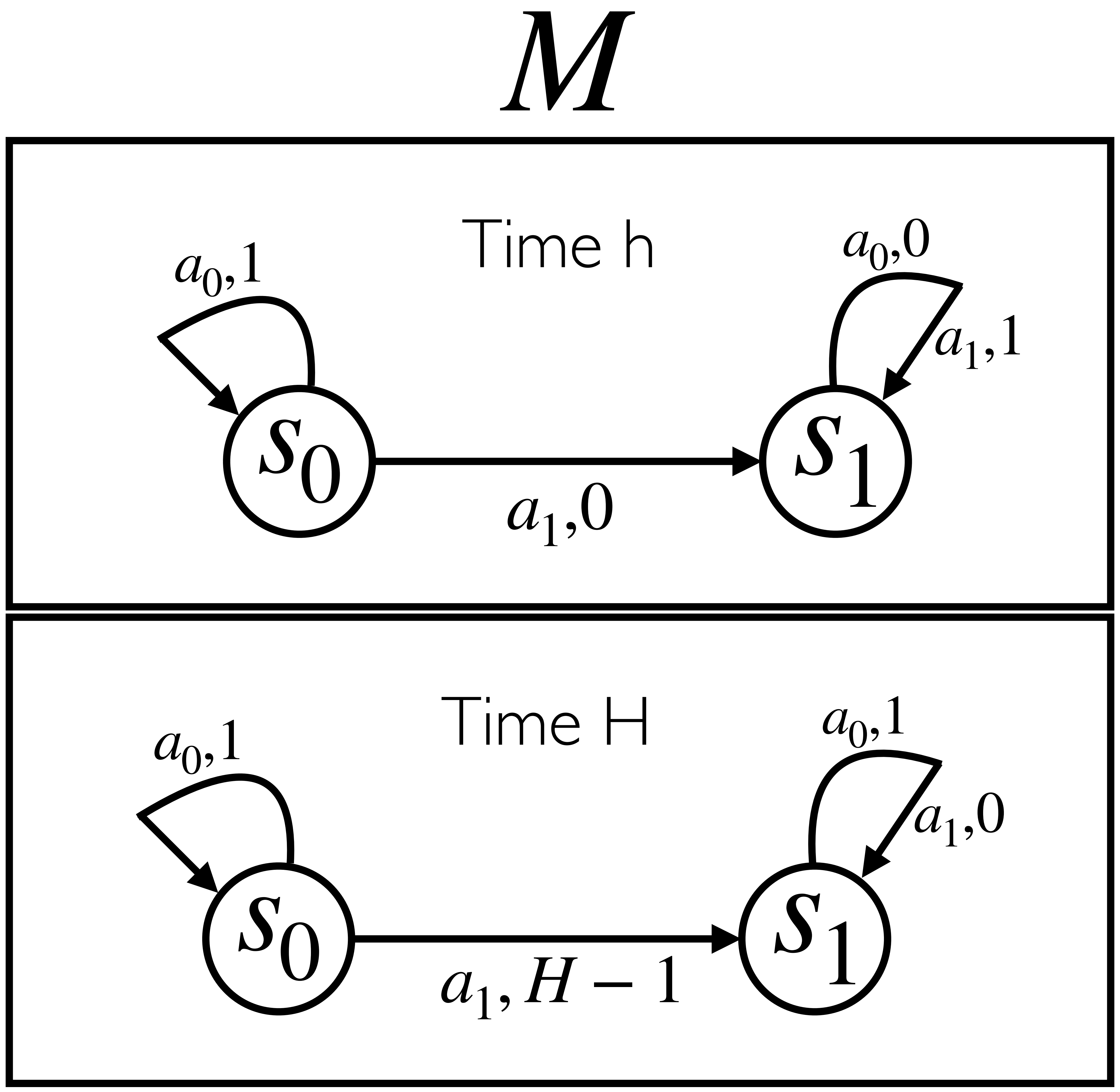
Example MDP

Disaster Relief with Autonomous Vehicles

- State Space is $\mathbb{R}^2$
- Action Space is $[-1,1]^2$
- New location is $s + a$
- Reward for finding people in need.

Performance of Optimal Policies

Performance of Optimal Policies

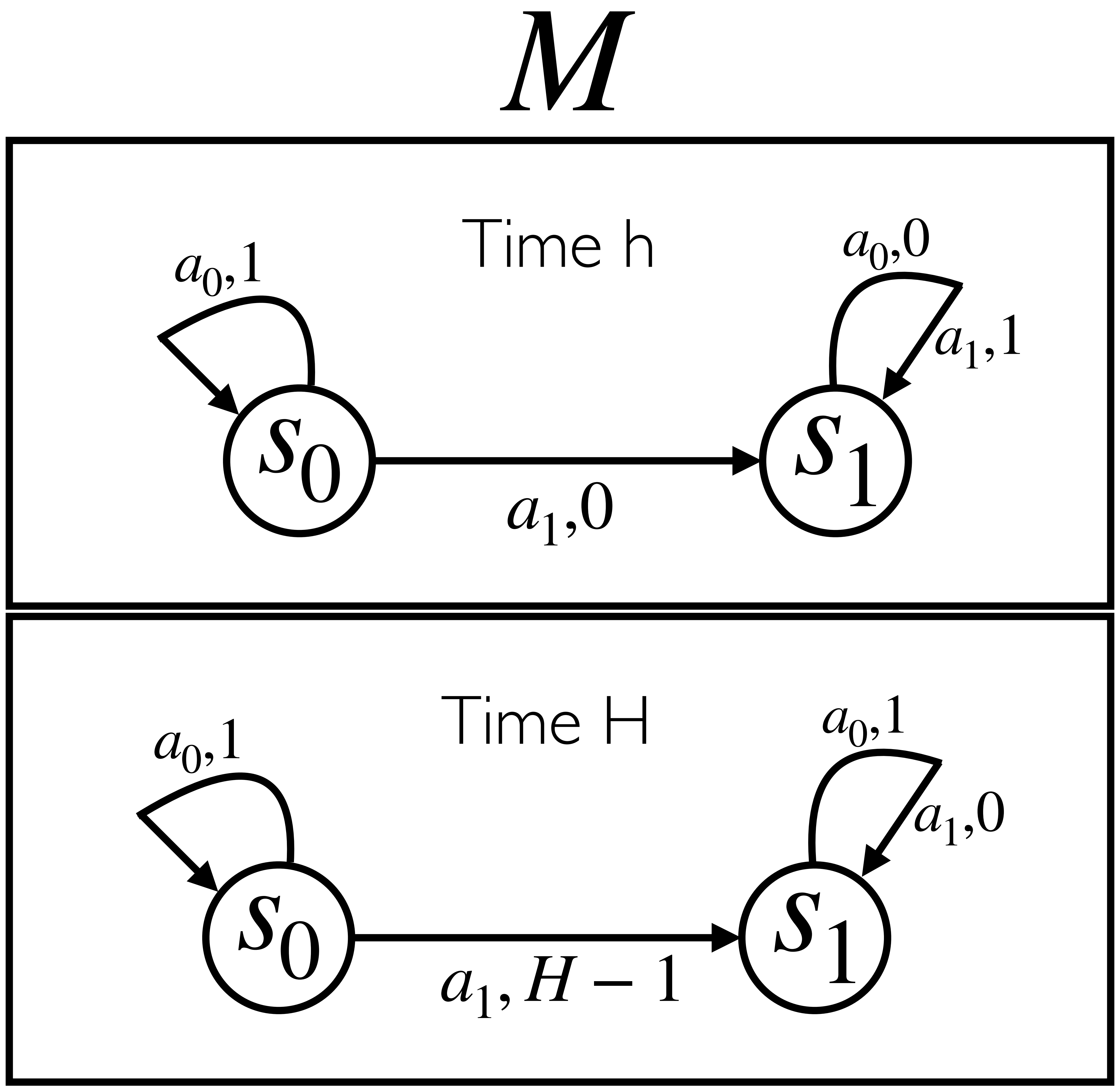


Performance of Optimal Policies

Unique optimal policy π^* is:

π^*

t/S	s_0	s_1
h	a_0	a_1
H	a_1	a_0



Performance of Optimal Policies

Unique optimal policy π^* is:

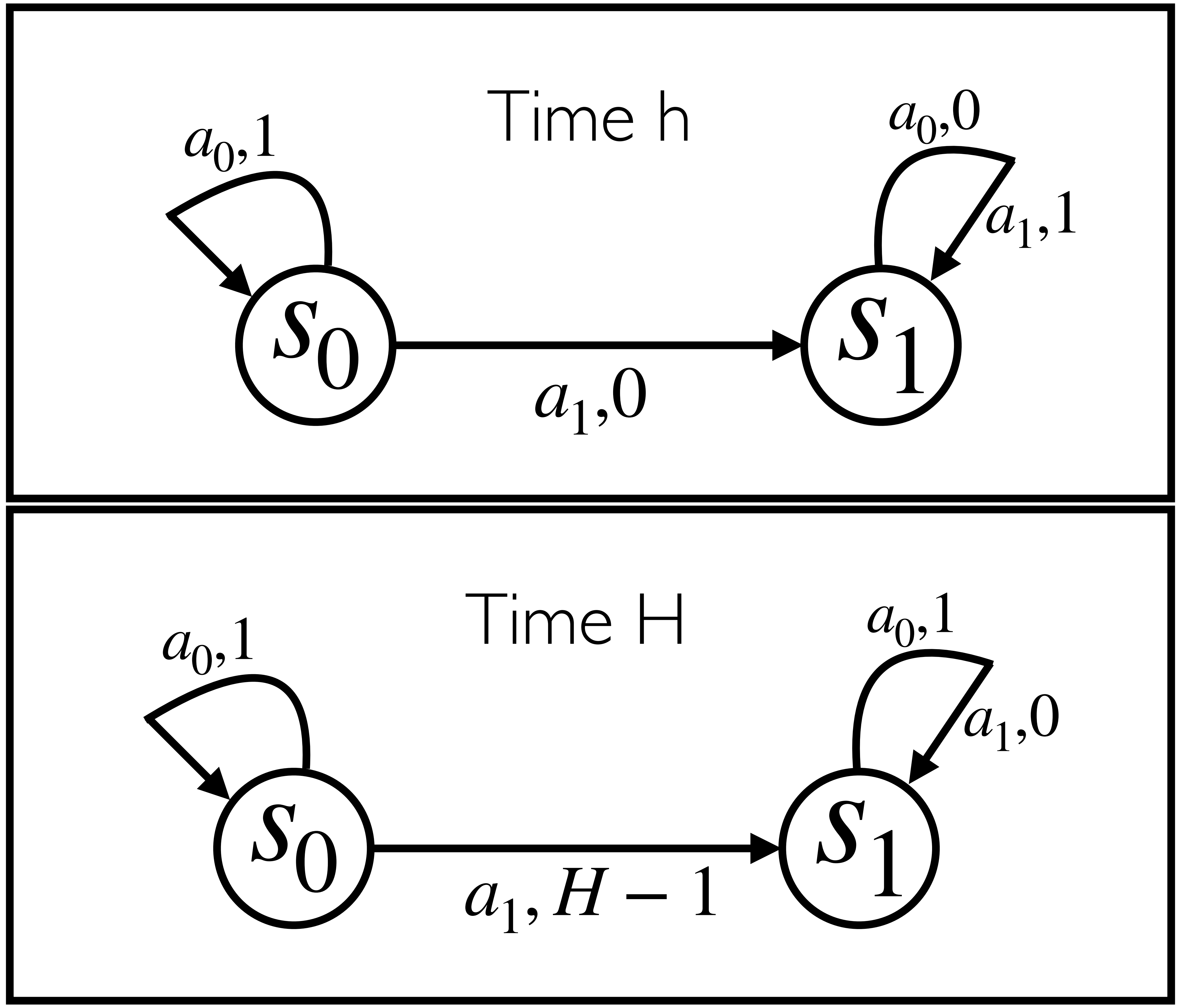
π^*

t/S	s_0	s_1
h	a_0	a_1
H	a_1	a_0

The optimal policy achieves value:

$$V_M^{\pi^*} = 2(H-1)$$

M



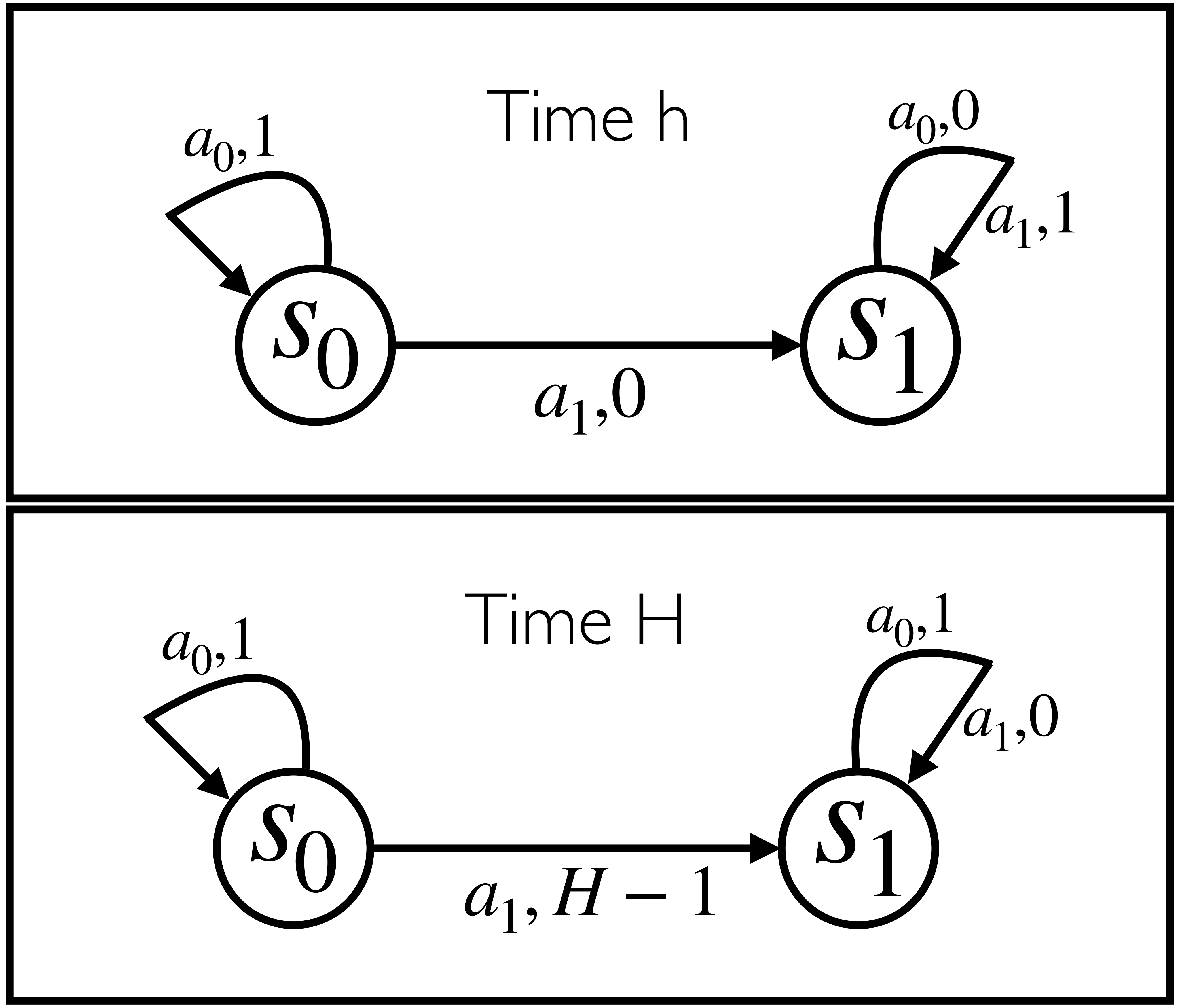
Optimal Policies are NOT Robust

Optimal Policies are NOT Robust

Optimal Policies may behave poorly under measurement noise or adversarial manipulations!

Optimal Policies are NOT Robust

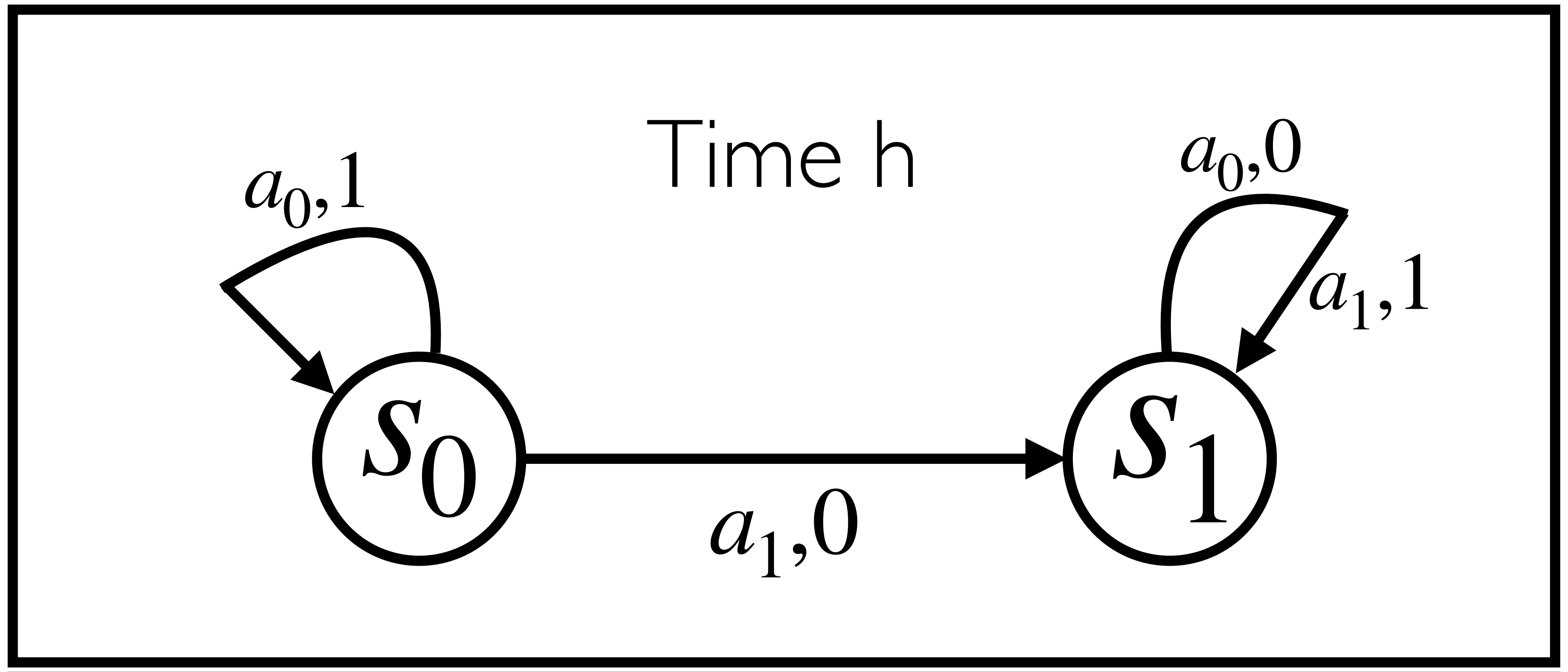
Optimal Policies may behave poorly under measurement noise or adversarial manipulations!

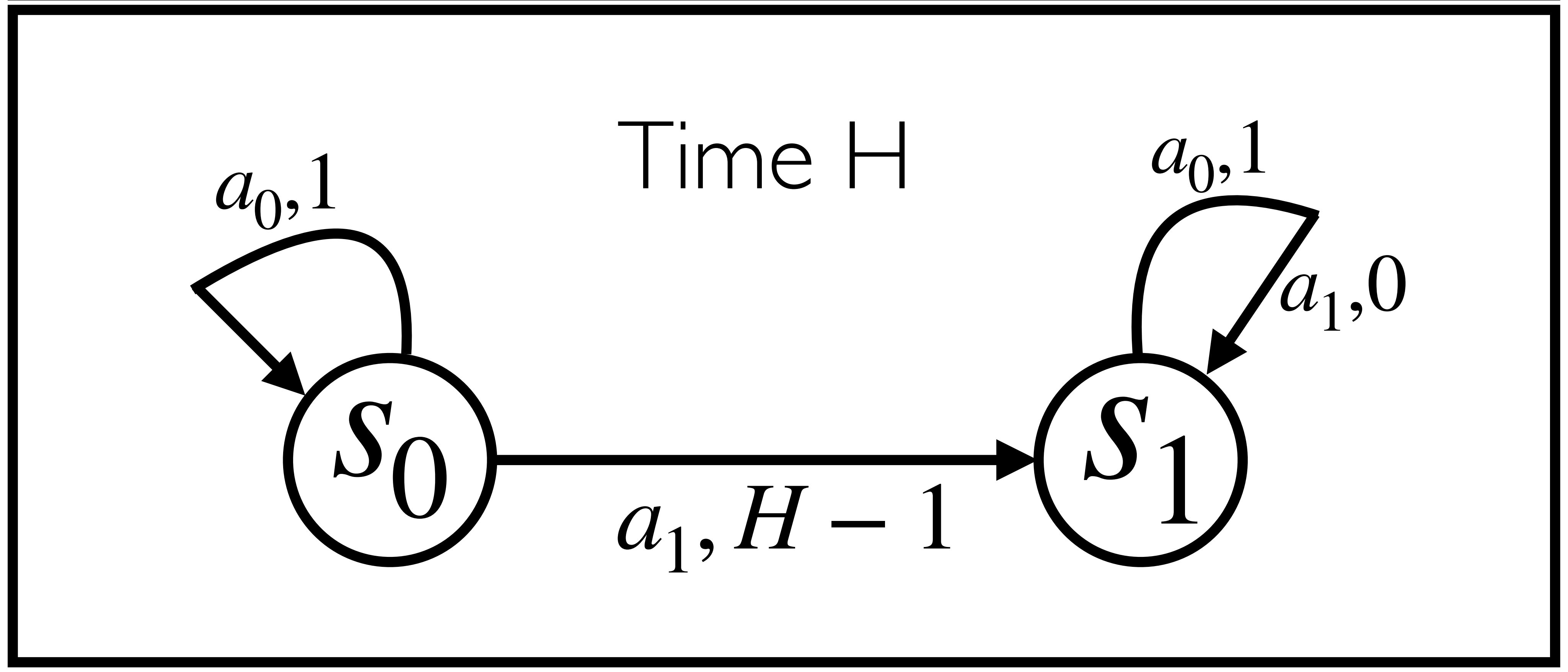


Optimal Policies are NOT Robust

Optimal Policies may behave poorly under measurement noise or adversarial manipulations!

- If first state is actually s_1 or M receives a_1 instead, π^* at best gets $1/2$ of its value.

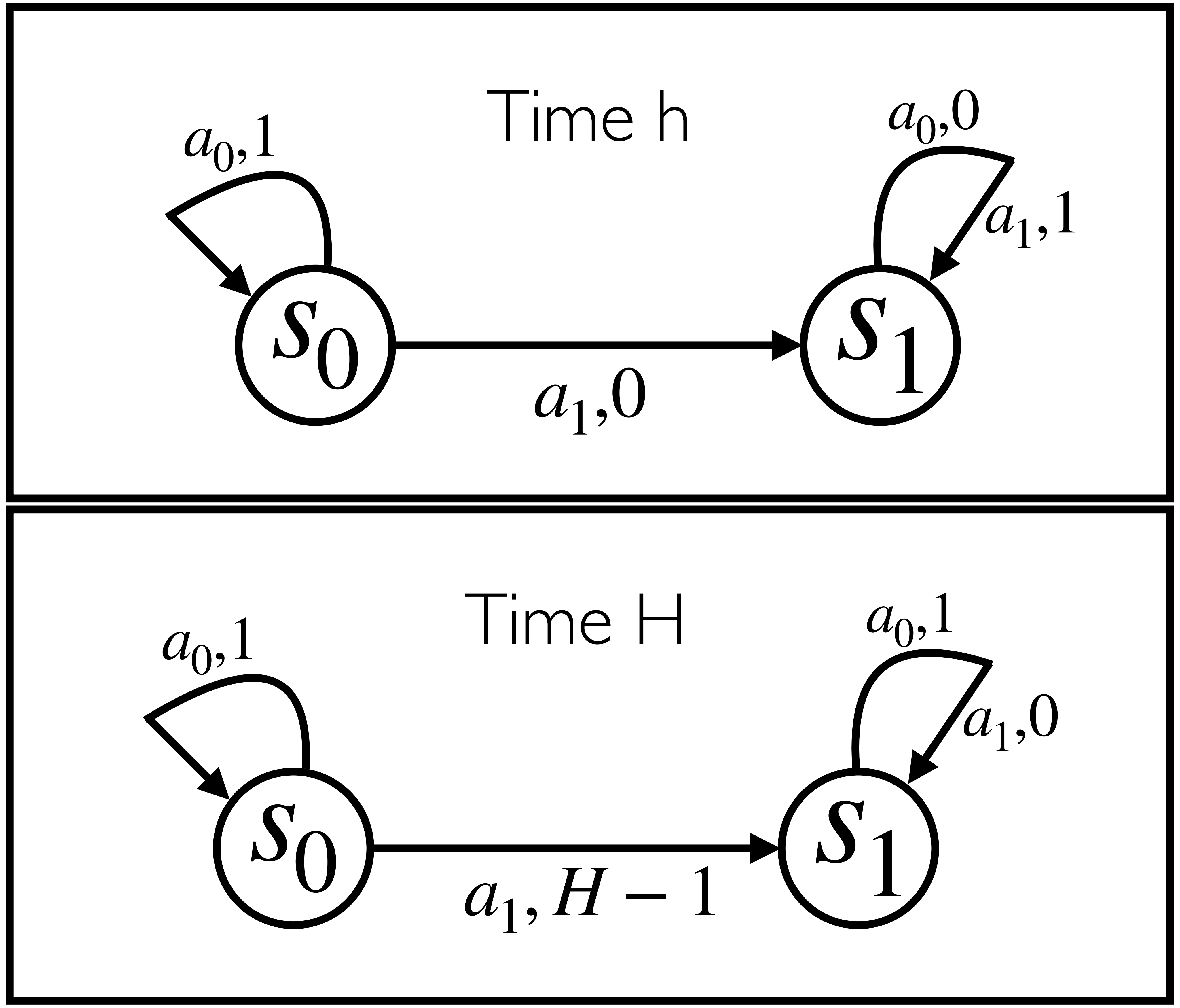




Optimal Policies are NOT Robust

Optimal Policies may behave poorly under measurement noise or adversarial manipulations!

- If first state is actually s_1 or M receives a_1 instead, π^* at best gets $1/2$ of its value.
- If states are swapped consistently, π^* gets no value!



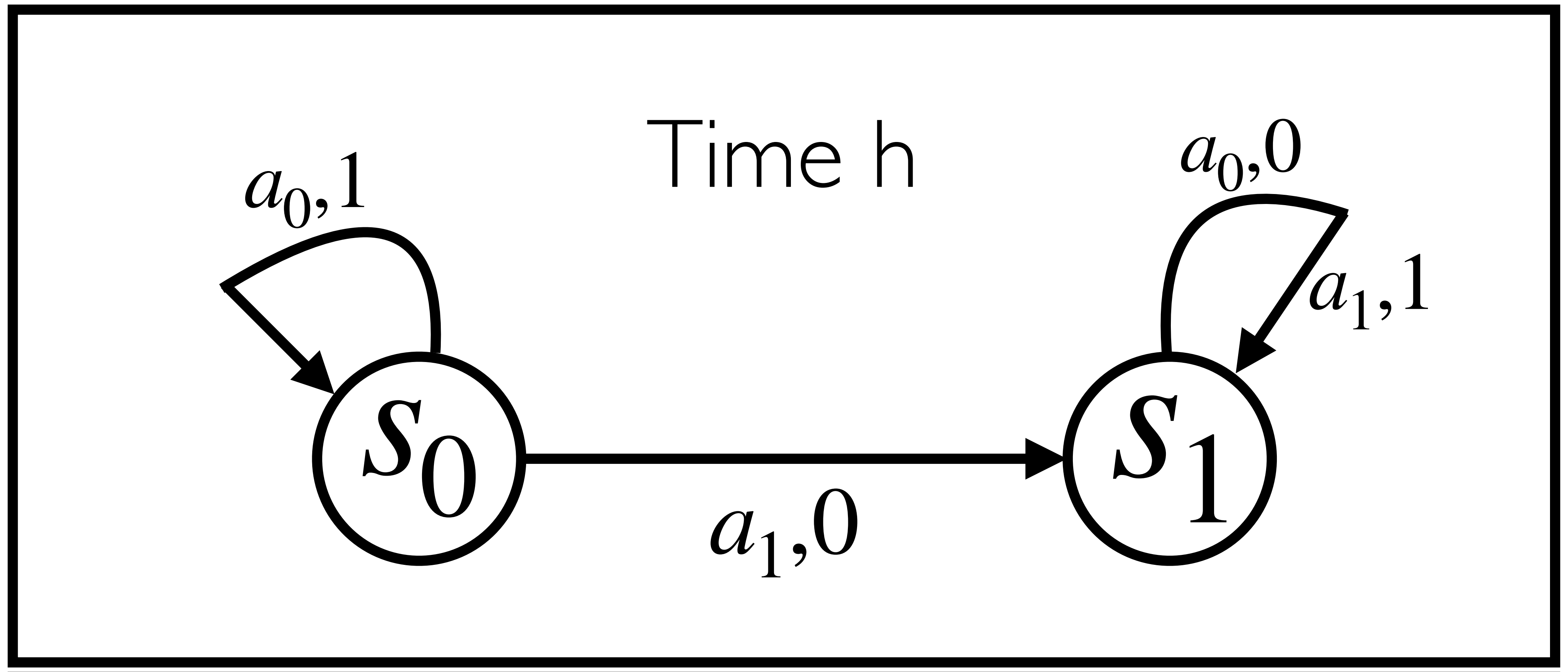
Optimal Policies are NOT Robust

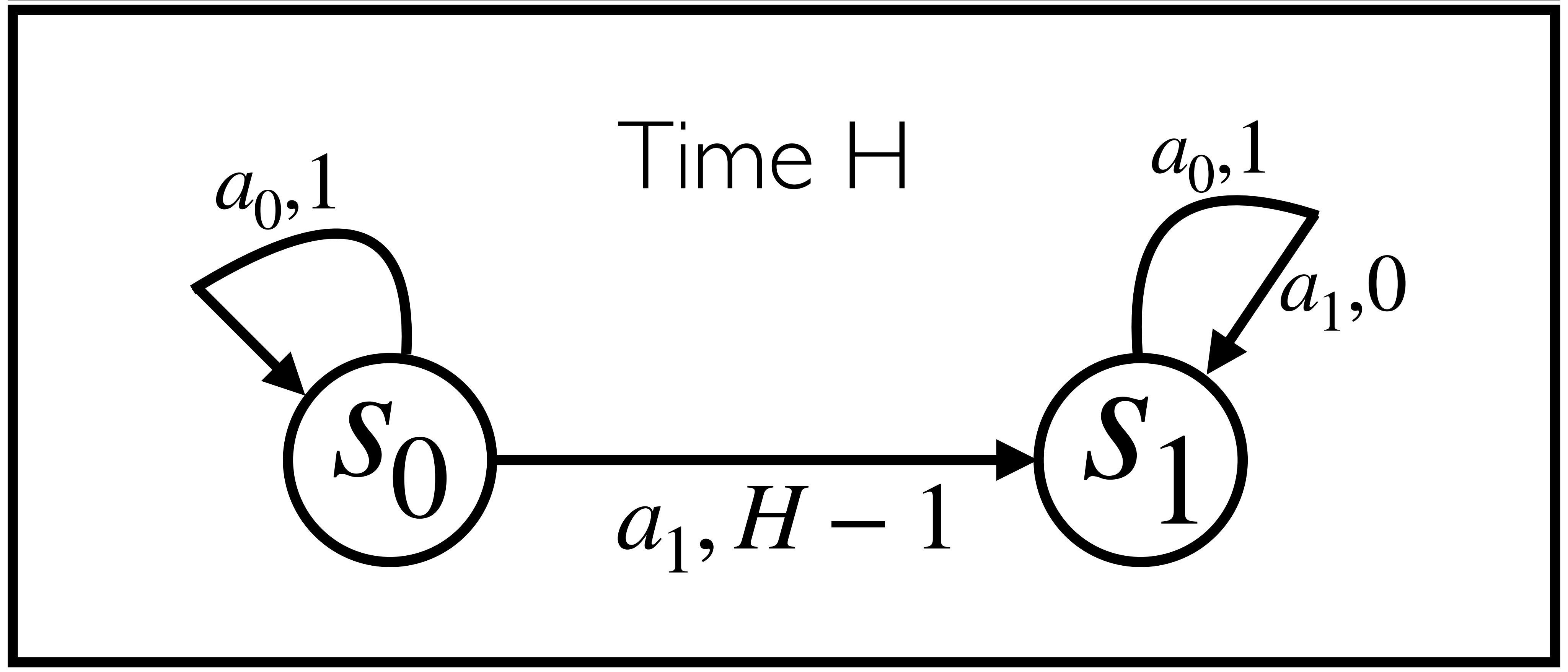
Optimal Policies may behave poorly under measurement noise or adversarial manipulations!

- If first state is actually s_1 or M receives a_1 instead, π^* at best gets $1/2$ of its value.
- If states are swapped consistently, π^* gets no value!

$\pi^* \circ \nu$

t/S	s_0	s_1
h	a_1	a_0
H	a_0	a_1





Security Threats to RL

Security Threats to RL

- Playing an optimal policy for the ideal environment is not always optimal for the real environment!

Security Threats to RL

- Playing an optimal policy for the ideal environment is not always optimal for the real environment!
- Strategies to compute robust policies are needed.

Security Threats to RL

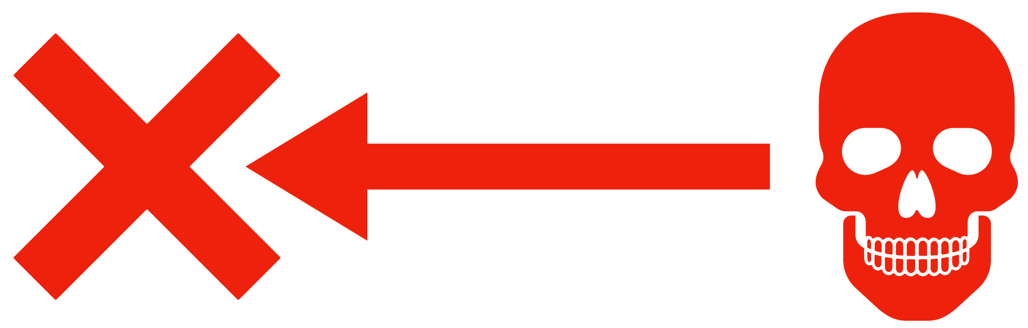
- Playing an optimal policy for the ideal environment is not always optimal for the real environment!
- Strategies to compute robust policies are needed.
- Inspiration for field of adversarial RL.

Adversarial RL

Adversarial RL

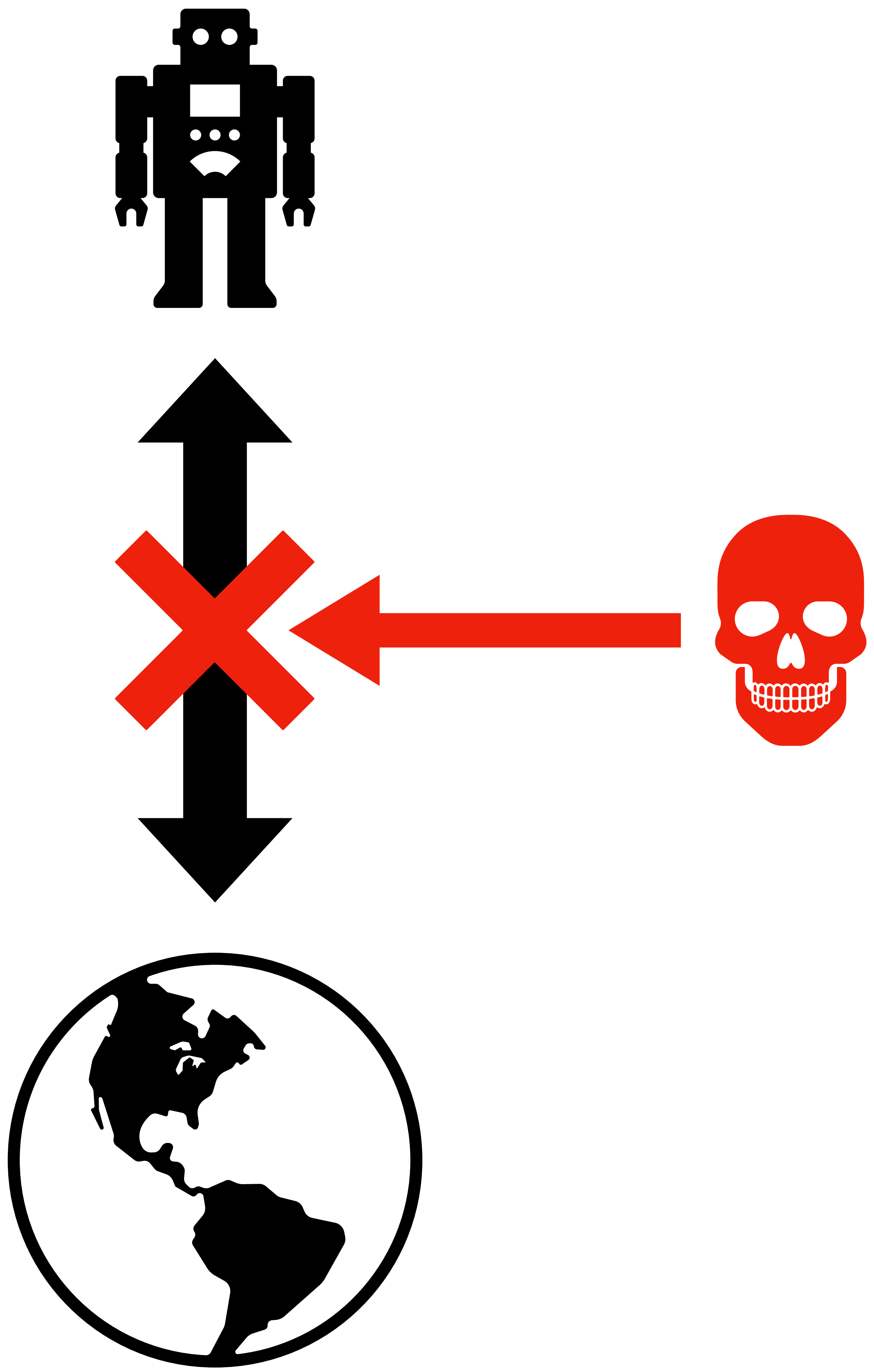
Adversarial RL

An external attacker can manipulate the interaction.



Adversarial RL

An external attacker can manipulate the interaction.



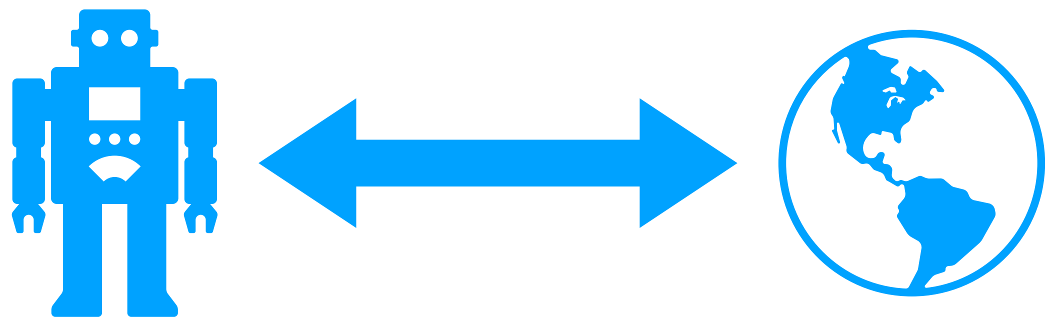
Attack Paradigms

Attack Paradigms

Training Time

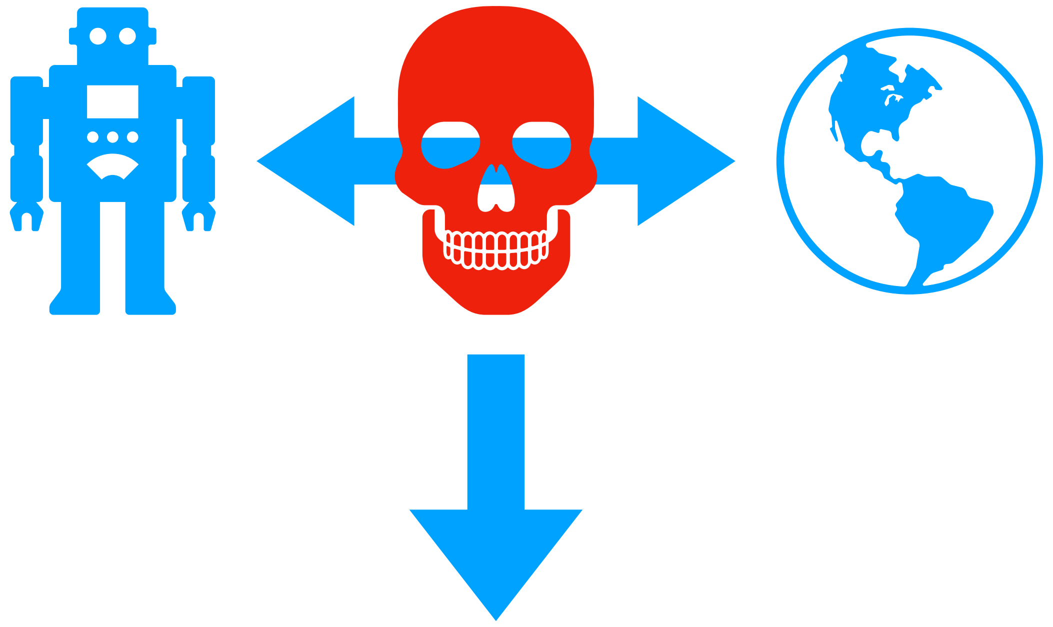
Attack Paradigms

Training Time



Attack Paradigms

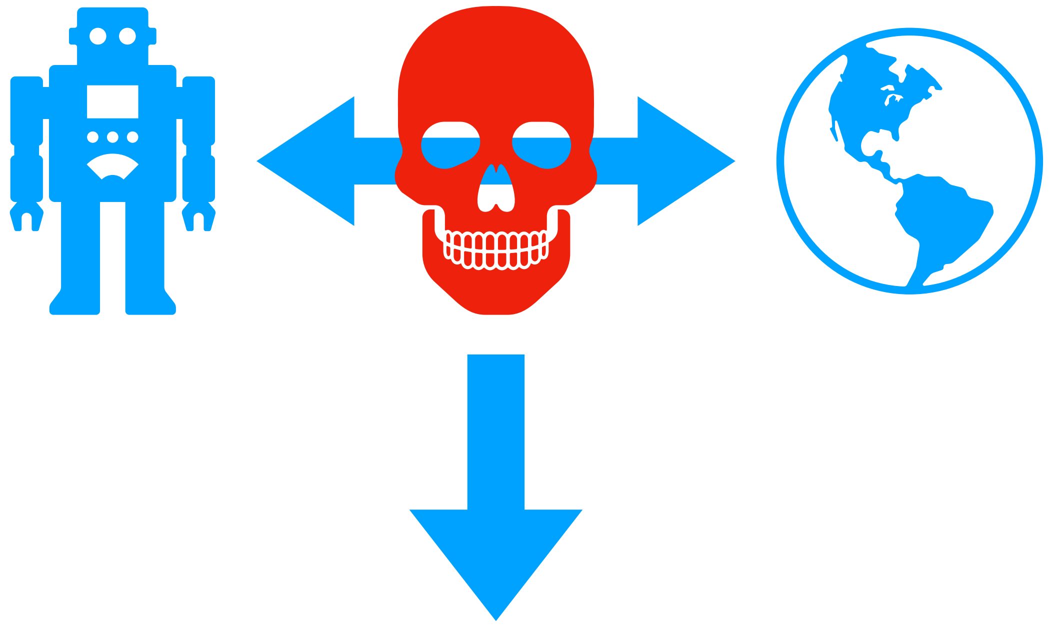
Training Time



Learn bad $\pi^\dagger$

Attack Paradigms

Training Time

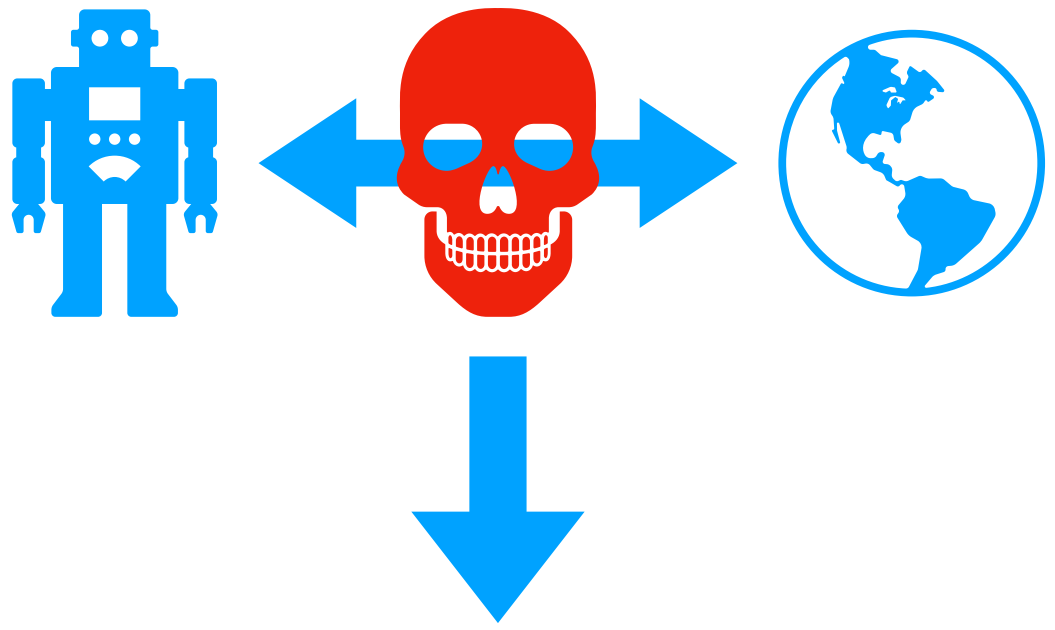


Learn bad $\pi^{\dagger}$

Test Time

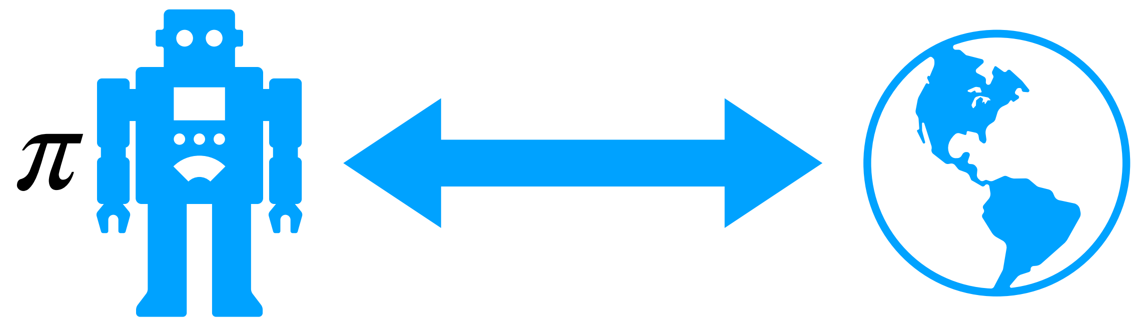
Attack Paradigms

Training Time



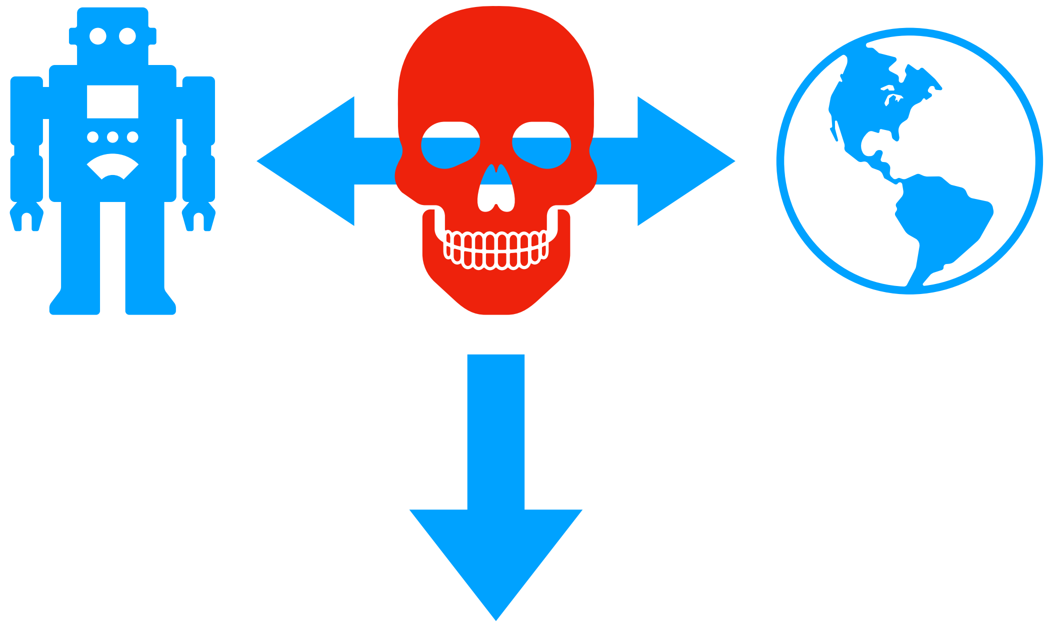
Learn bad $\pi^\dagger$

Test Time



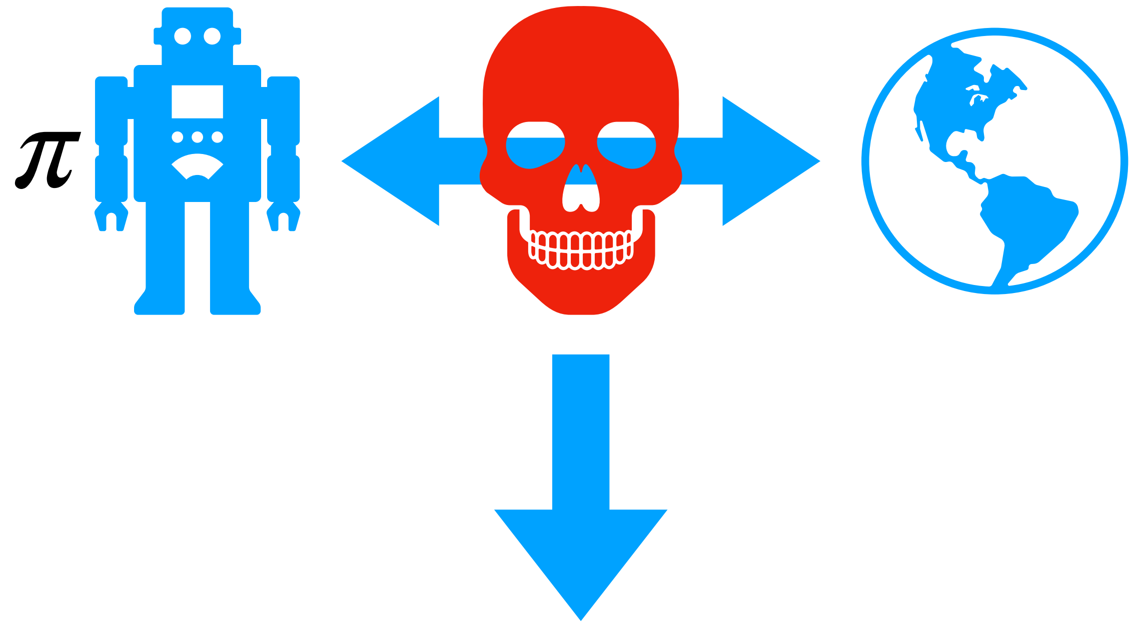
Attack Paradigms

Training Time



Learn bad $\pi^\dagger$

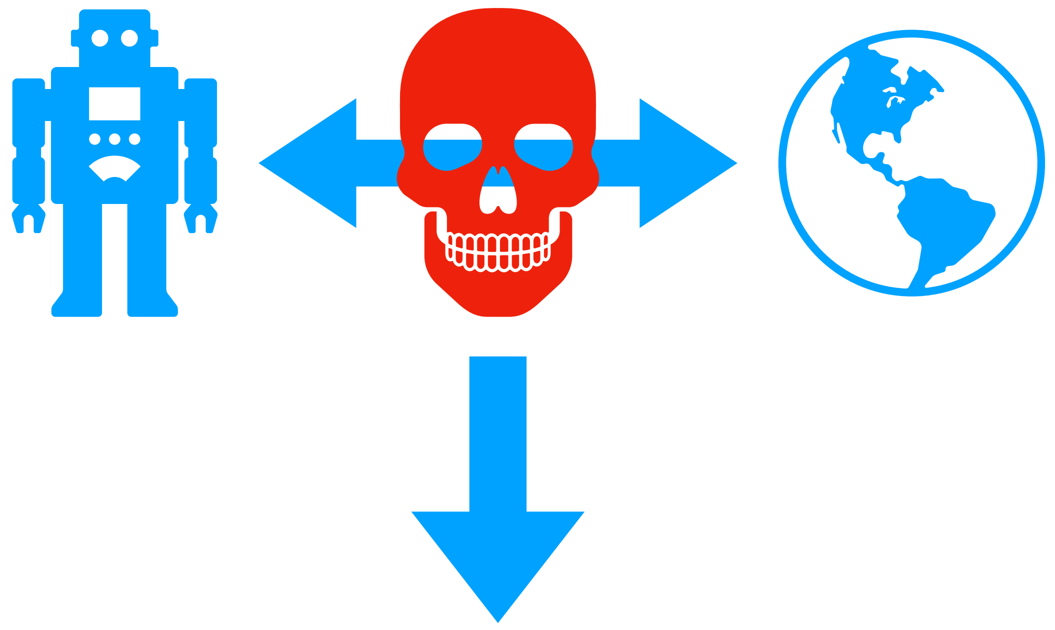
Test Time



Cause bad outcomes

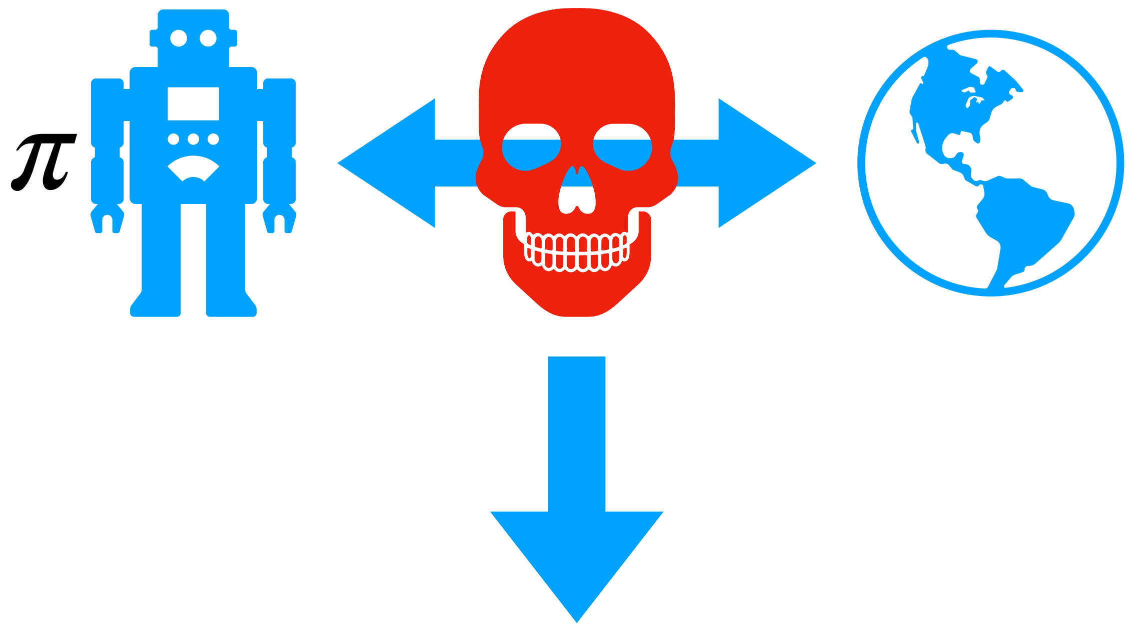
Attack Paradigms

Training Time



Learn bad $\pi^\dagger$

Test Time

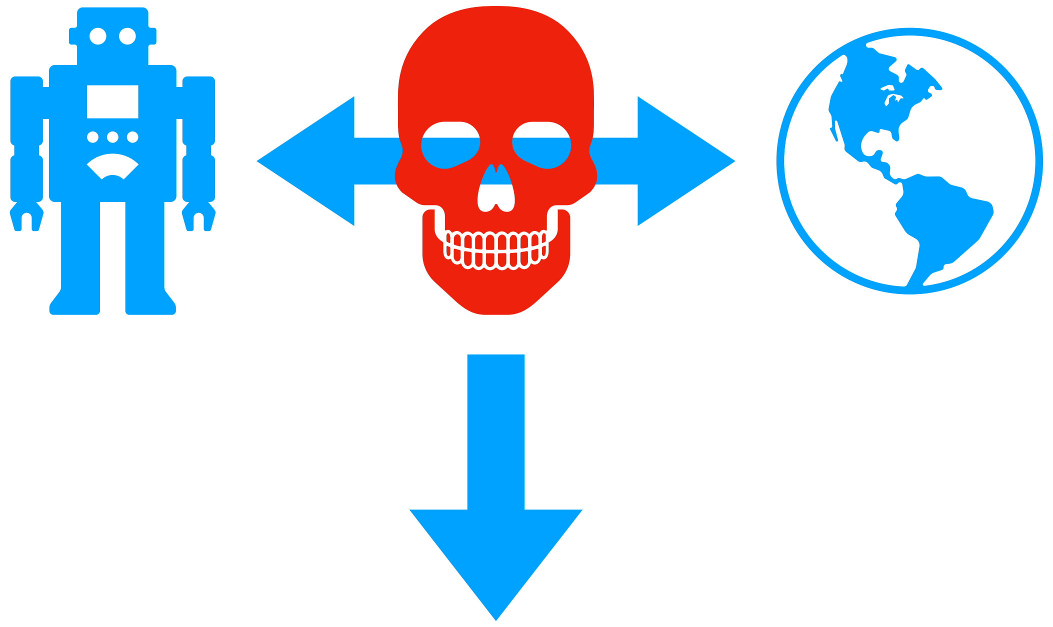


Cause bad outcomes

Trojan

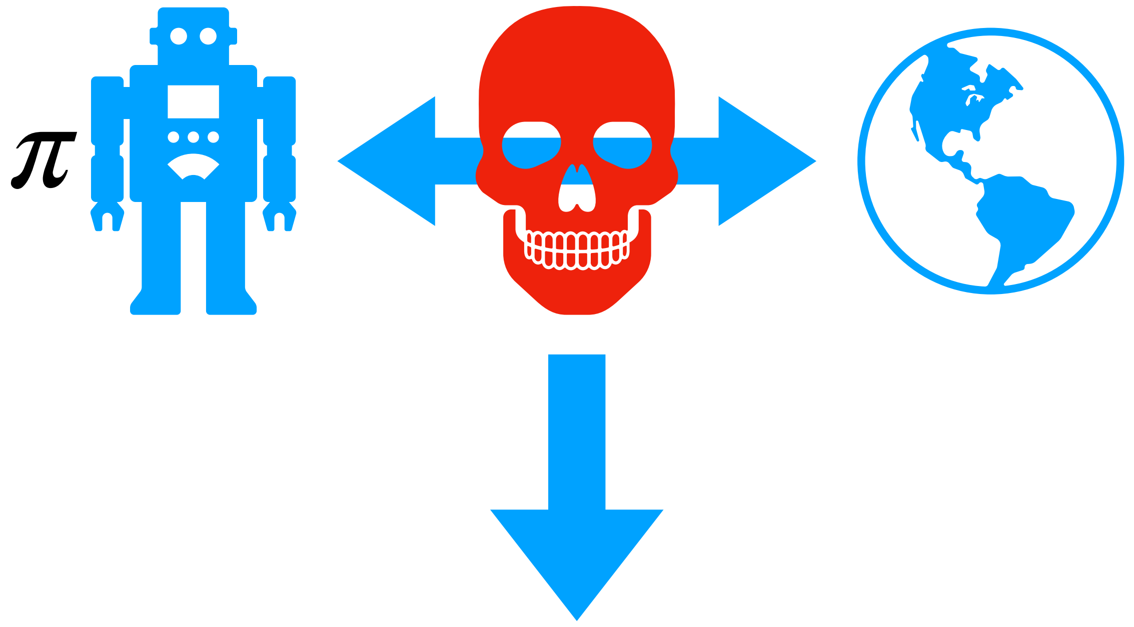
Attack Paradigms

Training Time



Learn bad $\pi^\dagger$

Test Time



Cause bad outcomes

Trojan

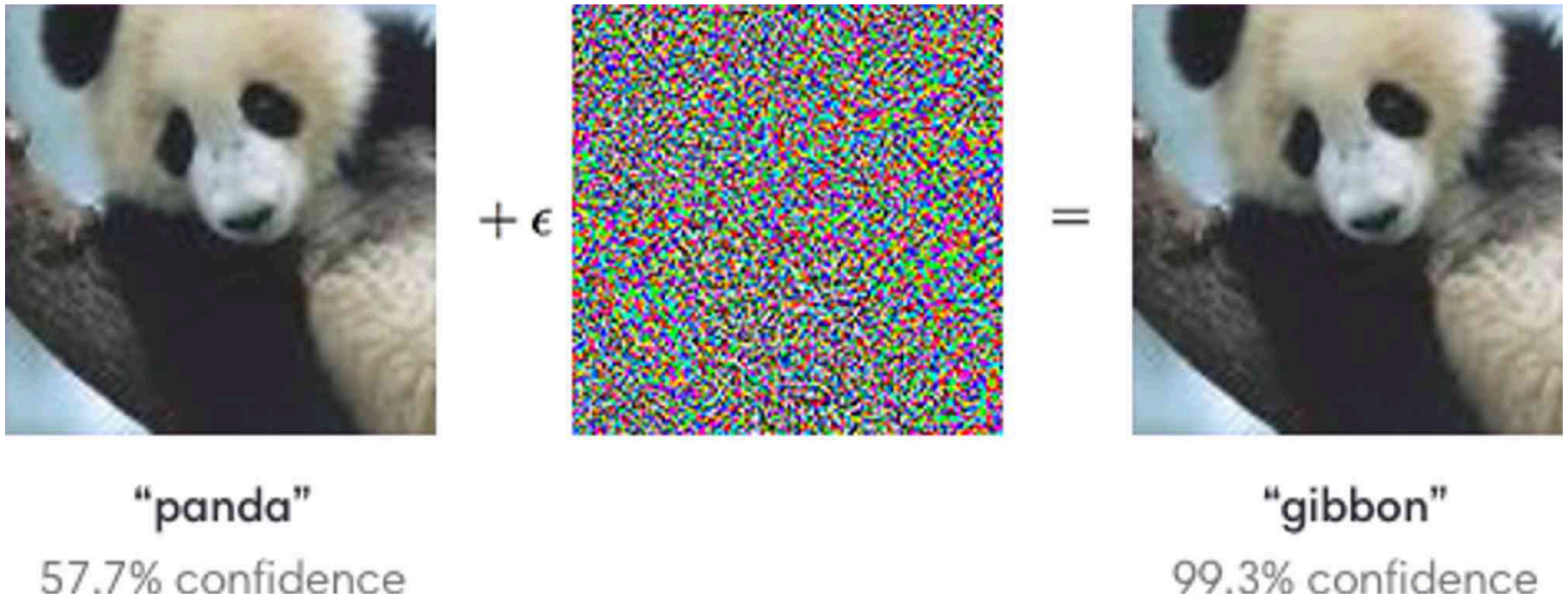
Hybrid: poison training to make policy easily test-time attackable

Panda Example

Panda Example

In Explaining and Harnessing Adversarial Examples, Goodfellow and his team added a small perturbation to the image of a panda, as seen below. The result was surprising. Not only did the classifier mark the panda as a gibbon, but did so with high confidence.

As you can see, a barely noticeable disturbance that appears normal to us can easily deceive an ML model into predicting an incorrect class.



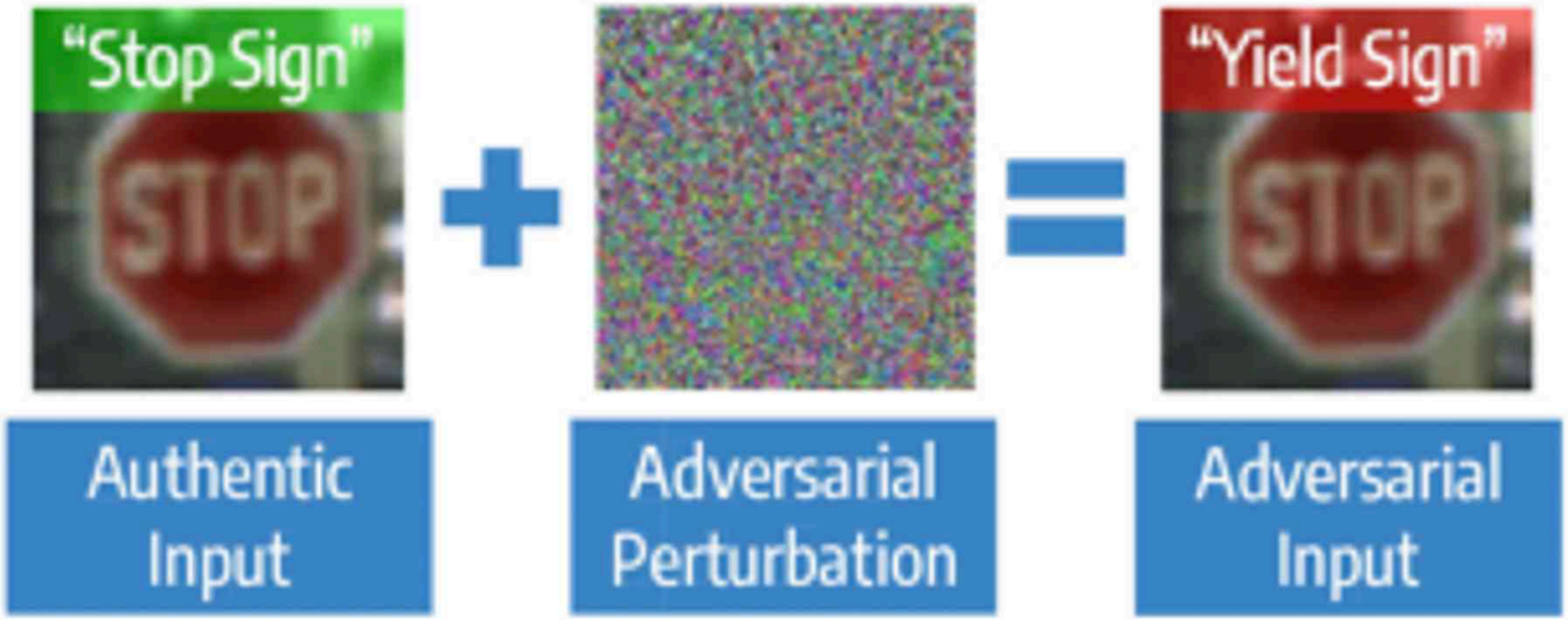
Source: Goodfellow et al, 2014

Car Crashing

Car Crashing

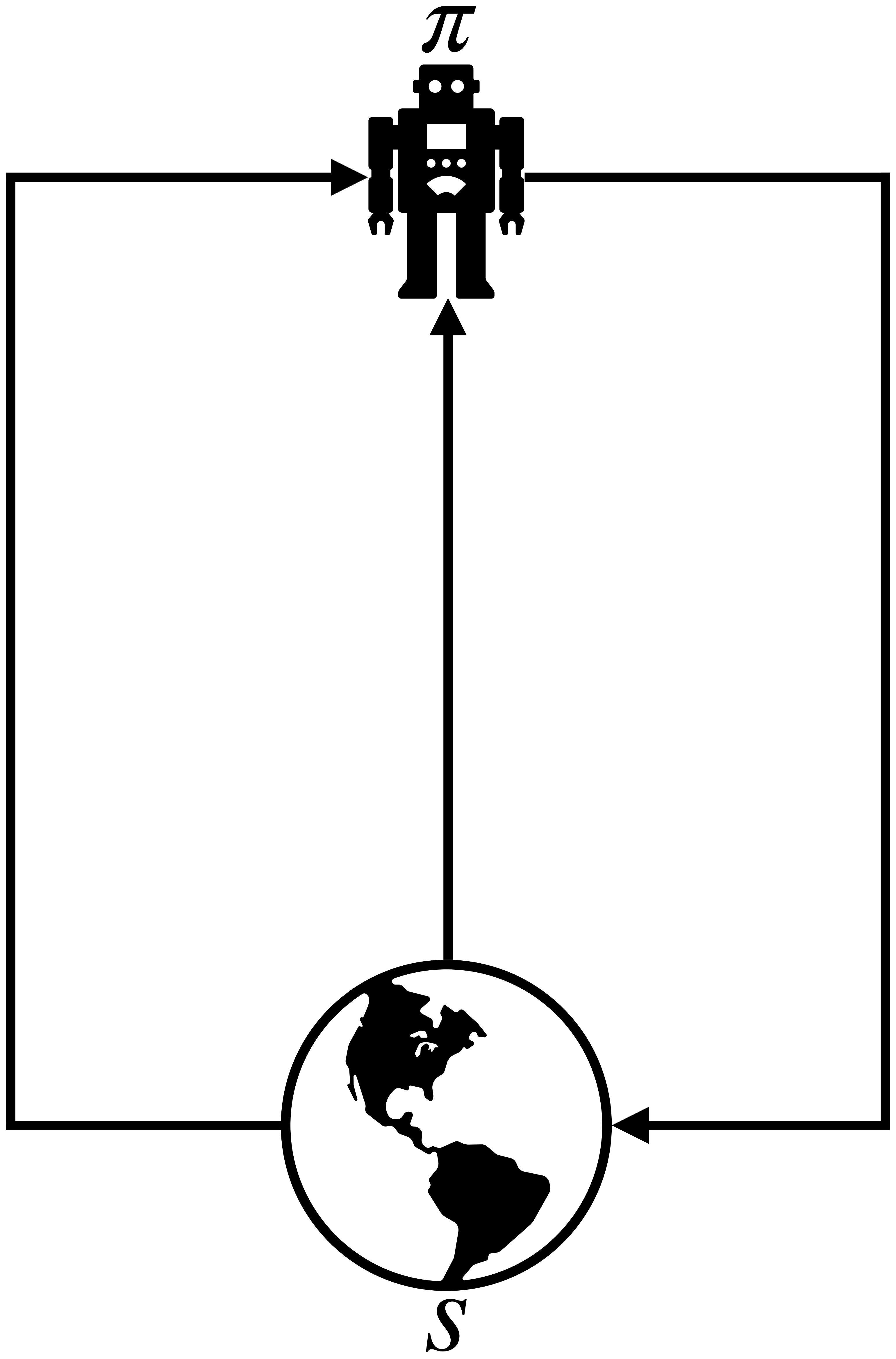
While the panda turned gibbon in the eyes of a machine is a harmless example of an adversarial attack, there are other forms of danger we must watch out for.

For instance, adversarial examples can also be used to hijack the ML models behind autonomous vehicles, causing them to misclassify 'stop' signs as 'yield', as seen below.

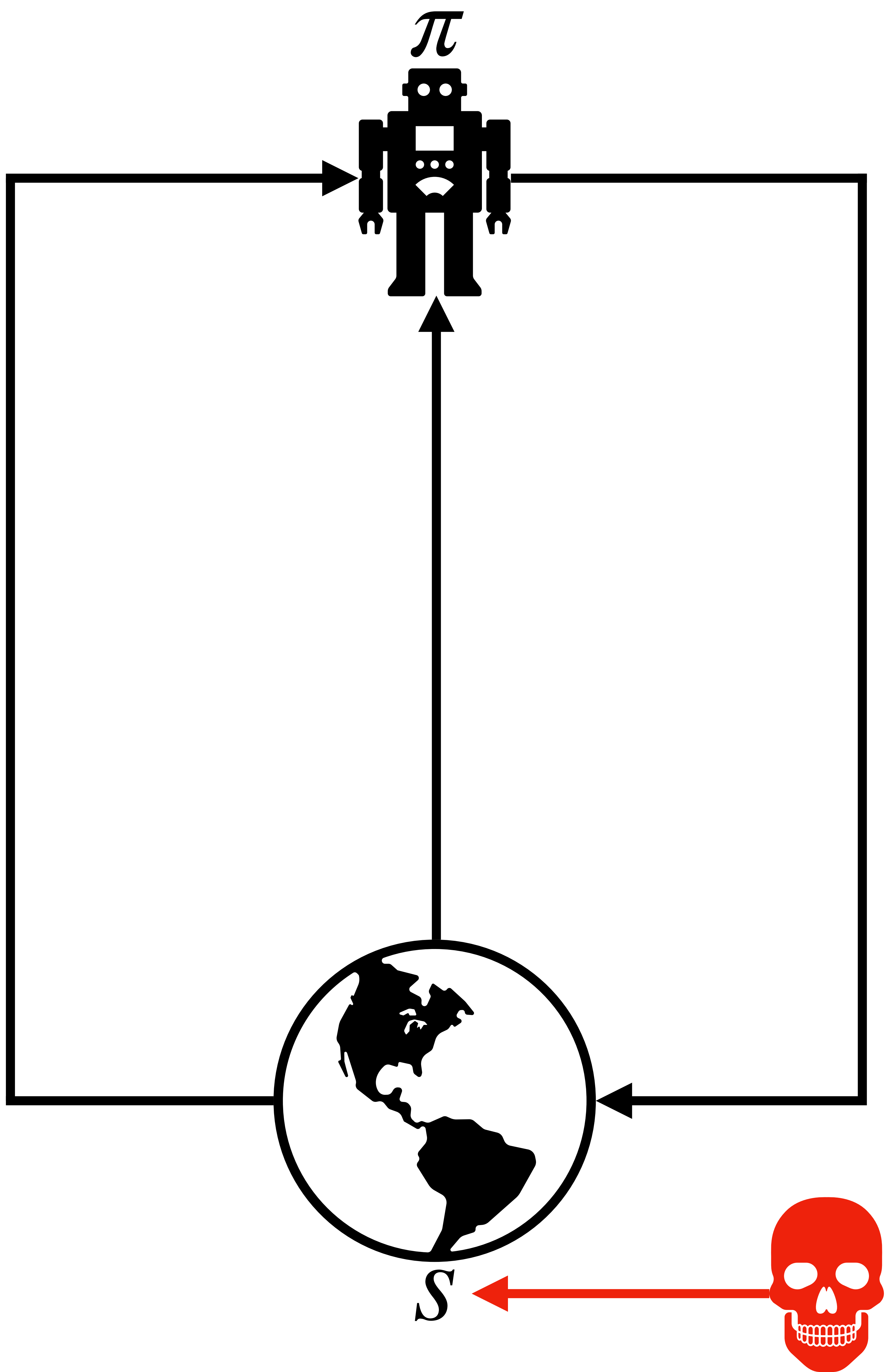


Source: Kumar et al, 2021

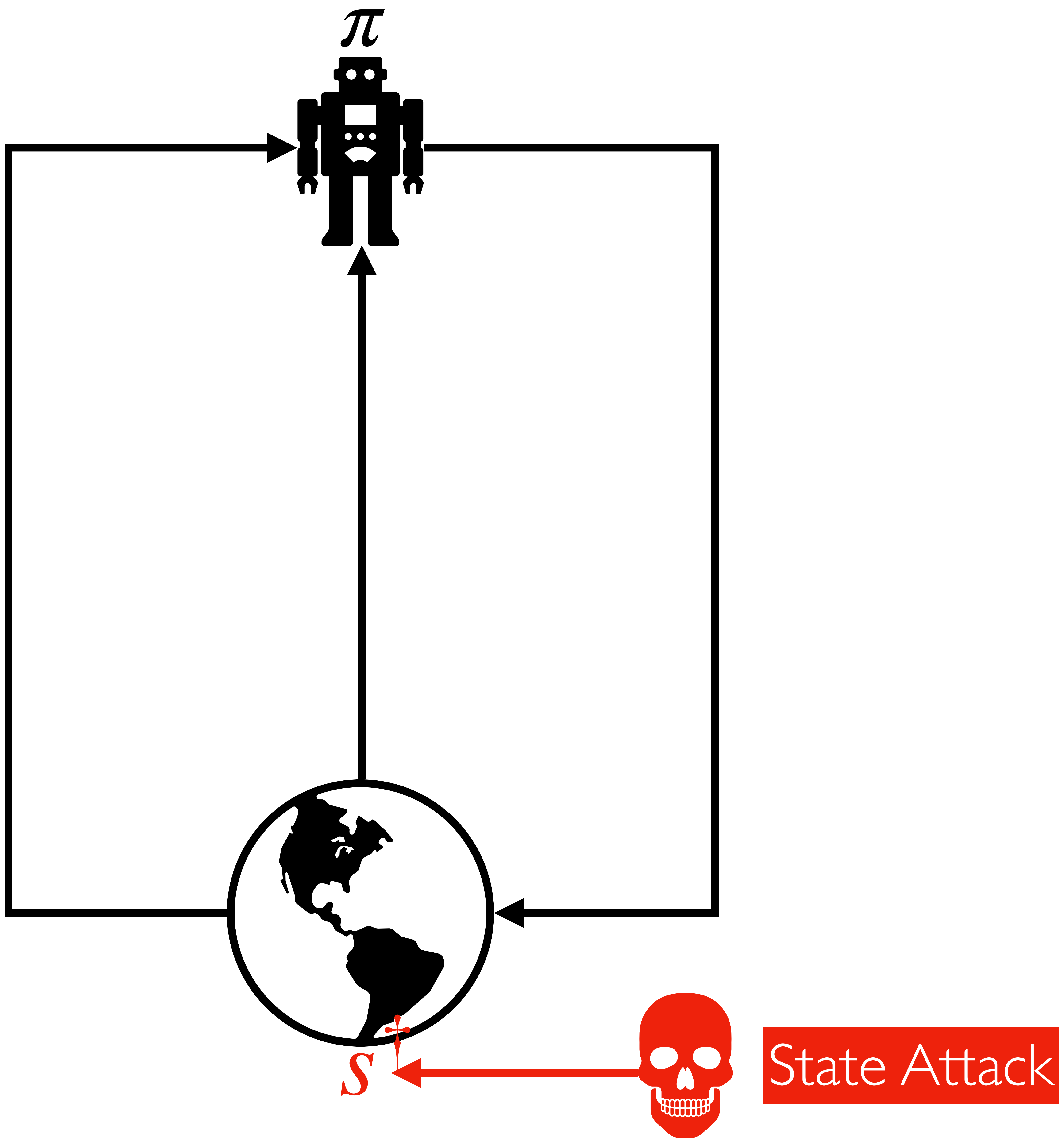
Attack Surfaces



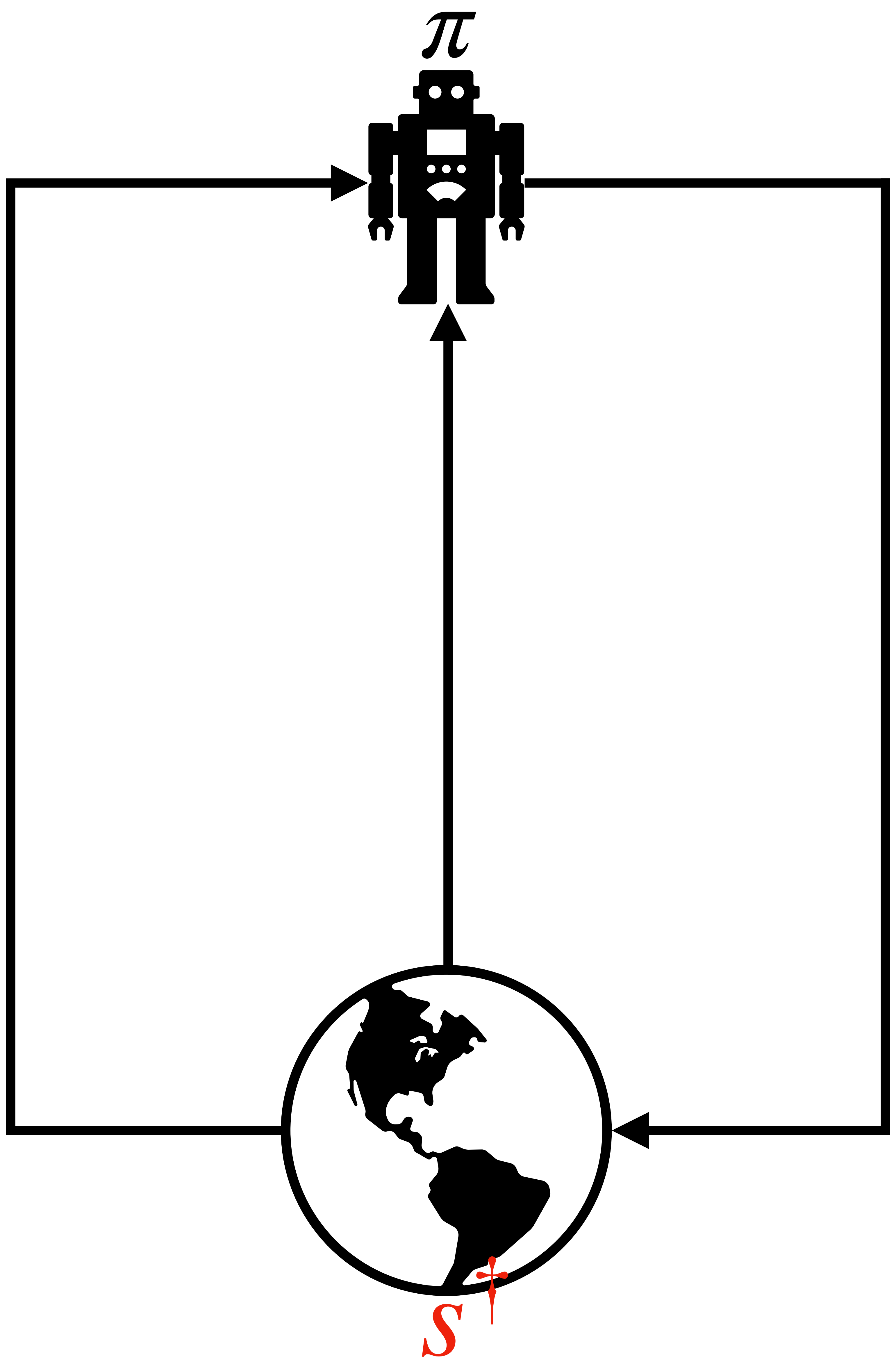
Attack Surfaces



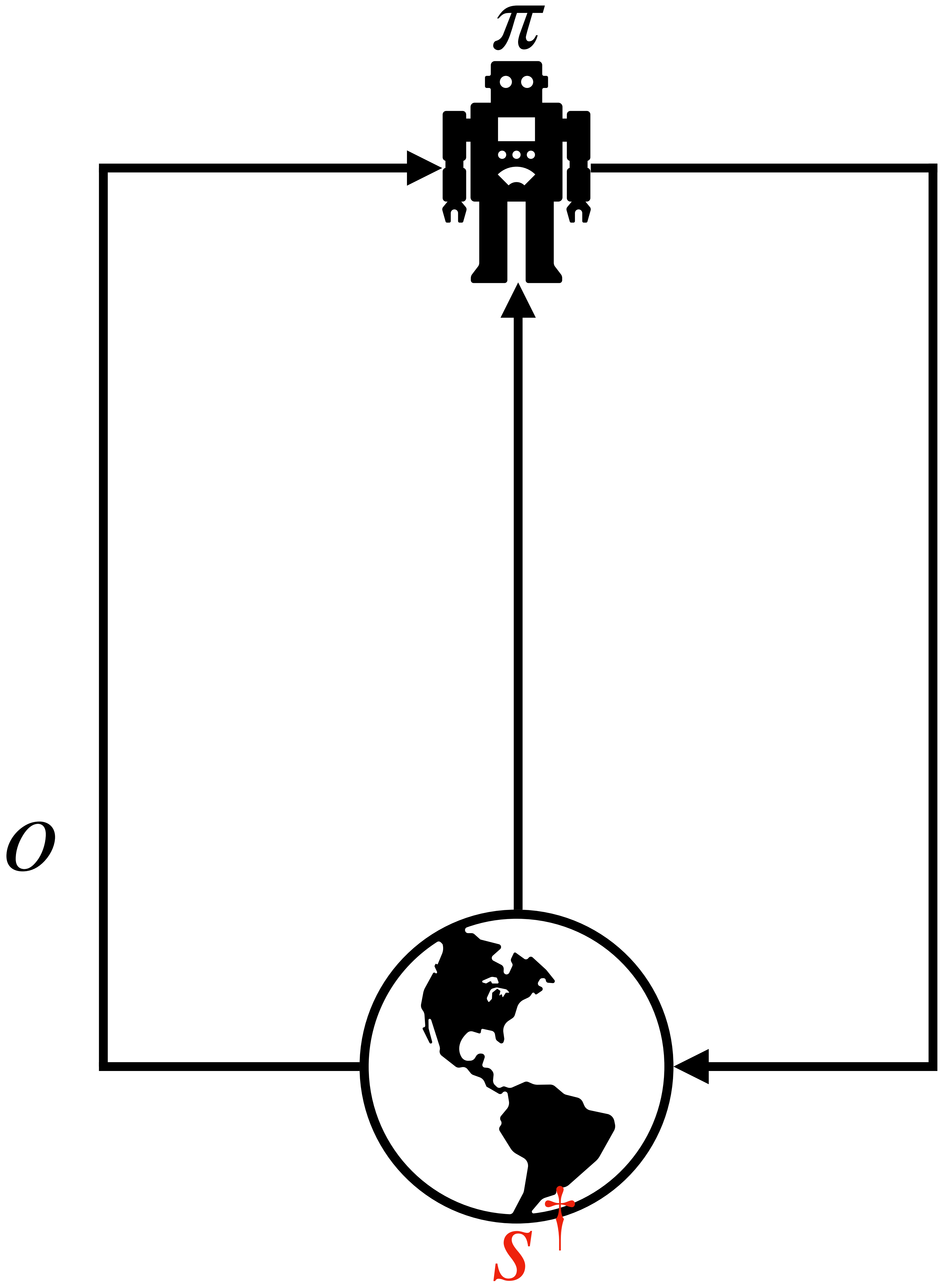
Attack Surfaces



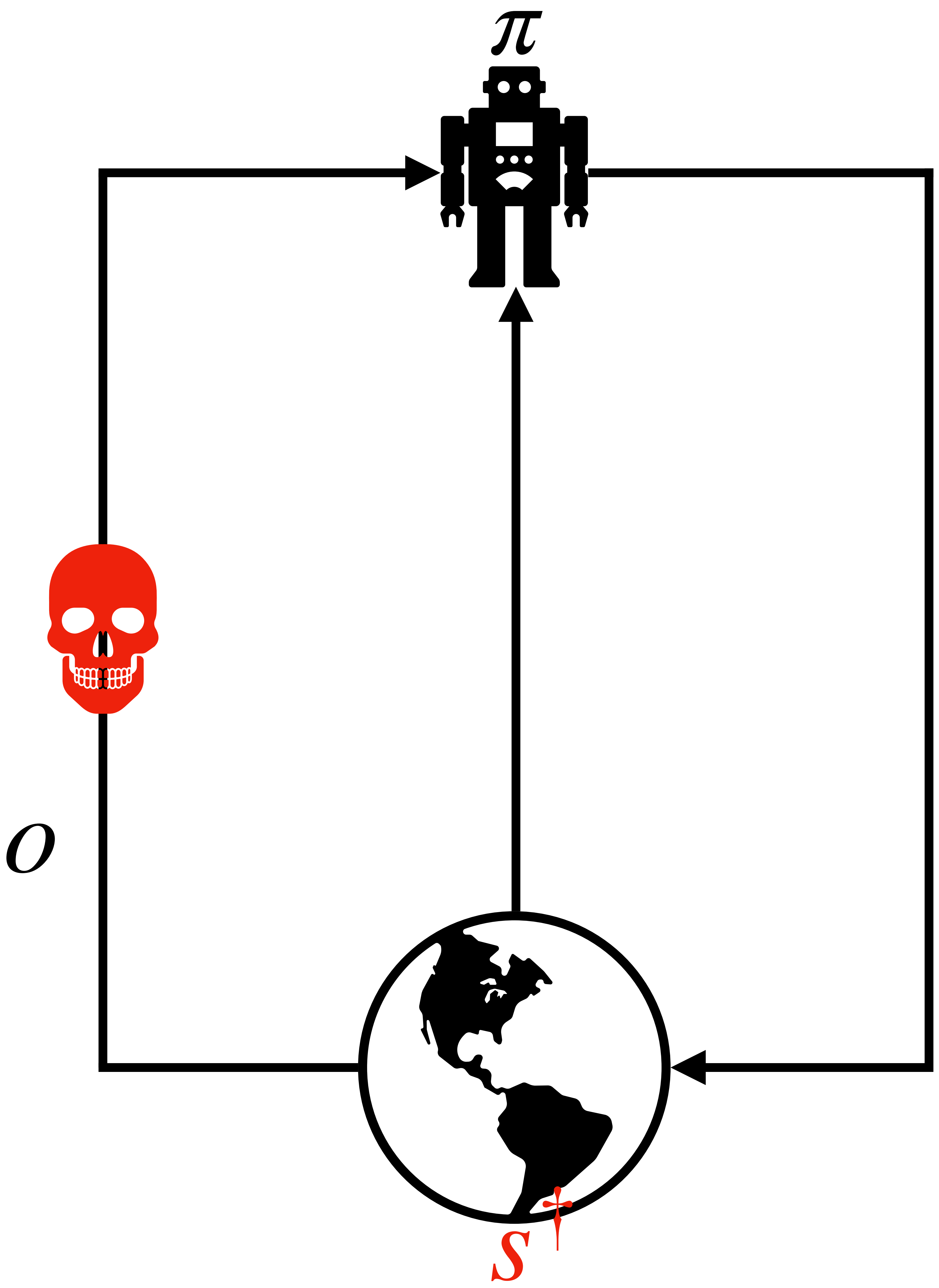
Attack Surfaces



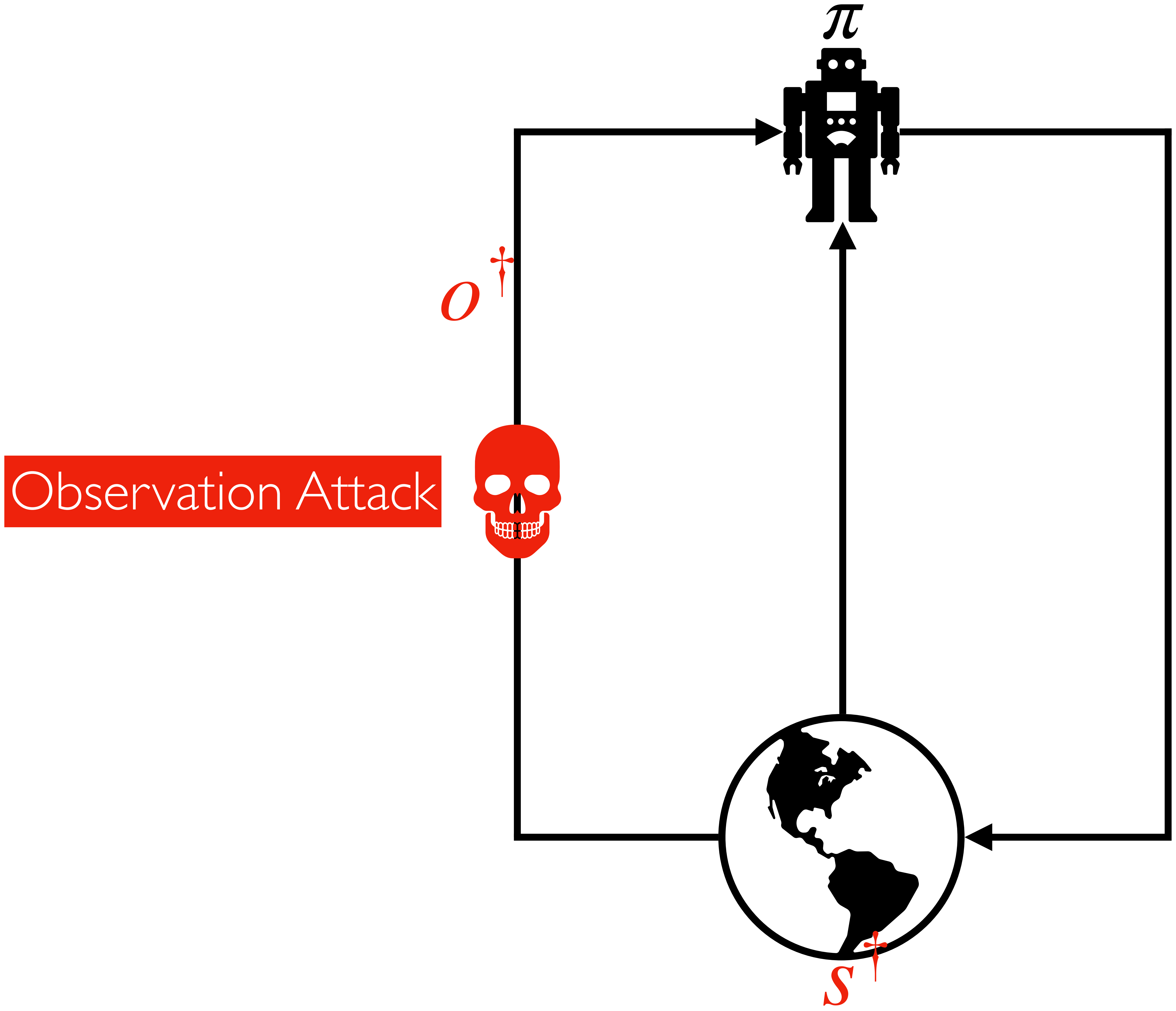
Attack Surfaces



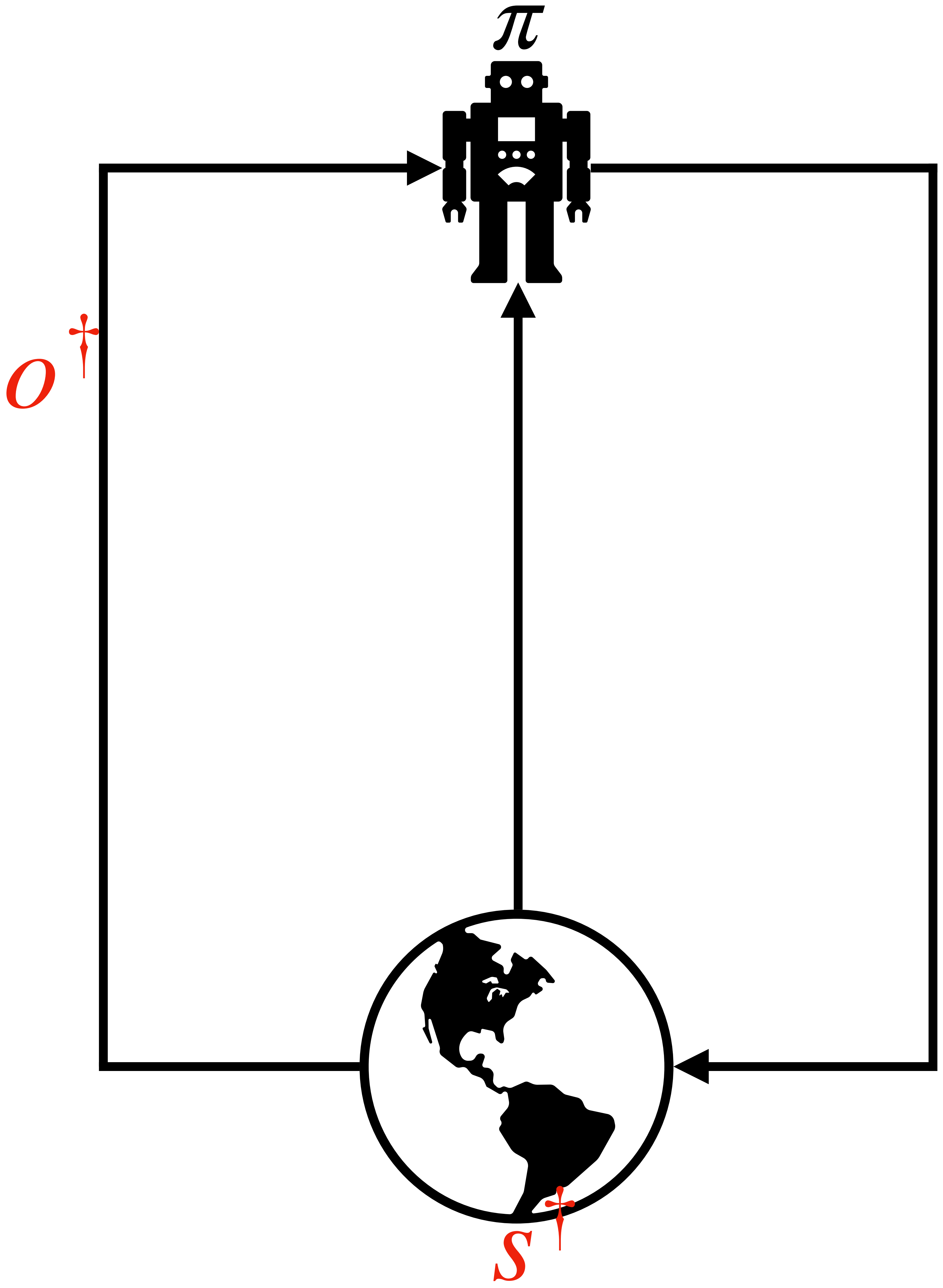
Attack Surfaces



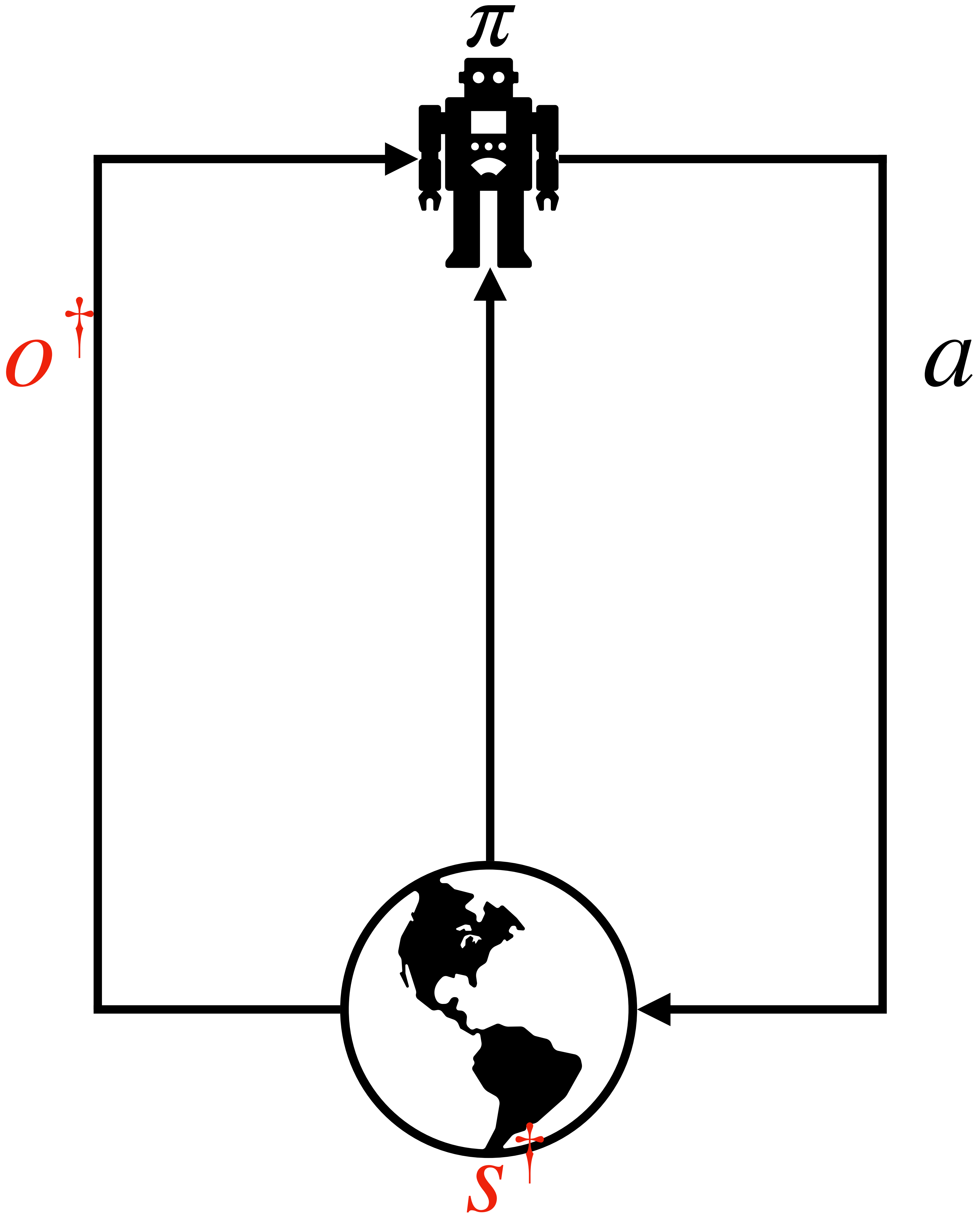
Attack Surfaces



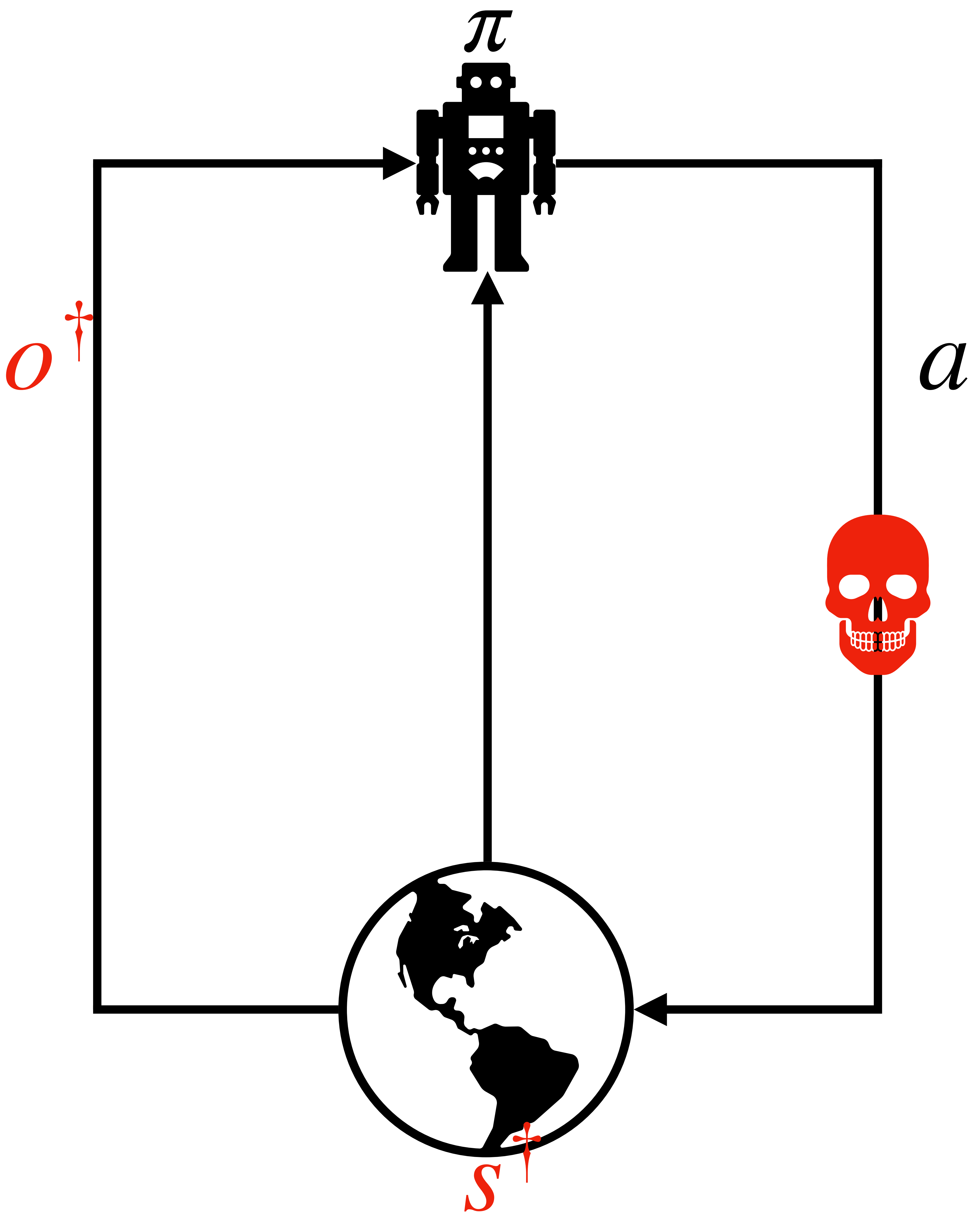
Attack Surfaces



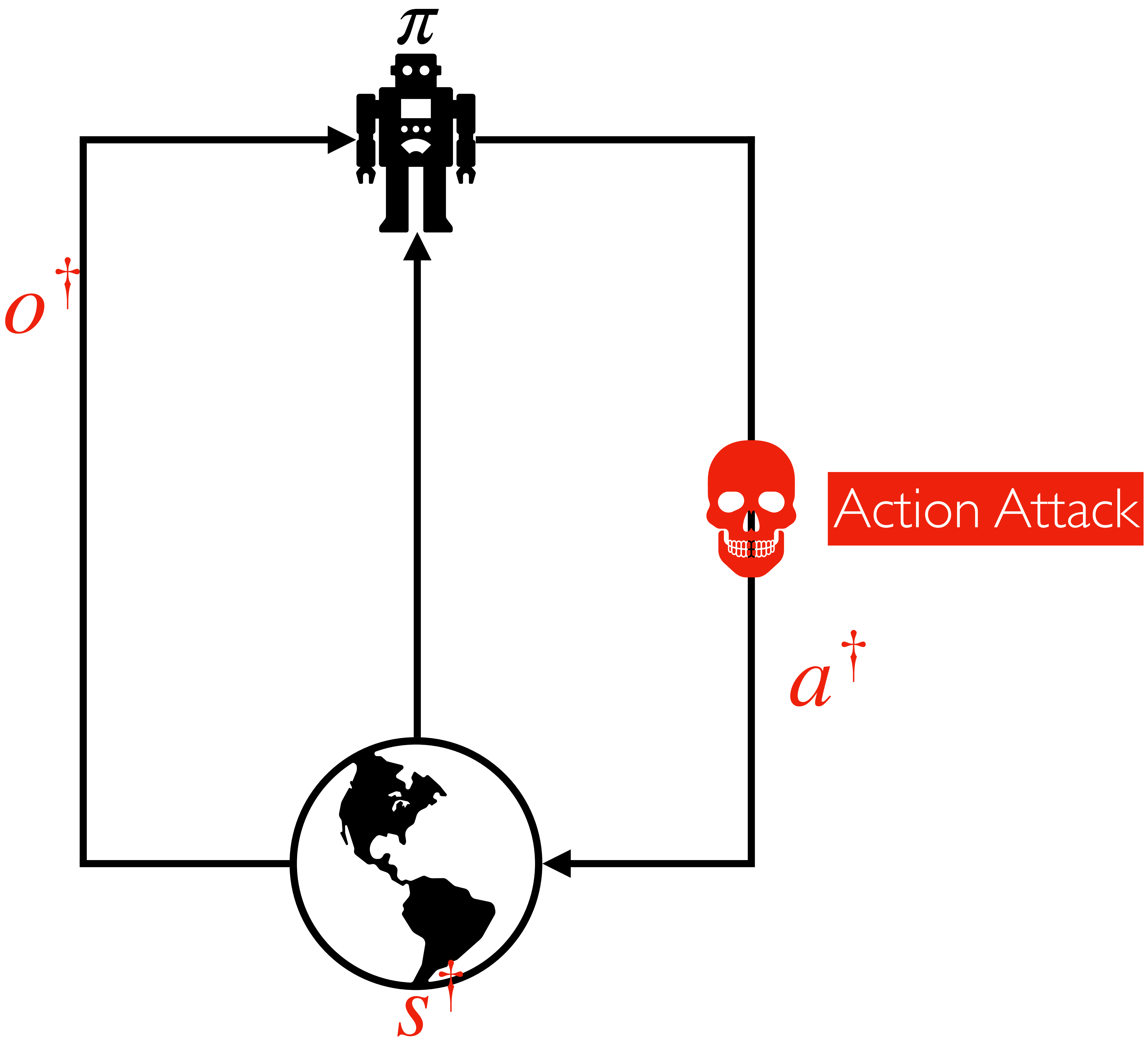
Attack Surfaces



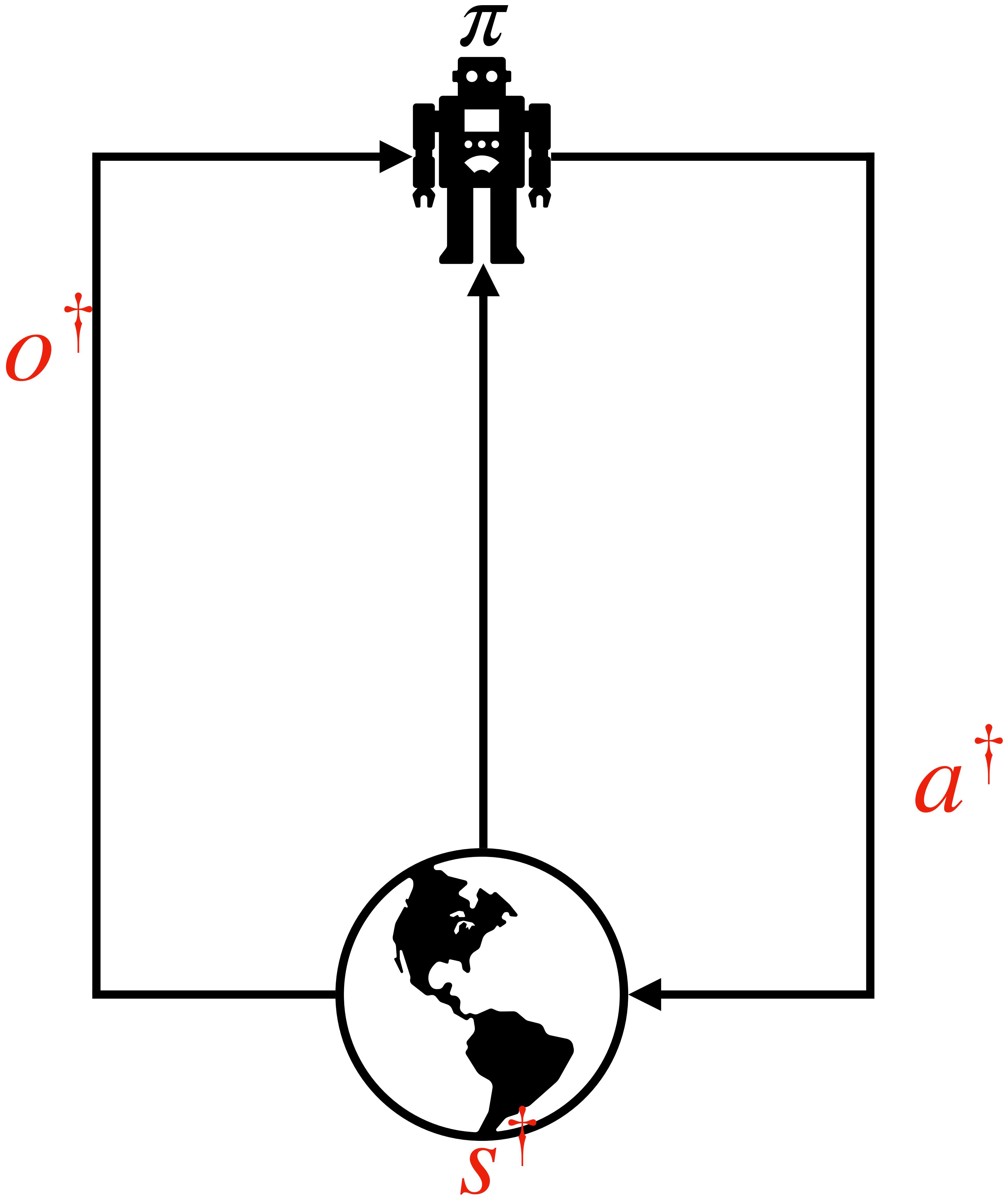
Attack Surfaces



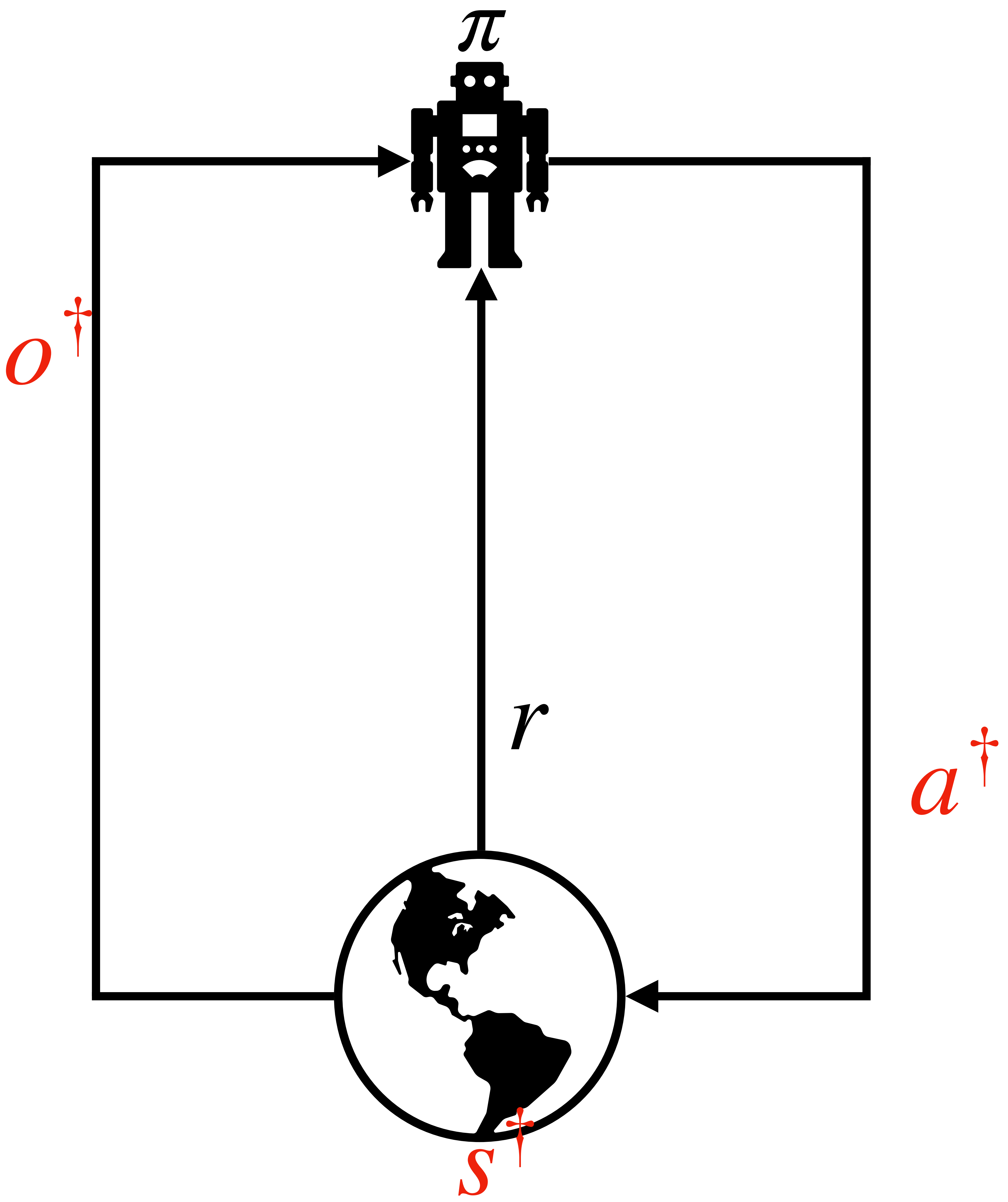
Attack Surfaces



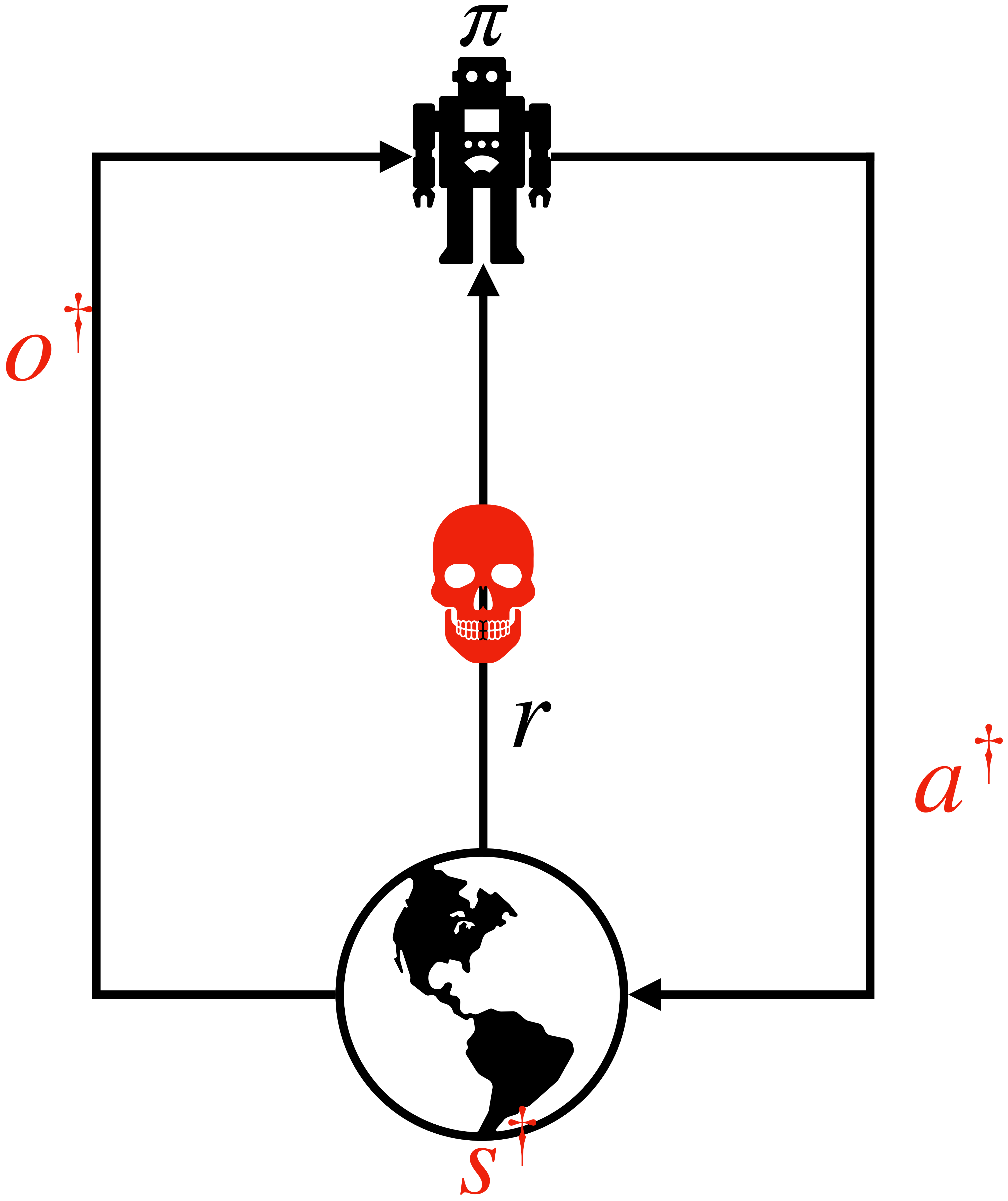
Attack Surfaces



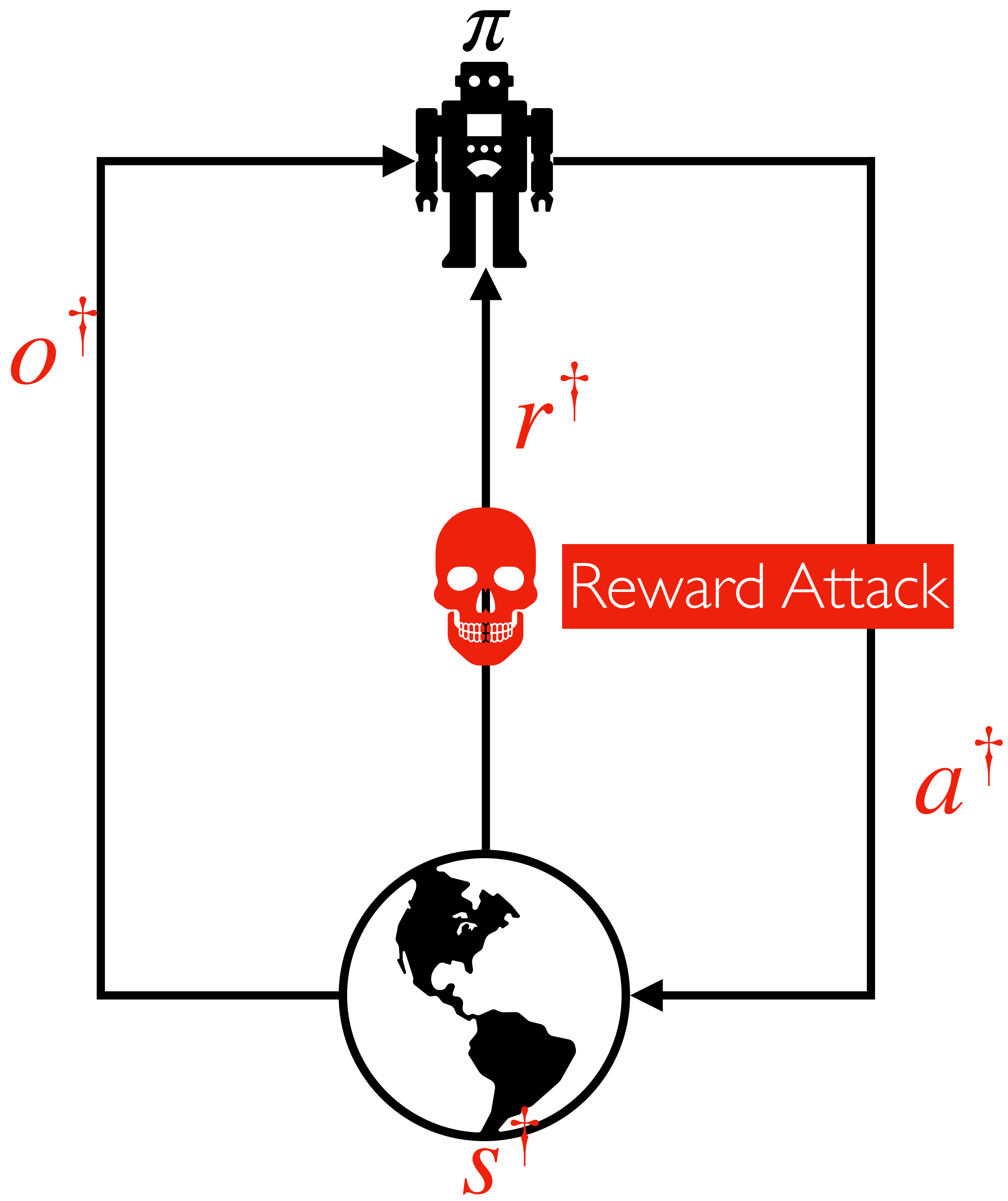
Attack Surfaces



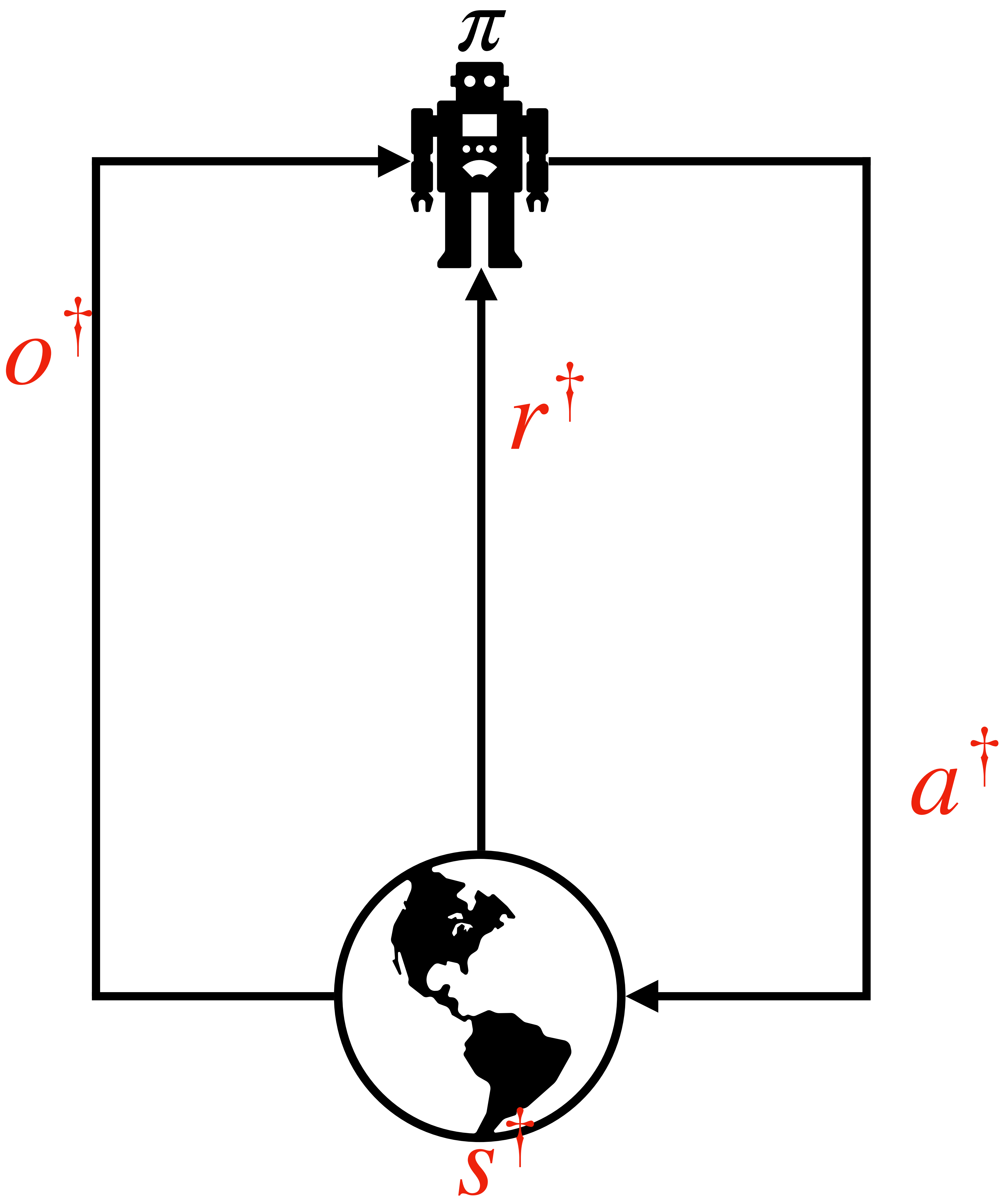
Attack Surfaces



Attack Surfaces



Attack Surfaces



Attack Surfaces

Attack Surfaces

- **State Attack:** changes the state of M from s_t to $s_t^\dagger$.

Attack Surfaces

- **State Attack:** changes the state of M from s_t to $s_t^\dagger$.
- **Observation Attack:** changes the agent's observation from o_t to $o_t^\dagger$.

Attack Surfaces

- **State Attack:** changes the state of M from s_t to $s_t^\dagger$.
- **Observation Attack:** changes the agent's observation from o_t to $o_t^\dagger$.
- **Action Attack:** changes the action M receives from a_t to $a_t^\dagger$.

Attack Surfaces

- **State Attack:** changes the state of M from s_t to $s_t^\dagger$.
- **Observation Attack:** changes the agent's observation from o_t to $o_t^\dagger$.
- **Action Attack:** changes the action M receives from a_t to $a_t^\dagger$.
- **Reward Attack:** changes the agent's reward from r_t to $r_t^\dagger$.

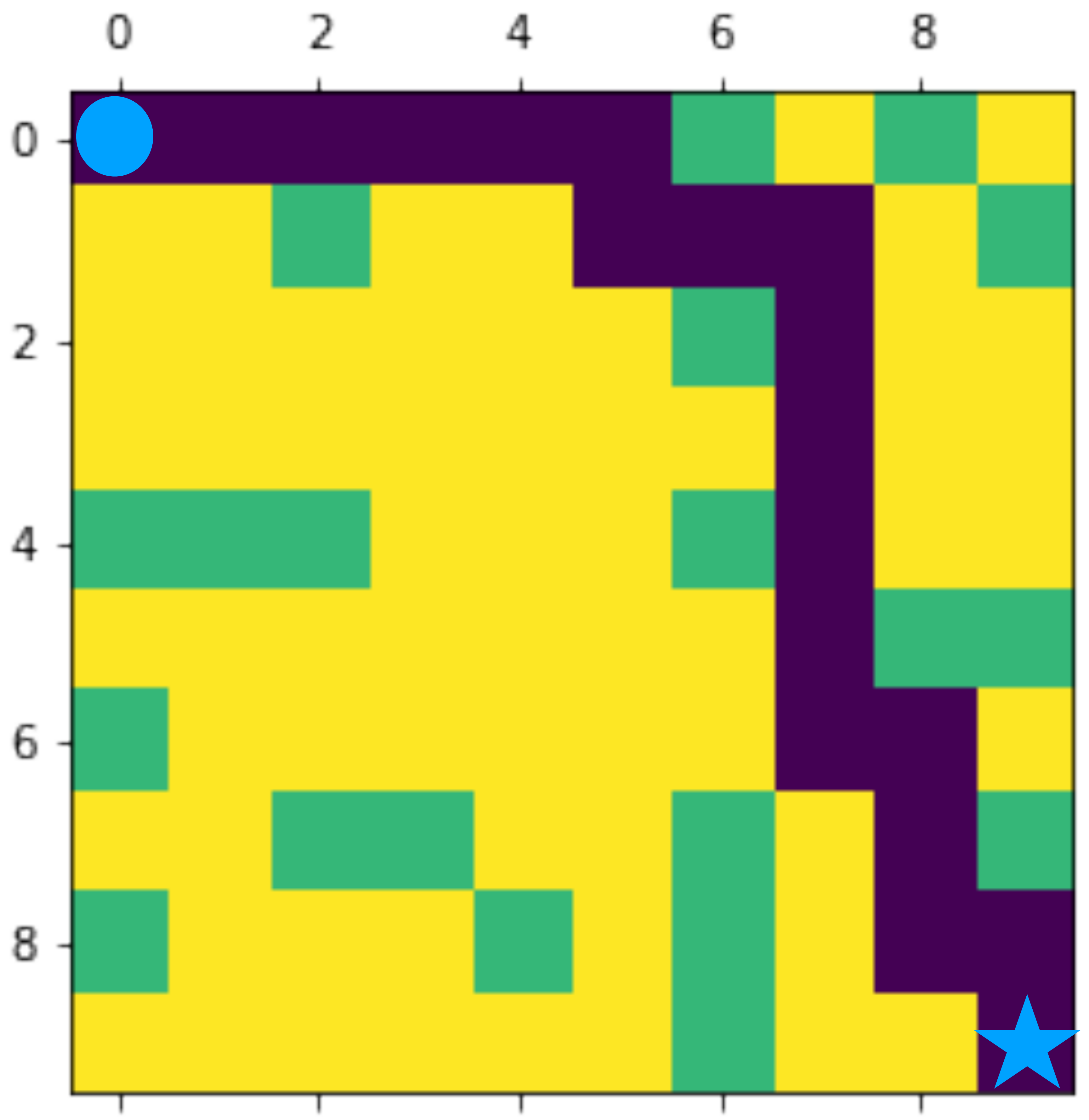
Attack Surfaces

- **State Attack:** changes the state of M from s_t to $s_t^\dagger$.
- **Observation Attack:** changes the agent's observation from o_t to $o_t^\dagger$.
- **Action Attack:** changes the action M receives from a_t to $a_t^\dagger$.
- **Reward Attack:** changes the agent's reward from r_t to $r_t^\dagger$.

The attacker can manipulate any element of the interaction tuple (s, a, r).

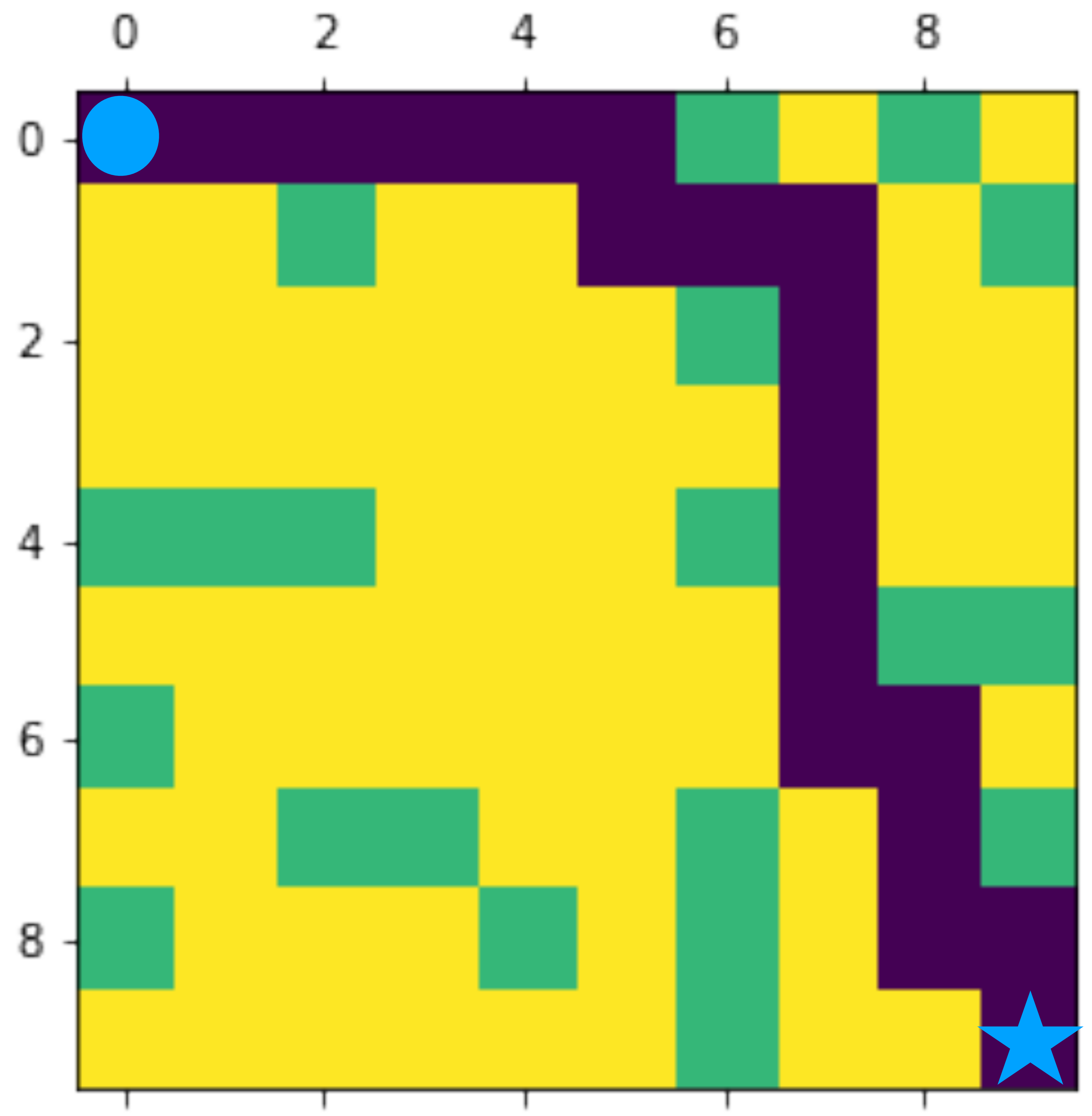
Maze Environment

Maze Environment



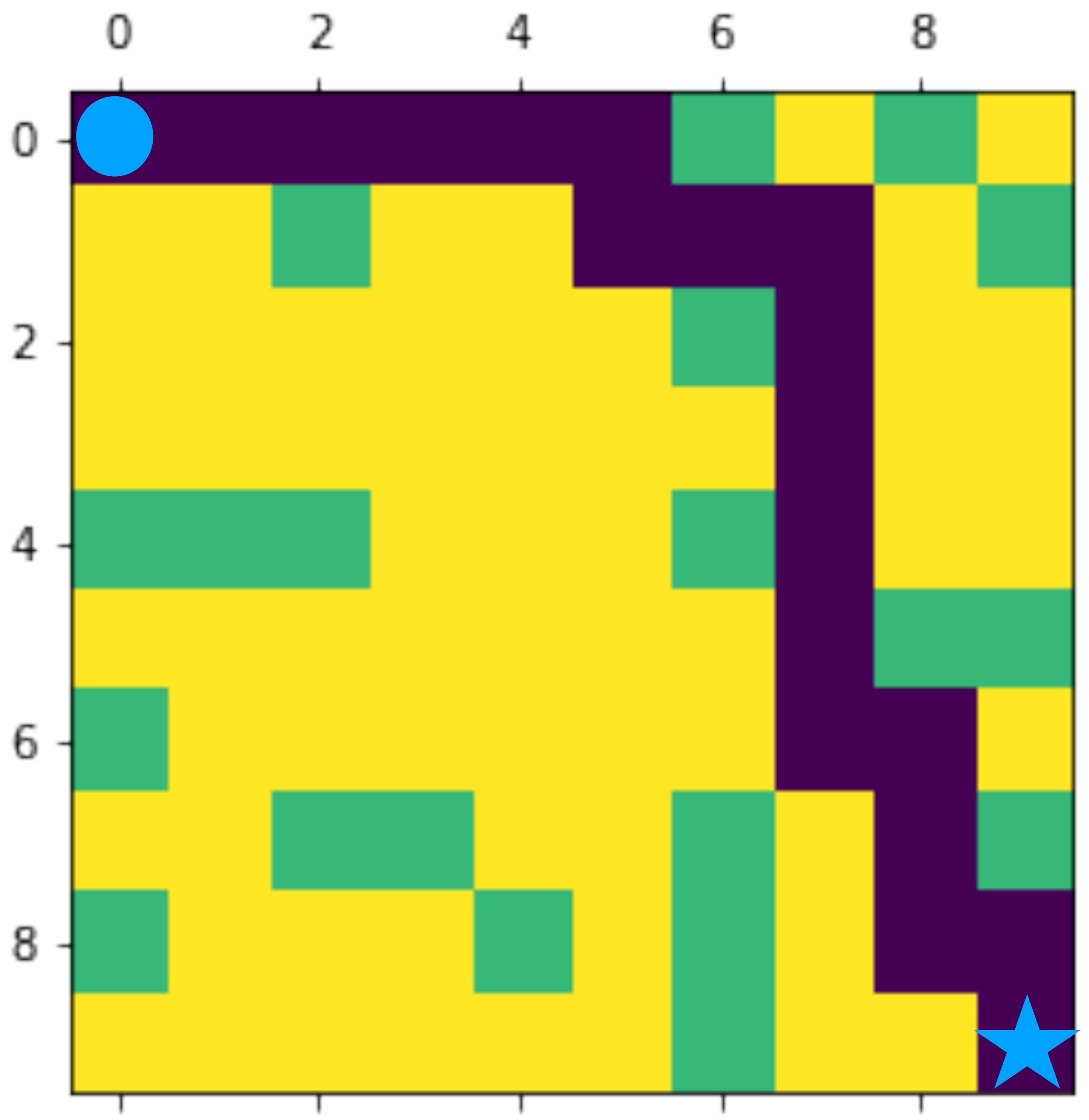
Maze Environment

- Green Squares are obstacles.



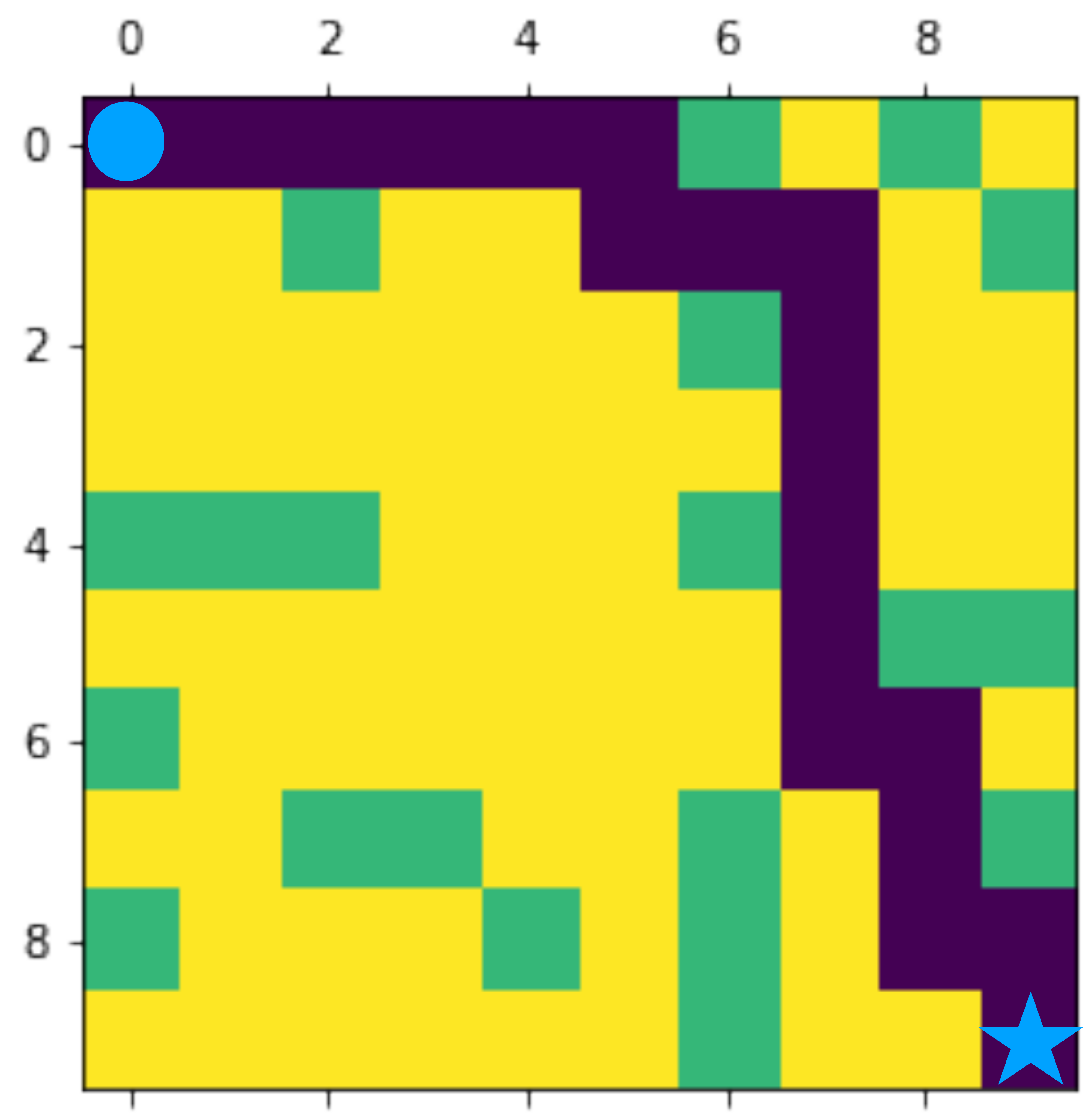
Maze Environment

- Green Squares are obstacles.
- Yellow Squares are traversable.



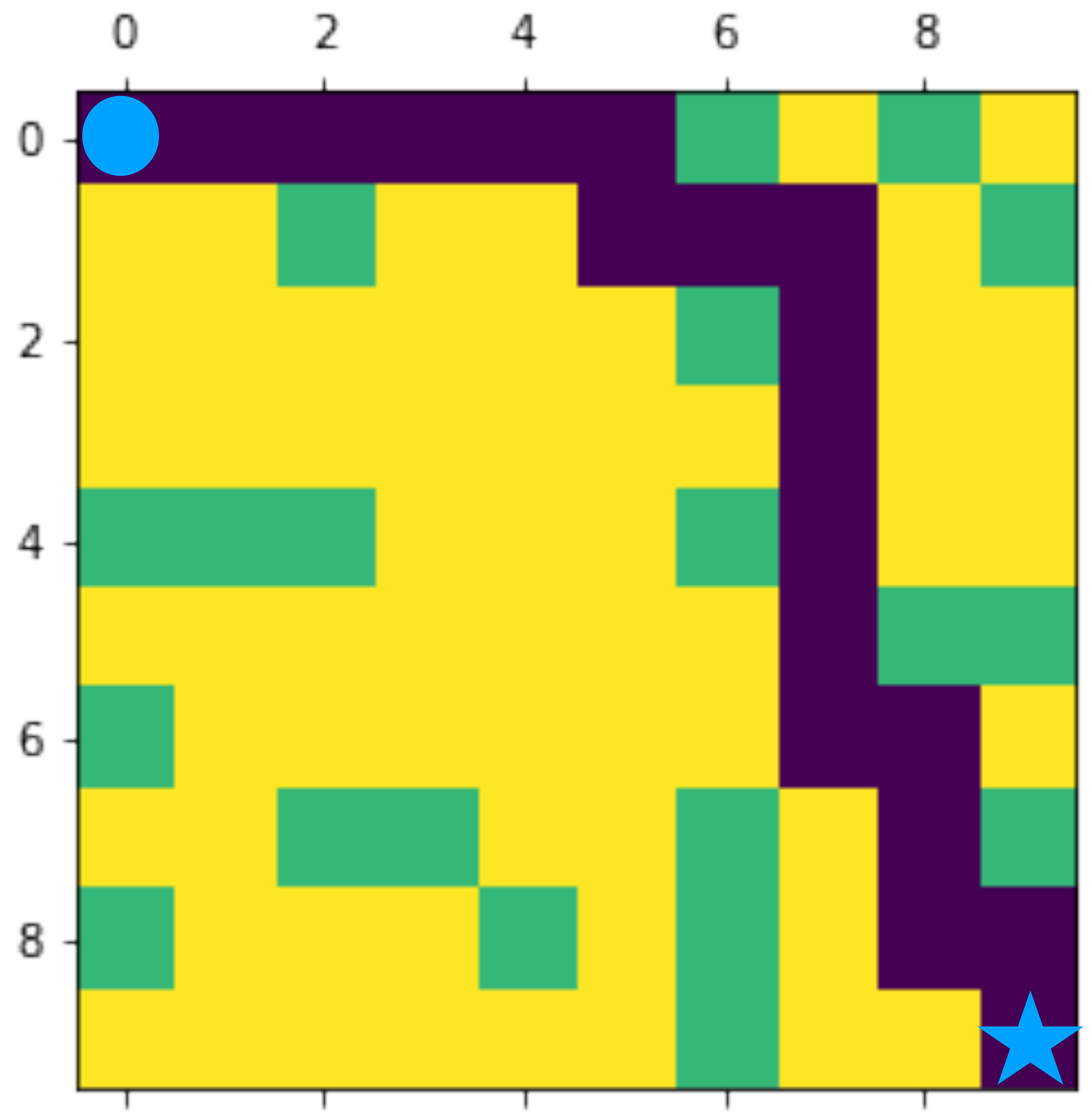
Maze Environment

- Green Squares are obstacles.
- Yellow Squares are traversable.
- The agent starts at top left corner.



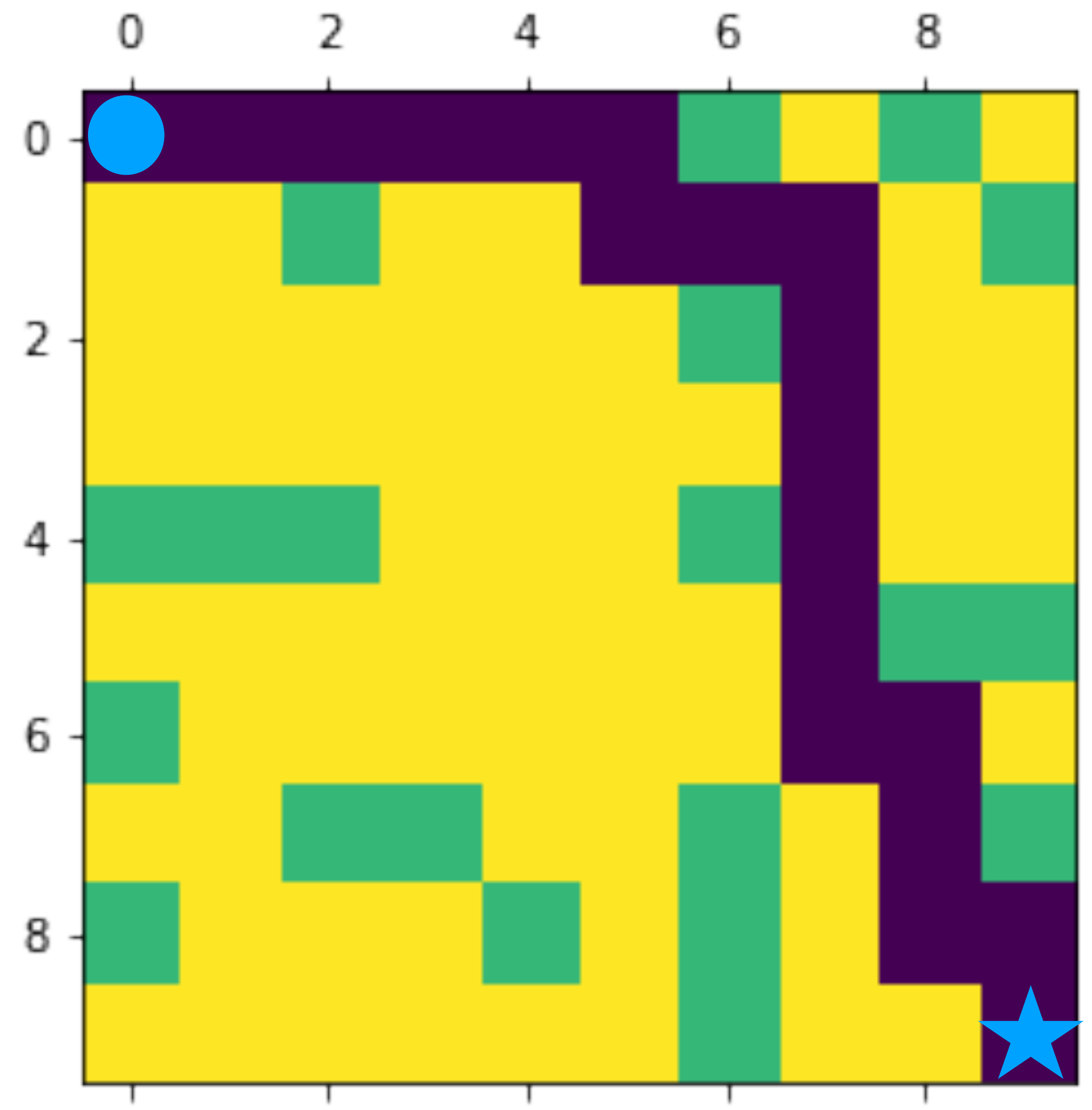
Maze Environment

- Green Squares are obstacles.
- Yellow Squares are traversable.
- The agent starts at top left corner.
- The agent receives reward only at the bottom right corner.



Maze Environment

- Green Squares are obstacles.
- Yellow Squares are traversable.
- The agent starts at top left corner.
- The agent receives reward only at the bottom right corner.
- An optimal (shortest path) policy for the agent is in purple.



Perceived-State Attack

Perceived-State Attack

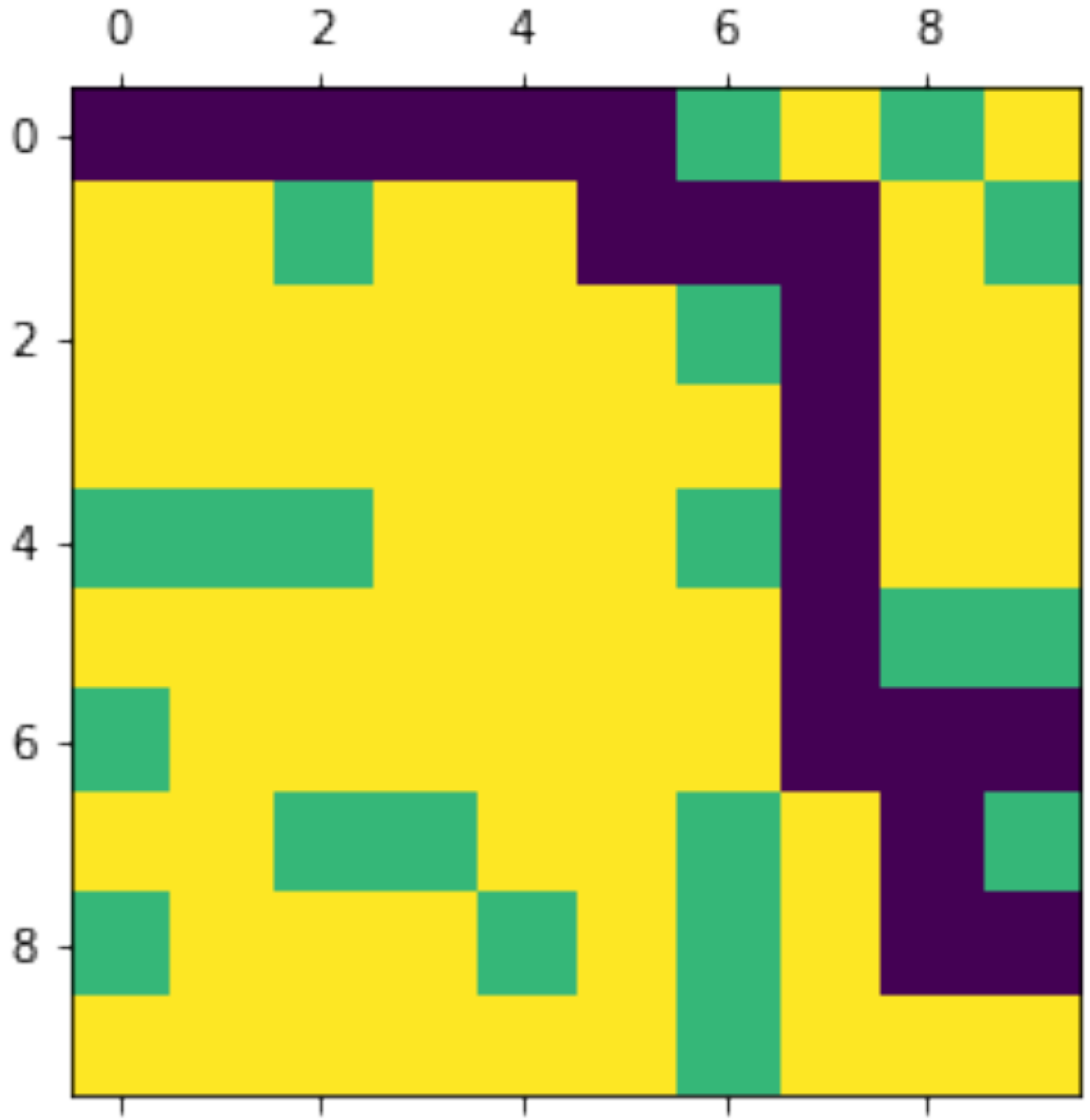
- Attacker shows agent $s^\dagger$.

Perceived-State Attack

- Attacker shows agent $s^\dagger$.
- Agent chooses action $\pi(s^\dagger)$ instead of $\pi(s)$

Perceived-State Attack

- Attacker shows agent $s^\dagger$.
- Agent chooses action $\pi(s^\dagger)$ instead of $\pi(s)$



Perceived-State Attack

Perceived-State Attack

Perceived-State Attack

Check out Shubham's full paper in Neurips22!
Provable Defense against Backdoor Policies in Reinforcement Learning

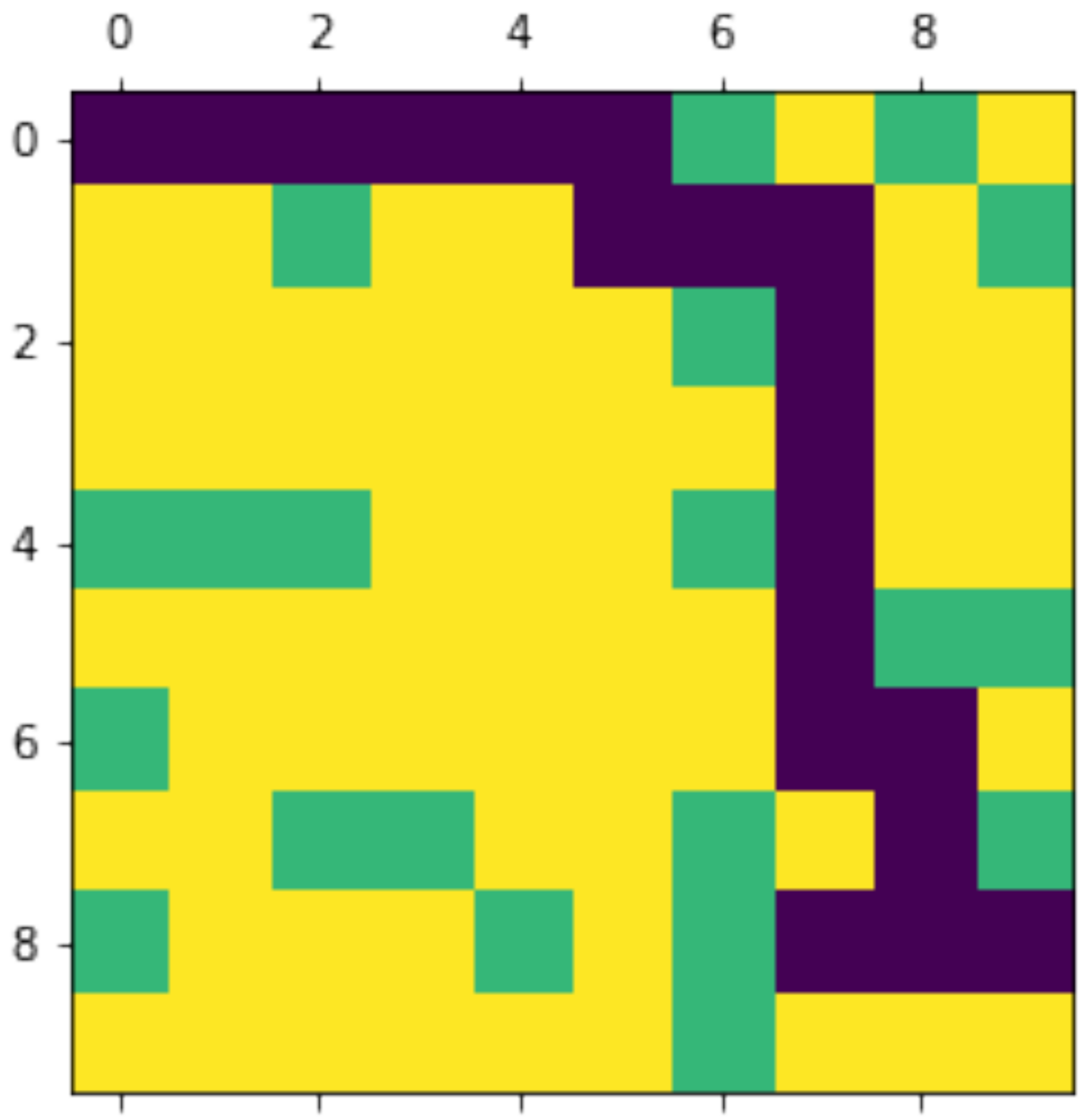
Action Attack

Action Attack

Attacker intercepts $a = \pi(s)$ and sends $a^\dagger$ to the environment instead.

Action Attack

Attacker intercepts $a = \pi(s)$ and sends $a^\dagger$ to the environment instead.



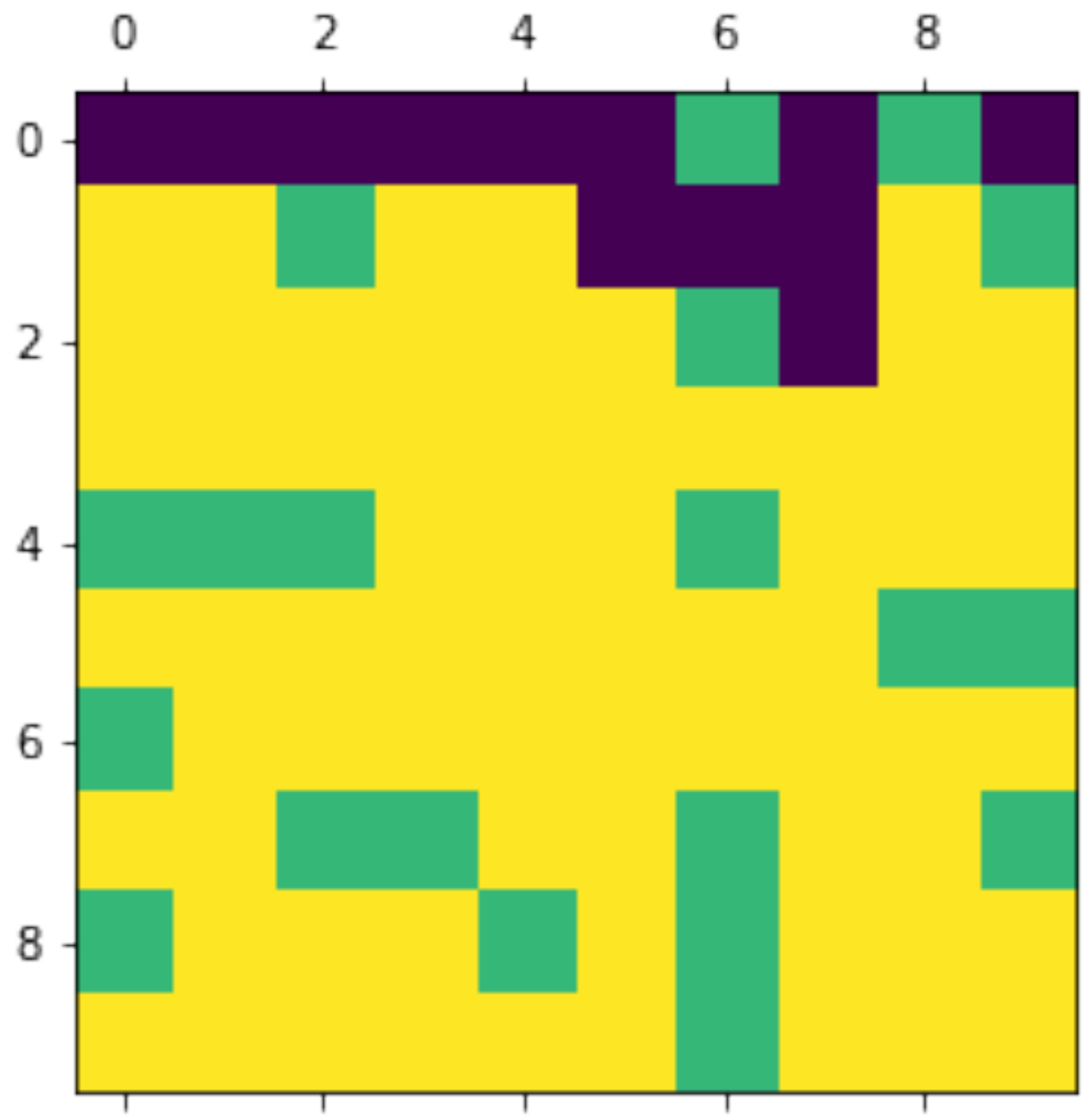
True-State Attack

True-State Attack

Attacker changes the environment's state to $s^\dagger$

True-State Attack

Attacker changes the environment's state to $s^\dagger$



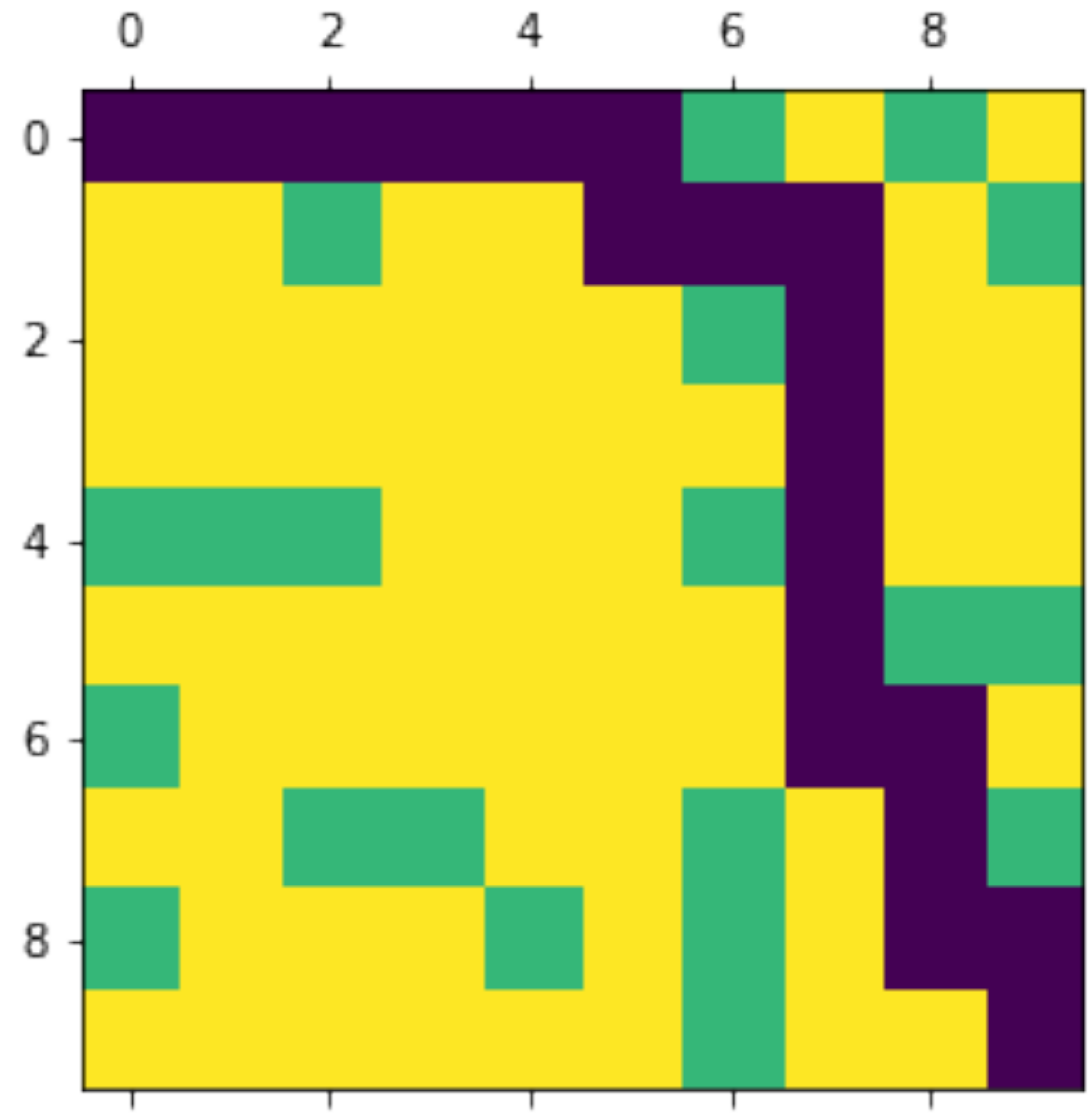
Reward Attack

Reward Attack

Attacker changes the reward the agent receives to $r^\dagger$

Reward Attack

Attacker changes the reward the agent receives to $r^\dagger$



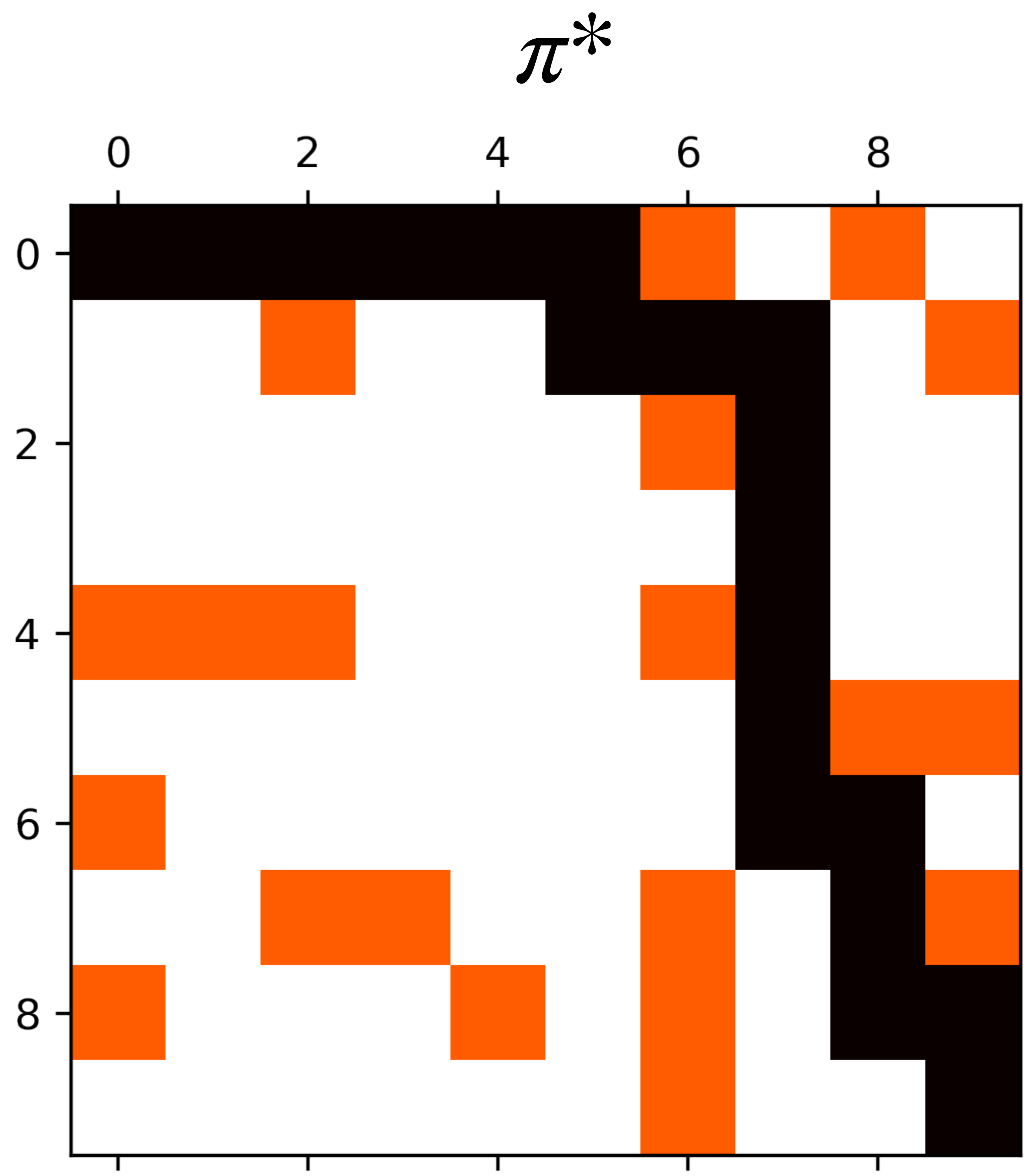
Motivation: Robust Policies

Motivation: Robust Policies

Optimal policies may be sensitive to noise or attacks.

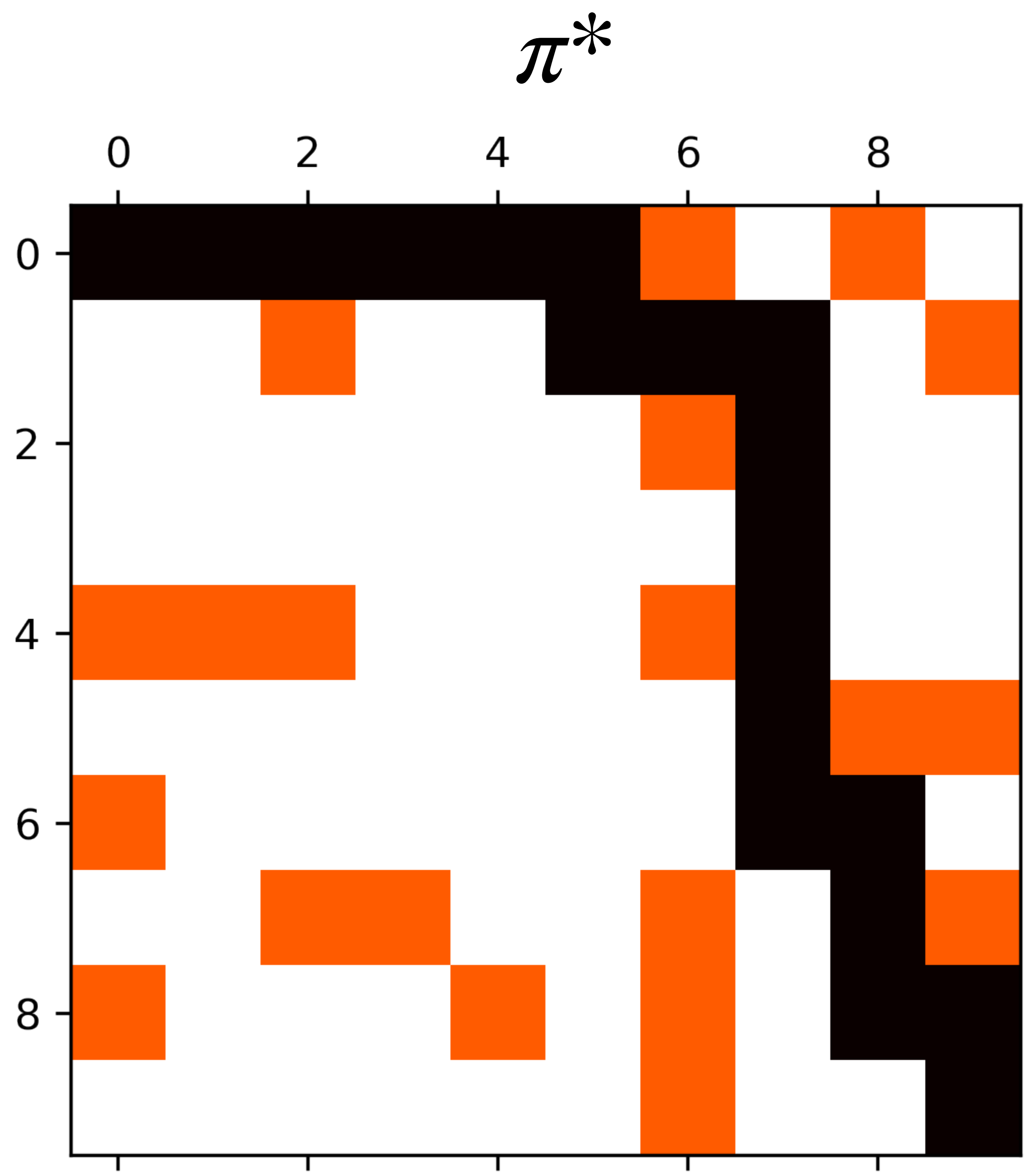
Motivation: Robust Policies

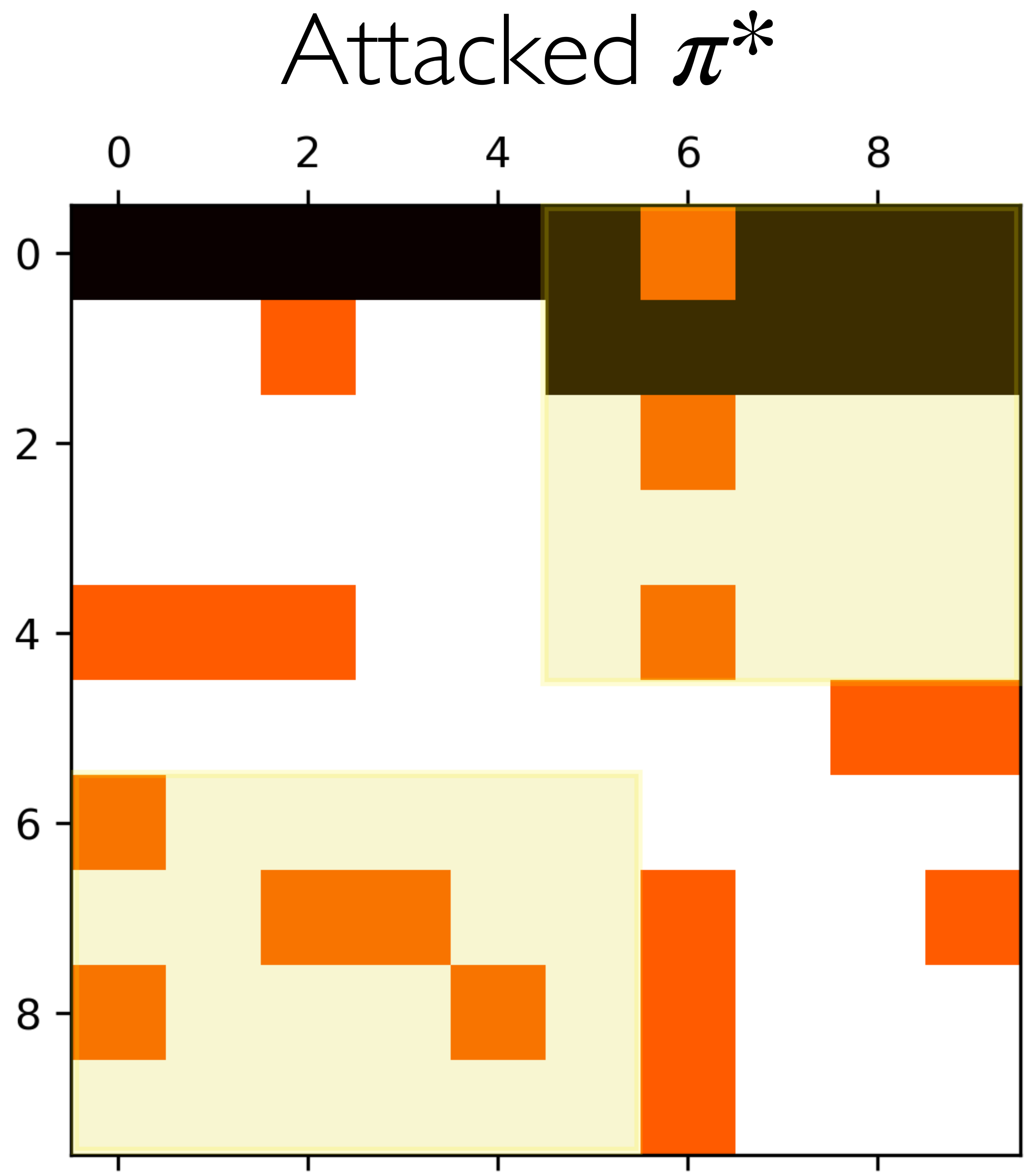
Optimal policies may be sensitive to noise or attacks.



Motivation: Robust Policies

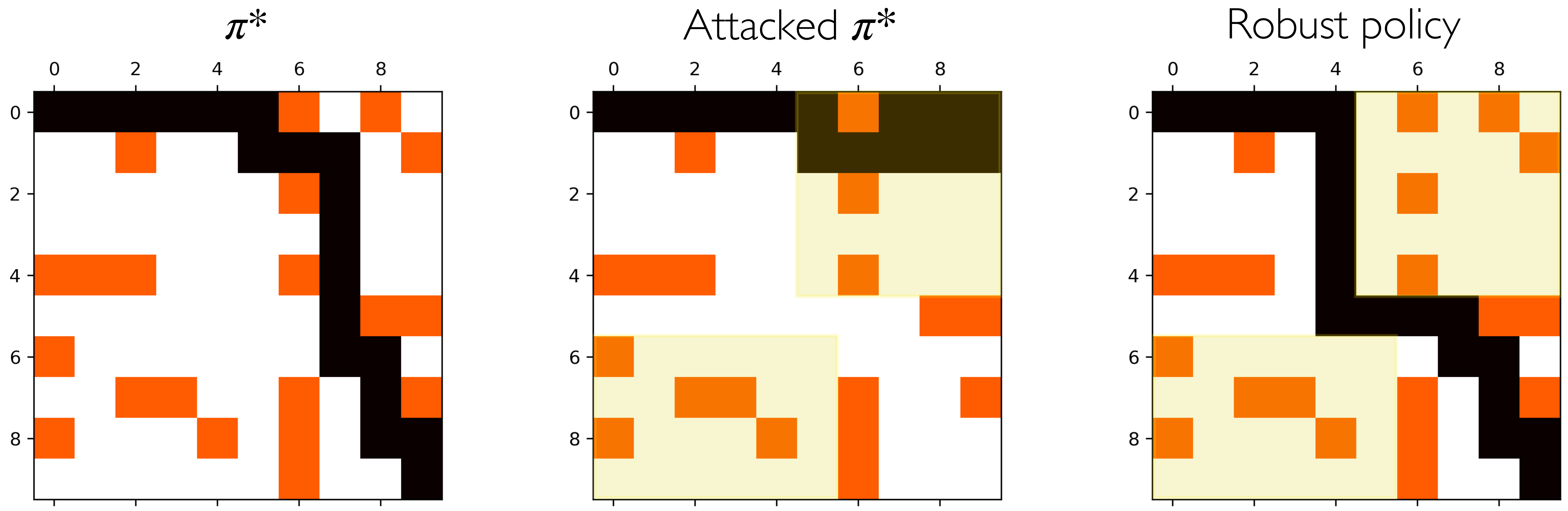
Optimal policies may be sensitive to noise or attacks.





Motivation: Robust Policies

Optimal policies may be sensitive to noise or attacks.



What's known?

What's known?

Optimal Observation Attacks

Robust Deep Reinforcement Learning against Adversarial Perturbations on State Observations

Huan Zhang[*,1] **Hongge Chen**[*,2] **Chaowei Xiao**[3]
Bo Li[4] **Mingyan Liu**[5] **Duane Boning**[2] **Cho-Jui Hsieh**[1]
[1]UCLA [2] MIT [3]NVIDIA [4]UIUC [5]University of Michigan
huan@huan-zhang.com, chenhg@mit.edu, chaoweix@nvidia.com,
lbo@illinois.edu,mingyan@umich.edu,boning@mtl.mit.edu,chohsieh@cs.ucla.edu

What's known?

Optimal Observation Attacks

Robust Deep Reinforcement Learning against Adversarial Perturbations on State Observations

Huan Zhang[*,1] **Hongge Chen**[*,2] **Chaowei Xiao**[3]
Bo Li[4] **Mingyan Liu**[5] **Duane Boning**[2] **Cho-Jui Hsieh**[1]
[1]UCLA [2] MIT [3]NVIDIA [4]UIUC [5]University of Michigan
huan@huan-zhang.com, chenhg@mit.edu, chaoweix@nvidia.com,
lbo@illinois.edu,mingyan@umich.edu,boning@mtl.mit.edu,chohsieh@cs.ucla.edu

[Training-time] Action and Reward Attacks

Understanding the Limits of Poisoning Attacks in Episodic Reinforcement Learning

Anshuka Rangi[1] , **Haifeng Xu**[2] , **Long Tran-Thanh**[3] , **Massimo Franceschetti**[1]
[1] University of California San Diego, USA
[2]University of Virginia, USA
[3]University of Warwick, UK
{arangi, mfranceschetti}@ucsd.edu, hx4ad@virginia.edu, long.tran-thanh@warwick.ac.uk

What's known?

Optimal Observation Attacks

Robust Deep Reinforcement Learning against Adversarial Perturbations on State Observations

Huan Zhang[*,1] **Hongge Chen**[*,2] **Chaowei Xiao**[3]
Bo Li[4] **Mingyan Liu**[5] **Duane Boning**[2] **Cho-Jui Hsieh**[1]
[1]UCLA [2] MIT [3]NVIDIA [4]UIUC [5]University of Michigan
huan@huan-zhang.com, chenhg@mit.edu, chaoweix@nvidia.com,
lbo@illinois.edu,mingyan@umich.edu,boning@mtl.mit.edu,chohsieh@cs.ucla.edu

[Training-time] Action and Reward Attacks

Understanding the Limits of Poisoning Attacks in Episodic Reinforcement Learning

Anshuka Rangi[1] , **Haifeng Xu**[2] , **Long Tran-Thanh**[3] , **Massimo Franceschetti**[1]
[1] University of California San Diego, USA
[2]University of Virginia, USA
[3]University of Warwick, UK
{arangi, mfranceschetti}@ucsd.edu, hx4ad@virginia.edu, long.tran-thanh@warwick.ac.uk

Defense against a specific reward attack algorithm

Defense Against Reward Poisoning Attacks in Reinforcement Learning

Kiarash Banihashem MPI-SWS kbanihas@mpi-sws.org
Adish Singla MPI-SWS adishs@mpi-sws.org
Goran Radanovic MPI-SWS gradanovic@mpi-sws.org

What's known?

Optimal Observation Attacks

Robust Deep Reinforcement Learning against Adversarial Perturbations on State Observations

Huan Zhang[*,1] Hongge Chen[*,2] Chaowei Xiao[3]
Bo Li[4] Mingyan Liu[5] Duane Boning[2] Cho-Jui Hsieh[1]
[1]UCLA [2] MIT [3]NVIDIA [4]UIUC [5]University of Michigan
huan@huan-zhang.com, chenhg@mit.edu, chaoweix@nvidia.com,
lbo@illinois.edu,mingyan@umich.edu,boning@mtl.mit.edu,chohsieh@cs.ucla.edu

[Training-time] Action and Reward Attacks

Understanding the Limits of Poisoning Attacks in Episodic Reinforcement Learning

Anshuka Rangi[1] , Haifeng Xu[2] , Long Tran-Thanh[3] , Massimo Franceschetti[1]
[1] University of California San Diego, USA
[2]University of Virginia, USA
[3]University of Warwick, UK
{arangi, mfranceschetti}@ucsd.edu, hx4ad@virginia.edu, long.tran-thanh@warwick.ac.uk

Defense against a specific reward attack algorithm

Defense Against Reward Poisoning Attacks in Reinforcement Learning

Kiarash Banihashem, MPI-SWS, kbanihas@mpi-sws.org
Adish Singla, MPI-SWS, adishs@mpi-sws.org
Goran Radanovic, MPI-SWS, gradanovic@mpi-sws.org

Not robust! Attacker can change its algorithm later.

The Attack Problem

The Attack Problem

Attacker has its own reward $g(s_t, a_t, r_t)$ that depends on the victim's.

The Attack Problem

Attacker has its own reward $g(s_t, a_t, r_t)$ that depends on the victim's.

Definition 1 (Attack Problem)**.** For any π, the attacker's seeks a policy $\nu^* \in N$ that maximizes its expected reward from the victim-attacker-M interaction:

$$\nu^* \in \arg\max_{\nu \in N} \mathbb{E}_M^{\pi,\nu} \left[\sum_{t=0}^{\infty} \gamma^t g(s_t, a_t, r_t) \right].$$

Adversarial Decomposition

Adversarial Decomposition

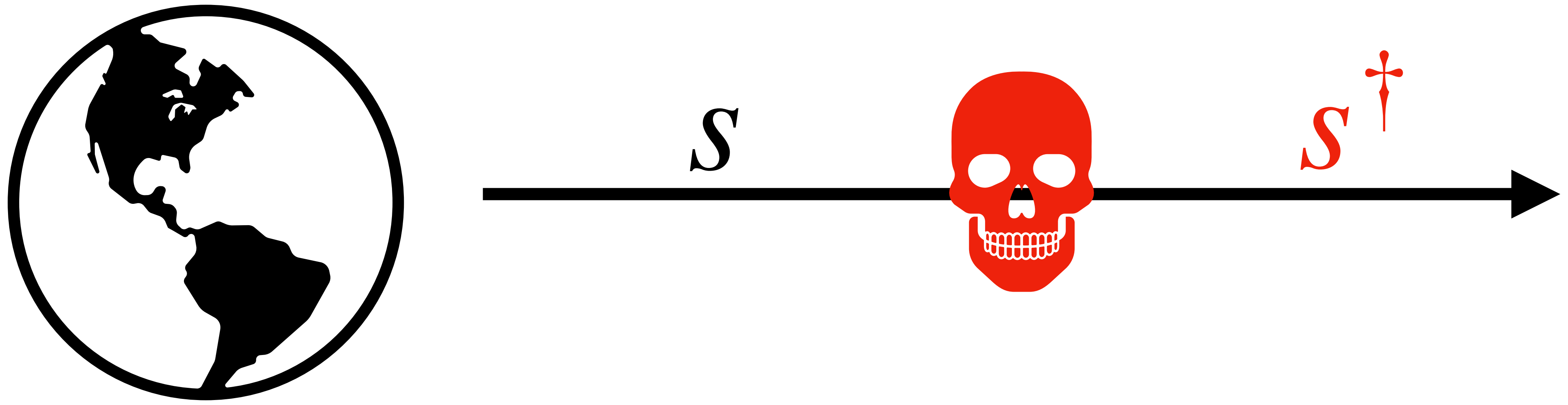
We decompose the attacked $\boldsymbol{\pi}$-M interaction based on the *flow of information*.

Adversarial Decomposition

We decompose the attacked $\boldsymbol{\pi}$-M interaction based on the *flow of information*.

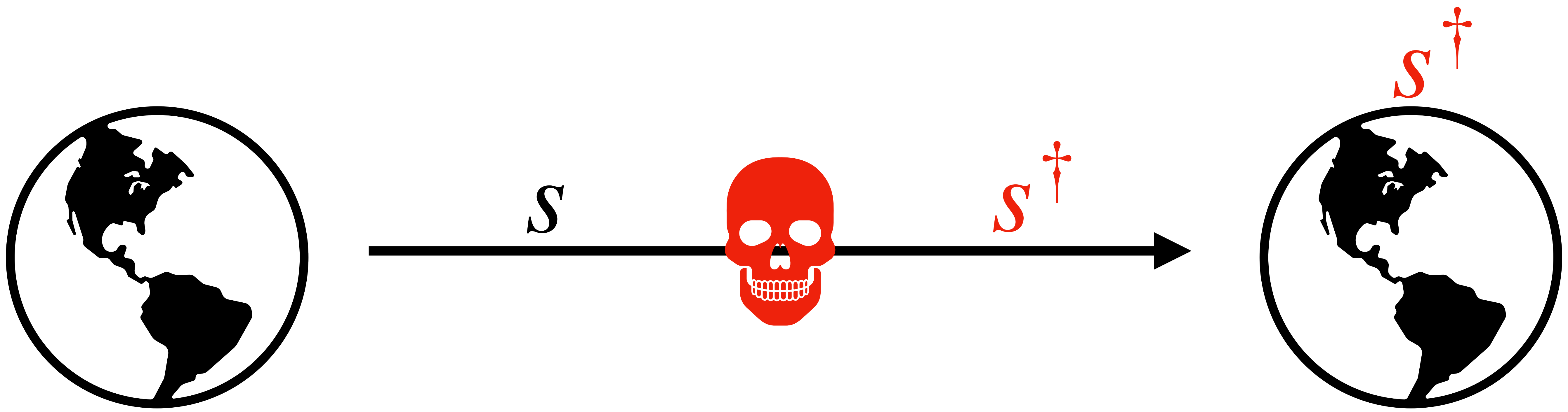
Adversarial Decomposition

We decompose the attacked $\boldsymbol{\pi}$-M interaction based on the *flow of information*.



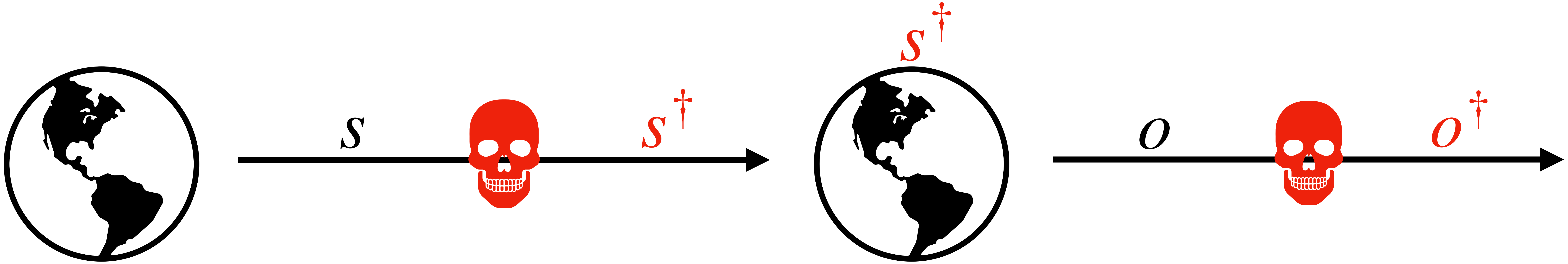
Adversarial Decomposition

We decompose the attacked $\boldsymbol{\pi}$-M interaction based on the *flow of information*.



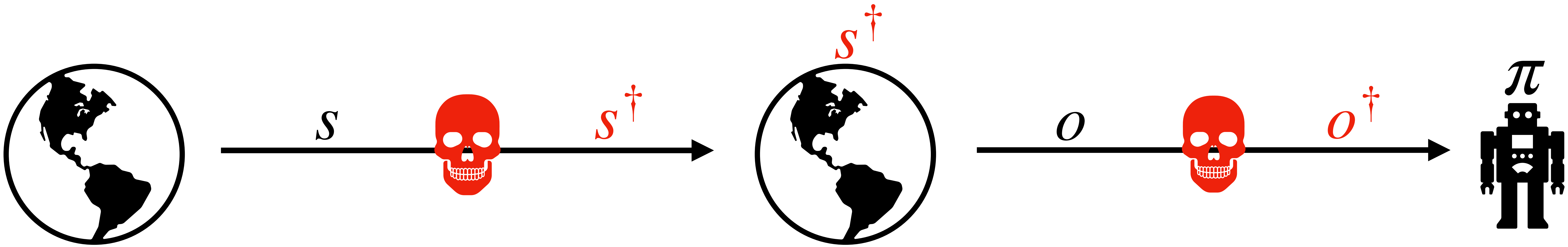
Adversarial Decomposition

We decompose the attacked $\boldsymbol{\pi}$-M interaction based on the *flow of information*.



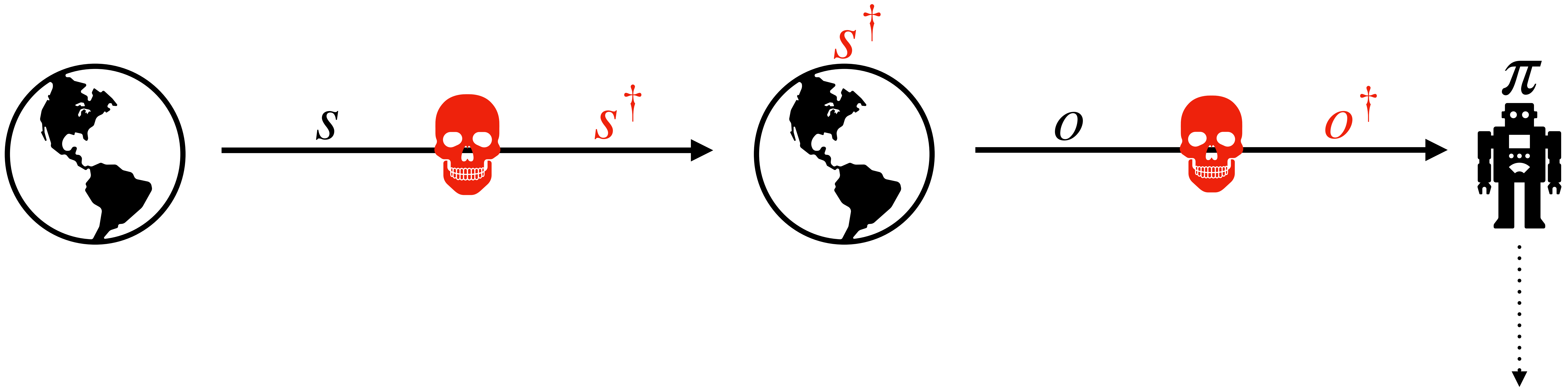
Adversarial Decomposition

We decompose the attacked π-M interaction based on the *flow of information*.



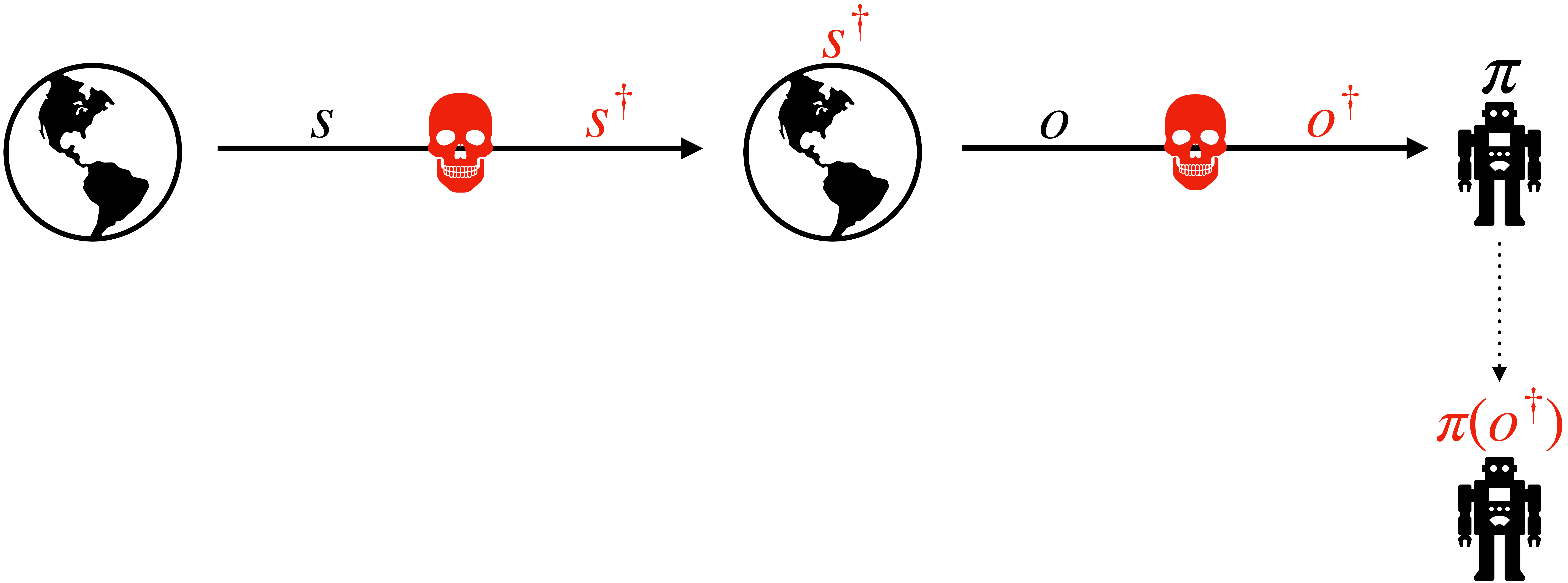
Adversarial Decomposition

We decompose the attacked π-M interaction based on the *flow of information*.



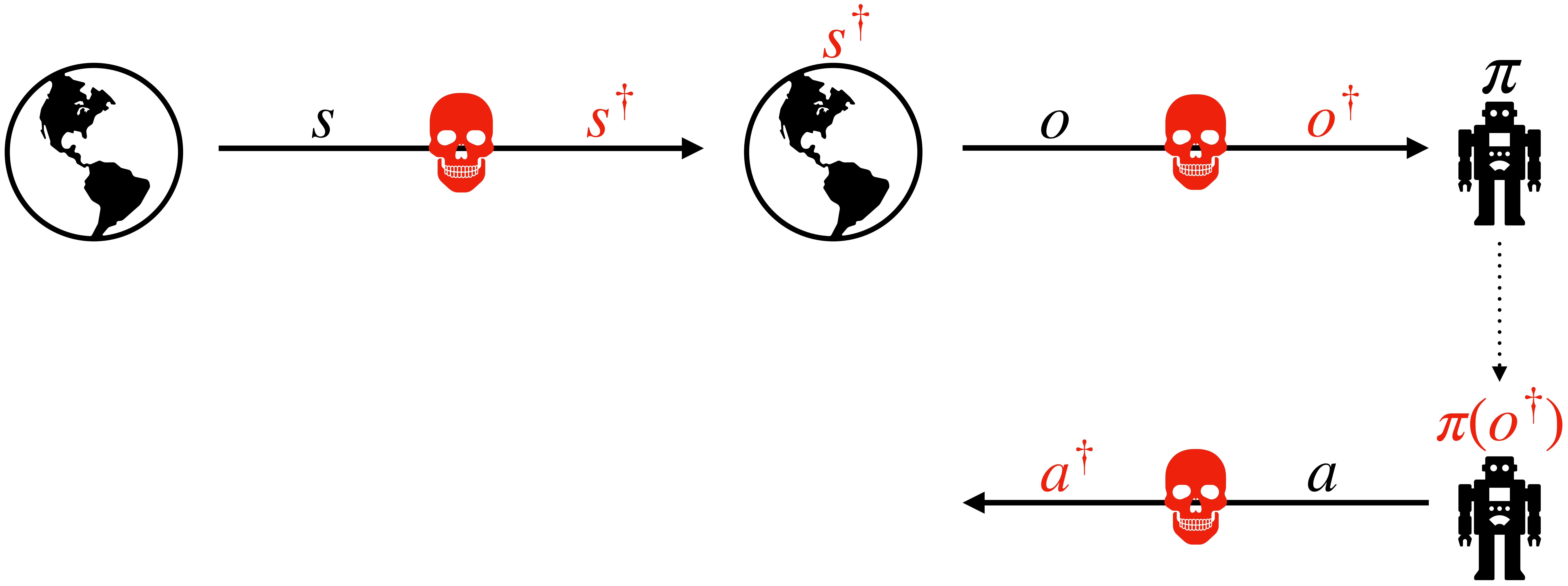
Adversarial Decomposition

We decompose the attacked $\boldsymbol{\pi}$-M interaction based on the *flow of information*.



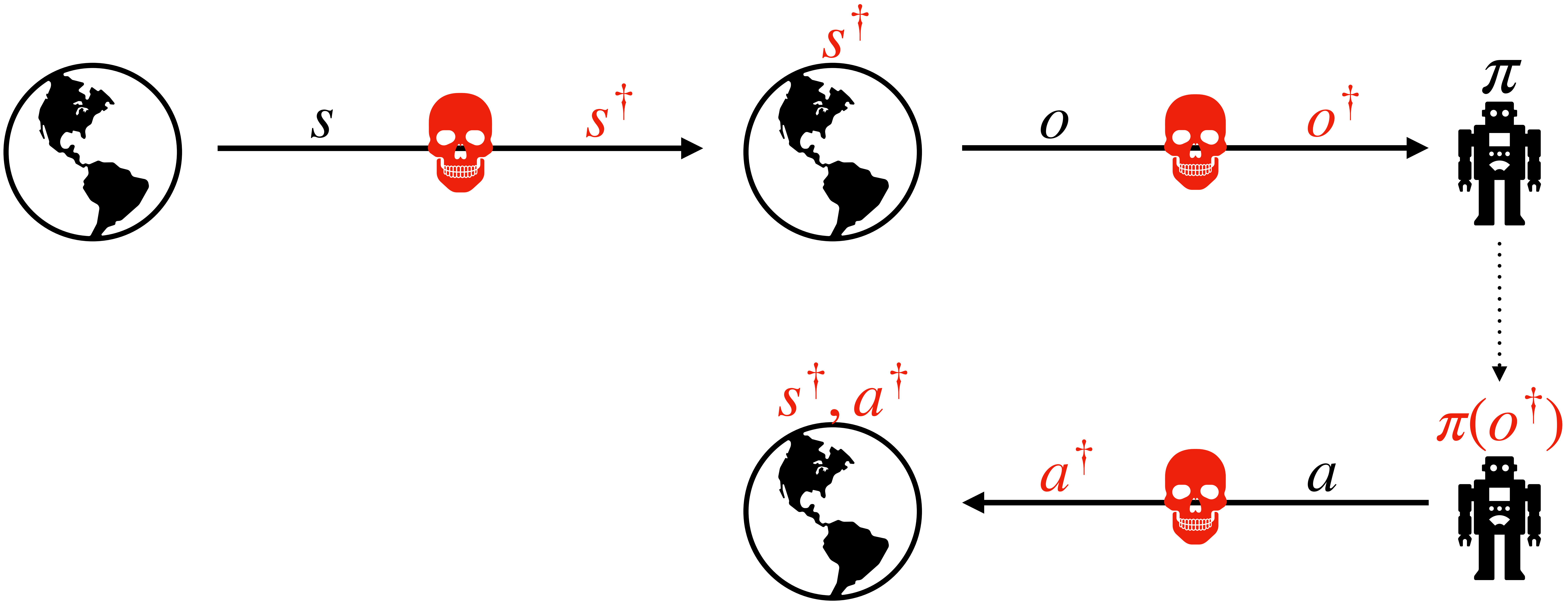
Adversarial Decomposition

We decompose the attacked π-M interaction based on the *flow of information*.



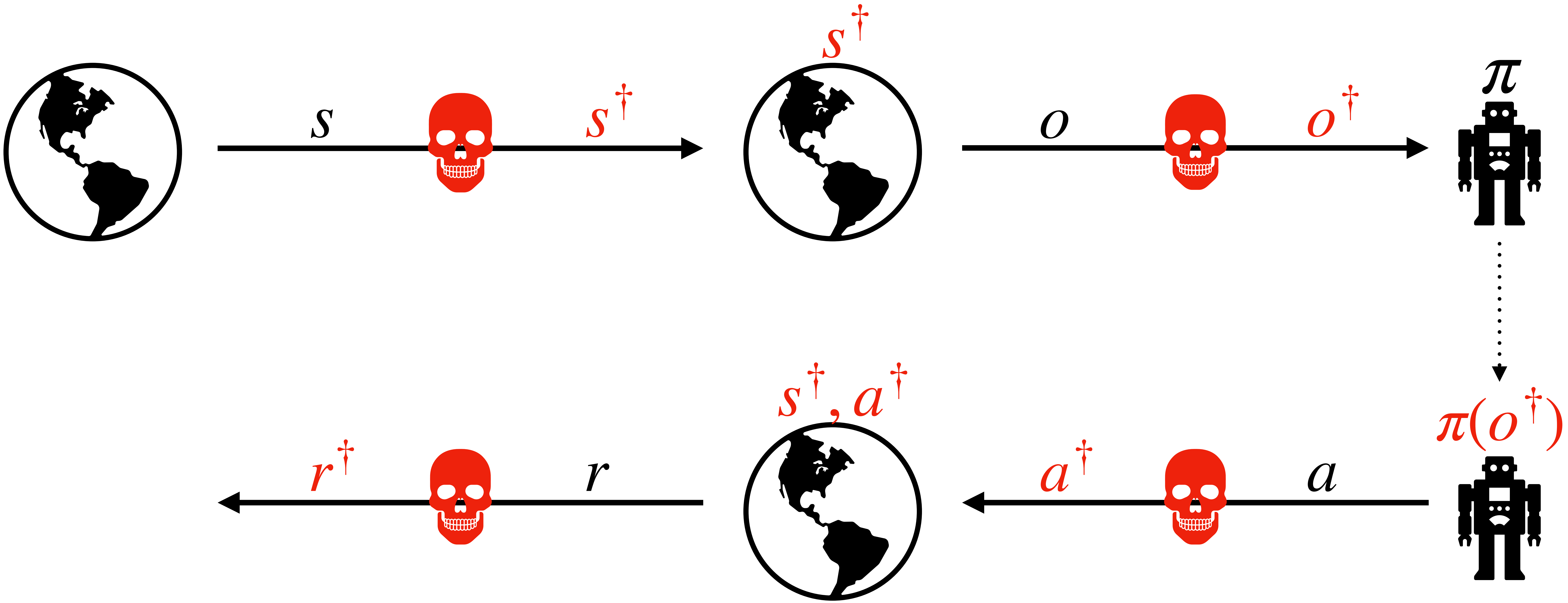
Adversarial Decomposition

We decompose the attacked π-M interaction based on the *flow of information*.



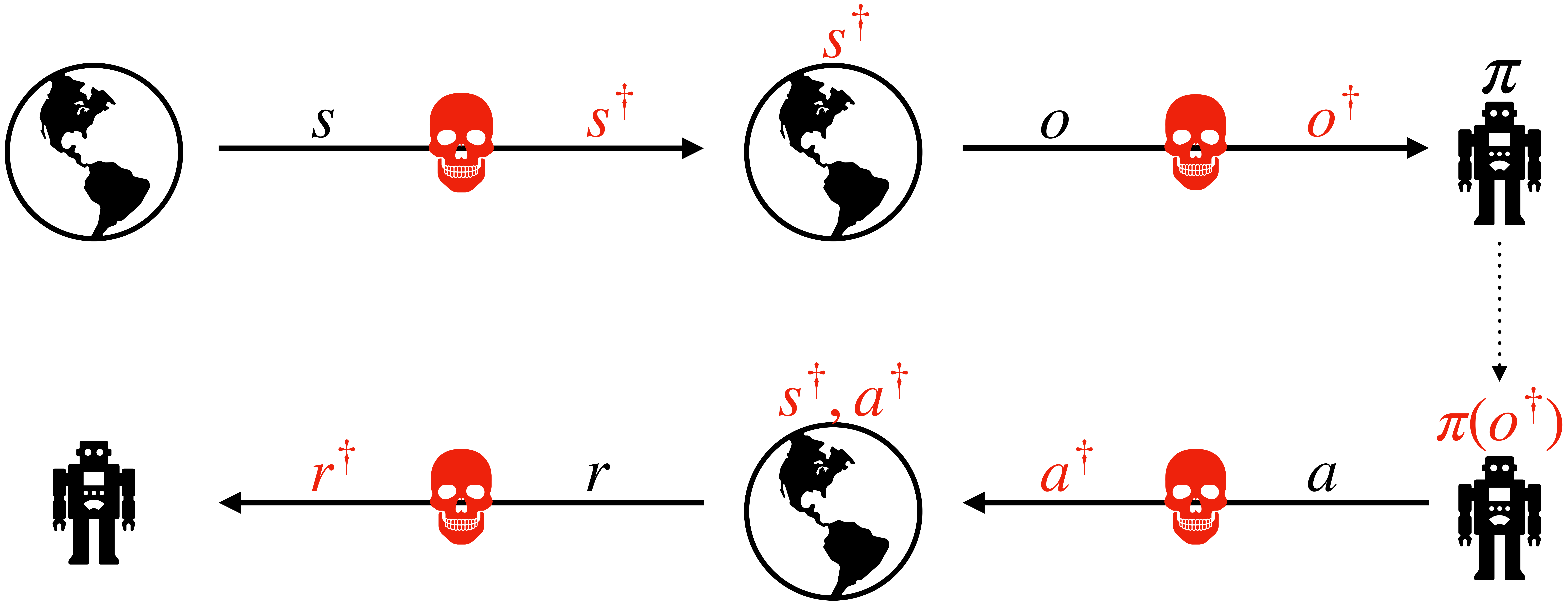
Adversarial Decomposition

We decompose the attacked π-M interaction based on the *flow of information*.



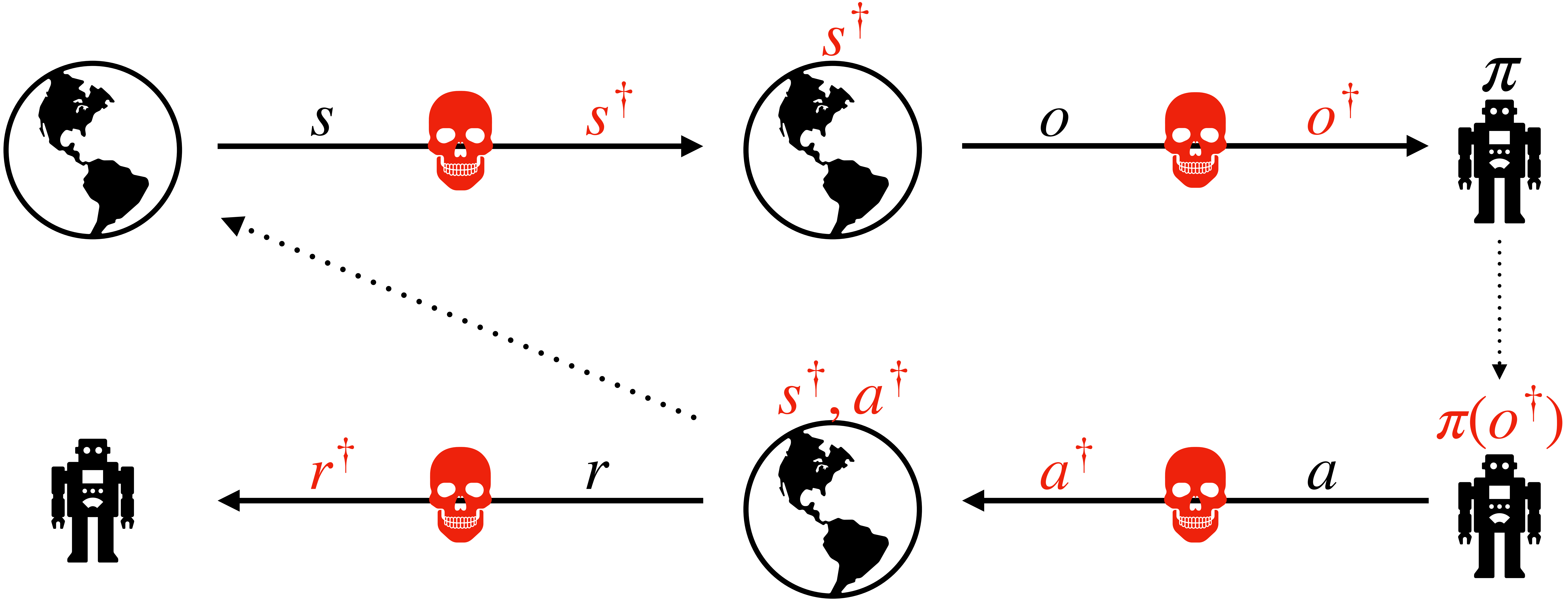
Adversarial Decomposition

We decompose the attacked $\boldsymbol{\pi}$-M interaction based on the *flow of information*.



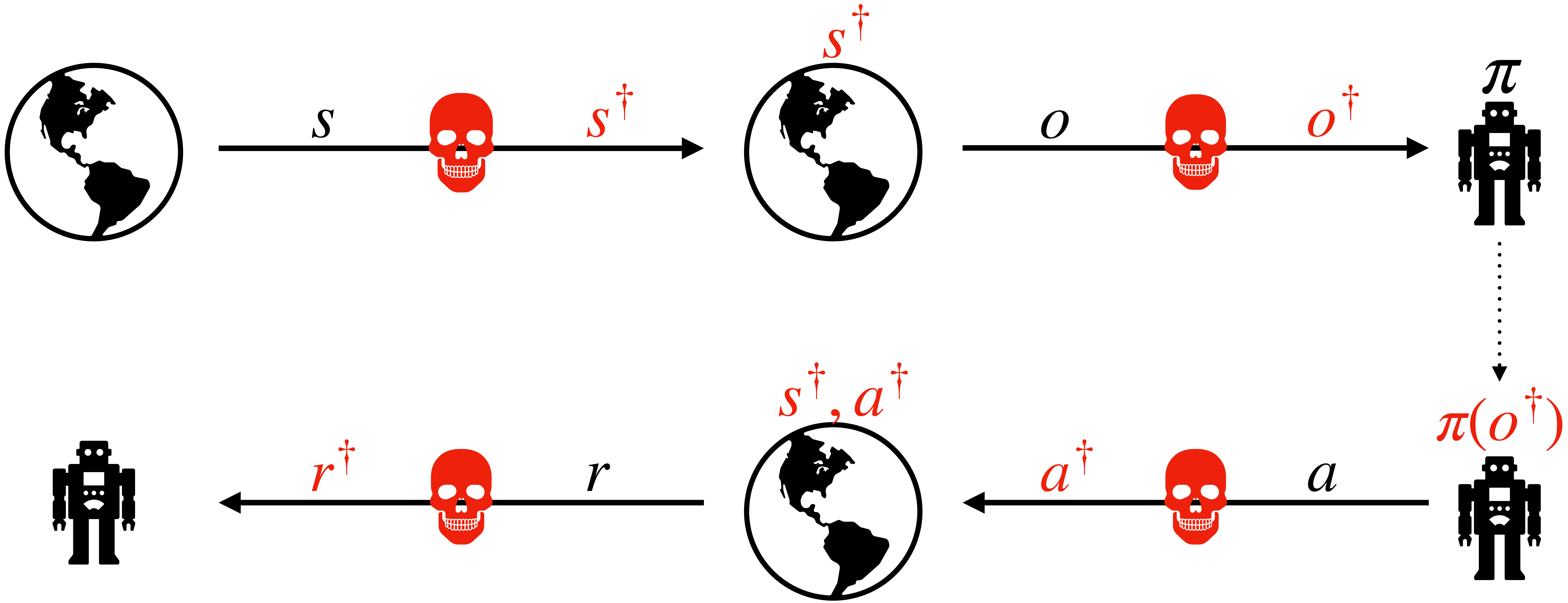
Adversarial Decomposition

We decompose the attacked $\boldsymbol{\pi}$-M interaction based on the *flow of information*.



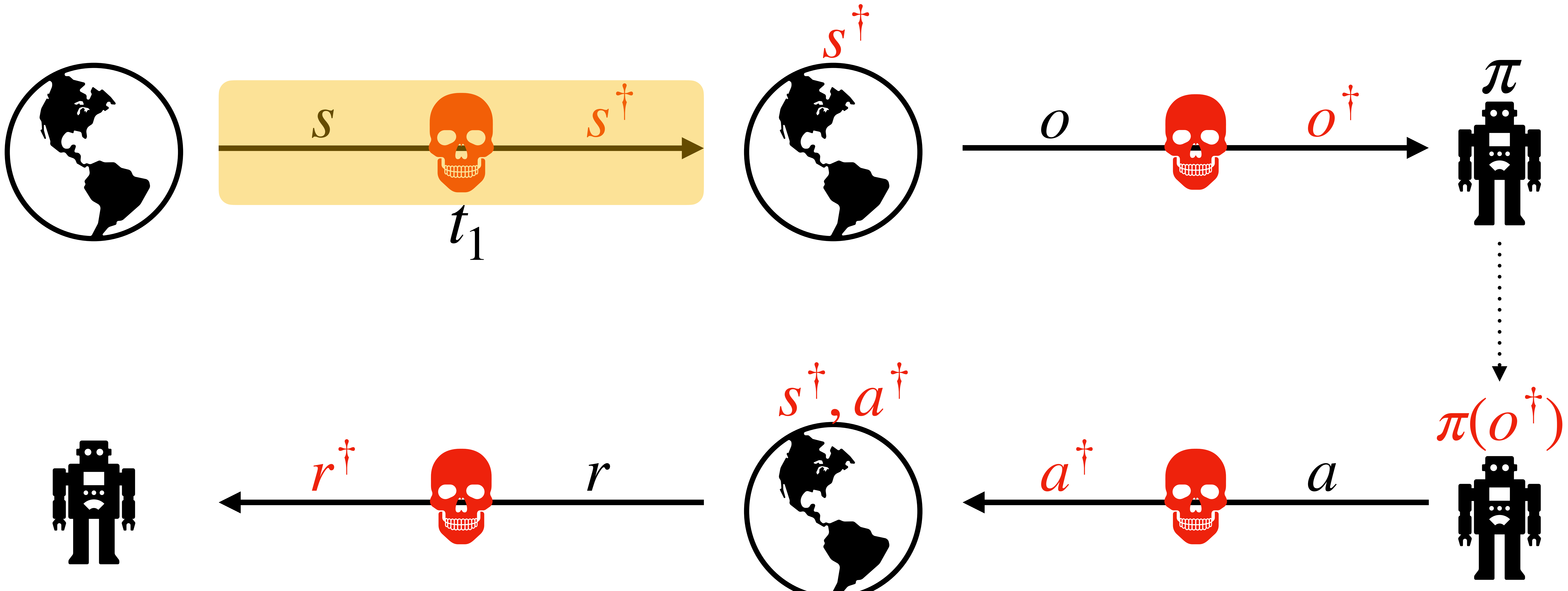
Adversarial Decomposition

We decompose the attacked $\boldsymbol{\pi}$-M interaction based on the *flow of information*.



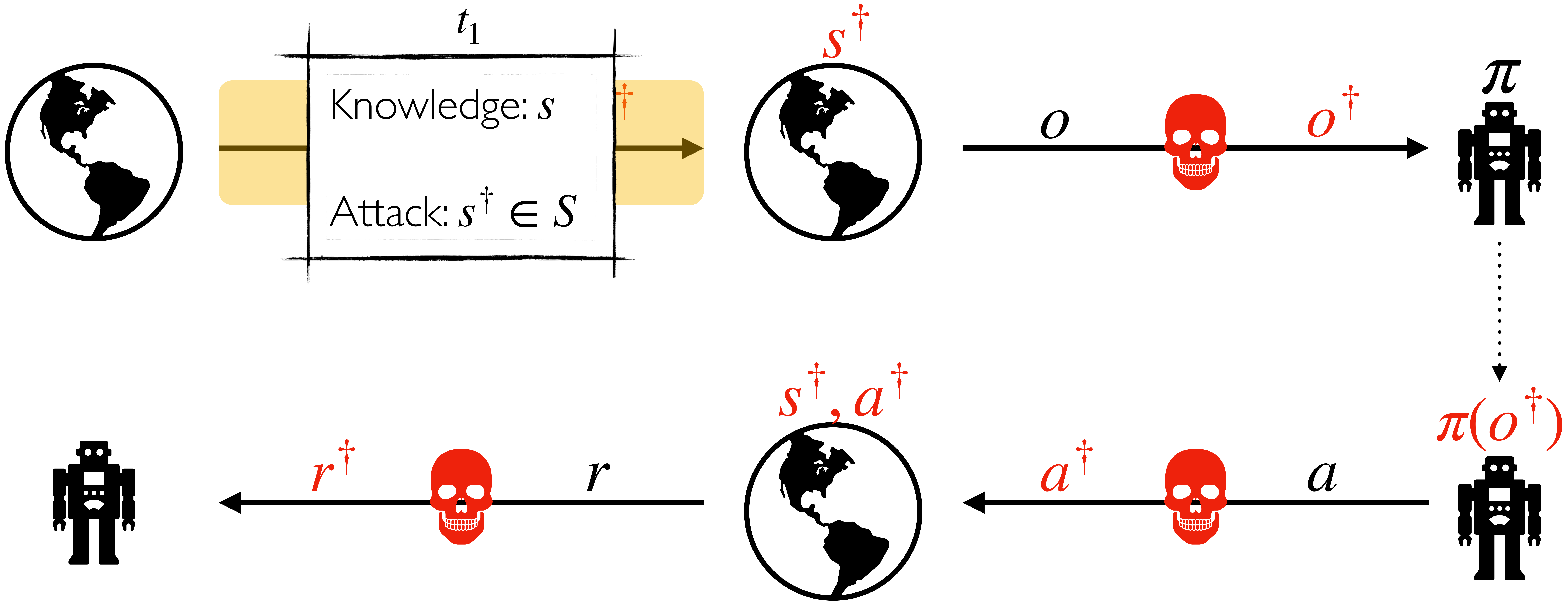
Adversarial Decomposition

We decompose the attacked **π**-M interaction based on the *flow of information*.



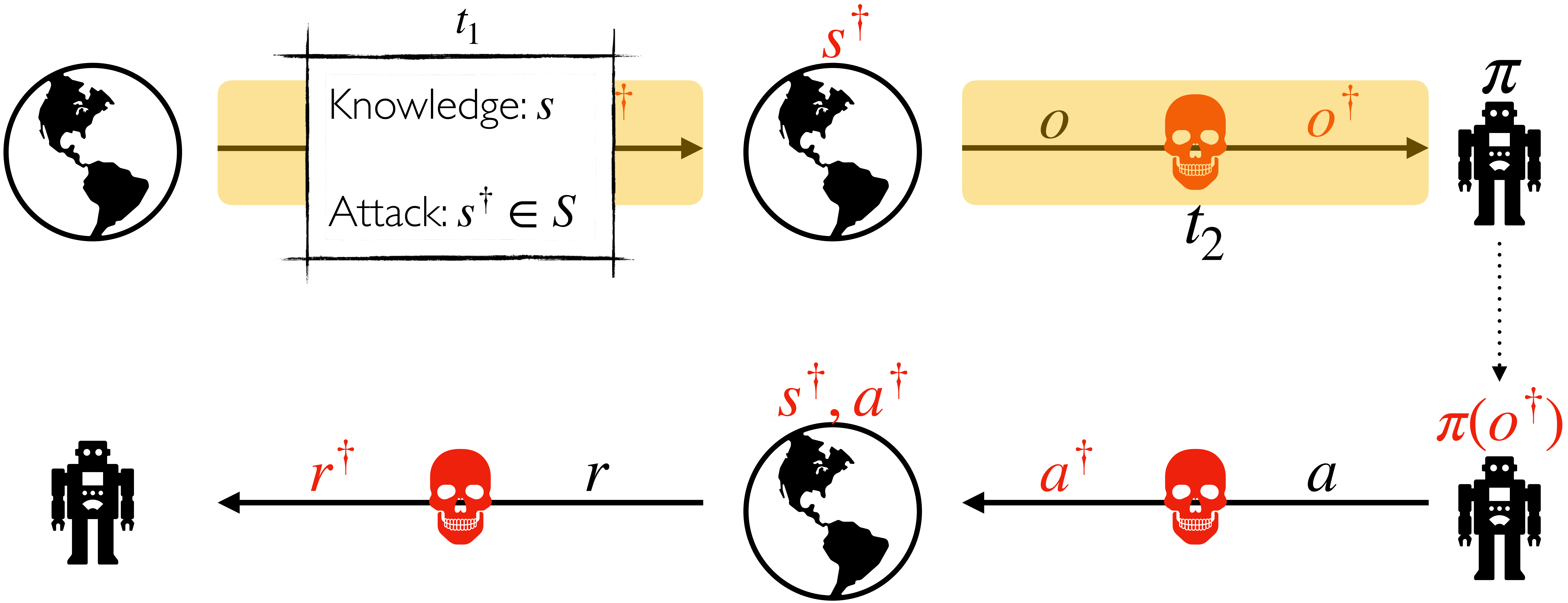
Adversarial Decomposition

We decompose the attacked π-M interaction based on the *flow of information*.



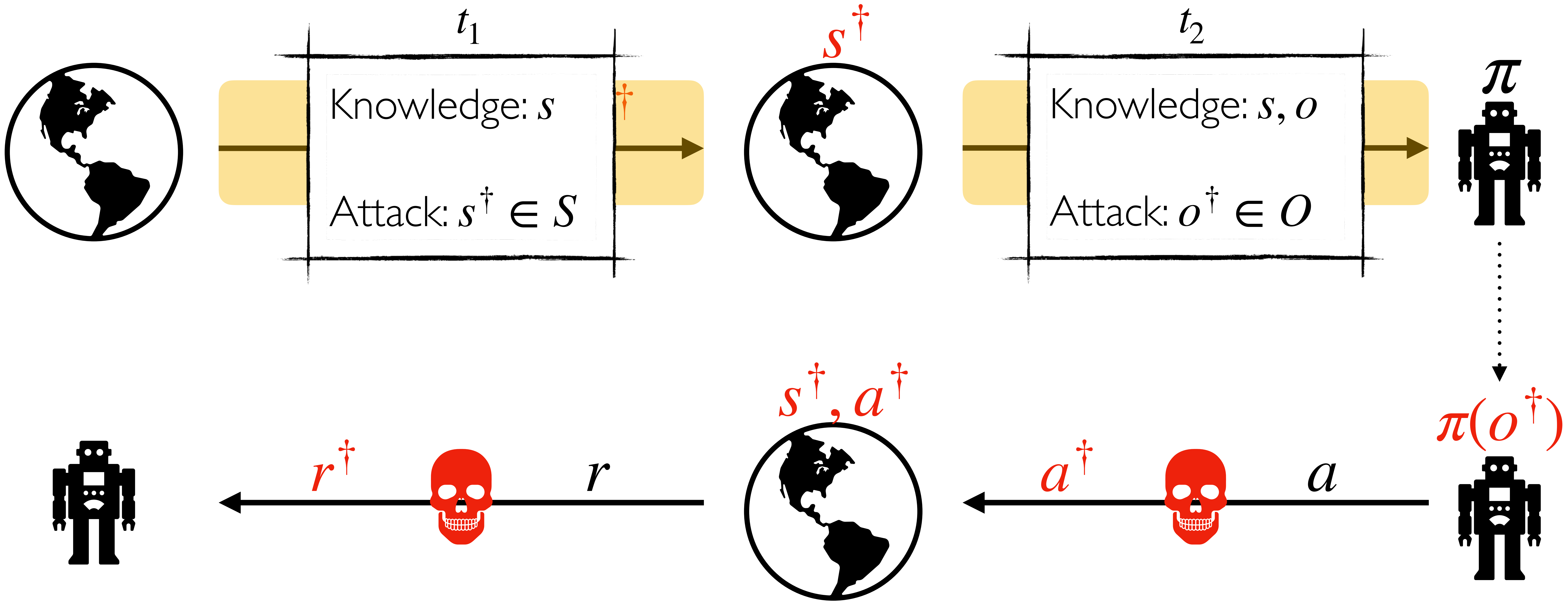
Adversarial Decomposition

We decompose the attacked π-M interaction based on the *flow of information*.



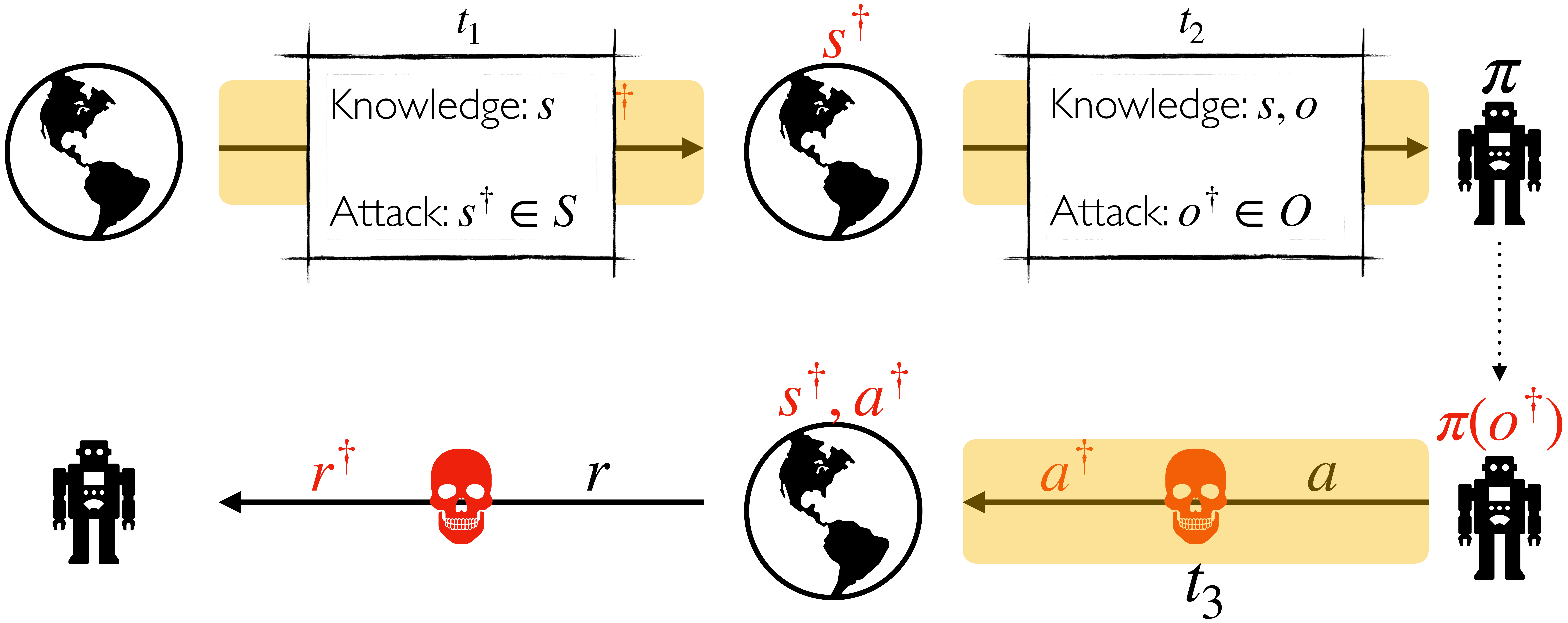
Adversarial Decomposition

We decompose the attacked π-M interaction based on the *flow of information*.



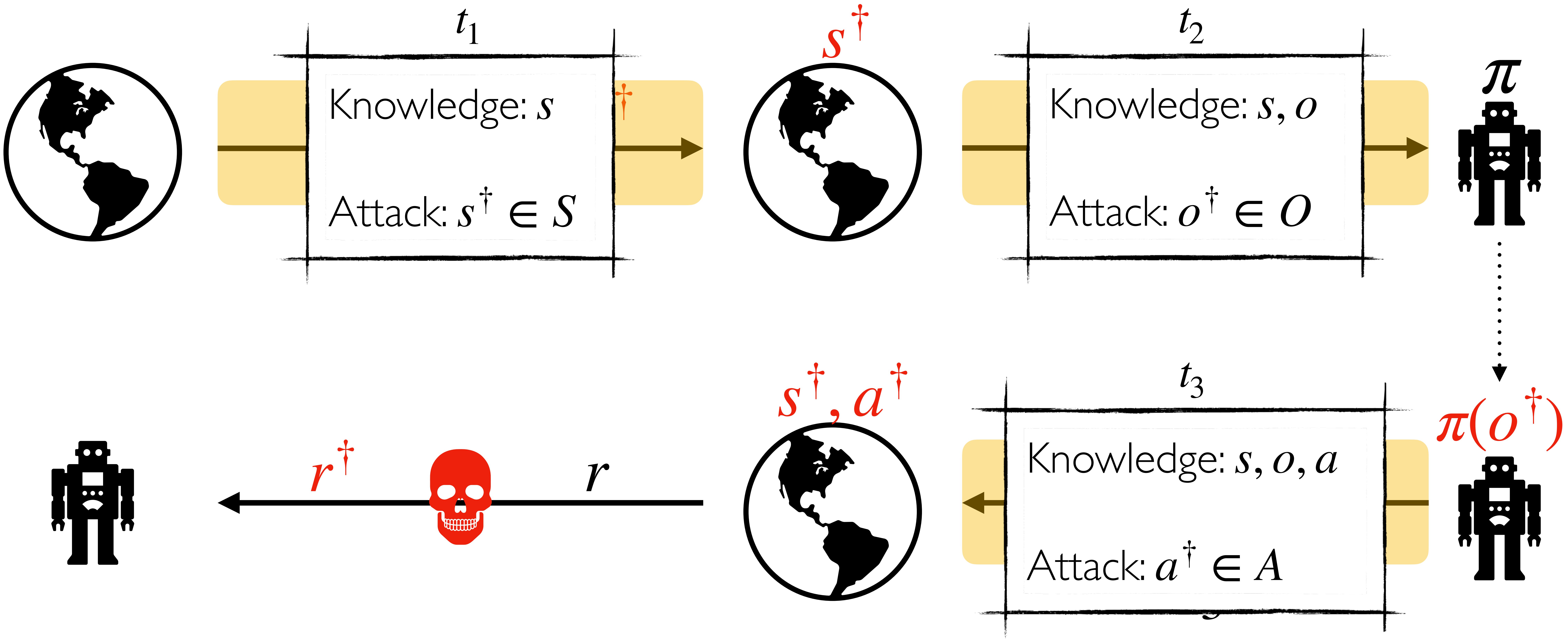
Adversarial Decomposition

We decompose the attacked π-M interaction based on the *flow of information*.



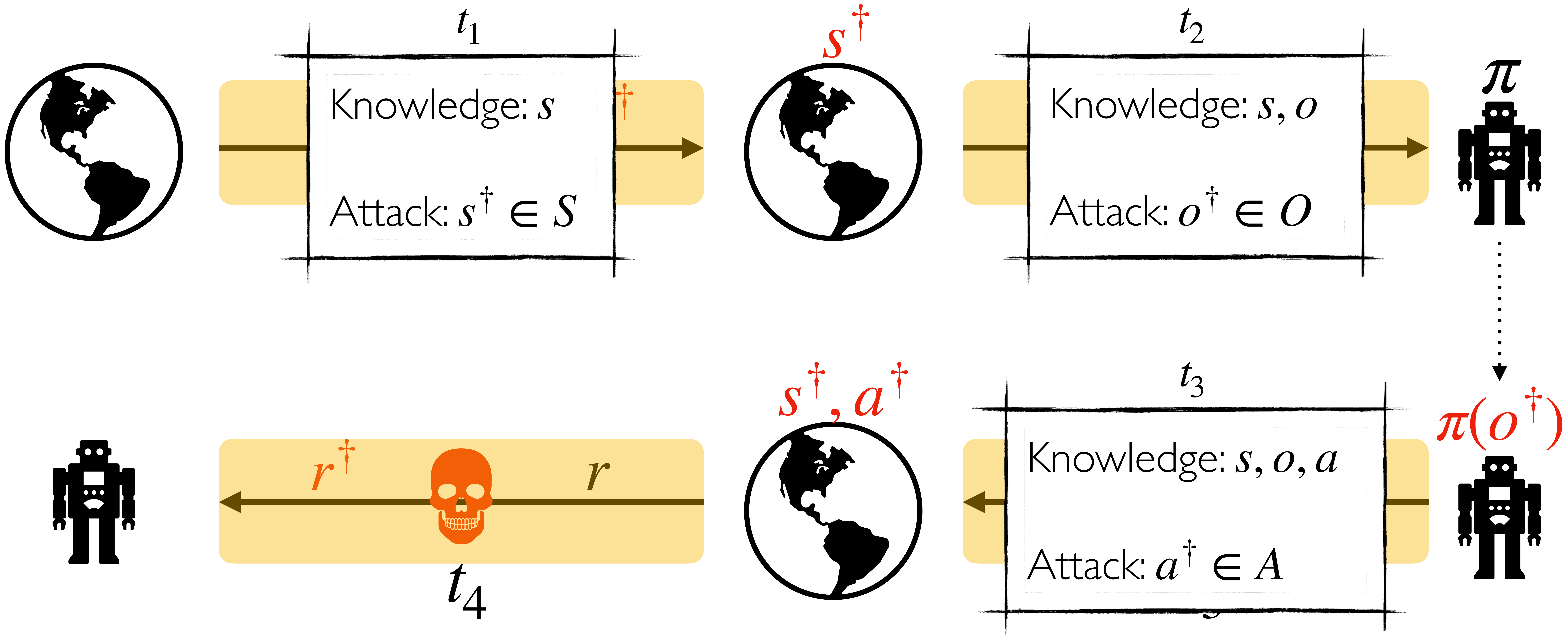
Adversarial Decomposition

We decompose the attacked π-M interaction based on the *flow of information*.



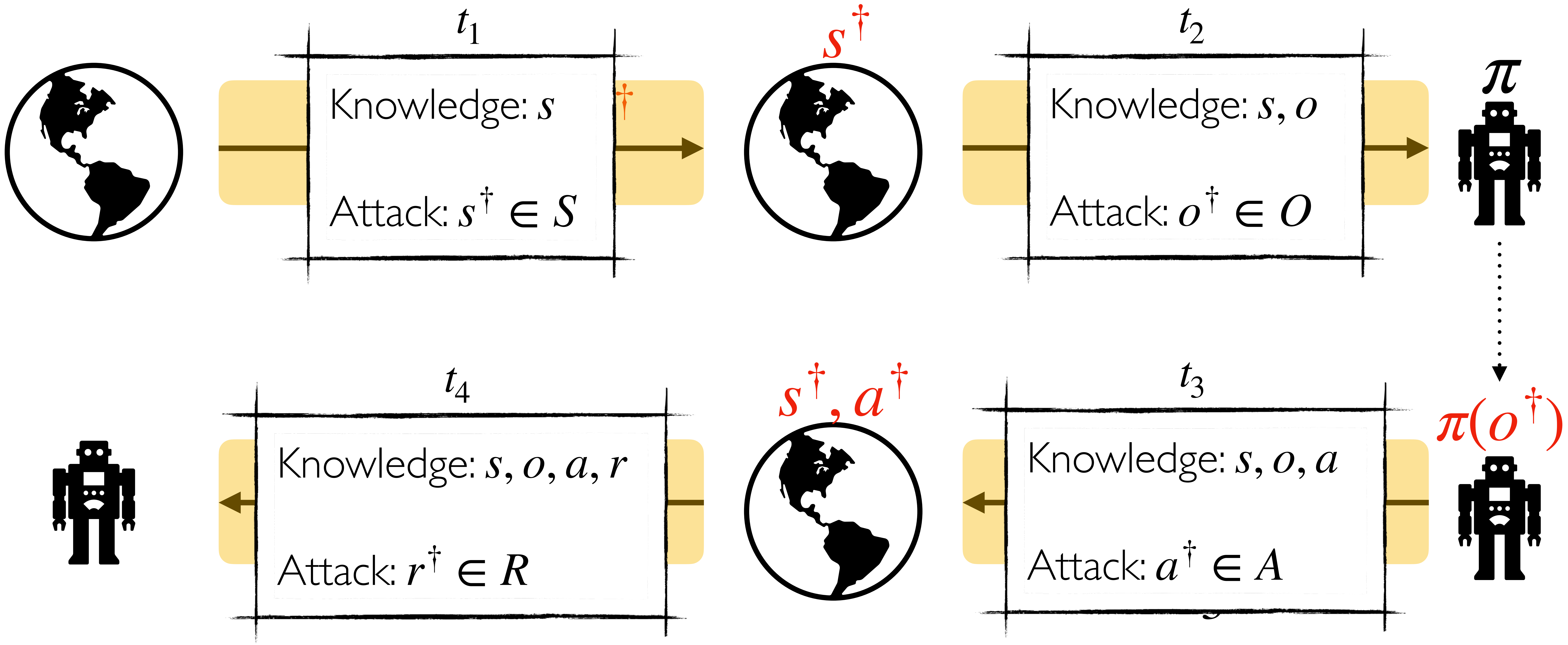
Adversarial Decomposition

We decompose the attacked π-M interaction based on the *flow of information*.



Adversarial Decomposition

We decompose the attacked π-M interaction based on the *flow of information*.



Attacker's Perspective

t_1

Knowledge: s

Attack: $s^{\dagger} \in S$

t_2

Knowledge: s, o

Attack: $o^{\dagger} \in O$

t_3

Knowledge: s, o, a

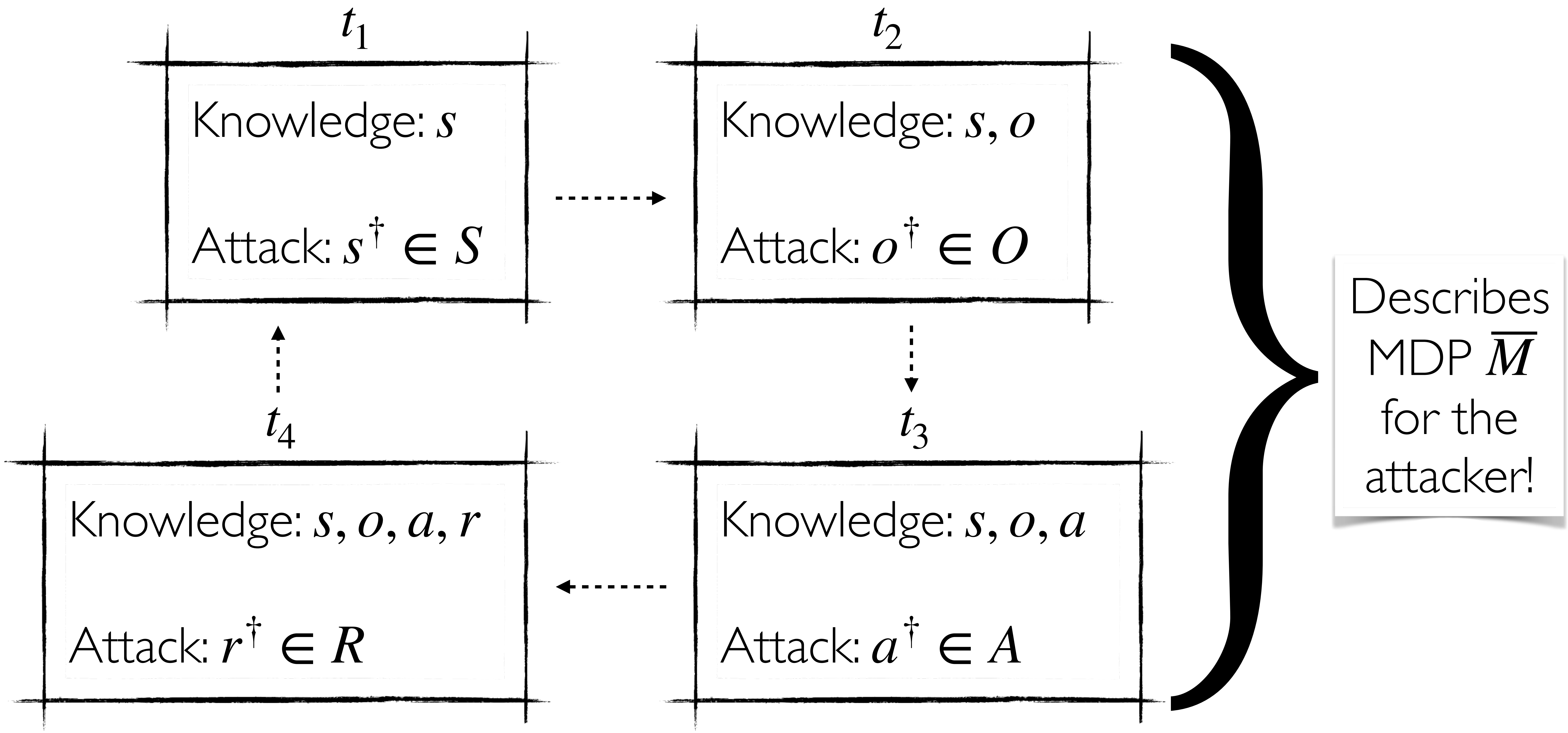
Attack: $a^{\dagger} \in A$

t_4

Knowledge: s, o, a, r

Attack: $r^{\dagger} \in R$

Attacker's Perspective



Meta MDP

Meta MDP

Attacker's interaction with $\boldsymbol{\pi}$ and $\boldsymbol{M}$ evolves according to MDP $\bar{\boldsymbol{M}}$.

Meta MDP

Attacker's interaction with π and M evolves according to MDP $\bar{M}$.

1. $\bar{S}$ records the attacker's information at any subperiod:

Meta MDP

Attacker's interaction with π and M evolves according to MDP $\bar{M}$.

1. $\bar{S}$ records the attacker's information at any subperiod:

$$\overline{\mathcal{S}} = \mathcal{S} \cup (\mathcal{S} \cup \mathcal{O}) \cup (\mathcal{S} \cup \mathcal{O} \cup \mathcal{A}) \cup (\mathcal{S} \cup \mathcal{O} \cup \mathcal{A} \cup \mathcal{R})$$

Meta MDP

Attacker's interaction with π and M evolves according to MDP $\bar{M}$.

1. $\bar{S}$ records the attacker's information at any subperiod:

$$\overline{\mathcal{S}} = \mathcal{S} \cup (\mathcal{S} \cup \mathcal{O}) \cup (\mathcal{S} \cup \mathcal{O} \cup \mathcal{A}) \cup (\mathcal{S} \cup \mathcal{O} \cup \mathcal{A} \cup \mathcal{R})$$

2. $\bar{A}$ captures the attacks available at any subperiod:

Meta MDP

Attacker's interaction with $\boldsymbol{\pi}$ and $\boldsymbol{M}$ evolves according to MDP $\bar{\boldsymbol{M}}$.

1. $\bar{S}$ records the attacker's information at any subperiod:

$$\overline{\mathcal{S}} = \mathcal{S} \cup (\mathcal{S} \cup \mathcal{O}) \cup (\mathcal{S} \cup \mathcal{O} \cup \mathcal{A}) \cup (\mathcal{S} \cup \mathcal{O} \cup \mathcal{A} \cup \mathcal{R})$$

2. $\bar{A}$ captures the attacks available at any subperiod:

$$\overline{\mathcal{A}}(s) \subseteq \mathcal{S},\ \overline{\mathcal{A}}(s,o) \subseteq \mathcal{O},\ \overline{\mathcal{A}}(s,o,a) \subseteq \mathcal{A},\ \overline{\mathcal{A}}(s,o,a,r) \subseteq \mathcal{R}$$

Meta MDP

Attacker's interaction with $\boldsymbol{\pi}$ and $\boldsymbol{M}$ evolves according to MDP $\bar{\boldsymbol{M}}$.

1. $\bar{S}$ records the attacker's information at any subperiod:

$$\overline{\mathcal{S}} = \mathcal{S} \cup (\mathcal{S} \cup \mathcal{O}) \cup (\mathcal{S} \cup \mathcal{O} \cup \mathcal{A}) \cup (\mathcal{S} \cup \mathcal{O} \cup \mathcal{A} \cup \mathcal{R})$$

2. $\bar{A}$ captures the attacks available at any subperiod:

$$\overline{\mathcal{A}}(s) \subseteq \mathcal{S},\ \overline{\mathcal{A}}(s,o) \subseteq \mathcal{O},\ \overline{\mathcal{A}}(s,o,a) \subseteq \mathcal{A},\ \overline{\mathcal{A}}(s,o,a,r) \subseteq \mathcal{R}$$

3. Transitions capture the evolution of information.

Meta MDP

Attacker's interaction with π and M evolves according to MDP $\bar{M}$.

1. $\bar{S}$ records the attacker's information at any subperiod:

$$\overline{\mathcal{S}} = \mathcal{S} \cup (\mathcal{S} \cup \mathcal{O}) \cup (\mathcal{S} \cup \mathcal{O} \cup \mathcal{A}) \cup (\mathcal{S} \cup \mathcal{O} \cup \mathcal{A} \cup \mathcal{R})$$

2. $\bar{A}$ captures the attacks available at any subperiod:

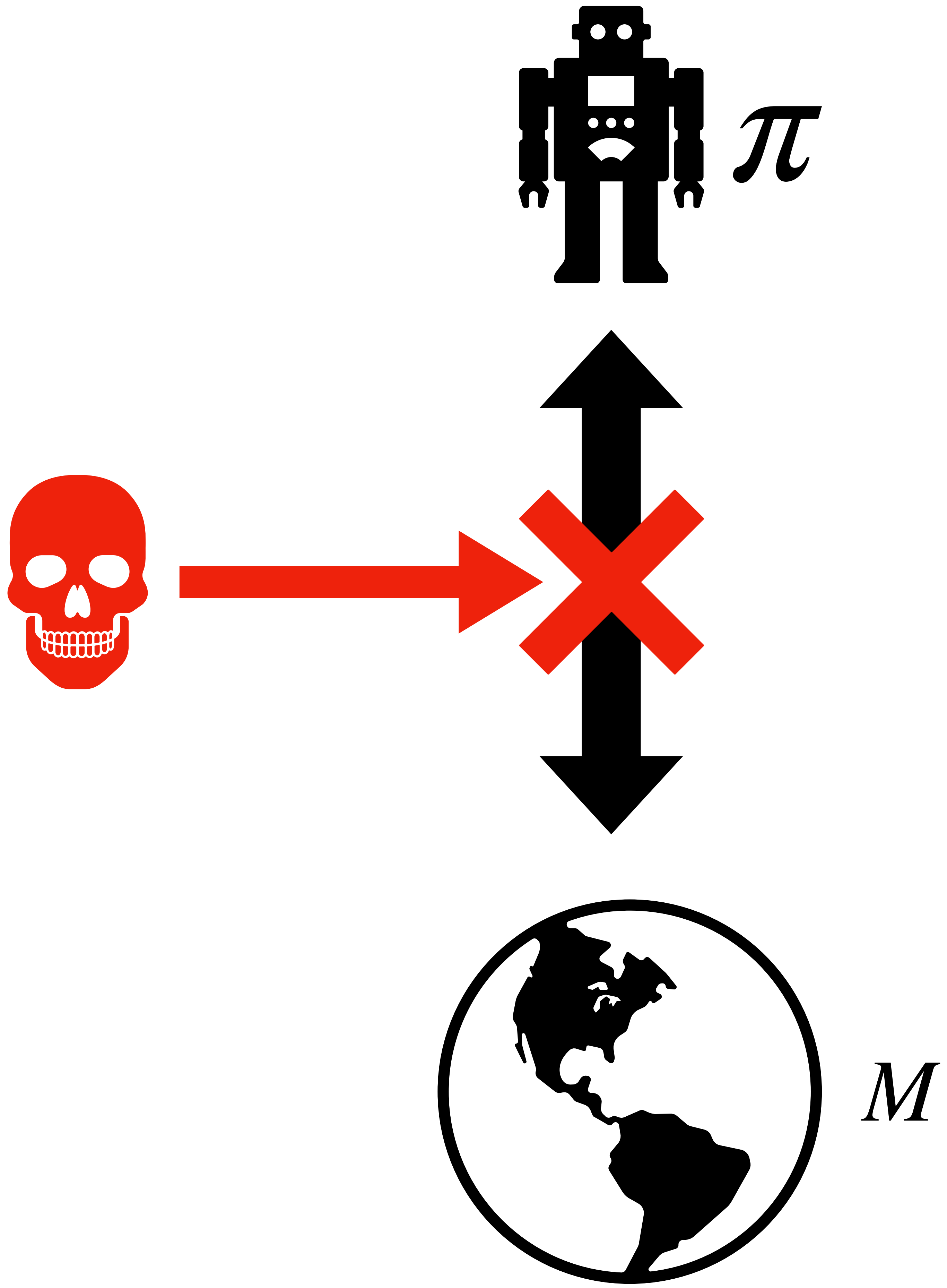
$$\overline{\mathcal{A}}(s) \subseteq \mathcal{S},\ \overline{\mathcal{A}}(s,o) \subseteq \mathcal{O},\ \overline{\mathcal{A}}(s,o,a) \subseteq \mathcal{A},\ \overline{\mathcal{A}}(s,o,a,r) \subseteq \mathcal{R}$$

3. Transitions capture the evolution of information.

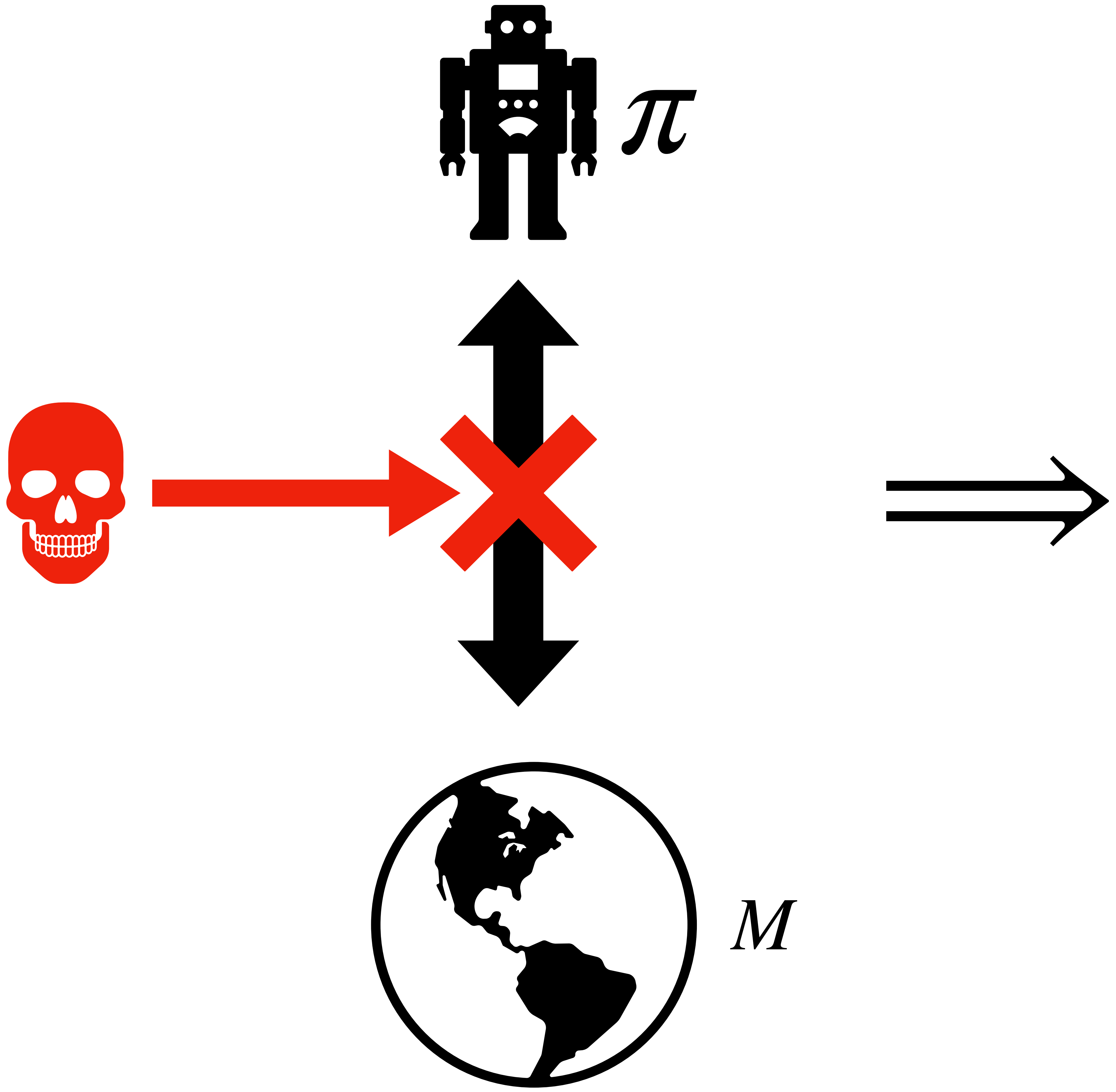
Proposition: Any optimal policy for $\overline{M}$ is an optimal attack policy.

Reduction to RL

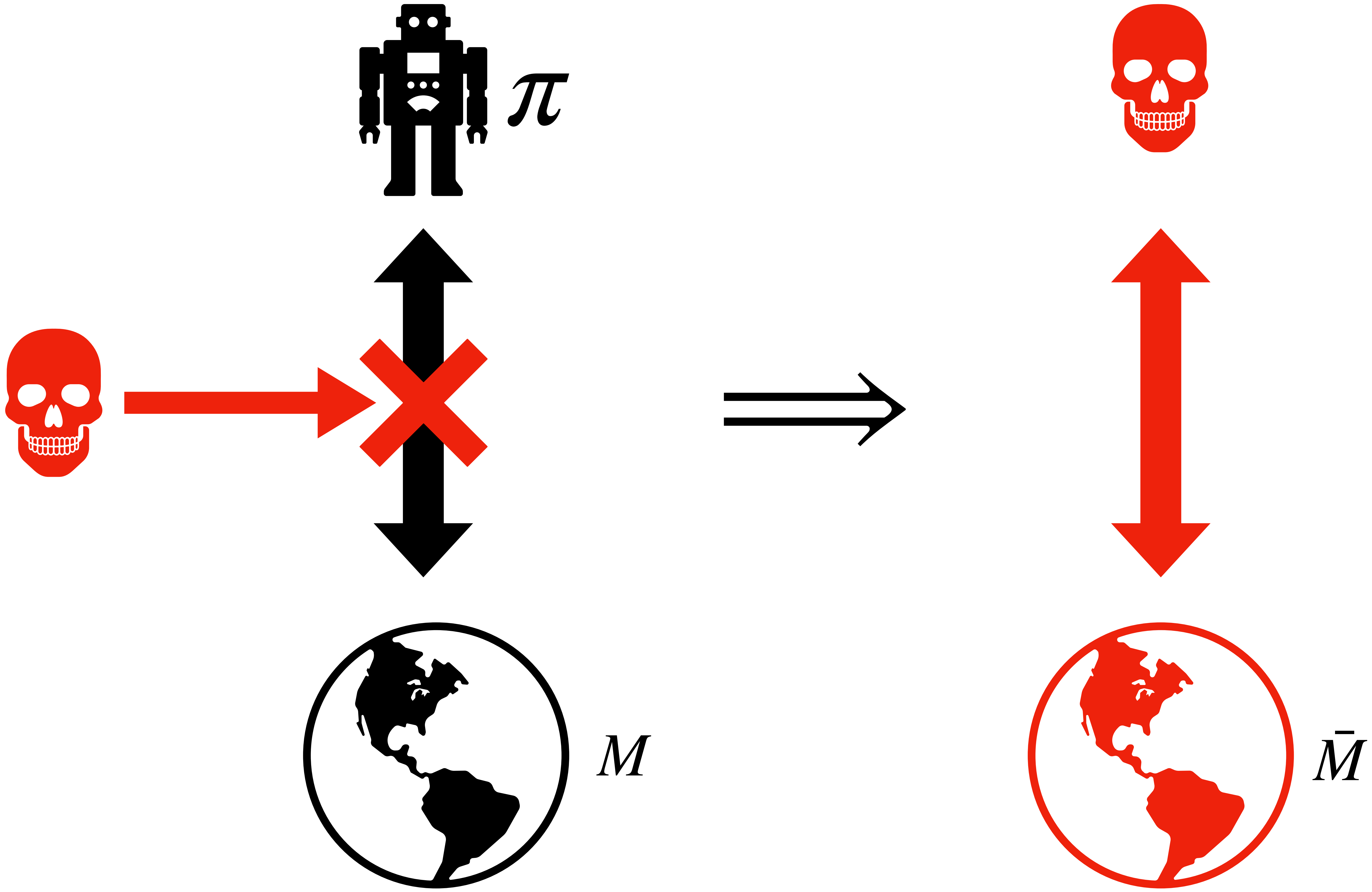
Reduction to RL



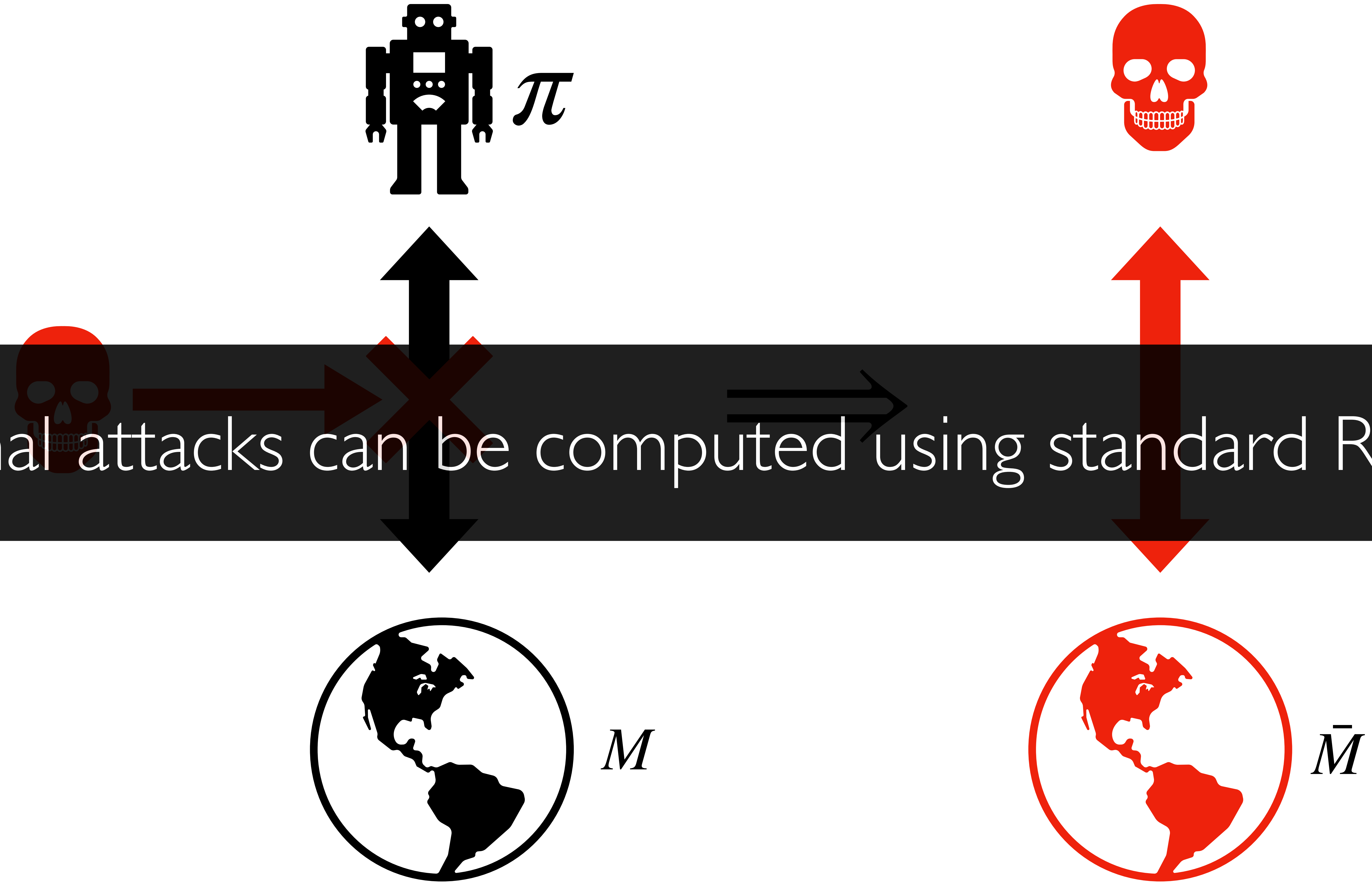
Reduction to RL



Reduction to RL



Reduction to RL



Optimal attacks can be computed using standard RL techniques!

Computational Efficiency

Computational Efficiency

$$|\bar{S}| \leq SOAR \quad \text{and} \quad |\bar{A}| \leq S + O + A + R$$

Computational Efficiency

$$|\bar{S}| \leq SOAR \quad \text{and} \quad |\bar{A}| \leq S + O + A + R$$

↓

$\overline{M}$ has only polynomially larger state and action space than M.

Computational Efficiency

$$|\bar{S}| \leq SOAR \quad \text{and} \quad |\bar{A}| \leq S + O + A + R$$

Attacking RL *efficiently* reduces to RL!

$\bar{M}$ has only polynomially larger state and action space than M.

Can we defend against attacks?

Defense

The Defense Problem

The Defense Problem

Let $(V_1^{\pi,\nu}, V_2^{\pi,\nu})$ denote the victim's and attacker's value, respectively.

The Defense Problem

Let $(V_1^{\pi,\nu}, V_2^{\pi,\nu})$ denote the victim's and attacker's value, respectively.

Definition 2 (Defense Problem)**.** The victim seeks a policy π^* that maximizes its expected reward from the victim-attacker-M interaction under the worst-case attack:

$$\pi^* \in \arg\max_{\pi \in \Pi} \min_{\nu \in BR(\pi)} V_1^{\pi,\nu}.$$

The Defense Problem

Let $(V_1^{\pi,\nu}, V_2^{\pi,\nu})$ denote the victim's and attacker's value, respectively.

Definition 2 (Defense Problem)**.** The victim seeks a policy π^* that maximizes its expected reward from the victim-attacker-M interaction under the worst-case attack:

$$\pi^* \in \arg\max_{\pi \in \Pi} \min_{\nu \in BR(\pi)} V_1^{\pi,\nu}.$$

$$BR(\pi) := \arg\max_{\nu \in N} V_2^{\pi,\nu}$$

The Defense Problem

Let $(V_1^{\pi,\nu}, V_2^{\pi,\nu})$ denote the victim's and attacker's value, respectively.

[redacted]

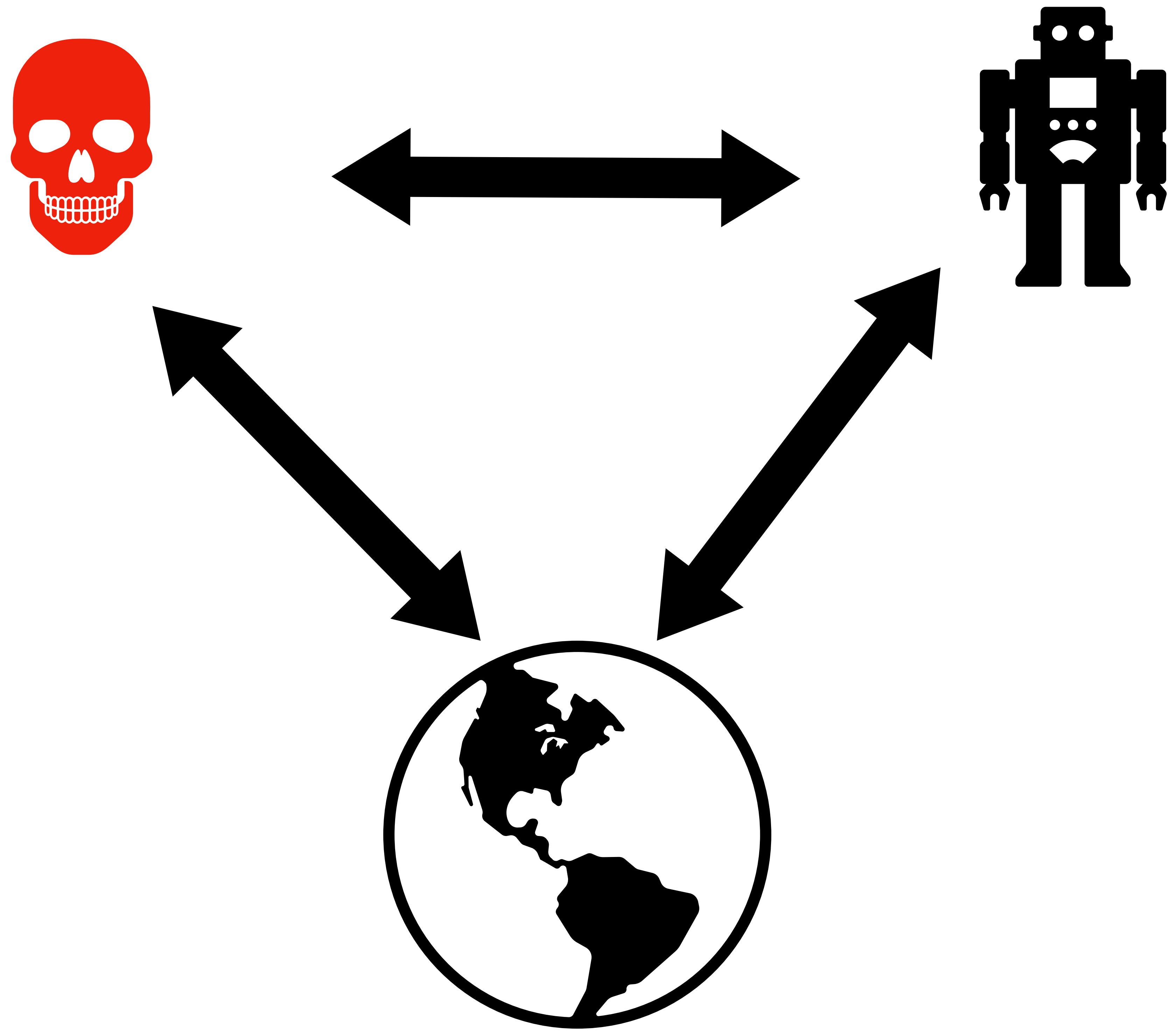
Avoids Cat and Mouse Game!

$$\pi^* \in \arg\max_{\pi \in \Pi} \min_{\nu \in BR(\pi)} V_1^{\pi,\nu}.$$

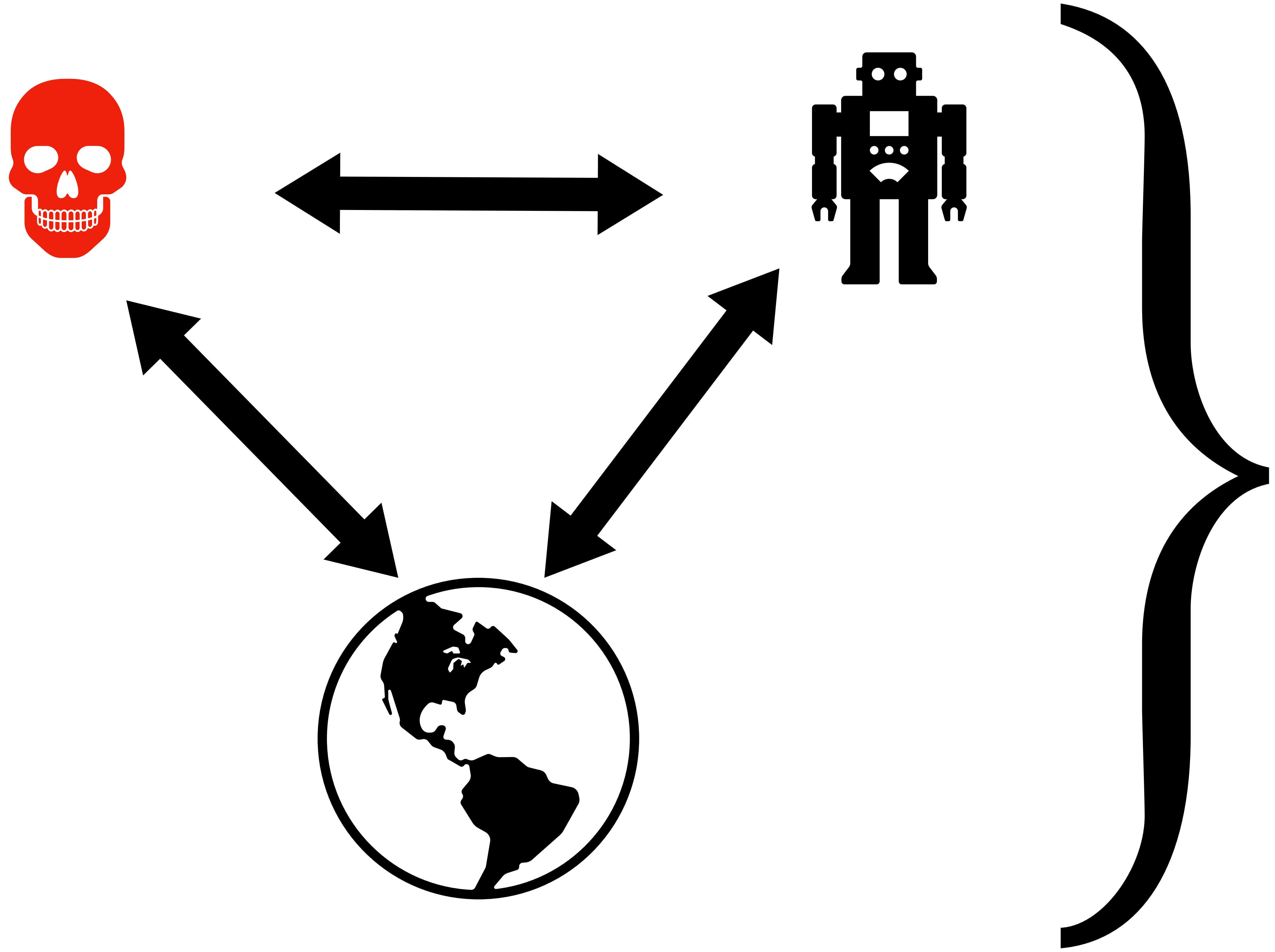
$$BR(\pi) := \arg\max_{\nu \in N} V_2^{\pi,\nu}$$

Reduction to MARL

Reduction to MARL

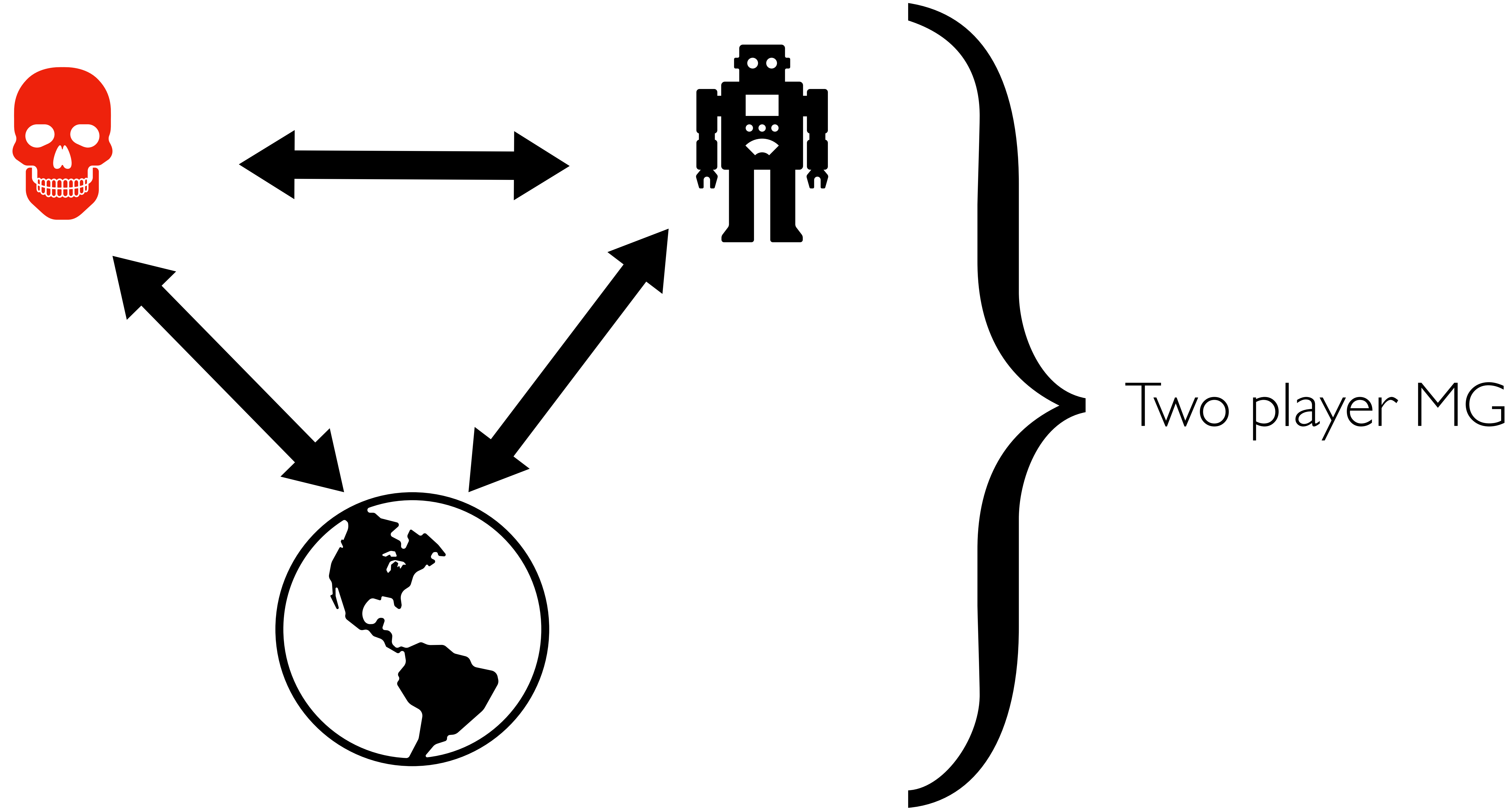


Reduction to MARL



Two player MG

Reduction to MARL



Defense corresponds to a Weak Stackelberg Equilibrium (WSE).

Challenges

Challenges

- WSE need not exist.

Challenges

- WSE need not exist.
- WSE are generally non-Markovian!

Challenges

- WSE need not exist.
- WSE are generally non-Markovian!

Proposition: The defense problem is as hard as solving POMDPs.

Thus, the defense problem is NP-hard to even approximate.

Special Structure: Sequential Play

Special Structure: Sequential Play

Key: restrict observation attacks.

Special Structure: Sequential Play

Key: restrict observation attacks.

P2

s

Special Structure: Sequential Play

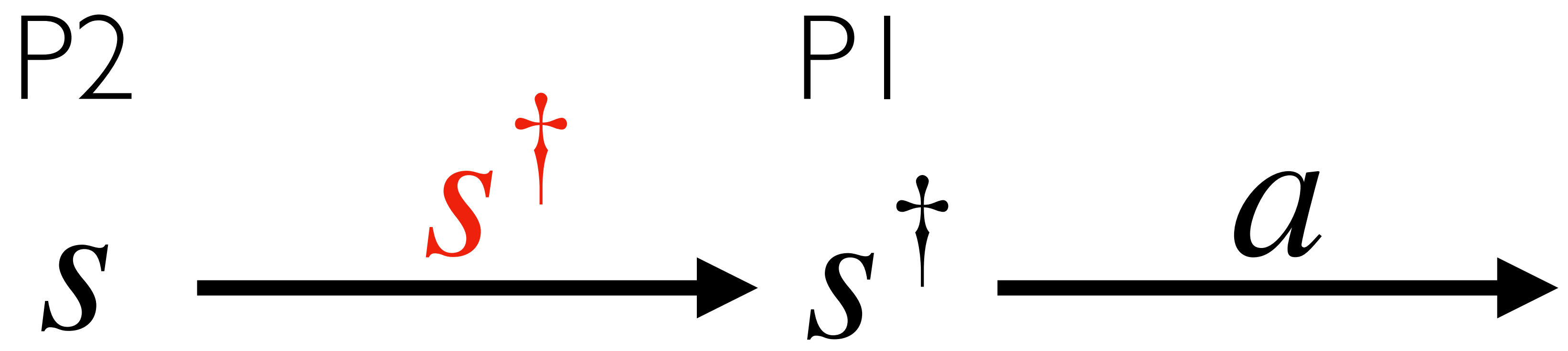
Key: restrict observation attacks.

Special Structure: Sequential Play

Key: restrict observation attacks.

Special Structure: Sequential Play

Key: restrict observation attacks.



Special Structure: Sequential Play

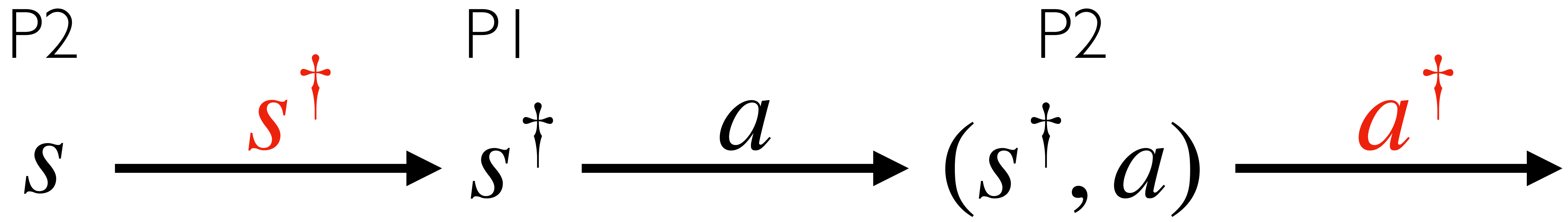
Key: restrict observation attacks.

P2 P1 P2

$$s \xrightarrow{\;s^{\dagger}\;} s^{\dagger} \xrightarrow{\;a\;} (s^{\dagger}, a)$$

Special Structure: Sequential Play

Key: restrict observation attacks.



Special Structure: Sequential Play

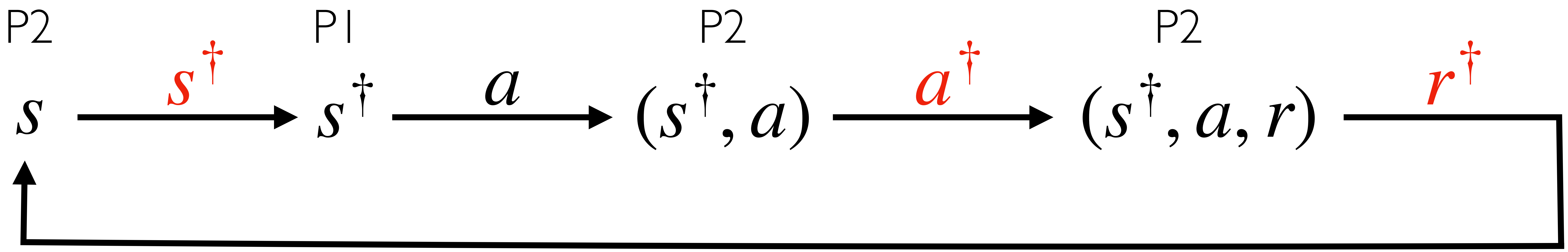
Key: restrict observation attacks.

P2 — P1 — P2 — P2

$$s \xrightarrow{\color{red}{s^\dagger}} s^\dagger \xrightarrow{a} (s^\dagger, a) \xrightarrow{\color{red}{a^\dagger}} (s^\dagger, a, r)$$

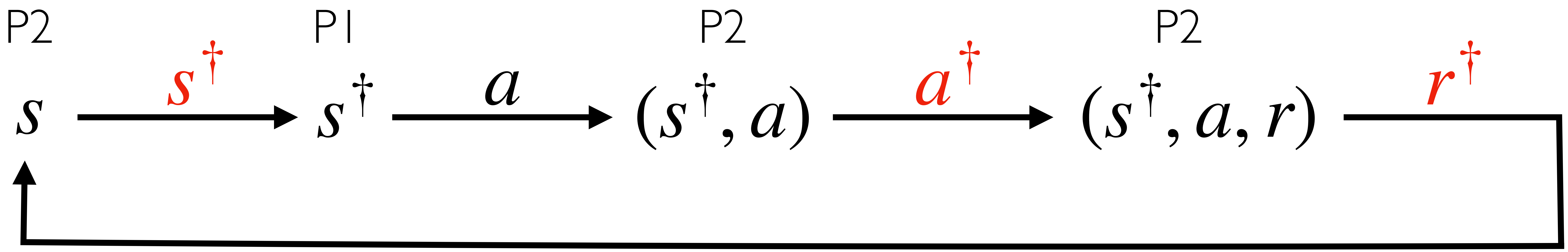
Special Structure: Sequential Play

Key: restrict observation attacks.



Special Structure: Sequential Play

Key: restrict observation attacks.



Game evolves like a *turn-based* Markov game $\overline{G}$.

Meta Turn-based Markov Game

1. $\bar{S}$ records the player's information at any subperiod:

$$\overline{\mathcal{S}}_1 = \mathcal{S} \quad \text{and} \quad \overline{\mathcal{S}}_2 = \mathcal{S} \cup (\mathcal{S} \cup \mathcal{A}) \cup (\mathcal{S} \cup \mathcal{A} \cup \mathcal{R})$$

2. $\bar{A}$ captures the actions available at any subperiod:

$$\overline{\mathcal{A}}_1 = \mathcal{A} \quad \text{and} \quad \overline{\mathcal{A}}_2(s) \subseteq \mathcal{S},\ \overline{\mathcal{A}}_2(s,a) \subseteq \mathcal{A},\ \overline{\mathcal{A}}_2(s,a,r) \subseteq \mathbb{R}$$

3. Transitions capture the evolution of information.

Meta Turn-based Markov Game

1. $\bar{S}$ records the player's information at any subperiod:

$$\overline{\mathcal{S}}_1 = \mathcal{S} \quad \text{and} \quad \overline{\mathcal{S}}_2 = \mathcal{S} \cup (\mathcal{S} \cup \mathcal{A}) \cup (\mathcal{S} \cup \mathcal{A} \cup \mathcal{R})$$

2. $\bar{A}$ captures the actions available at any subperiod:

$$\overline{\mathcal{A}}_1 = \mathcal{A} \quad \text{and} \quad \overline{\mathcal{A}}_2(s) \subseteq \mathcal{S},\ \overline{\mathcal{A}}_2(s,a) \subseteq \mathcal{A},\ \overline{\mathcal{A}}_2(s,a,r) \subseteq \mathbb{R}$$

3. Transitions capture the evolution of information.

Proposition: Any WSE for $\overline{G}$ is an optimal defense policy.

Efficient Reduction to MARL

Efficient Reduction to MARL

$\overline{G}$

Efficient Reduction to MARL

$\overline{G}$

Zero-sum:

Efficient Reduction to MARL

$\overline{G}$

Zero-sum:

Defense → WSE

Efficient Reduction to MARL

$\overline{G}$

Zero-sum:

Defense → WSE → MPNE

Efficient Reduction to MARL

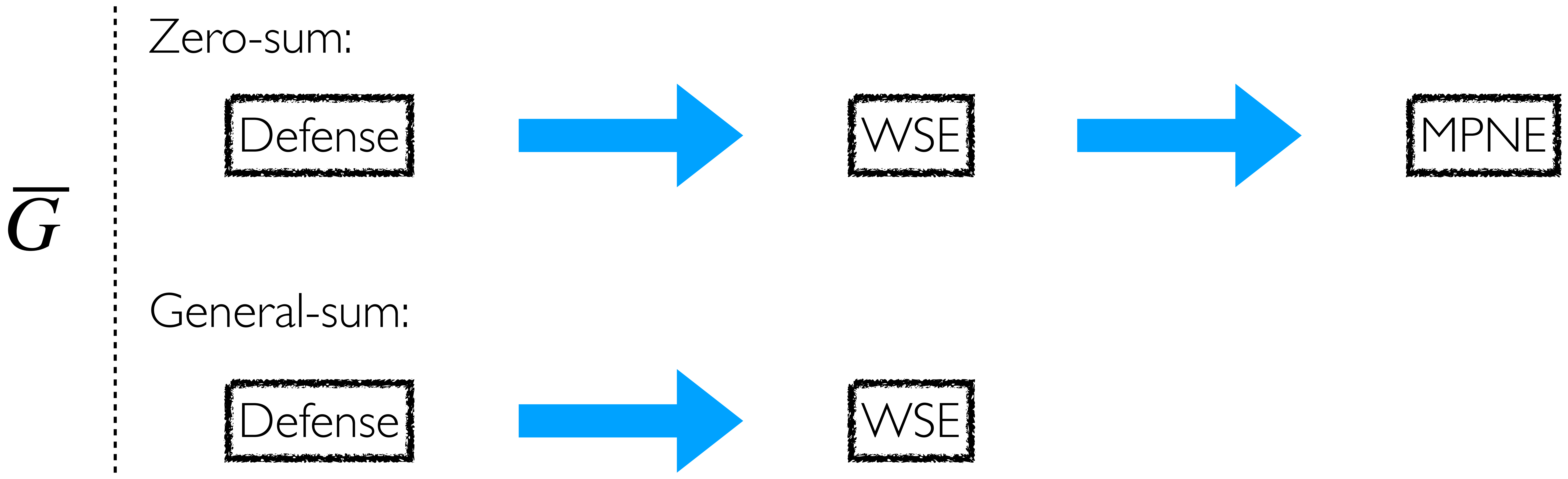
$\overline{G}$

Zero-sum:

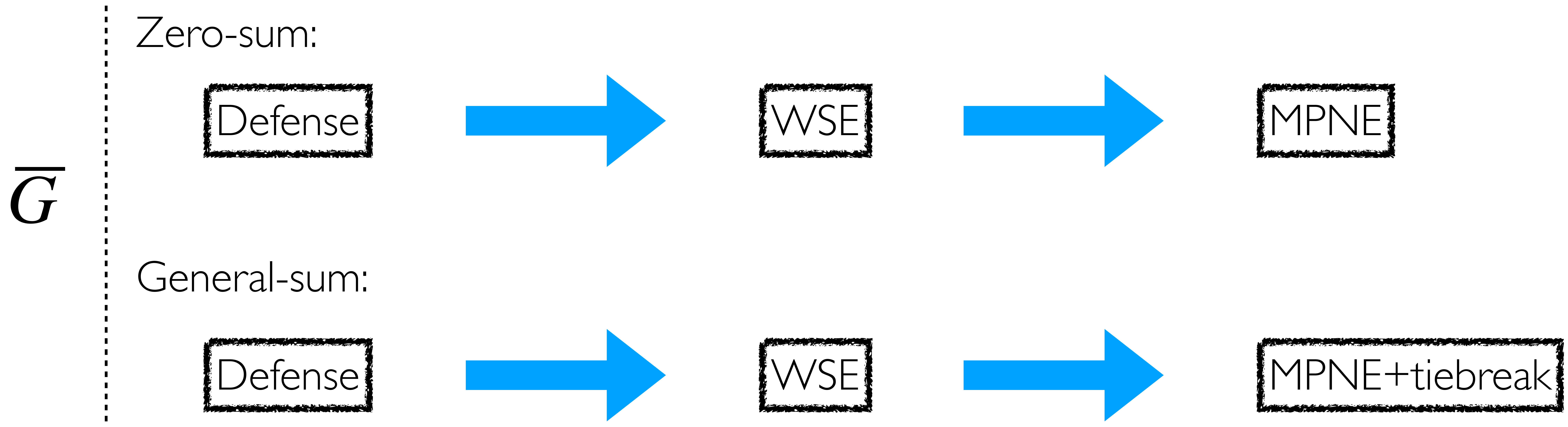
Defense → WSE → MPNE

General-sum:

Efficient Reduction to MARL



Efficient Reduction to MARL



Rollback Algorithm

Rollback Algorithm

Special Case: Action Attacks

Rollback Algorithm

Special Case: Action Attacks

1. Victim determines Attacker's best response to any action a:

$$BR_h(s, a) = \arg\max_{a^\dagger \in \overline{\mathcal{A}}(s,a)} \left[g_h(s, a, r_h(s, a)) + \mathbb{E}_{s' \sim P_h(s,a^\dagger)} \left[V^*_{h+1,2}(s', \pi^*_{h+1}(s'))\right]\right]$$

Rollback Algorithm

Special Case: Action Attacks

1. Victim determines Attacker's best response to any action a:

$$BR_h(s,a) = \arg\max_{a^\dagger \in \overline{\mathcal{A}}(s,a)} \left[g_h(s,a,r_h(s,a)) + \mathbb{E}_{s' \sim P_h(s,a^\dagger)} \left[V^*_{h+1,2}(s', \pi^*_{h+1}(s')) \right] \right]$$

2. Victim picks a based on the worst-case best-response:

$$V^*_{h,1}(s) = \max_{a \in \mathcal{A}} \min_{a^\dagger \in BR_h(s,a)} \left[r_h(s,a^\dagger) + \mathbb{E}_{s' \sim P_h(s,a^\dagger)} \left[V^*_{h+1,1}(s') \right] \right]$$

Guarantees

Guarantees

Theorem: An optimal defense can be computed or learned in polynomial time if observation attacks are not permitted, and

Guarantees

Theorem: An optimal defense can be computed or learned in polynomial time if observation attacks are not permitted, and

1. $\overline{G}$ is zero-sum, or

Guarantees

Theorem: An optimal defense can be computed or learned in polynomial time if observation attacks are not permitted, and

1. $\overline{G}$ is zero-sum, or
2. $\overline{G}$ has finite-horizon.

Guarantees

Theorem: An optimal defense can be computed or learned in polynomial time if observation attacks are not permitted, and

Complete characterization: hard $\iff$ observation attacks!

1. $\overline{G}$ is zero-sum, or
2. $\overline{G}$ has finite-horizon.

Conclusions

Conclusions

- Optimal attacks can be efficiently computed for all attack surfaces.

Conclusions

- Optimal attacks can be efficiently computed for all attack surfaces.
- The defense problem is NP-hard to even approximate.

Conclusions

- Optimal attacks can be efficiently computed for all attack surfaces.
- The defense problem is NP-hard to even approximate.
- Absent observation attacks, optimal defenses can be efficiently computed.

Conclusions

Thank you!

- Optimal attacks can be efficiently computed for all attack surfaces.
- The defense problem is NP-hard to even approximate.
- Absent observation attacks, optimal defenses can be efficiently computed.