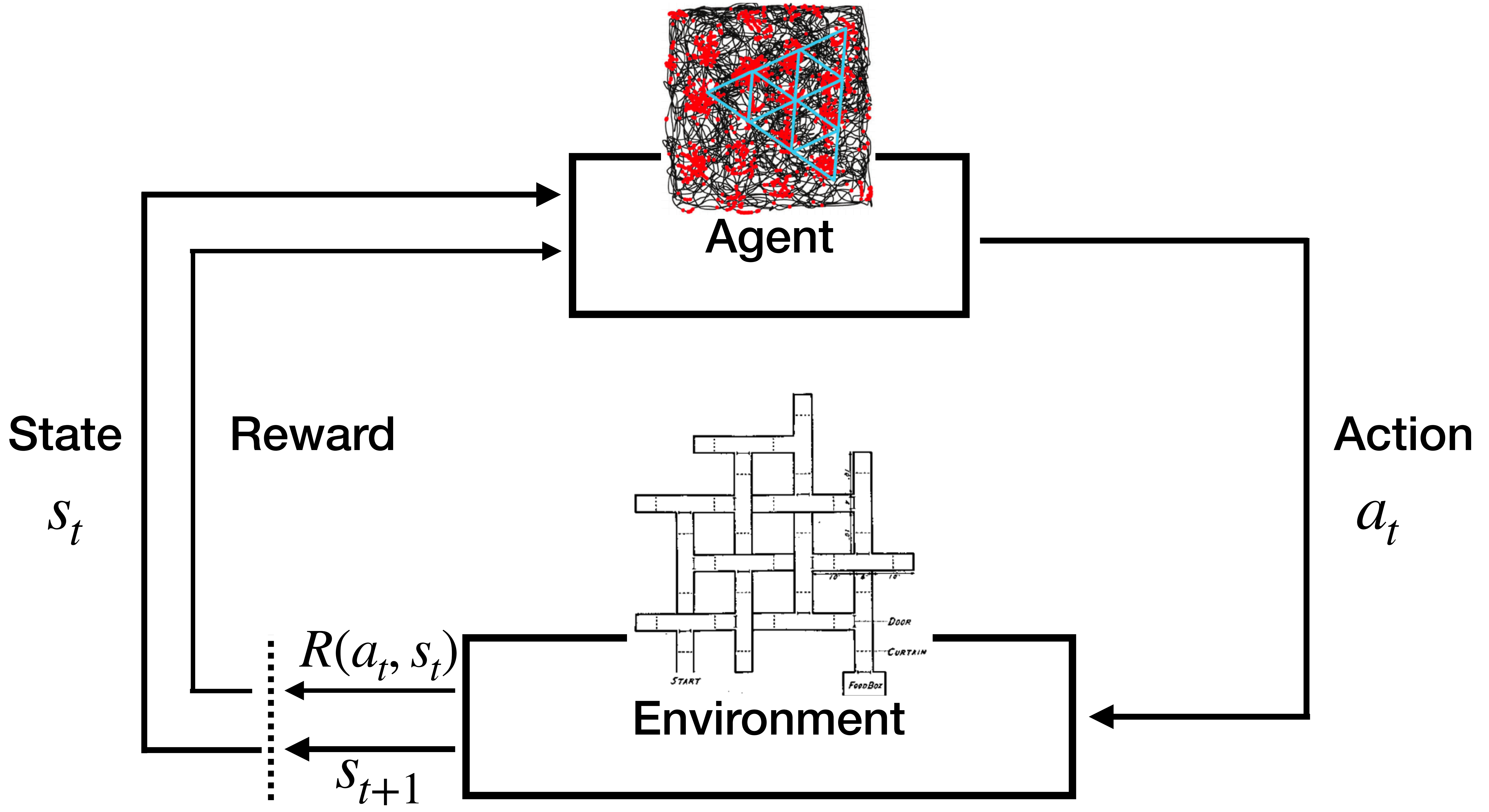


Intro to Reinforcement Learning

Cognitive Maps Seminar

Nov 2nd, 2022

https://hmc-lab.com/Cogmaps.html

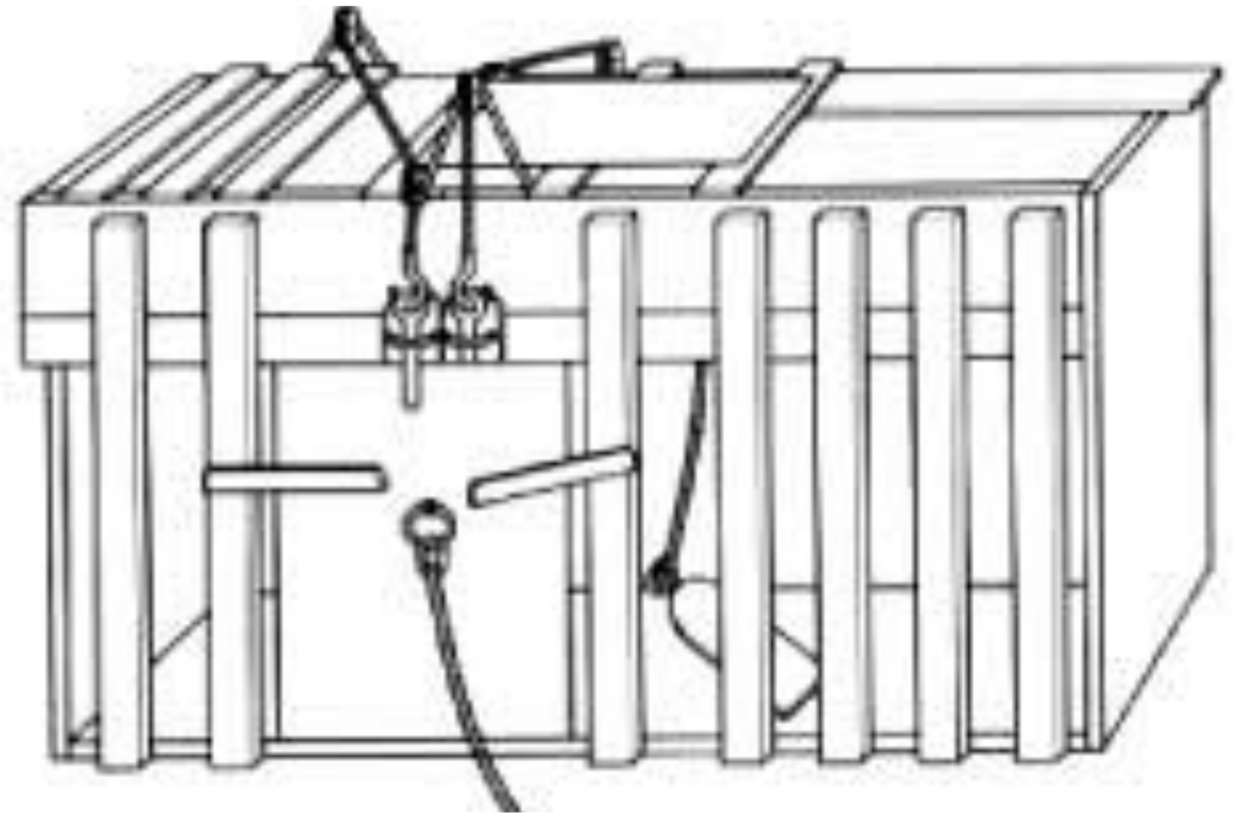
Location changes

- We will use the ground floor seminar whenever possible
- It is bigger and has ventilation
- Check the schedule on the course website for the most up-to-date info

Date	Location	Host	Topic	Required Readings
19. Oct 2022	4th floor	Charley	Introduction to cognitive maps (slides)	Tolman, E. C. (1948). Cognitive maps in rats and men. Psychological review, 55(4), 189.
26. Oct 2022	4th floor	Philipp	What is a cognitive map? An overview of modern neuroscientific discoveries (slides)	Epstein, R. A., Patai, E. Z., Julian, J. B., & Spiers, H. J. (2017). The cognitive map in humans: spatial navigation and beyond. Nature neuroscience, 20(11), 1504-1513.
2. Nov 2022	Ground floor	Charley	Introduction to Reinforcement Learning	Niv, Y. (2009). Reinforcement learning in the brain. Journal of Mathematical Psychology, 53(3), 139–154. [Section 1 only] Dolan, R. J., & Dayan, P. (2013). Goals and habits in the brain. Neuron, 80(2), 312–325. [Focus on generation 3]
9. Nov 2022	4th floor	Philipp	Neuroscience of RL	Lee, D., Seo, H., & Jung, M. W. (2012). Neural basis of reinforcement learning and decision making. Annual review of neuroscience, 35, 287.
16. Nov 2022	Ground floor	Nir Moneta (MPI Berlin)	Cognitive maps beyond spatial stimuli	Doeller, C. F., Barry, C., & Burgess, N. (2010). Evidence for grid cells in a human memory network. Nature, 463(7281), 657-661.
23. Nov 2022	Ground floor	Noémi	From maps to behavior and back again	Stachenfeld, K. L., Botvinick, M. M., & Gershman, S. J. (2017). The hippocampus as a predictive map. Nature neuroscience, 20(11), 1643-1653.
30. Nov 2022	Ground floor	Georgy Antonov (MPI BC)	Linking memory and navigation	Eichenbaum, H. (2017). On the integration of space, time, and memory. Neuron, 95(5), 1007-1018.
7. Dec 2022	4th floor	Philipp	Student led presentation	See list of recommended papers
14. Dec 2022	Ground floor	Philipp	Student led presentation 2	

The story so far …

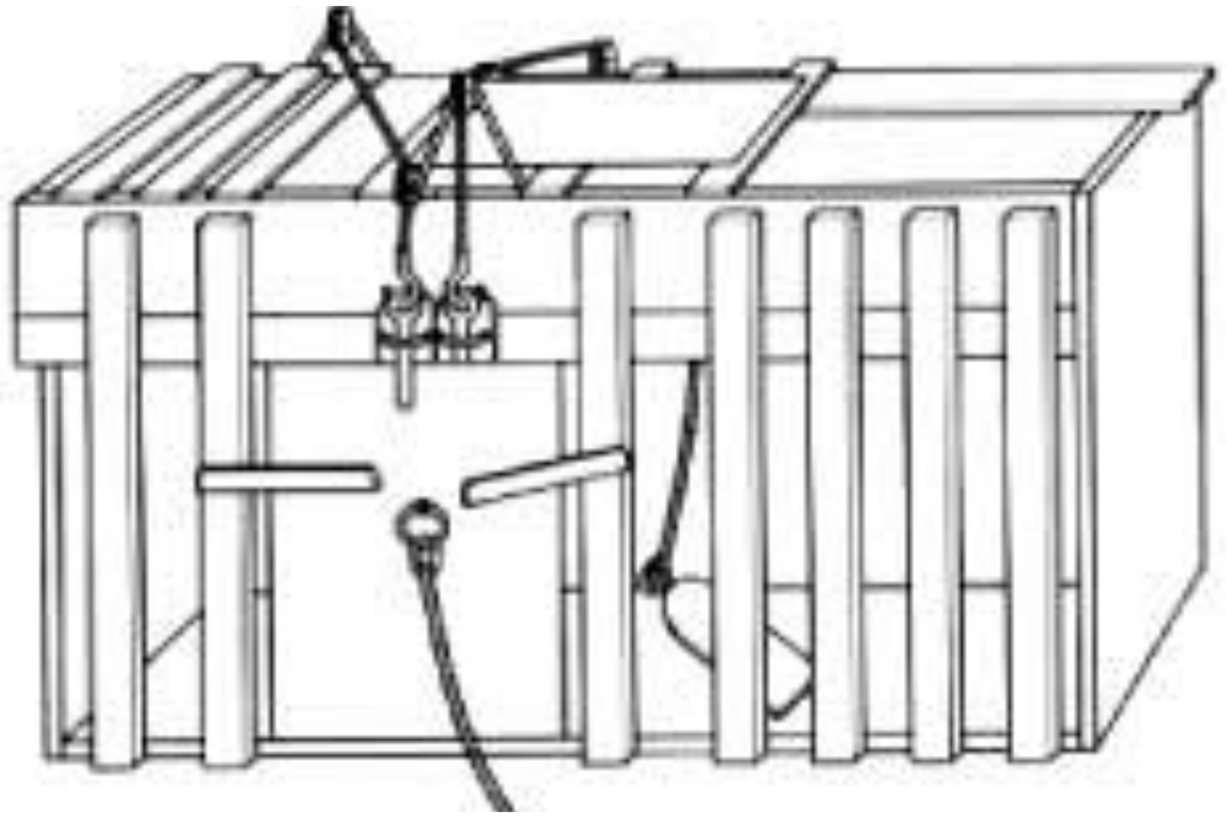
Thorndike's (1898) Law of Effect



Puzzle Box

Thorndike's (1898) Law of Effect

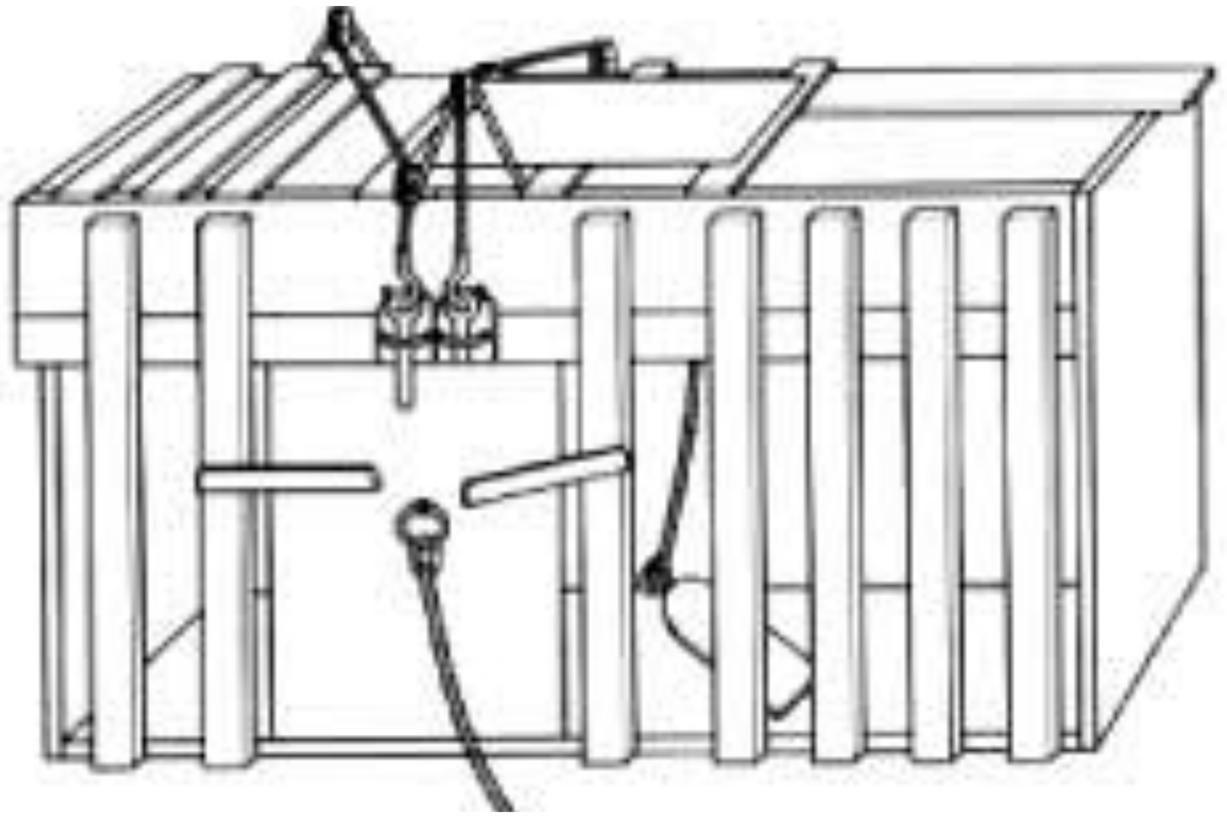
Cat



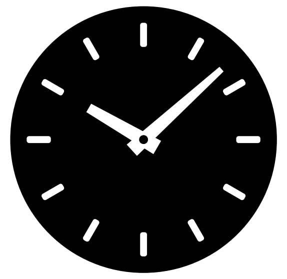
Puzzle Box

Thorndike’s (1898) Law of Effect

Cat



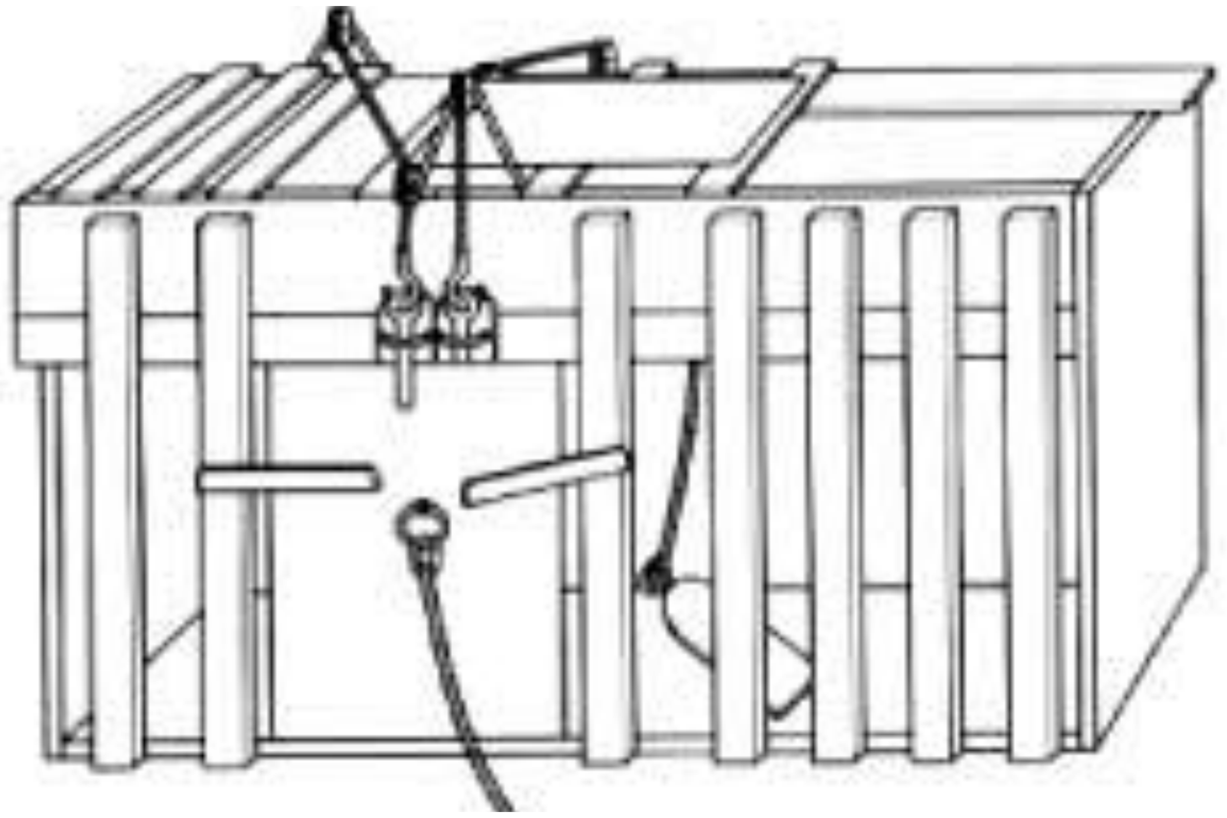
Puzzle Box



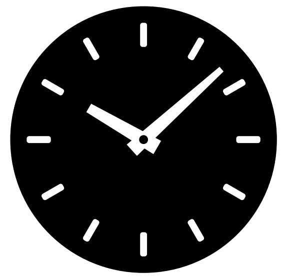
Time to escape

Thorndike’s (1898) Law of Effect

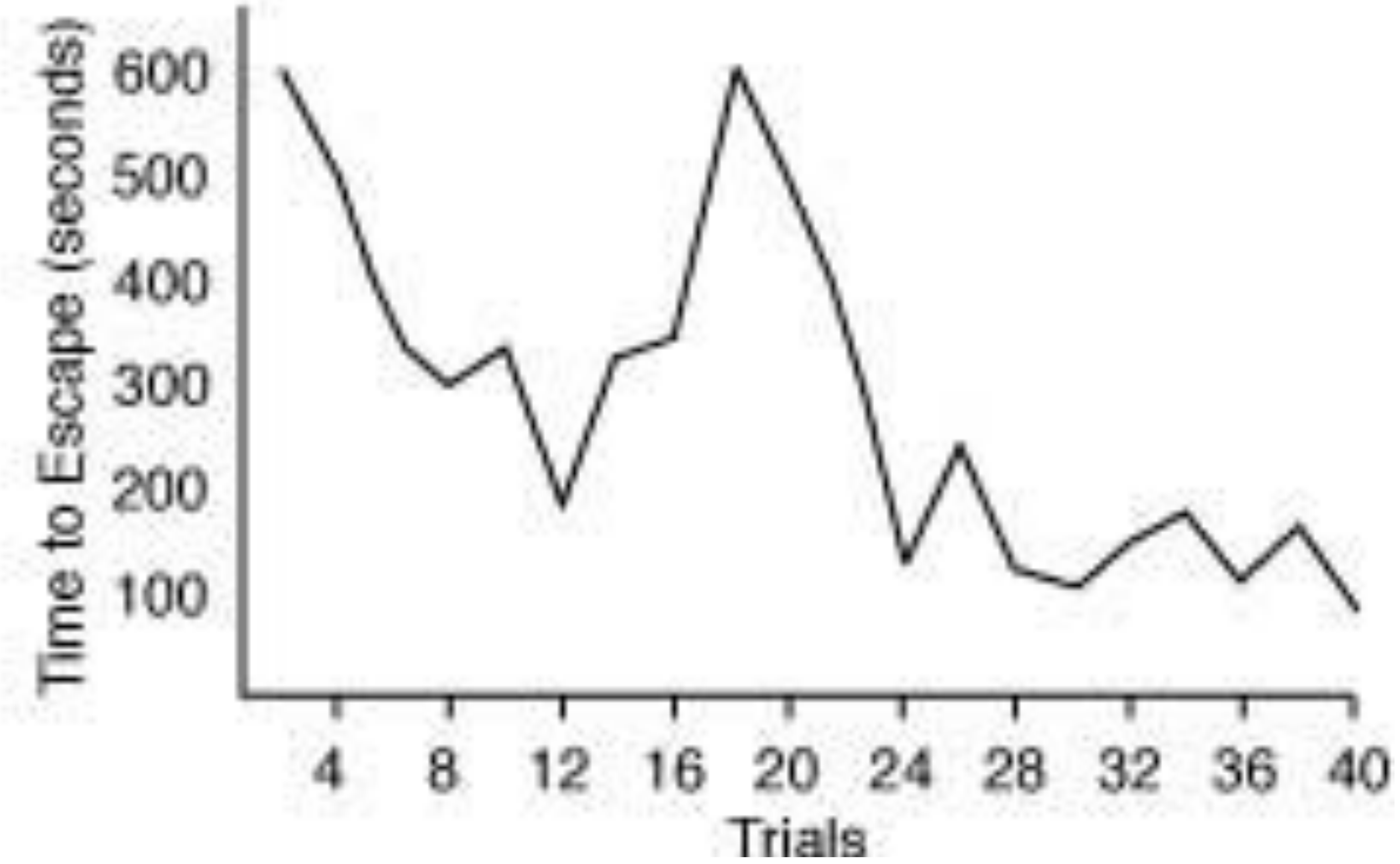
Cat



Puzzle Box



Time to escape

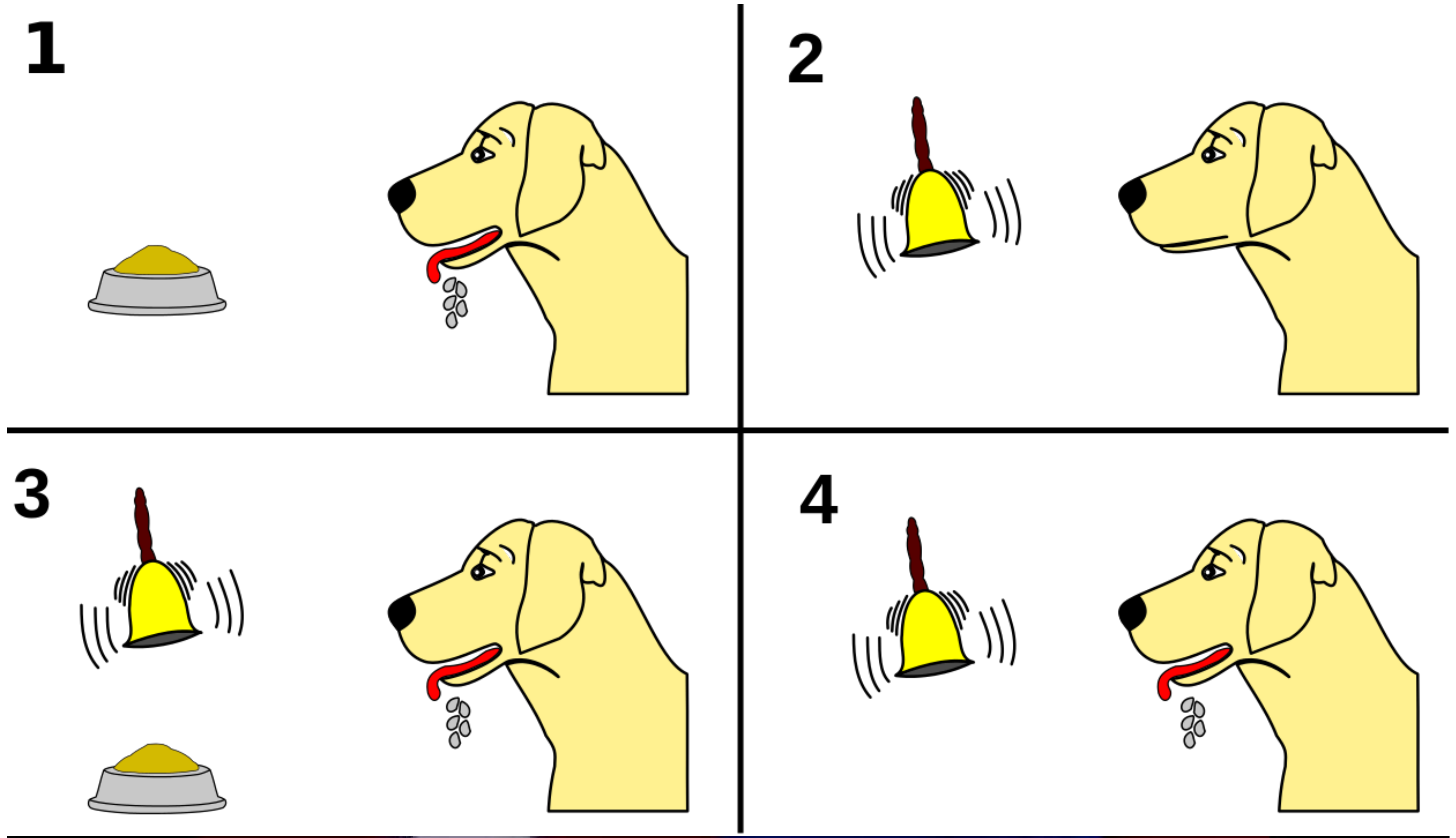


Actions associated with satisfaction are strengthened, while those associated with discomfort become weakened.

Classical and Operant Conditioning

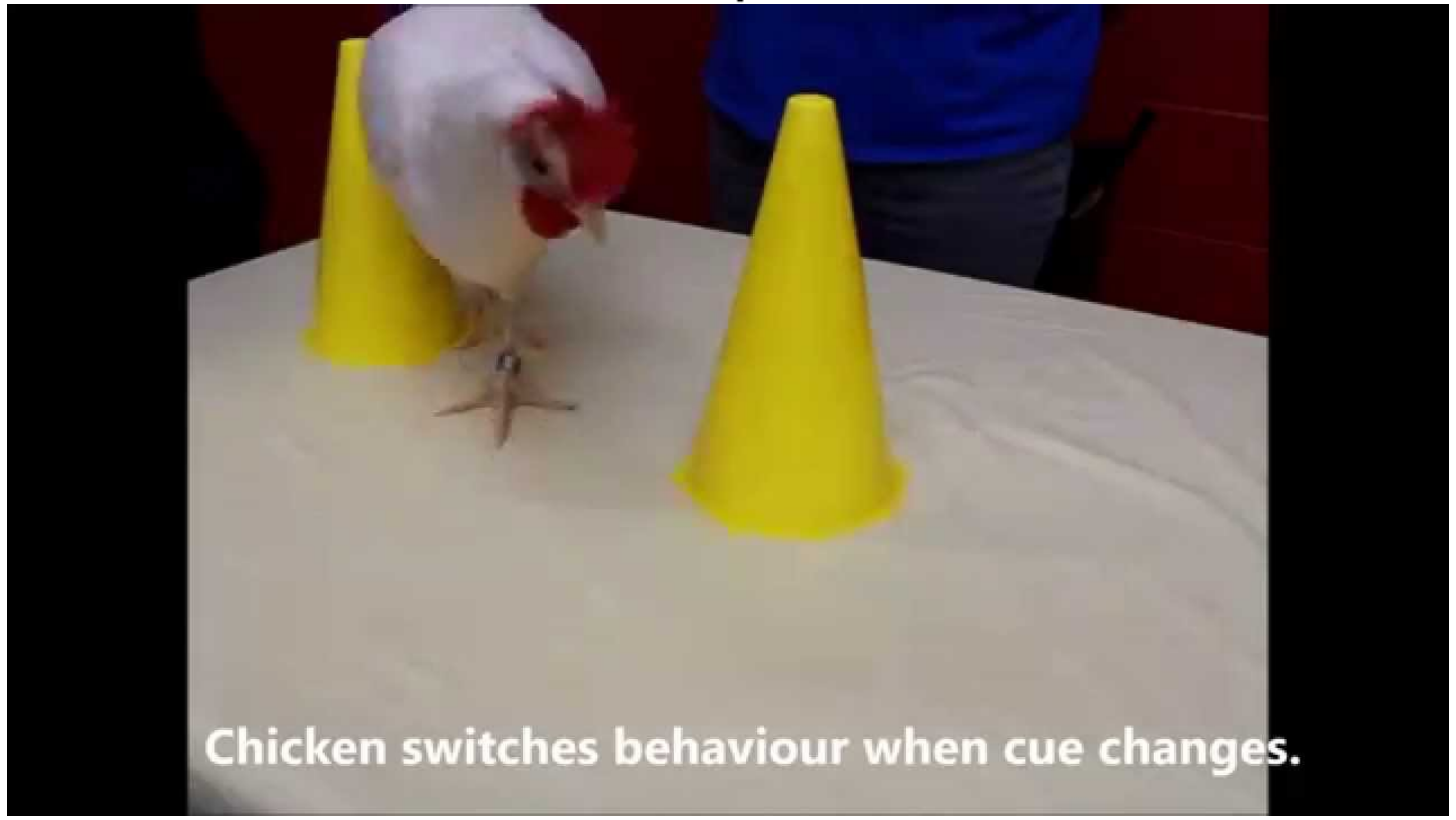
Classical Condition (Pavlov, 1927)

Learning as the *passive* coupling of stimulus (bell ringing) and response (salivation), anticipating future rewards



Operant Condition (Skinner, 1938)

Skinner (1938): Learning as the *active* shaping of behavior in response to rewards or punishments

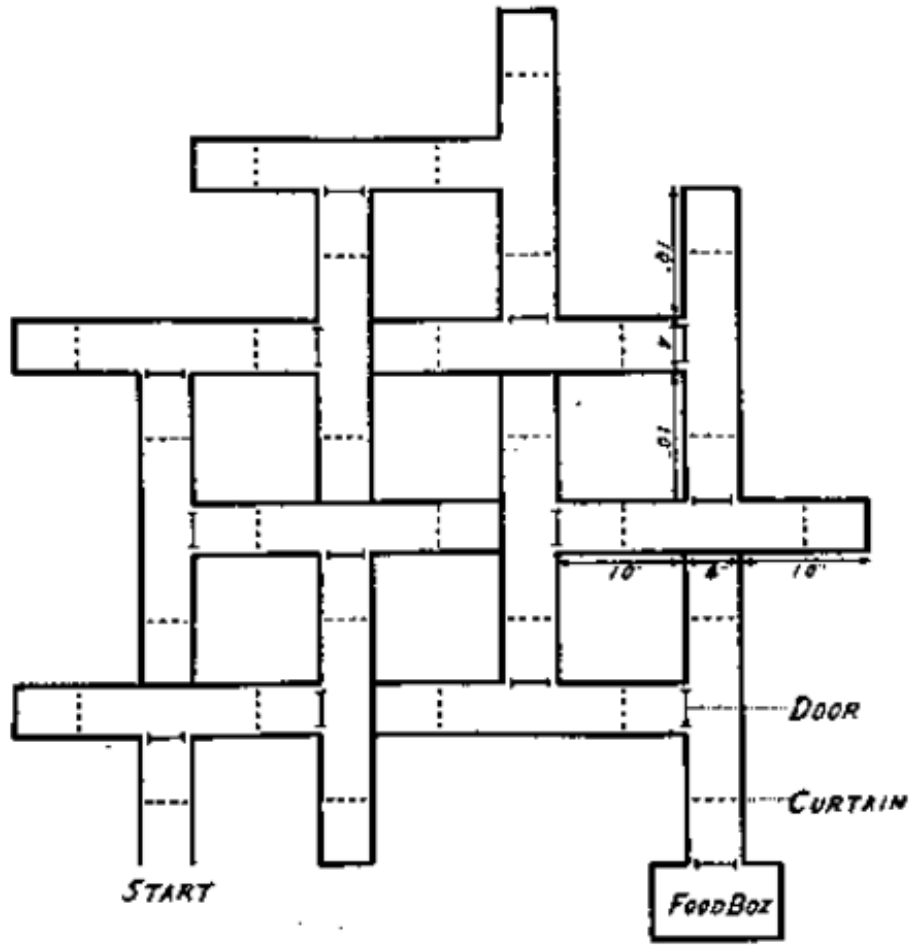


Tolman and Cognitive maps

- Learning is not just a telephone switchboard connecting incoming sensory signals to outgoing responses (S-R Learning)
- Rather, “latent learning” establishes something like a “field map of the environment” gets etablished (S-S learning)

Stimulus-Response (S-R) Learning

Stimulus-Stimulus (S-S) Learning



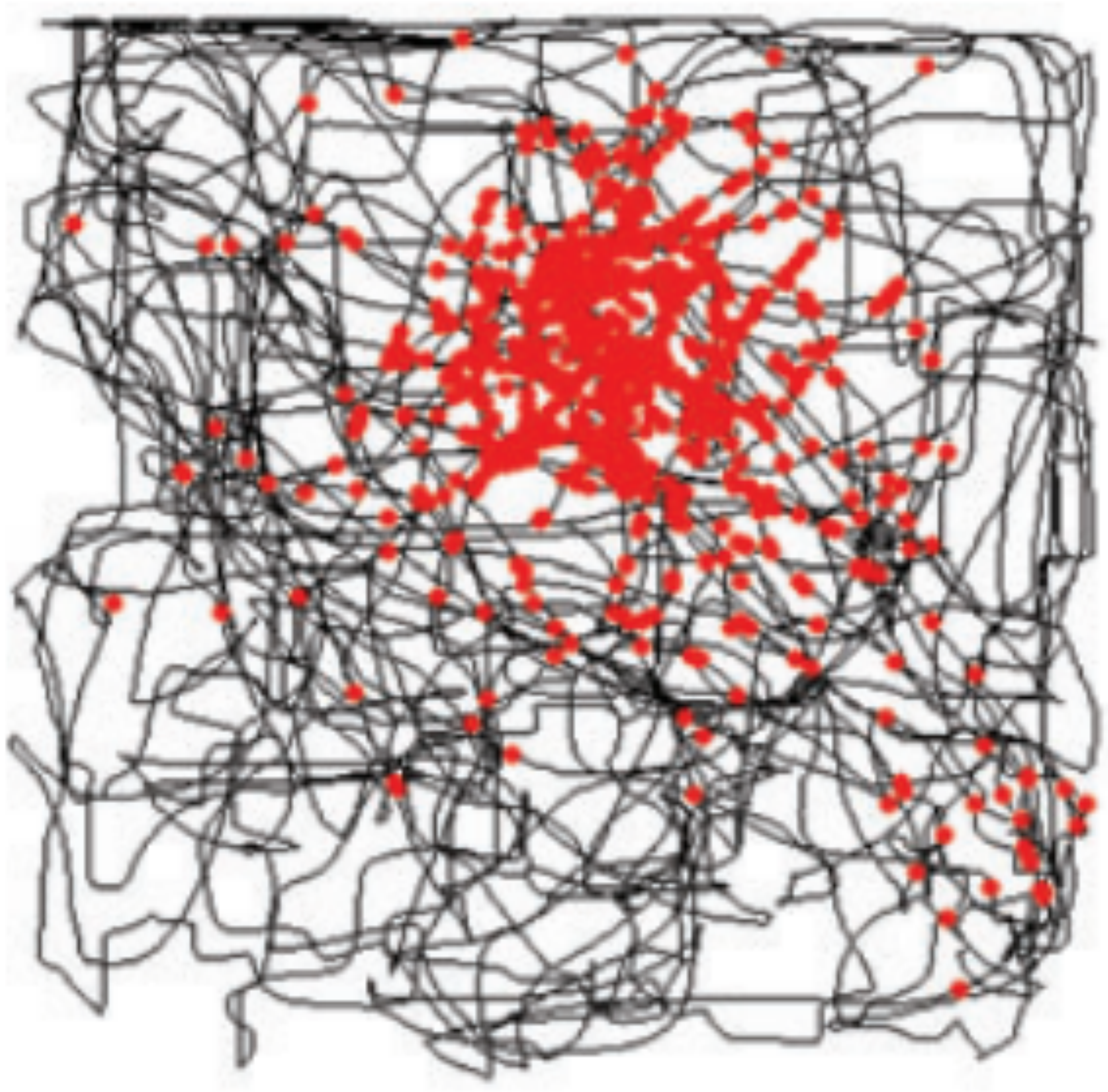
Plan of maze
14-Unit T-Alley Maze

FIG. 1

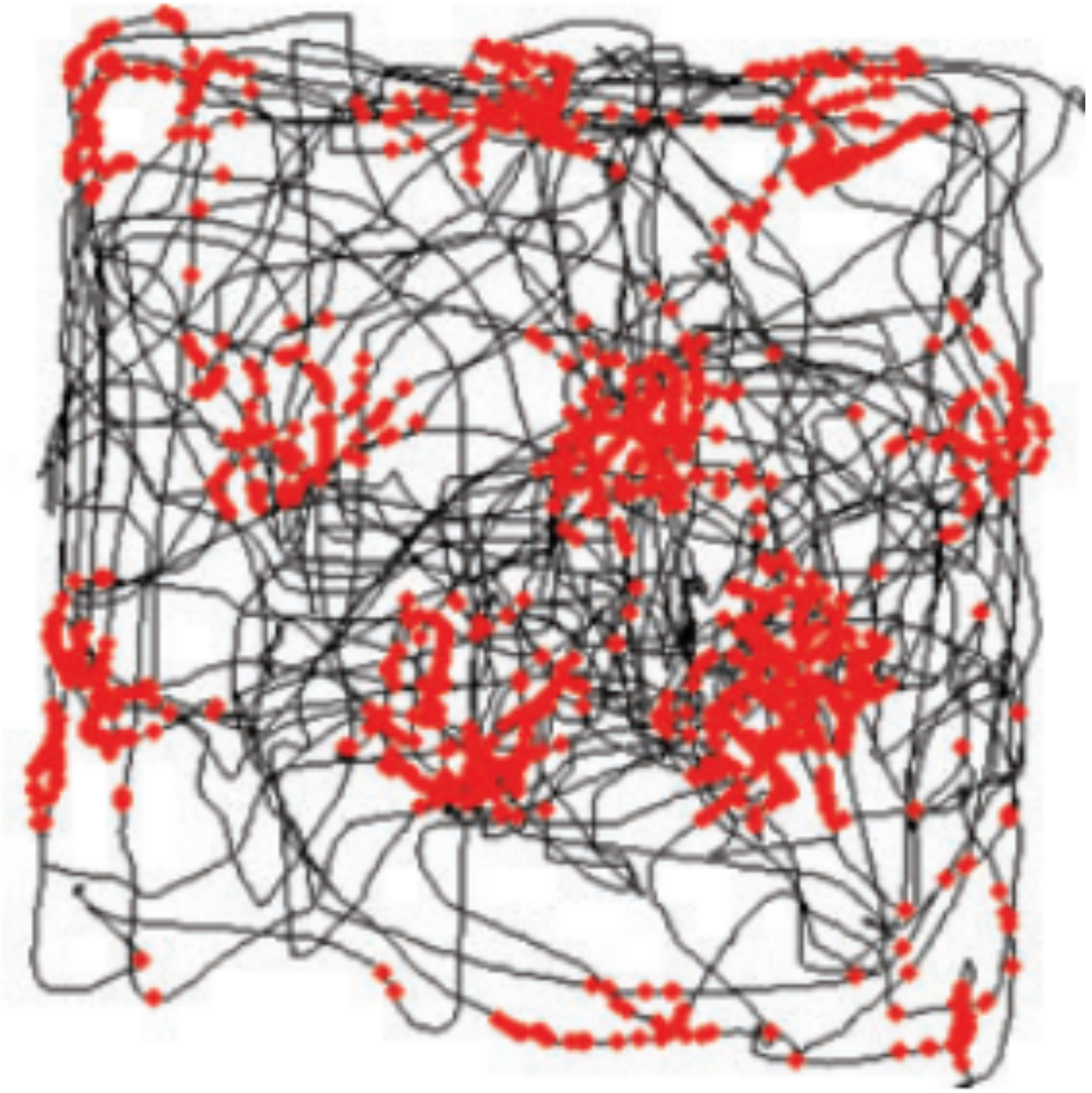
(From M. H. Elliott, The effect of change of reward on the maze performance of rats. *Univ. Calif. Publ. Psychol.*, 1928, 4, p. 20.)

Cognitive maps in biological brains

Place cells in the hippocampus

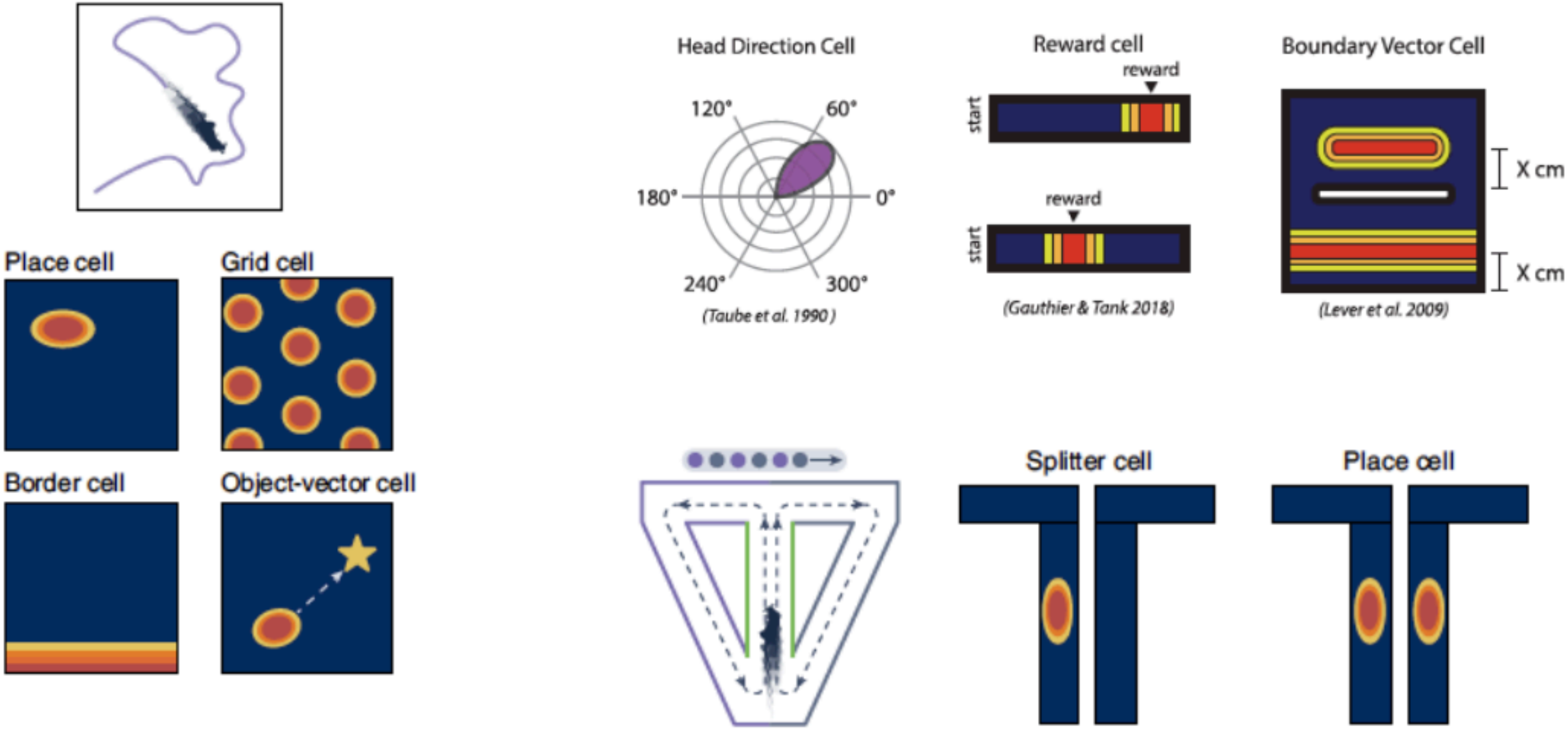


Grid cells in the medial entorhinal cortex



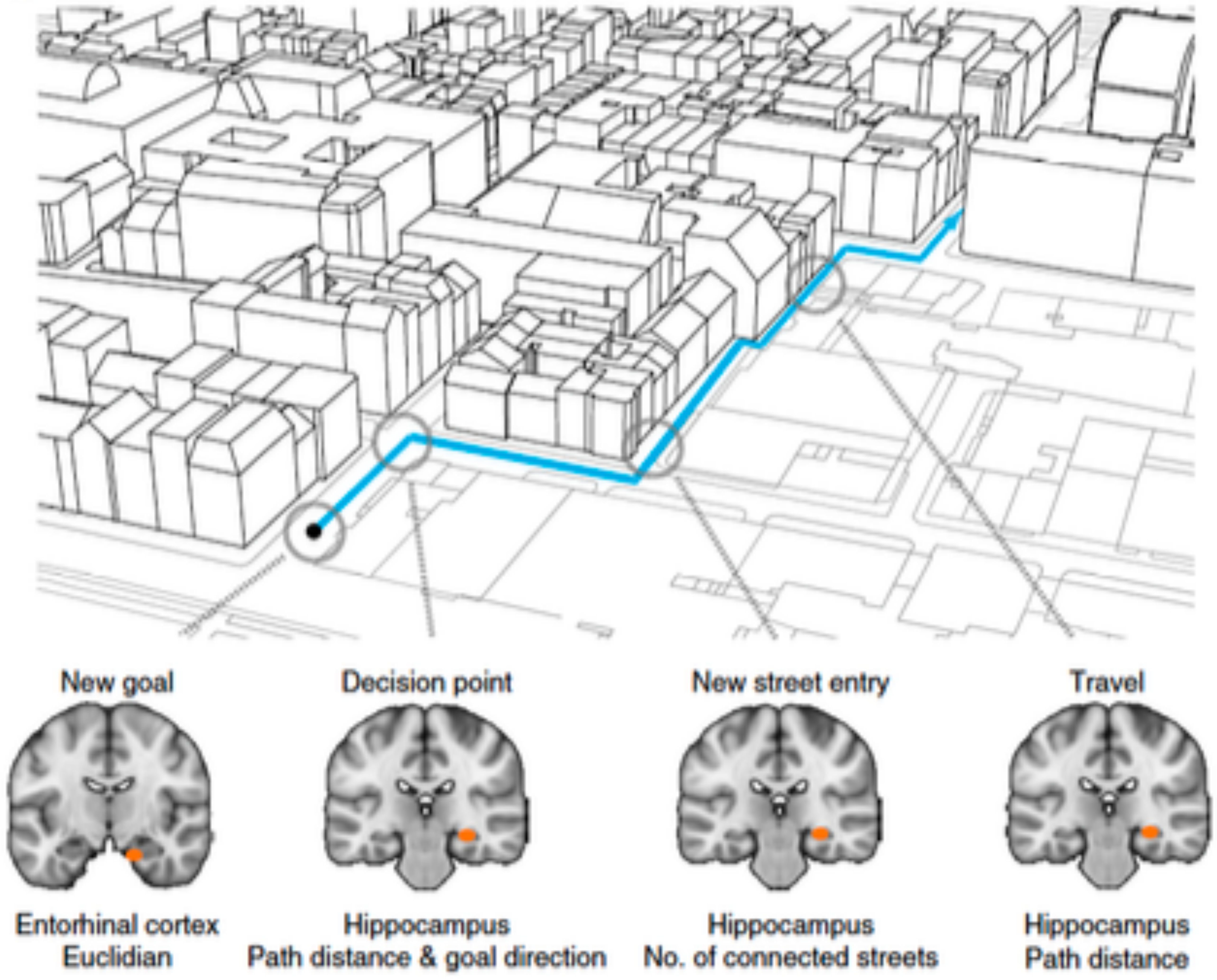
Moser et al., (*Ann Rev Neuro* 2008)

“Hippocampal zoo”



Behrens et al., (*Neuron* 2018)
Whittington et al,. (*Nat Neuro* 2022)

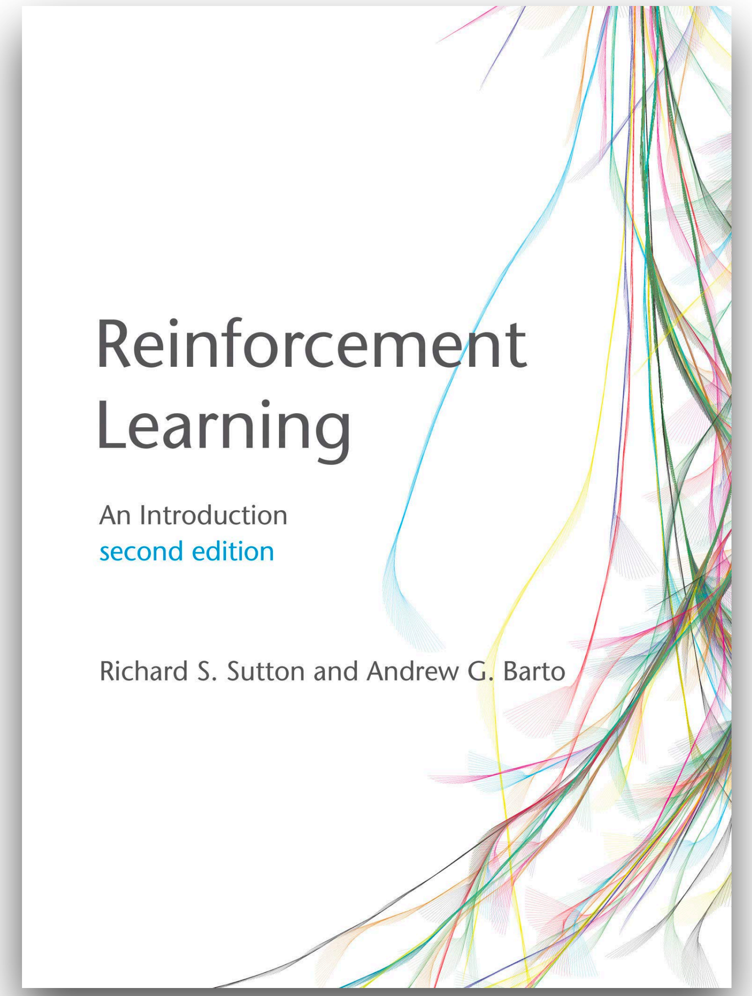
Cognitive maps support navigation and planning



Agenda for today: From Tolman to Reinforcement Learning

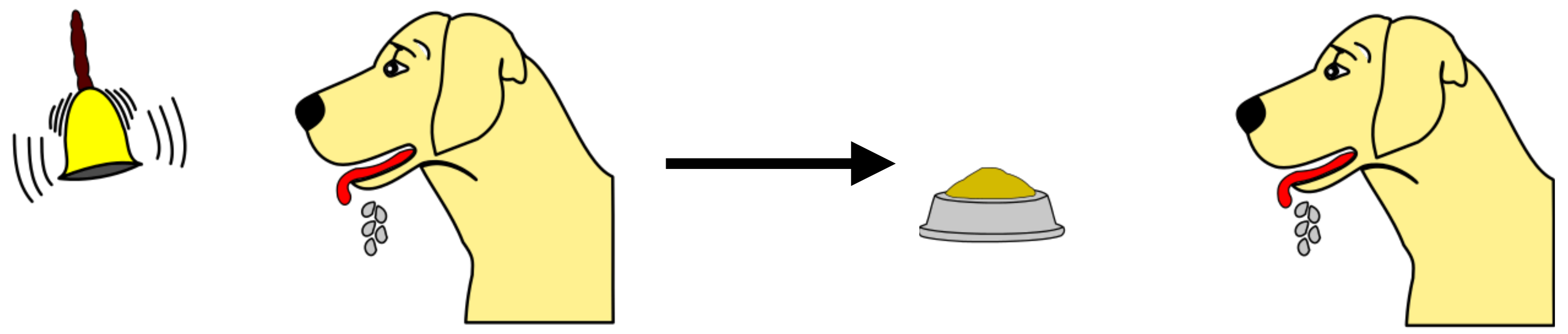
- **Part 1**: Introduce RL framework, origins, and terminology (Niv, 2009)

- **Part 2:** Model-free vs. model-based RL (Dolan & Dayan, 2013)

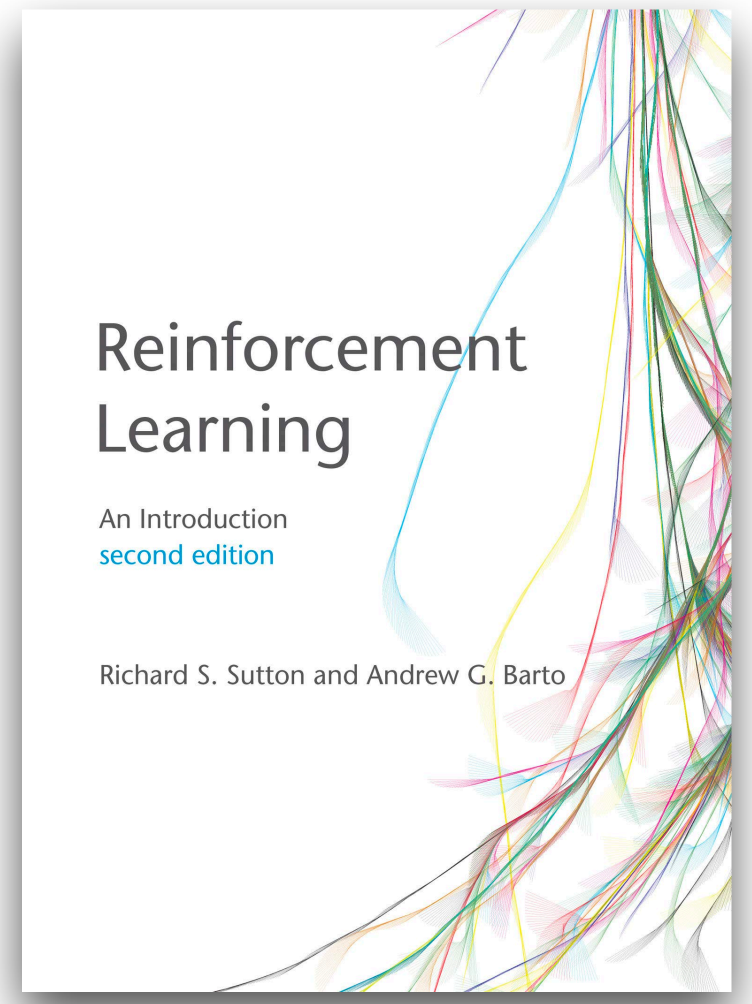


Reinforcement Learning

Pavlovian (classical) conditioning

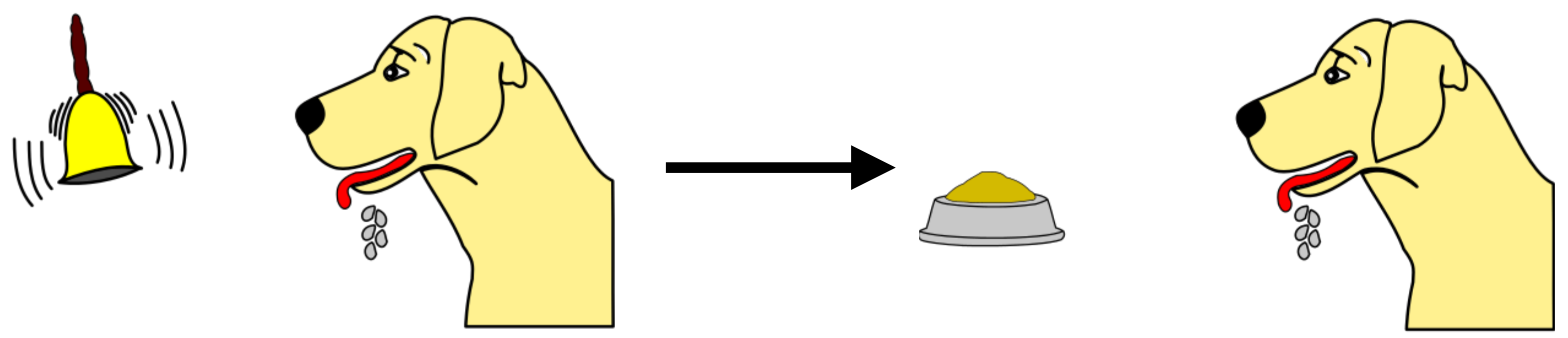


Learn which environmental cues *predict* reward

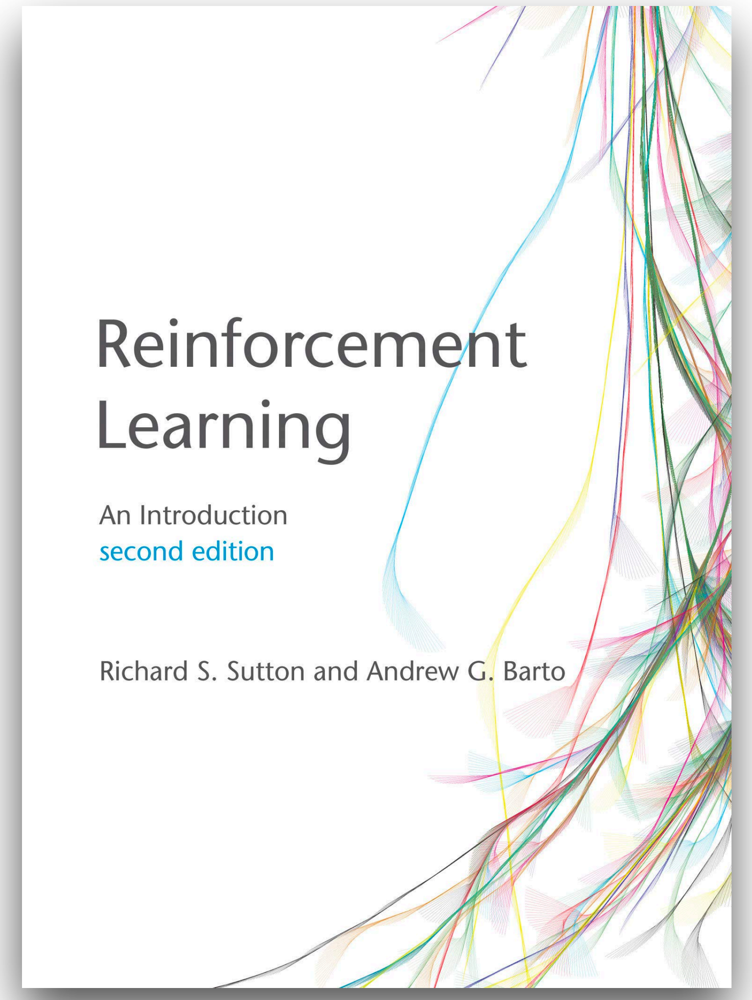


Reinforcement Learning

Pavlovian (classical) conditioning

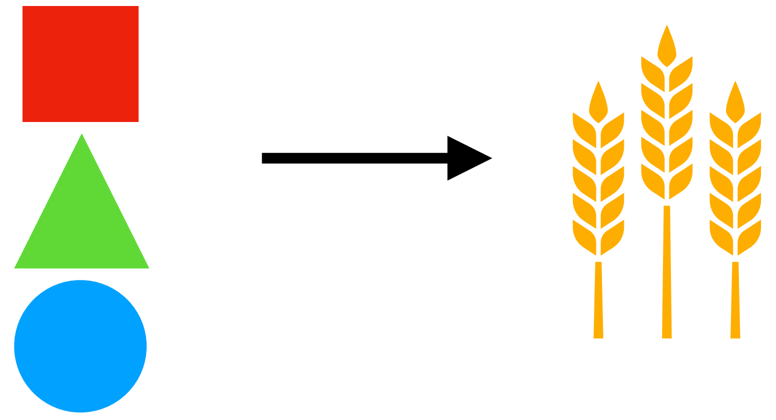


Learn which environmental cues *predict* reward



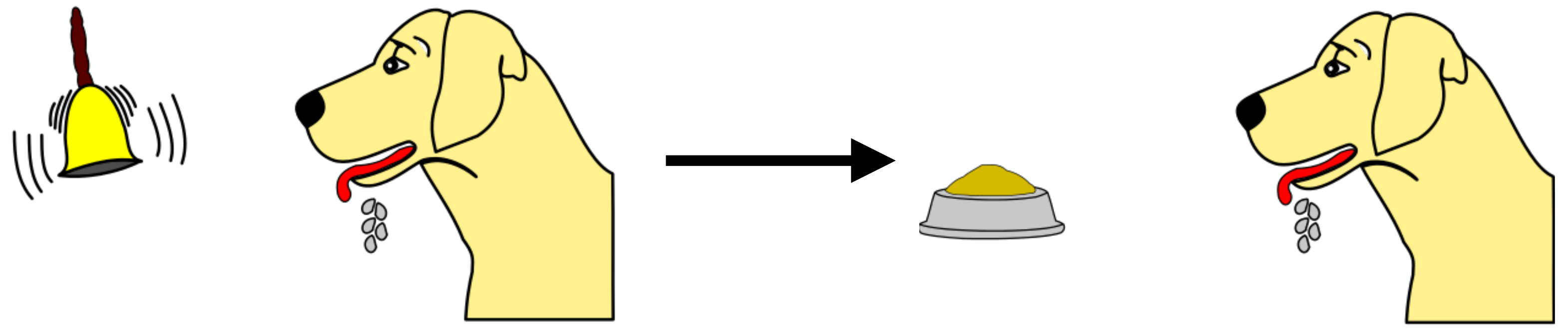
Reinforcement Learning

Operant (instrumental) conditioning

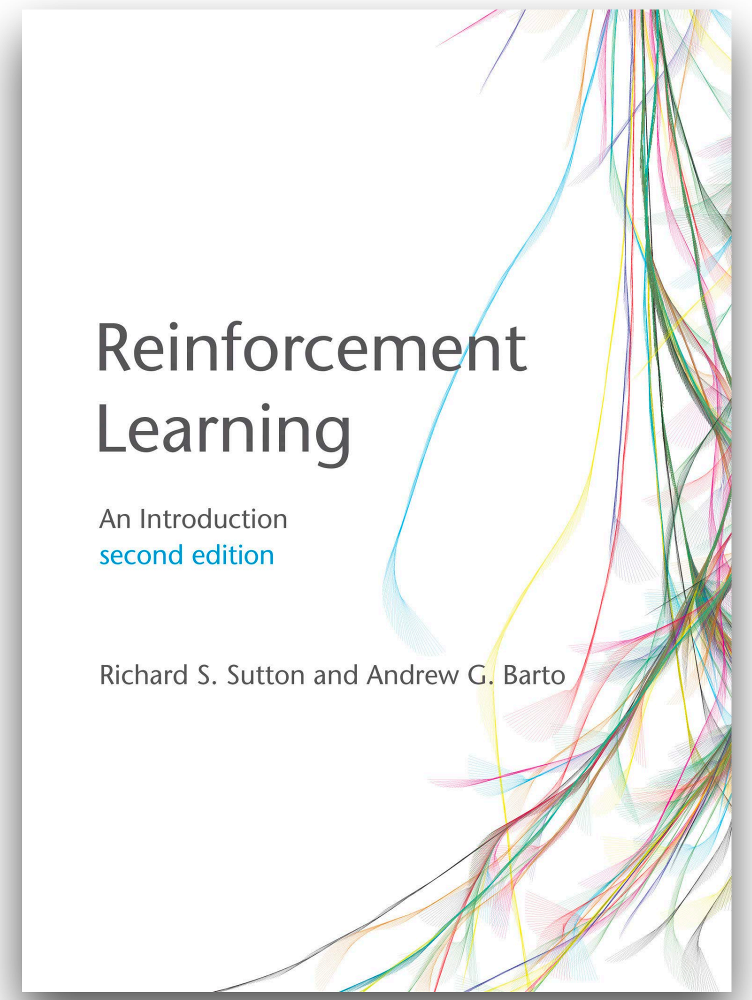


Learn which actions *predict* reward

Pavlovian (classical) conditioning

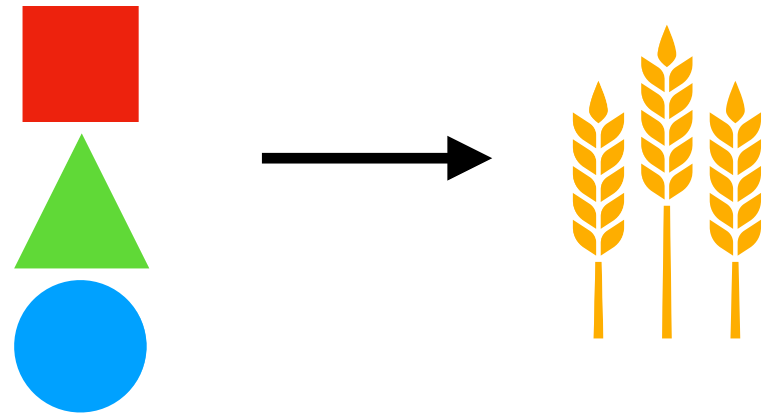


Learn which environmental cues *predict* reward



Reinforcement Learning

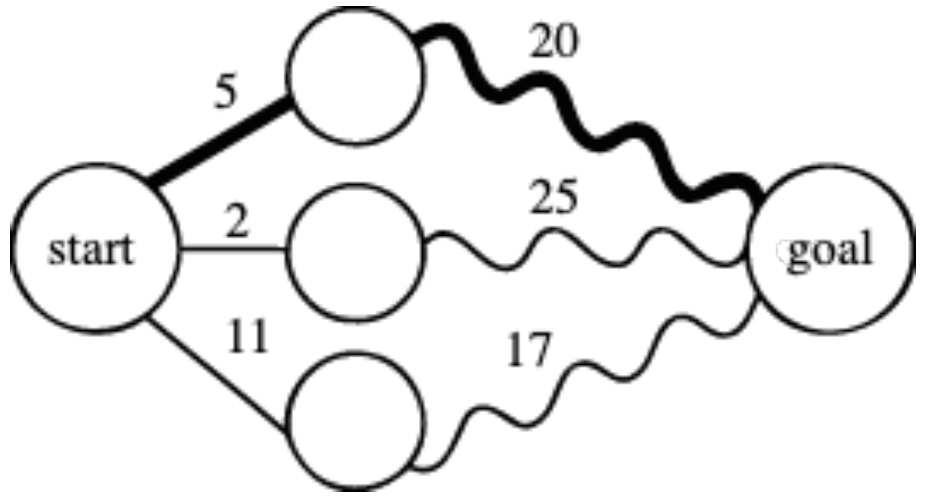
Operant (instrumental) conditioning



Learn which actions *predict* reward

Neuro-dynamic programing
Bertsekas & Tsitsiklis (1996)

Stochastic approximations to dynamic programing problems



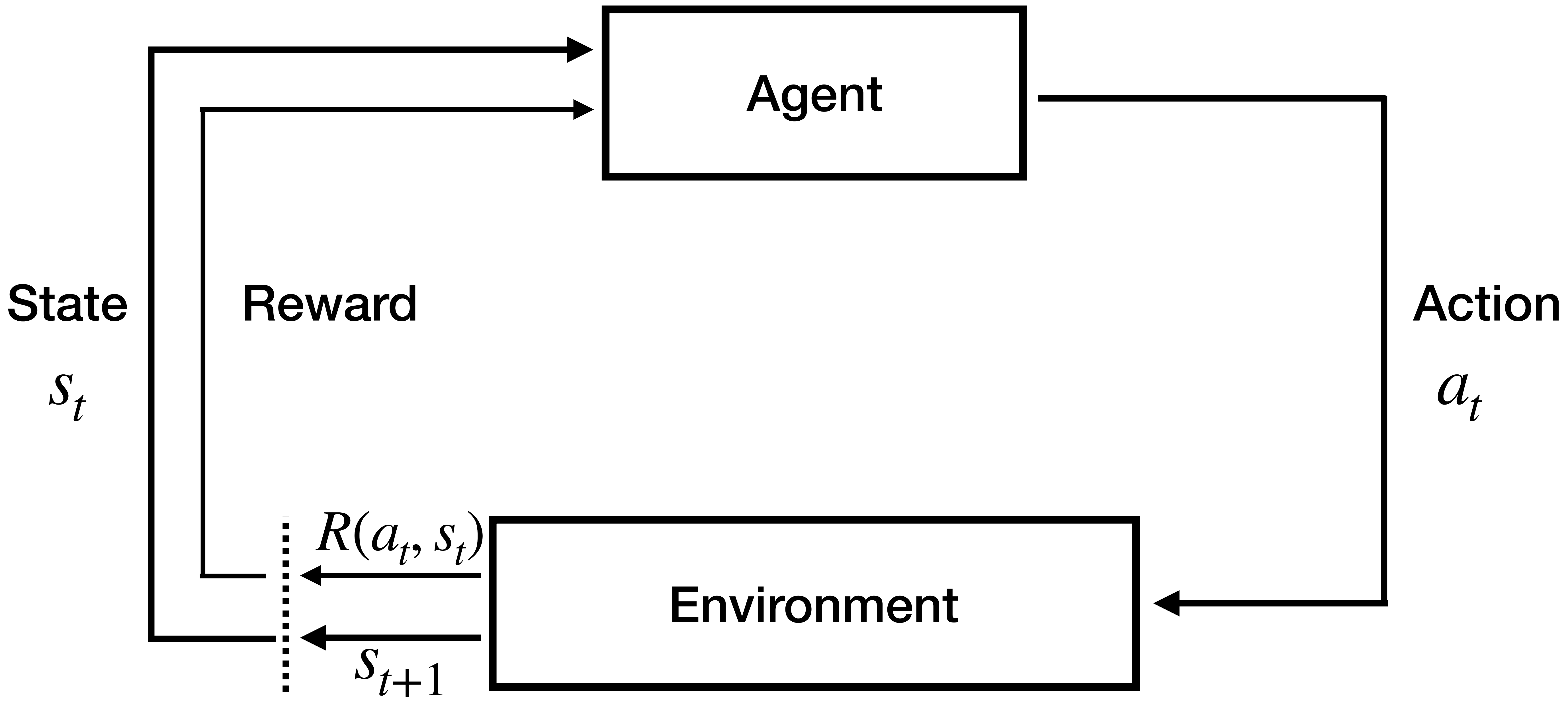
Reinforcement Learning

The Agent:

- Iteratively selects actions a_t
- Receives feedback from the environment in terms of new states s_{t+1} and rewards $R(a_t, s_t)$
- Updates internal representations

The Environment:

- governs the transition between states $s_t \rightarrow s_{t+1}$
- provides rewards $R(a_t, s_t)$



Sutton and Barto (2018 [1998])


Agent

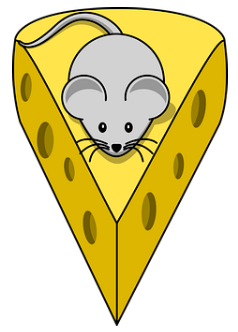
Agent

- **Experiences Rewards**

- **Learns a Policy**

Agent

- **Experiences Rewards**

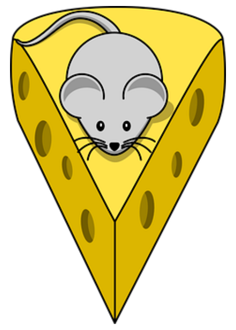


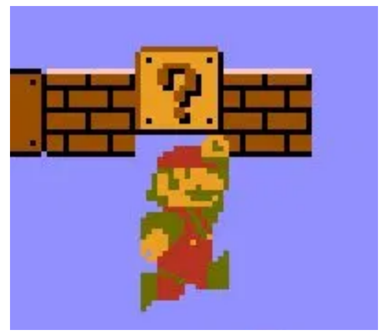
  - How good is a given state? $V(s_t)$

- **Learns a Policy**

Agent

- **Experiences Rewards**
  - How good is a given state? $V(s_t)$
  - How good is a state-action pair? $Q(s_t, a_t)$

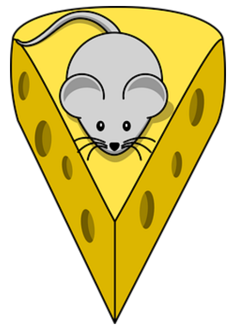


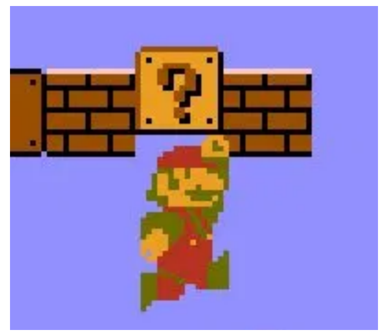


- **Learns a Policy**

Agent

- **Experiences Rewards**
  - How good is a given state? $V(s_t)$
  - How good is a state-action pair? $Q(s_t, a_t)$
  - How good is a *trajectory* $\tau = (s_0, a_0, s_1, a_1, \ldots)$?

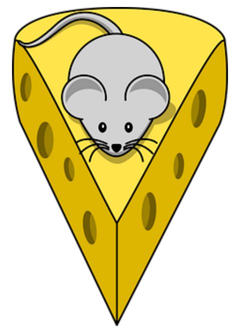


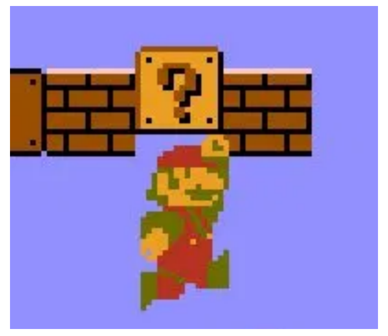


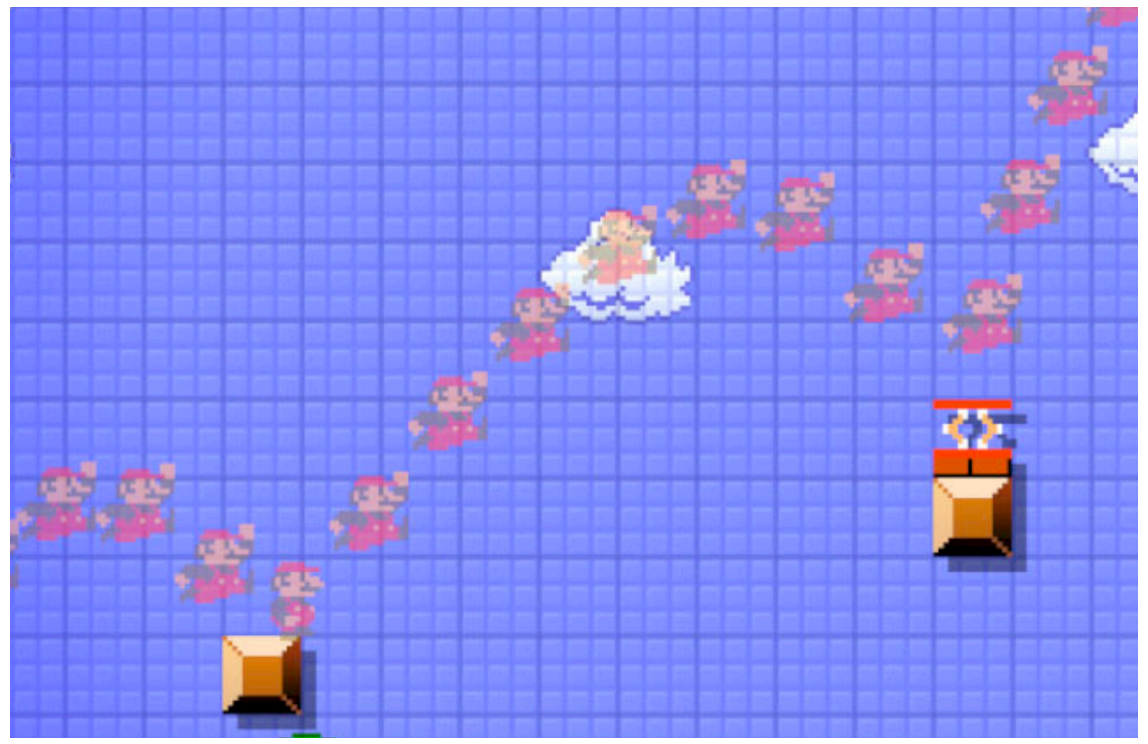
- **Learns a Policy**

Agent

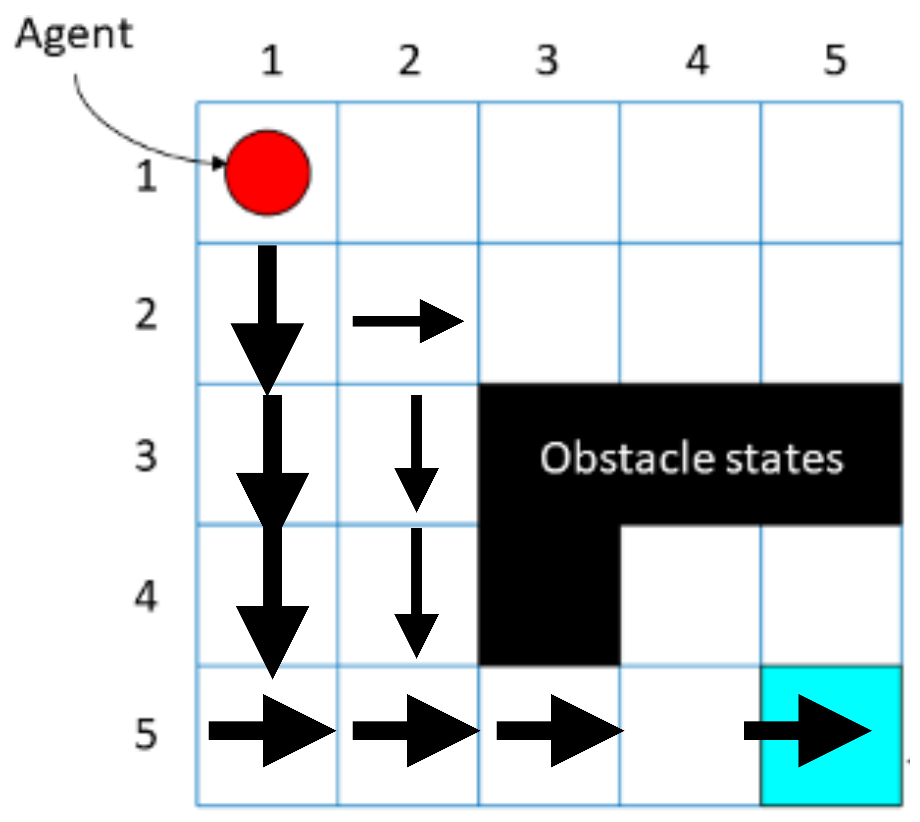
- **Experiences Rewards**
  - How good is a given state? $V(s_t)$
  - How good is a state-action pair? $Q(s_t, a_t)$
  - How good is a *trajectory* $\tau = (s_0, a_0, s_1, a_1, \ldots)$?







- **Learns a Policy**
  - π defines how to act, where $\pi(a \mid s)$ is the probability of selecting action a in state s
  - sample actions from the policy $a_t \sim \pi$



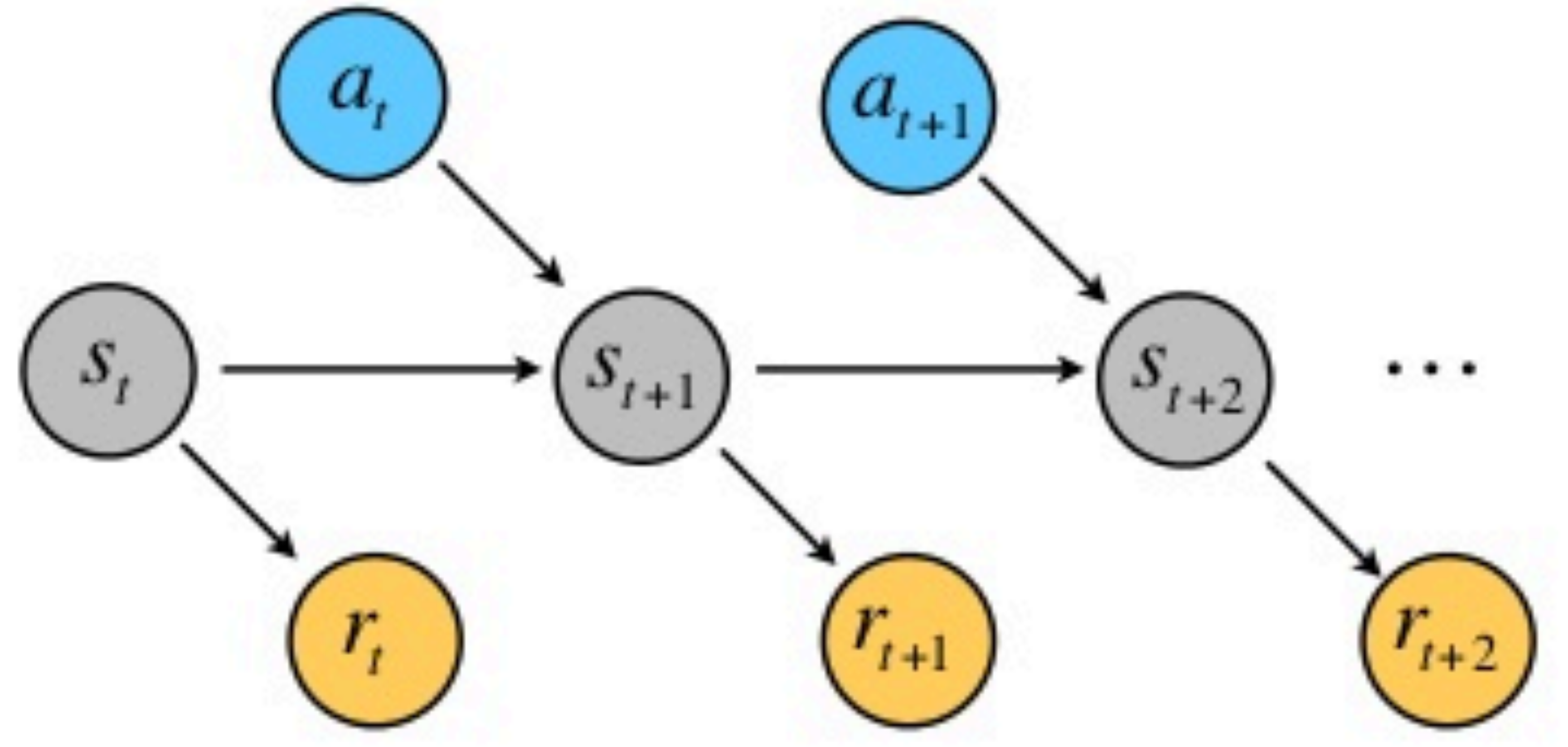
Grid World

Environment

actions

states

reward



Markov Decision Process (MDP)

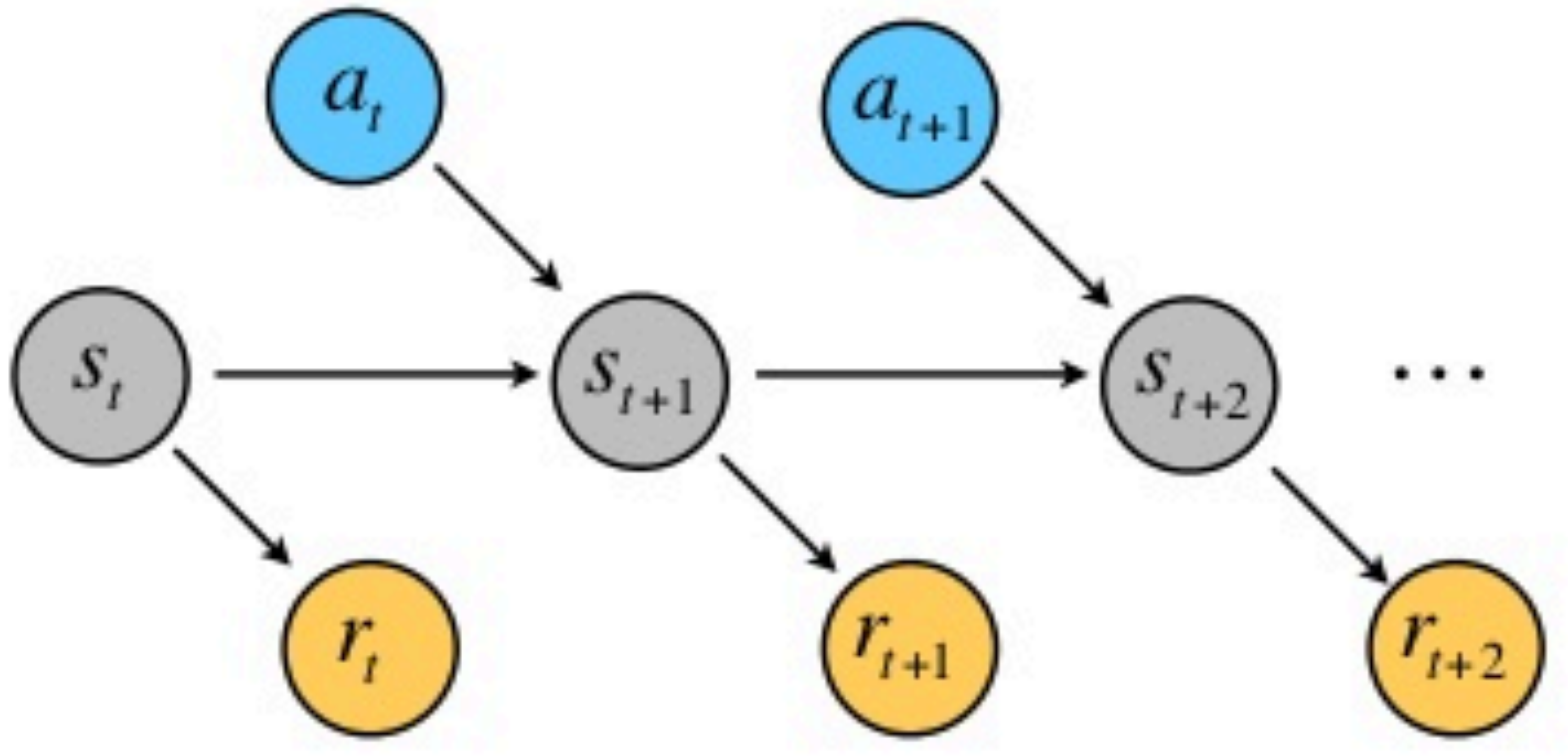
- Simplifying assumption that the system is fully defined by only the previous state (i.e., Markov Principle): $P(s_{t+1} \mid s_t, a_t)$

Environment

actions

states

reward

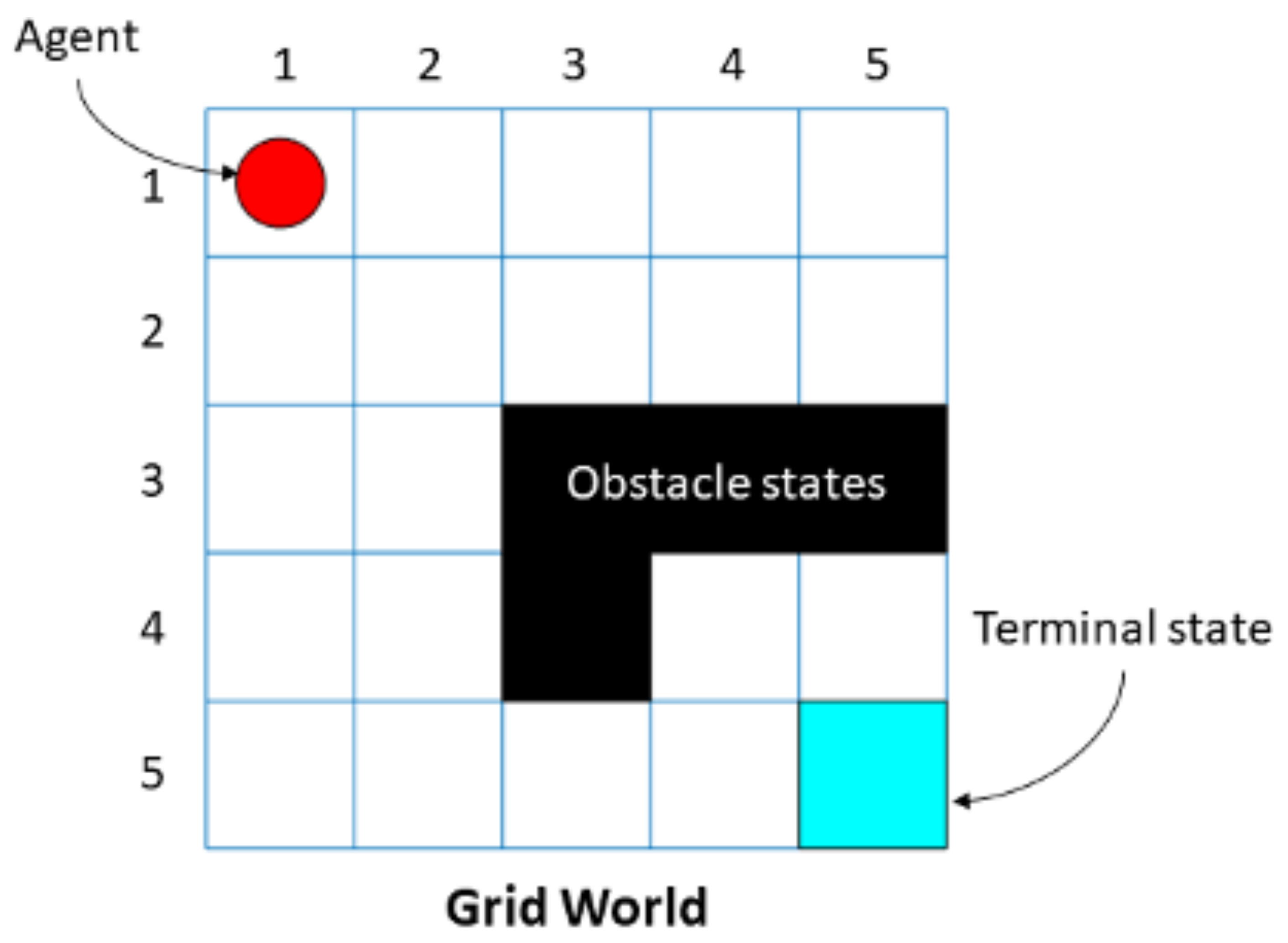


Markov Decision Process (MDP)

- Simplifying assumption that the system is fully defined by only the previous state (i.e., Markov Principle): $P(s_{t+1} \mid s_t, a_t)$

What are the states?

- Discrete locations, pixels on a screen, a set of feature values, etc…

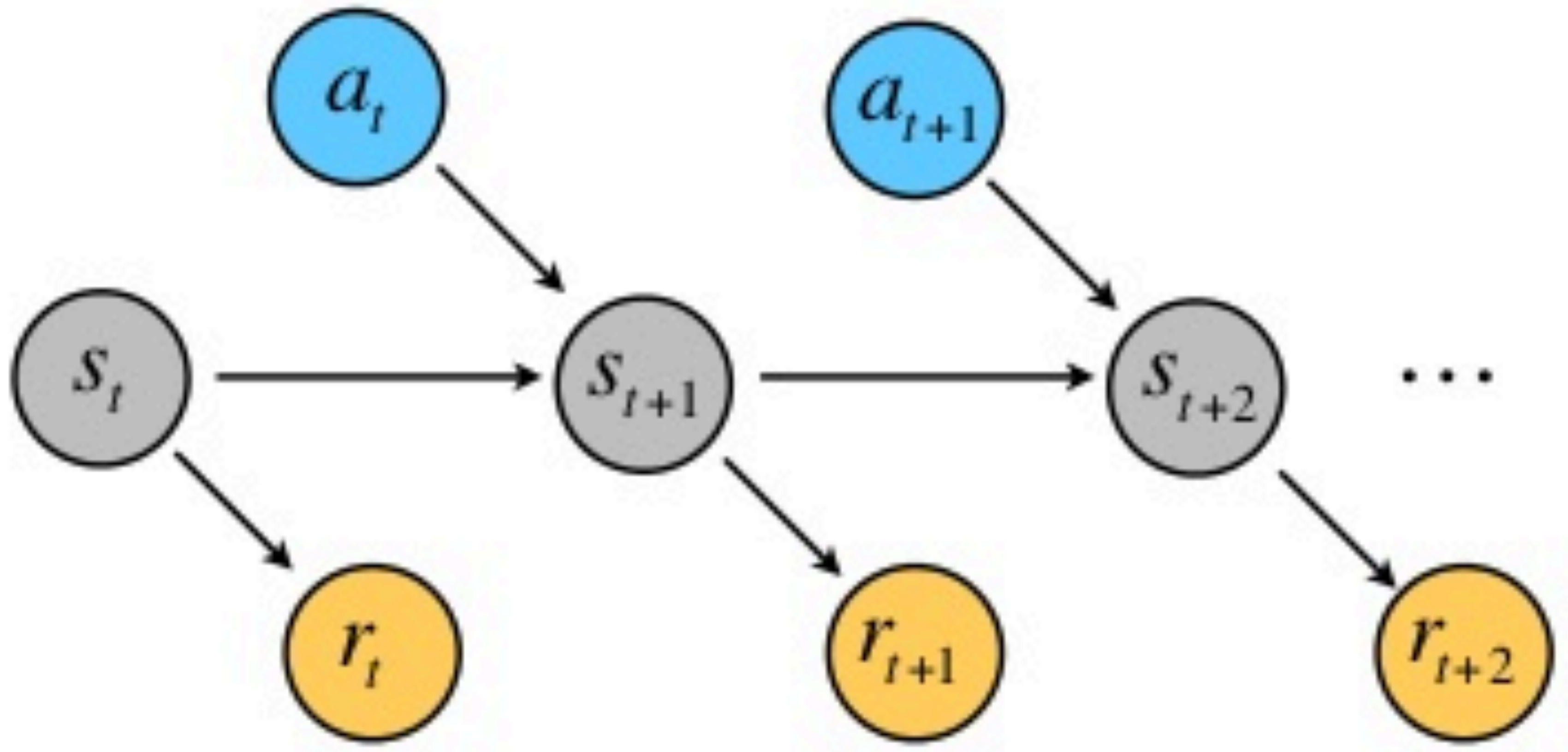


Environment

actions

states

reward

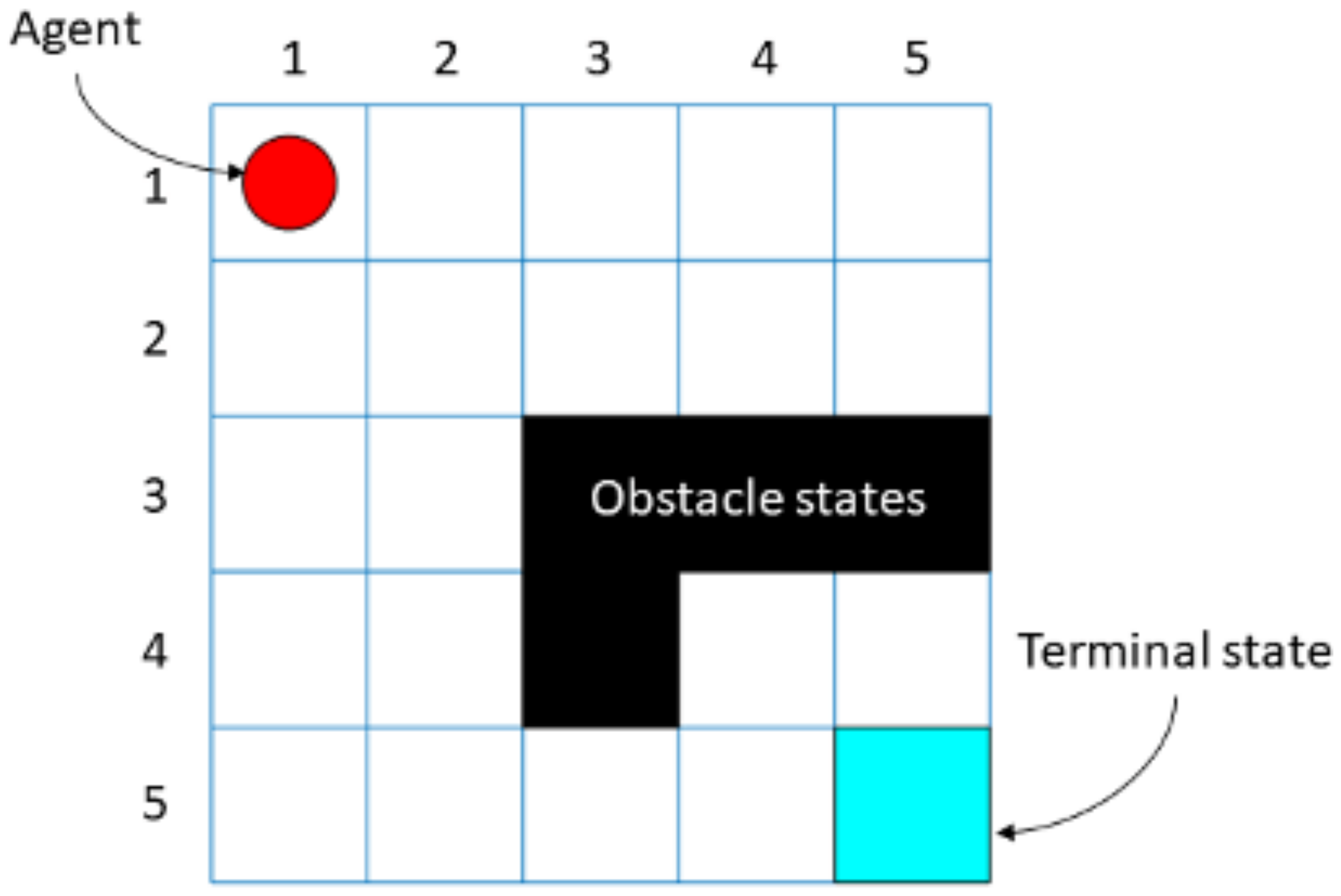


Markov Decision Process (MDP)

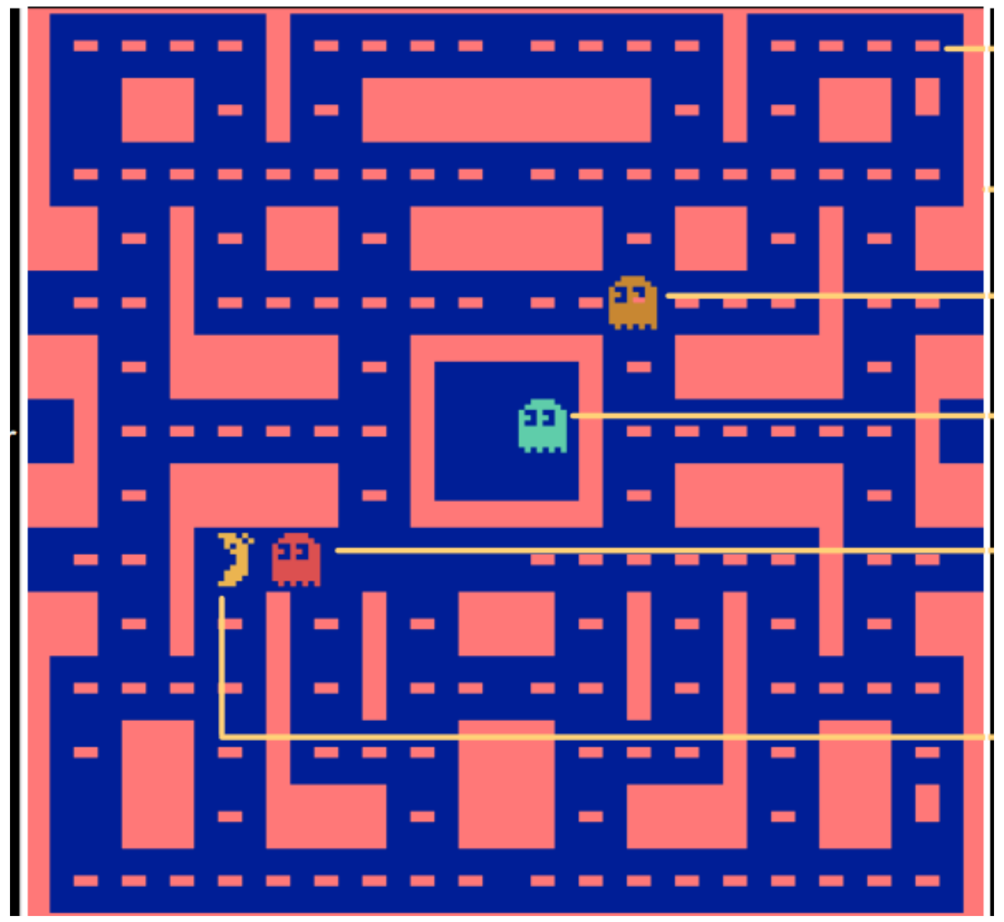
- Simplifying assumption that the system is fully defined by only the previous state (i.e., Markov Principle): $P(s_{t+1} \mid s_t, a_t)$

What are the states?

- Discrete locations, pixels on a screen, a set of feature values, etc…



Grid World

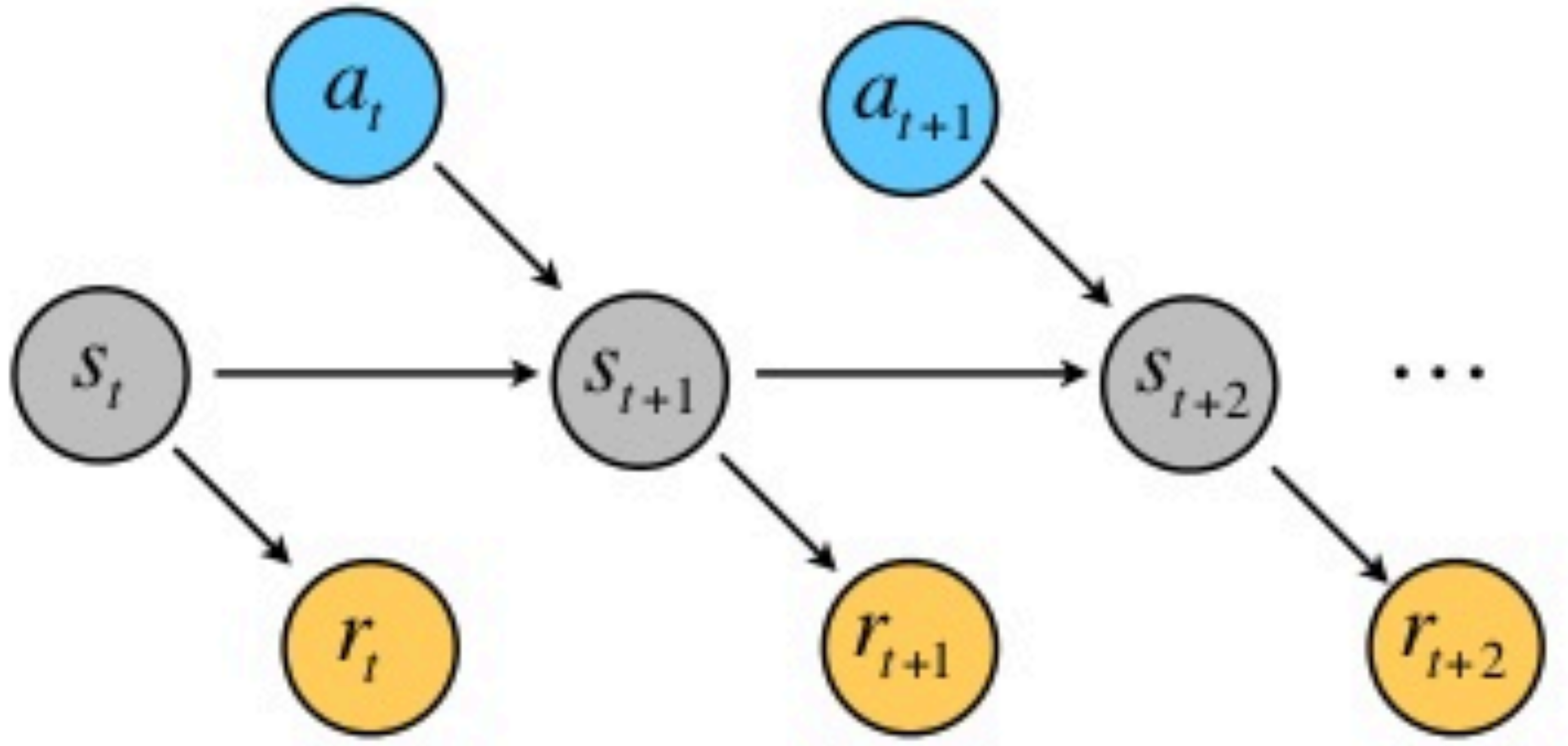


Environment

actions

states

reward

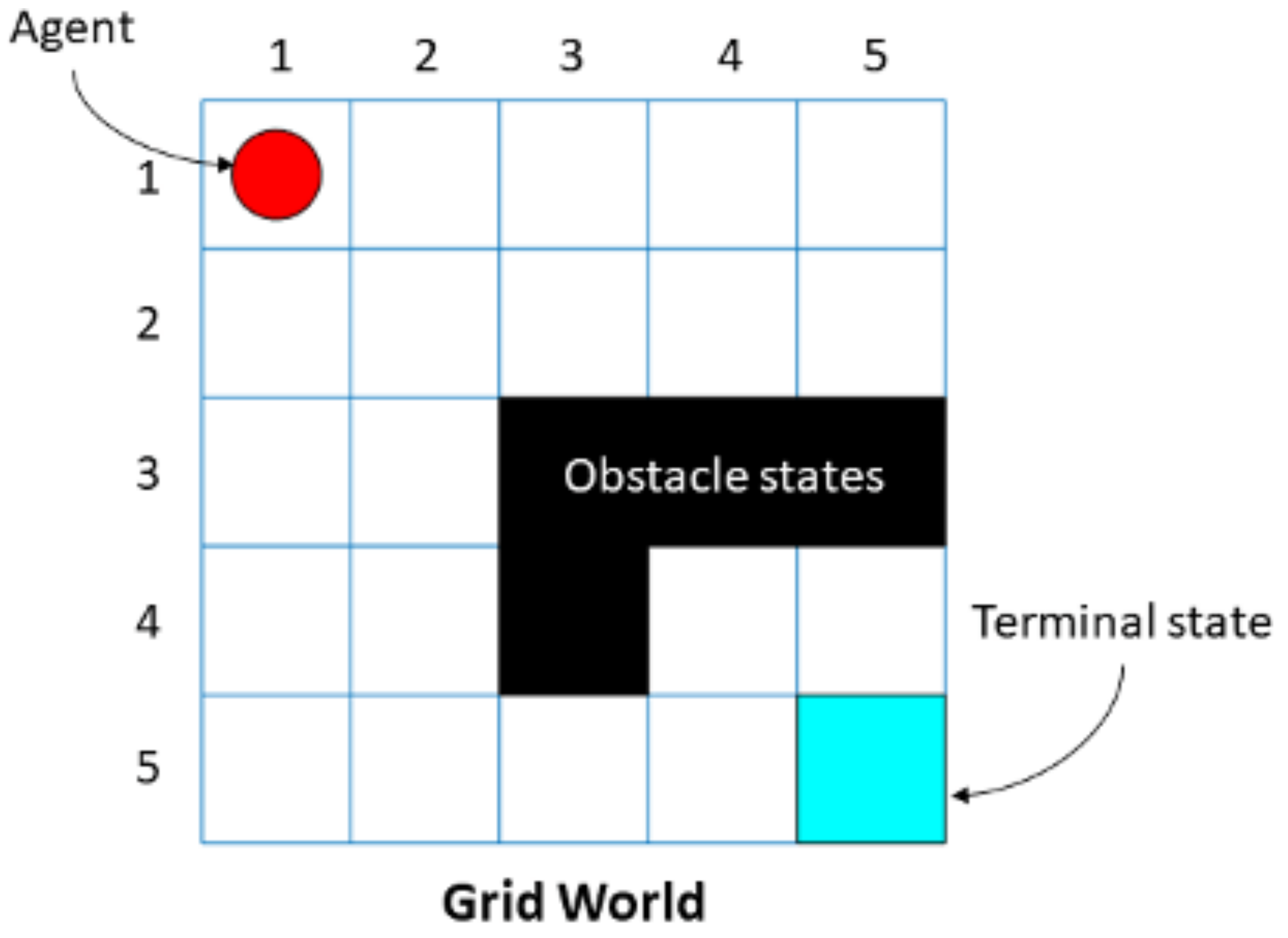


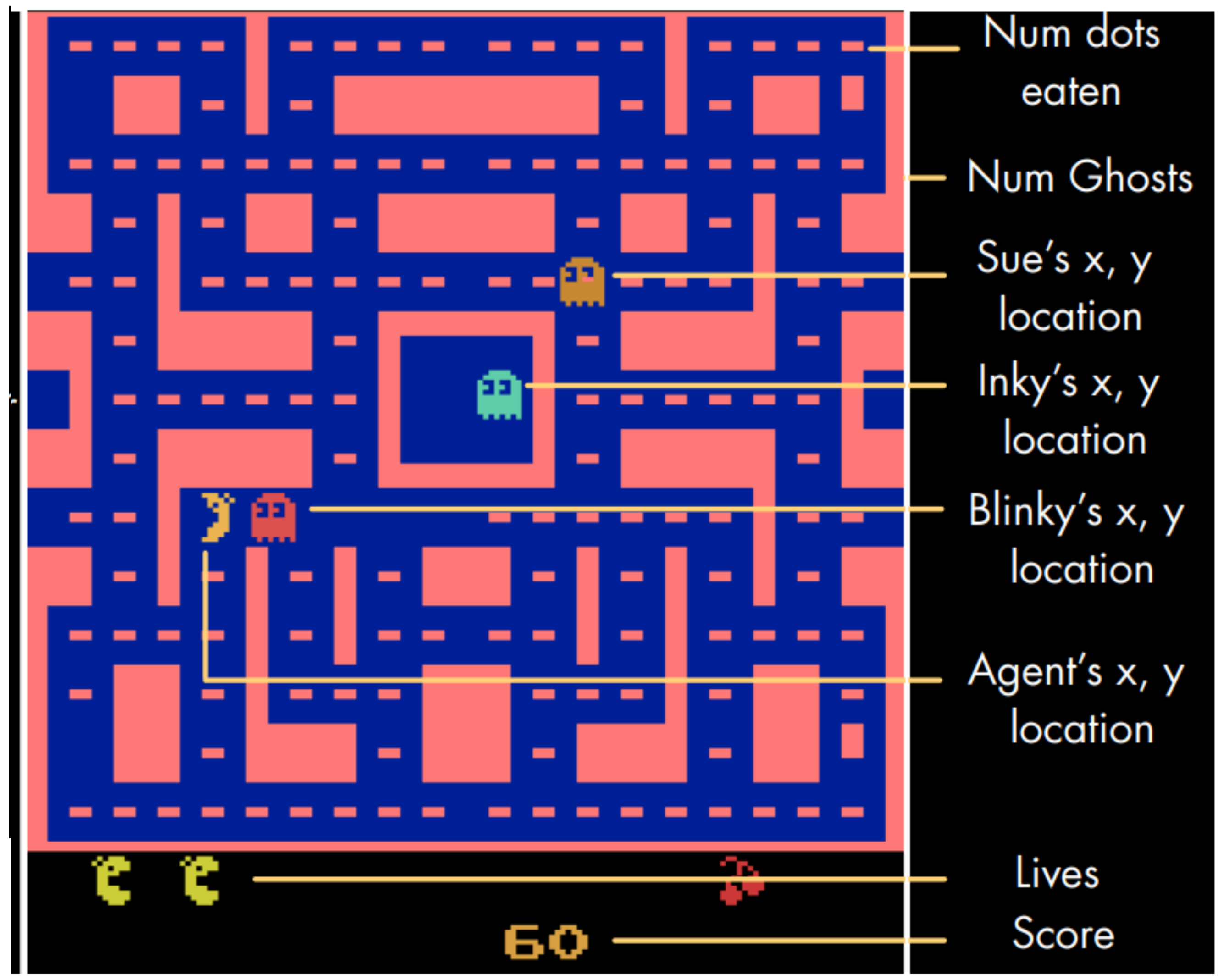
Markov Decision Process (MDP)

- Simplifying assumption that the system is fully defined by only the previous state (i.e., Markov Principle): $P(s_{t+1} \mid s_t, a_t)$

What are the states?

- Discrete locations, pixels on a screen, a set of feature values, etc…



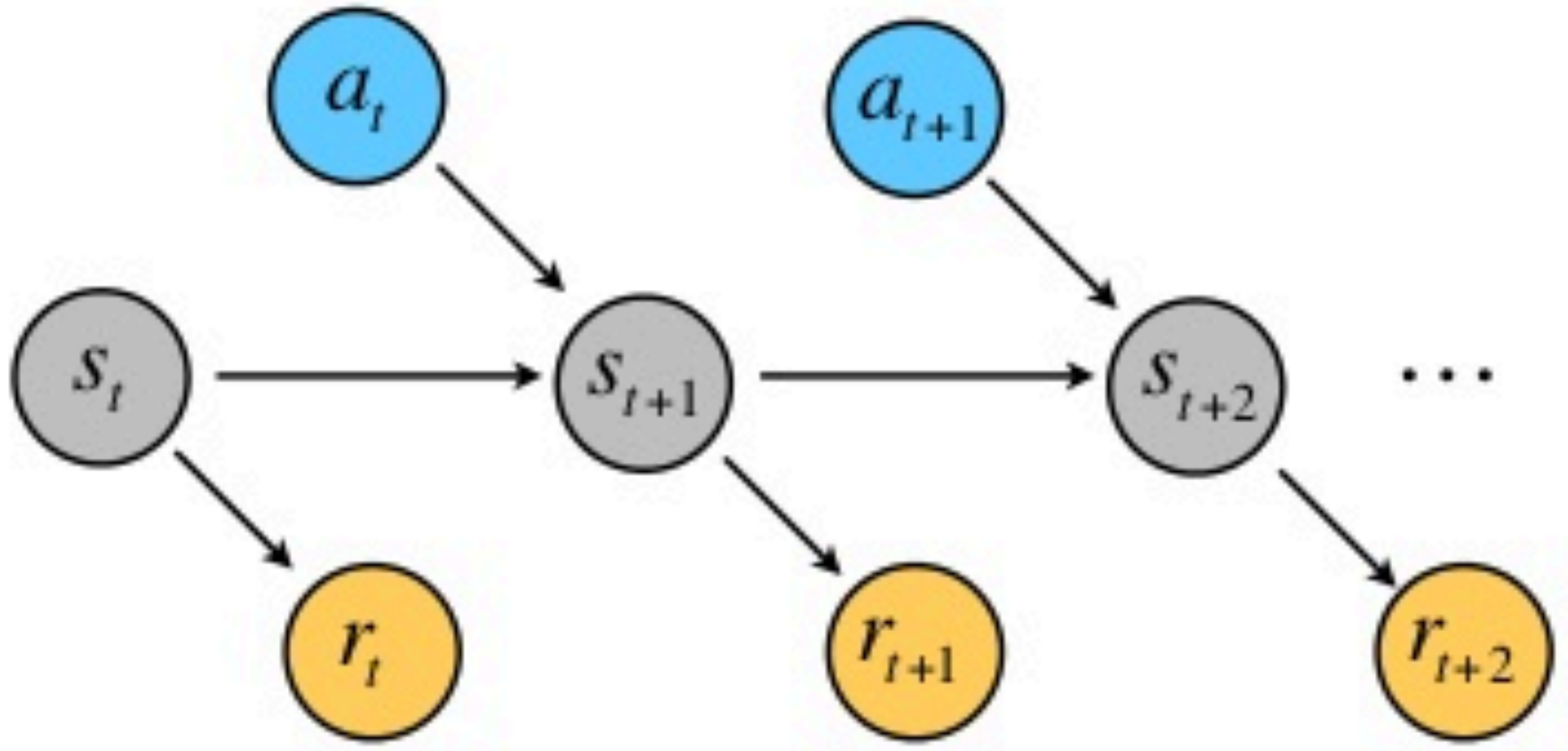


Environment

actions

states

reward

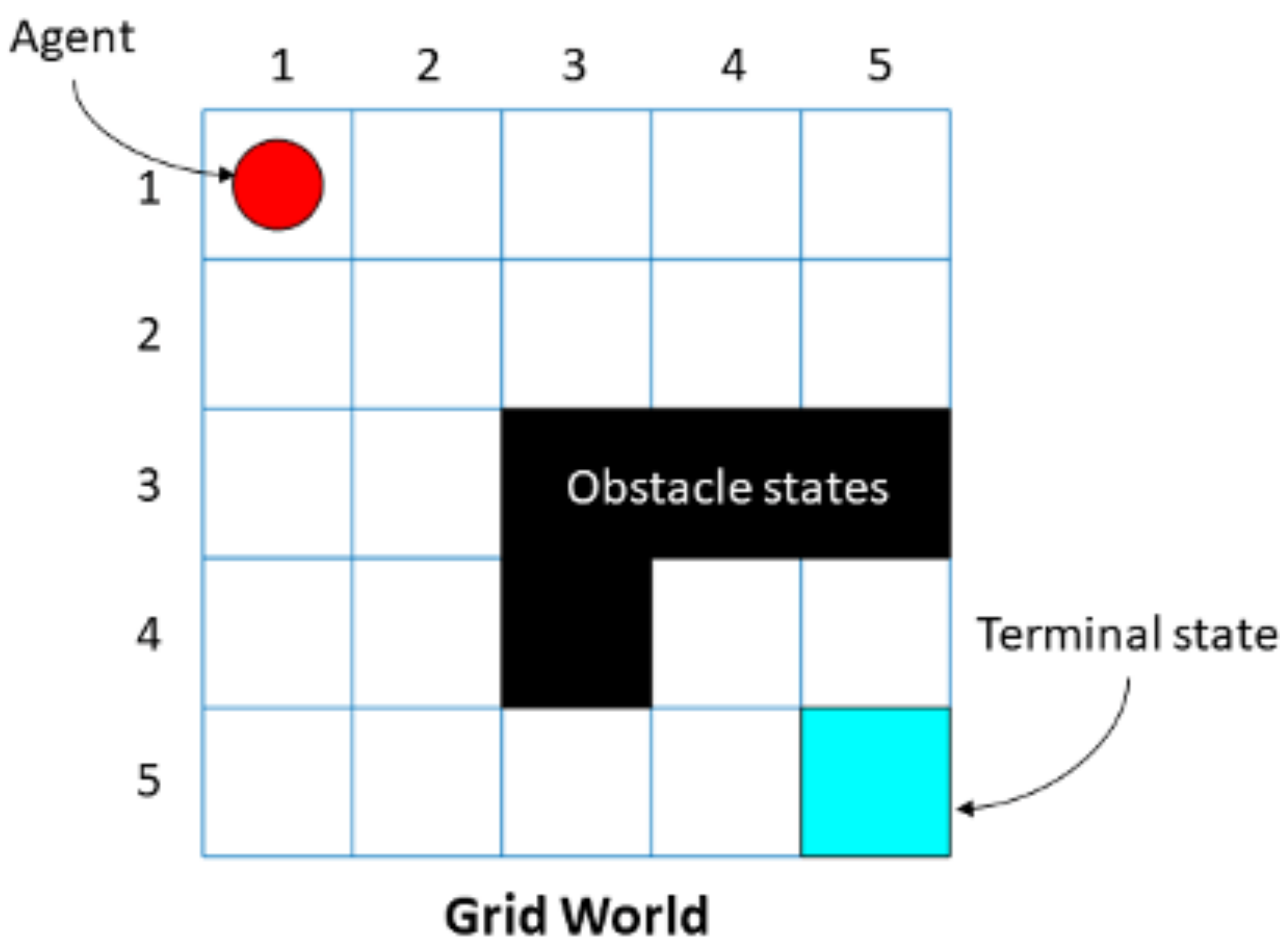


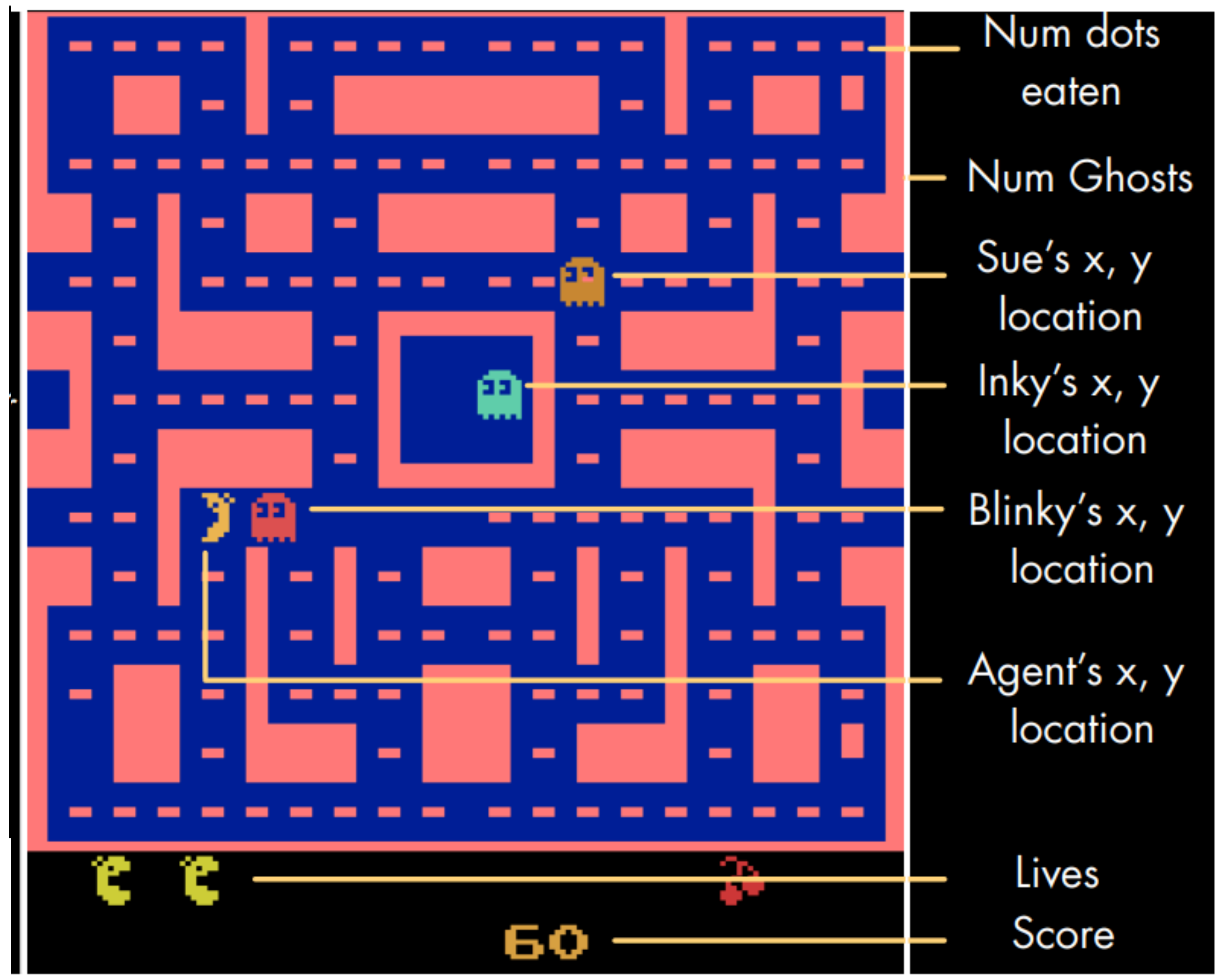
Markov Decision Process (MDP)

- Simplifying assumption that the system is fully defined by only the previous state (i.e., Markov Principle): $P(s_{t+1} \mid s_t, a_t)$

What are the states?

- Discrete locations, pixels on a screen, a set of feature values, etc…





Partially Observable MDP (POMDP)

Normative vs. Descriptive

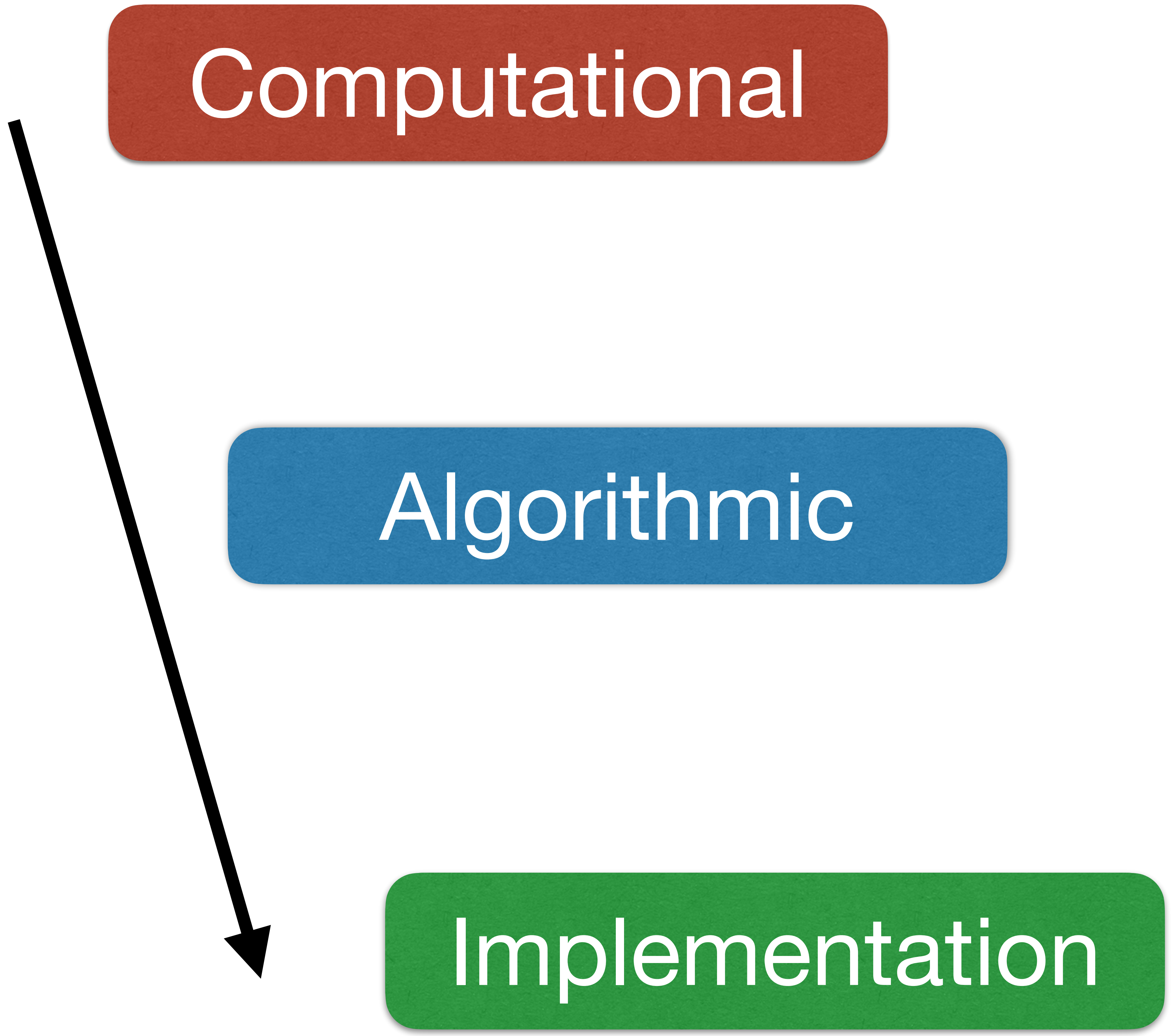
RL as a **normative** framework:

- How *should* a rational agent behave when learning from the environment?
- Which learning mechanisms and which policies lead to better outcomes?

RL as a **descriptive** framework:

- How *does* an agent update beliefs and select actions when learning from the environment?
- Which learning mechanisms and which policies provide better descriptions of behavior

Marr’s Levels of Analysis (1982)



Marr's Levels of Analysis (1982)

Computational

What is the goal of the system?
How does it behave?

Algorithmic

Implementation

Marr’s Levels of Analysis (1982)

Computational

What is the goal of the system?
How does it behave?

Algorithmic

Which representations
and computations?

Implementation

Marr's Levels of Analysis (1982)

Computational

What is the goal of the system?
How does it behave?

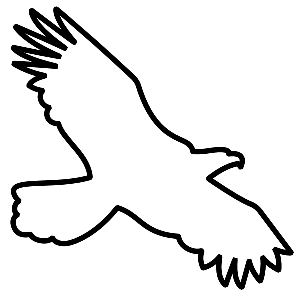
Algorithmic

Which representations
and computations?

Implementation

How is the system realized?

Marr’s Levels of Analysis (1982)



Flight

Computational

What is the goal of the system?
How does it behave?

Flapping

Algorithmic

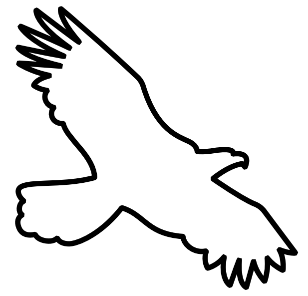
Which representations
and computations?

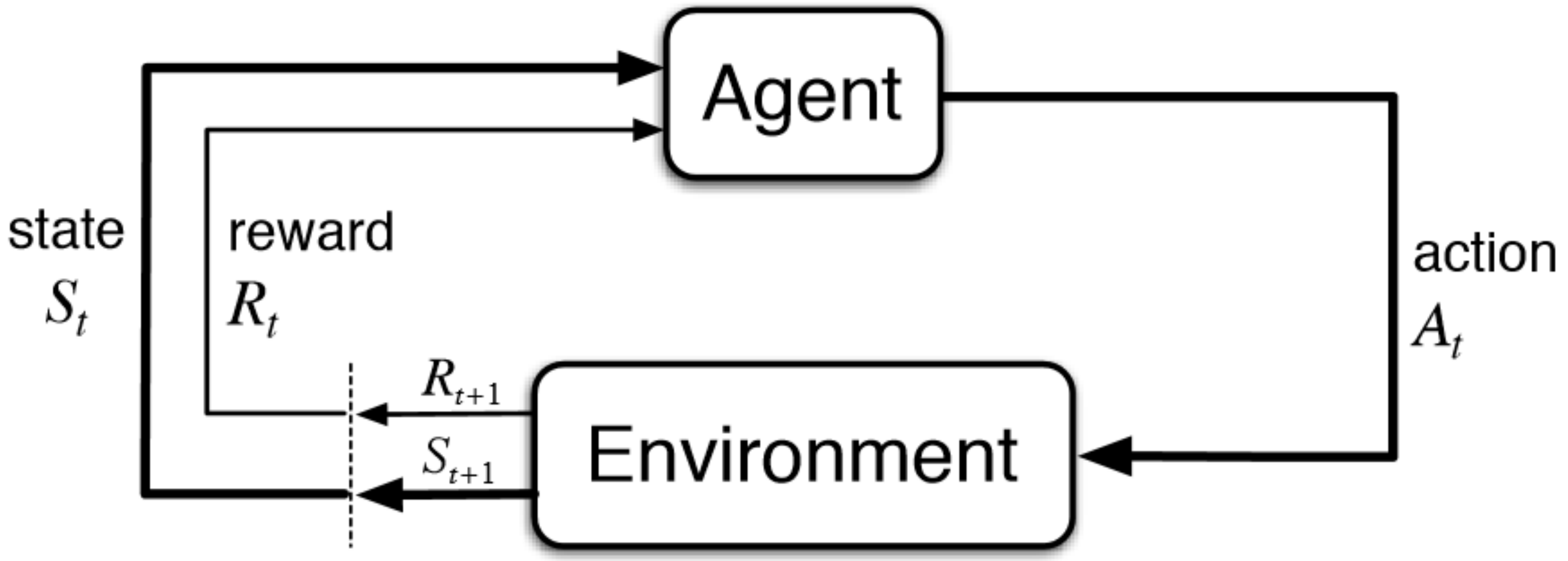
Feathers

Implementation

How is the system realized?

Marr's Levels of Analysis (1982)





Flight

Computational

What is the goal of the system?
How does it behave?

Flapping

Algorithmic

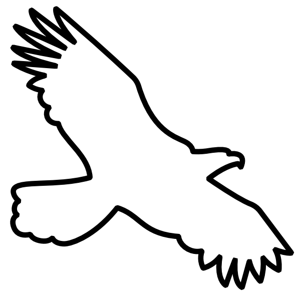
Which representations
and computations?

Feathers

Implementation

How is the system realized?

Marr's Levels of Analysis (1982)



Flight

Computational

What is the goal of the system?
How does it behave?

Flapping

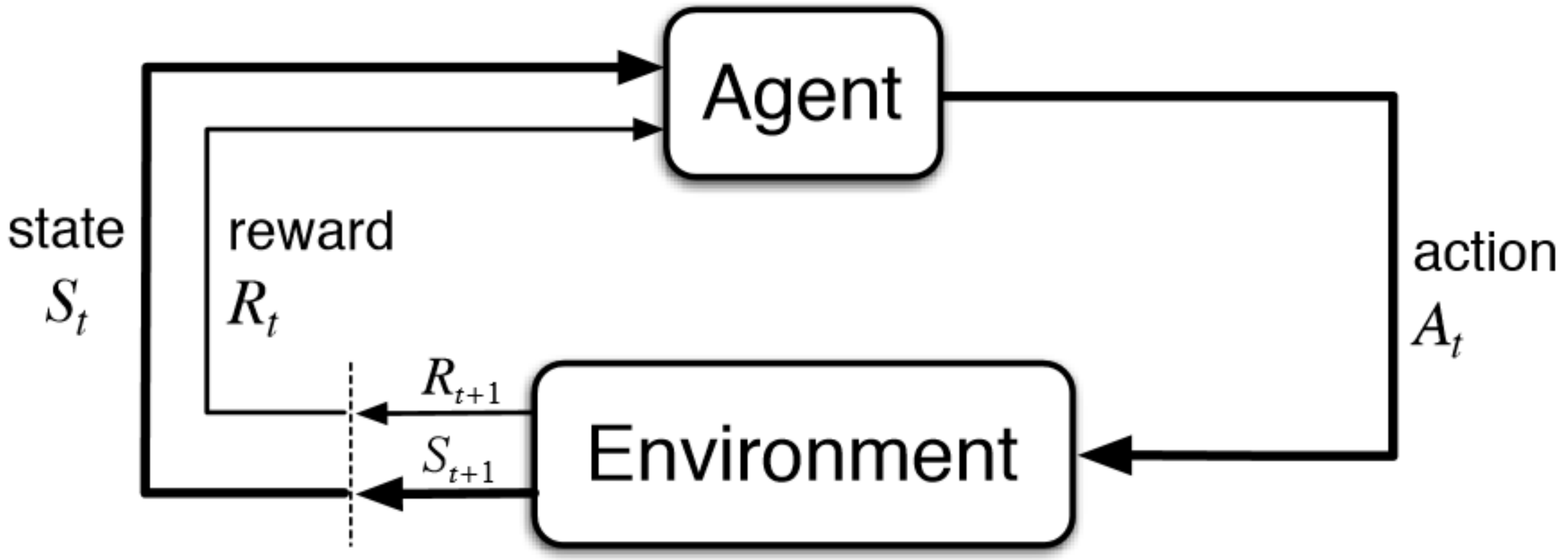
Algorithmic

Which representations
and computations?

Feathers

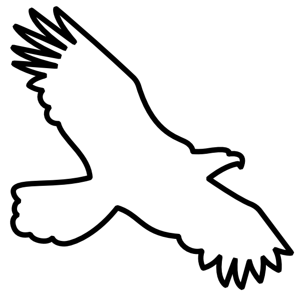
Implementation

How is the system realized?



Initialize $Q(s,a)$ arbitrarily
Repeat (for each episode):
  Initialize s
  Repeat (for each step of episode):
    Choose a from s using policy derived from Q (e.g., ε-greedy)
    Take action a, observe r, s'
    $Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right]$
    $s \leftarrow s'$;
  until s is terminal

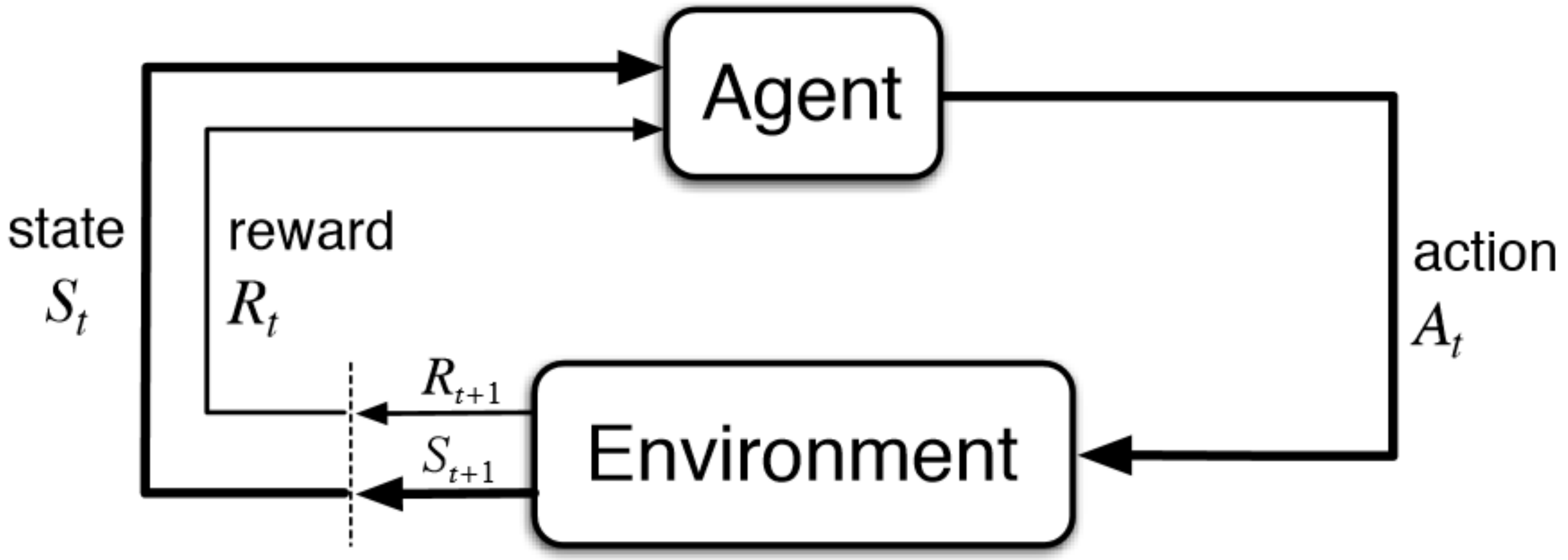
Marr's Levels of Analysis (1982)



Flight

Computational

What is the goal of the system?
How does it behave?

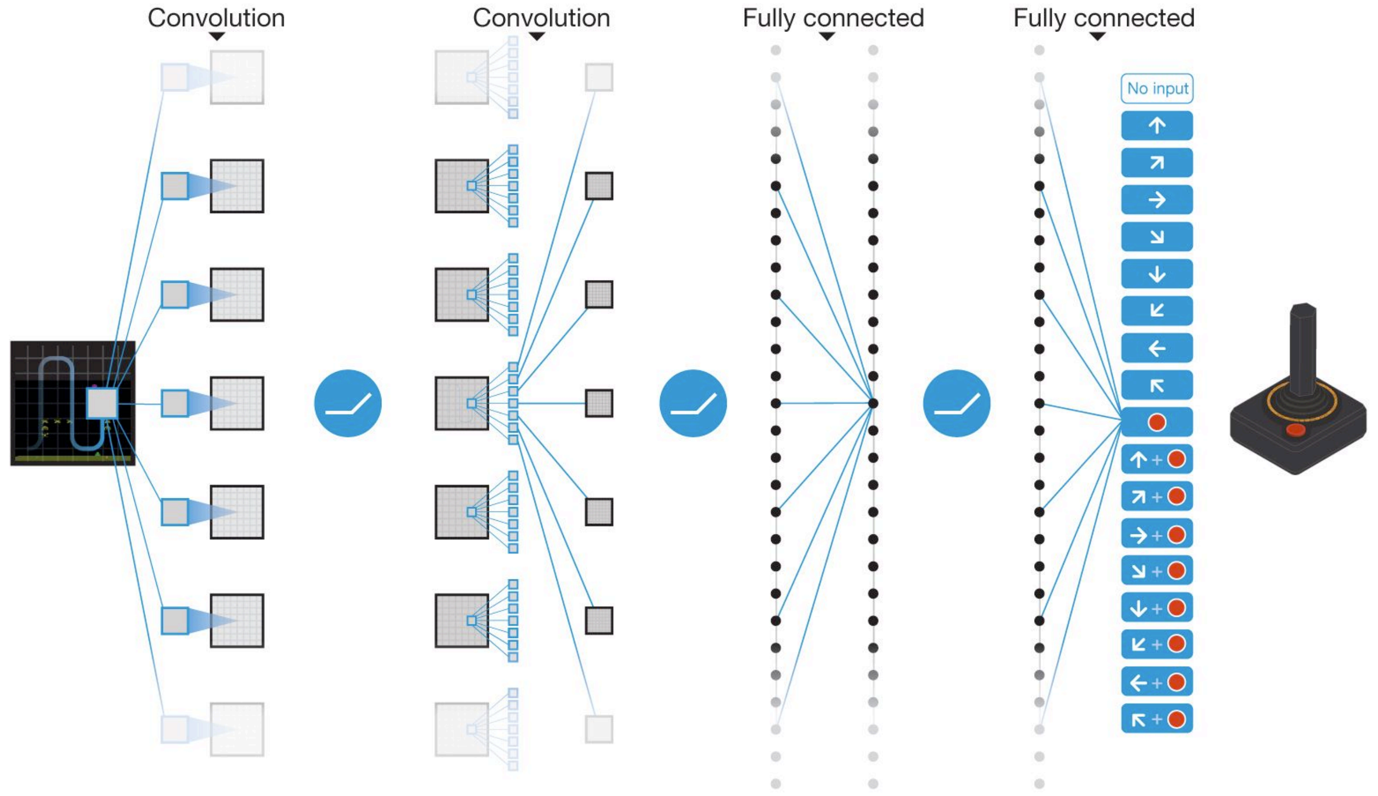


Initialize $Q(s,a)$ arbitrarily
Repeat (for each episode):
  Initialize s
  Repeat (for each step of episode):
    Choose a from s using policy derived from Q (e.g., ε-greedy)
    Take action a, observe r, s'
    $Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right]$
    $s \leftarrow s'$;
  until s is terminal

Flapping

Algorithmic

Which representations and computations?



Feathers

Implementation

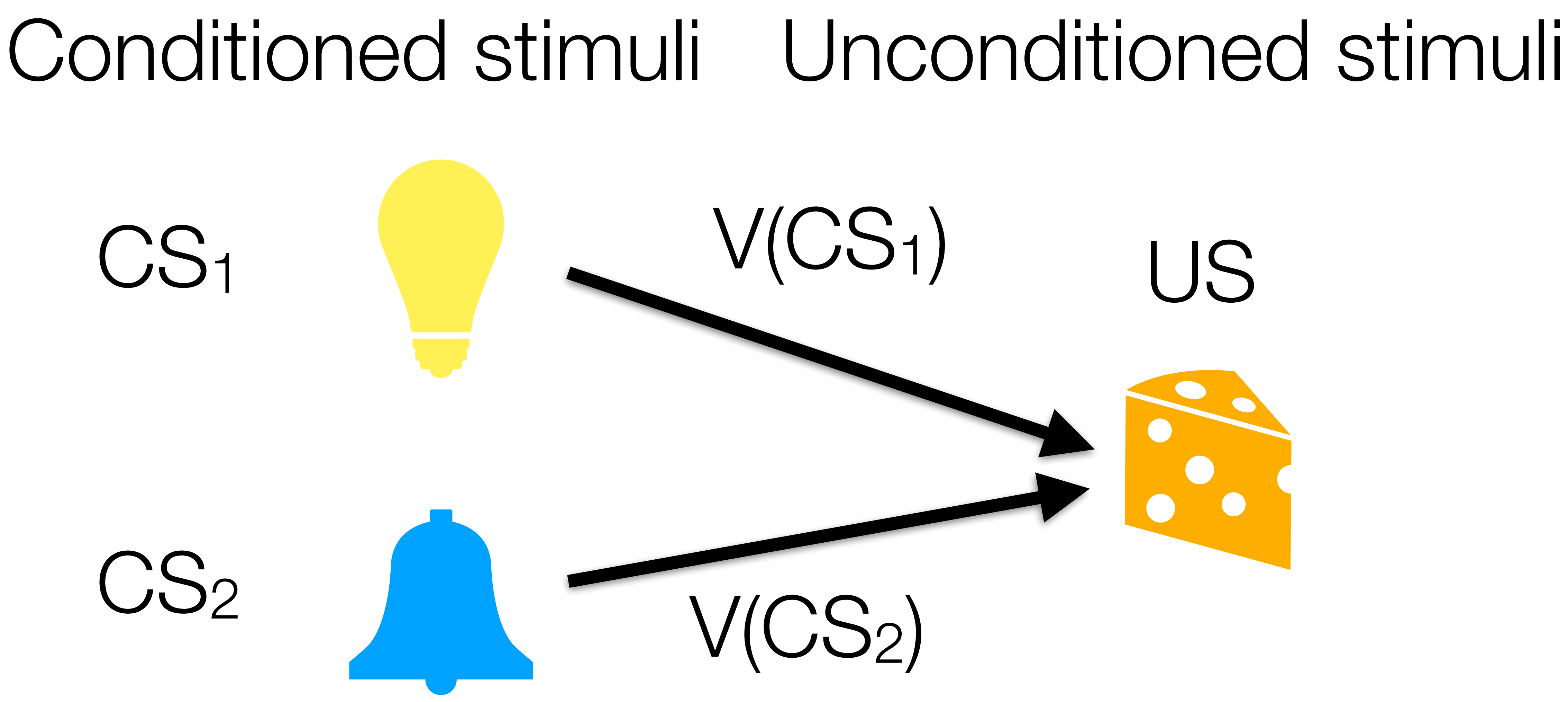
How is the system realized?

Rescorla-Wagner (proto-RL)

Rescorla-Wagner model

(Bush & Mosteller, 1951; Rescorla & Wagner, 1972)

$$V_{new}(CS_i) = V_{old}(CS_i) + \eta \left[\lambda_{US} - \sum_i V_{old}(CS_i) \right]$$

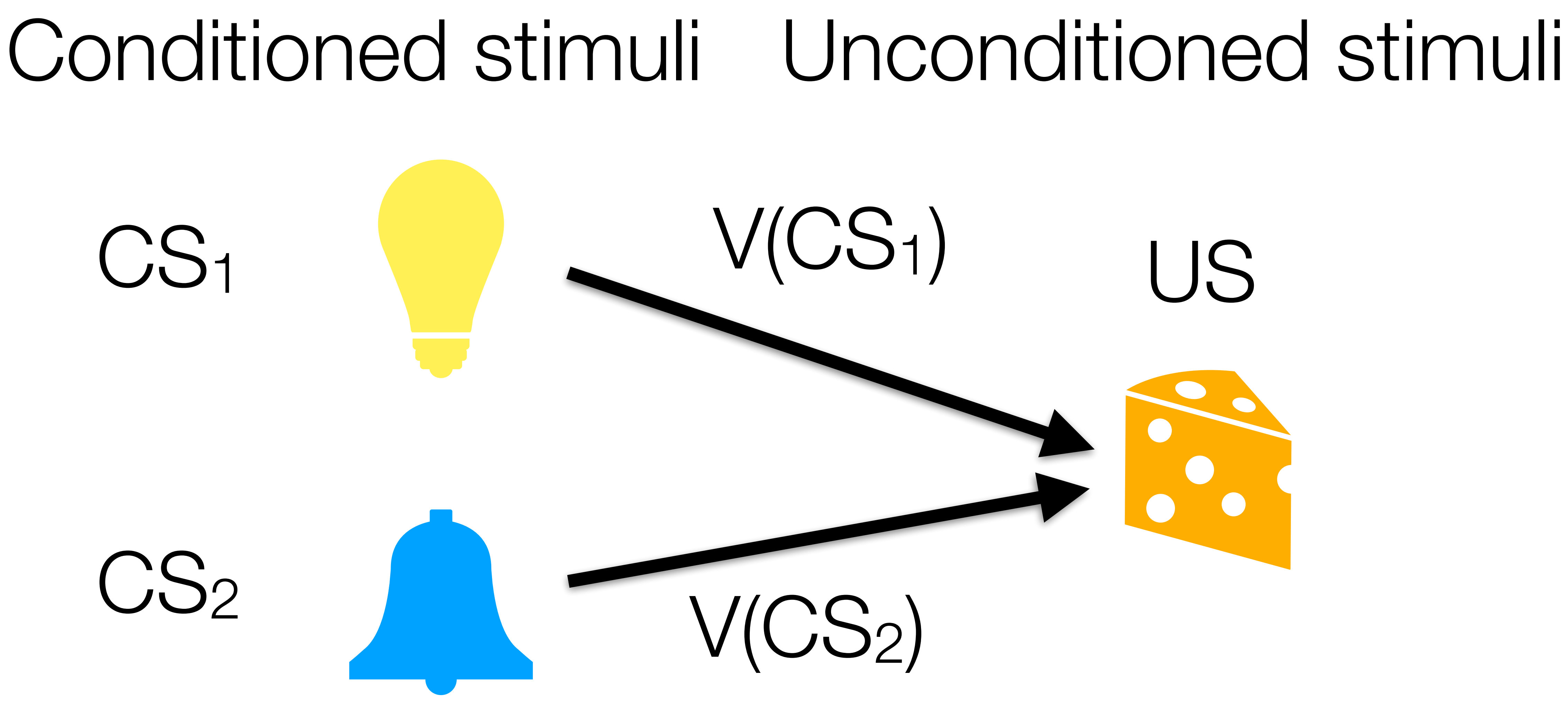


Rescorla-Wagner (proto-RL)

Rescorla-Wagner model

(Bush & Mosteller, 1951; Rescorla & Wagner, 1972)

$$V_{new}(CS_i) = V_{old}(CS_i) + \eta \left[\underbrace{\lambda_{US}}_{\text{Observed outcome}} - \sum_i \underbrace{V_{old}(CS_i)}_{\text{Predicted outcome}} \right]$$



Rescorla-Wagner (proto-RL)

Rescorla-Wagner model

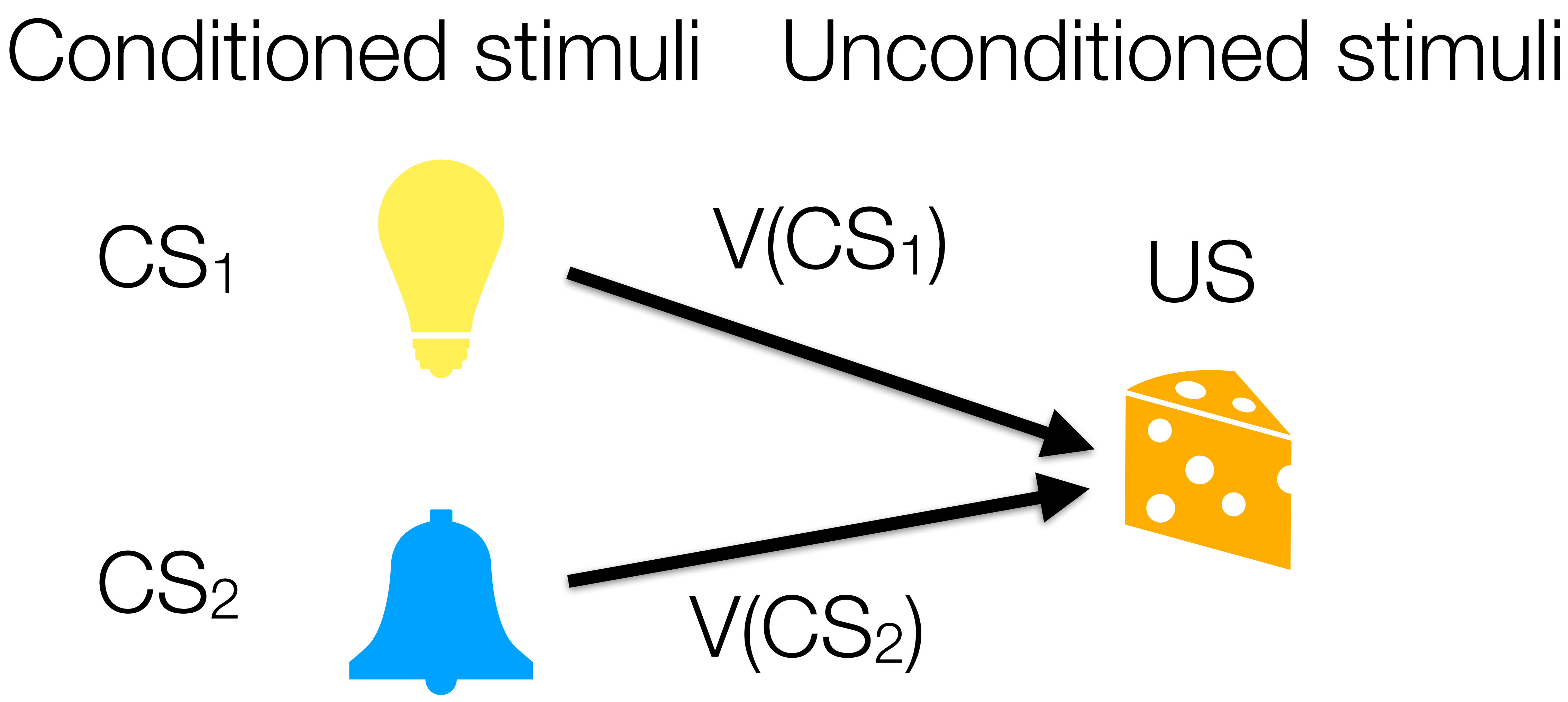
(Bush & Mosteller, 1951; Rescorla & Wagner, 1972)

$$V_{new}(CS_i) = V_{old}(CS_i) + \eta \left[\lambda_{US} - \sum_i V_{old}(CS_i) \right]$$

learning rate

Observed outcome

Predicted outcome



Rescorla-Wagner (proto-RL)

Rescorla-Wagner model

(Bush & Mosteller, 1951; Rescorla & Wagner, 1972)

$$V_{new}(CS_i) = V_{old}(CS_i) + \eta \left[\underbrace{\lambda_{US} - \sum_i V_{old}(CS_i)}_{\delta} \right]$$

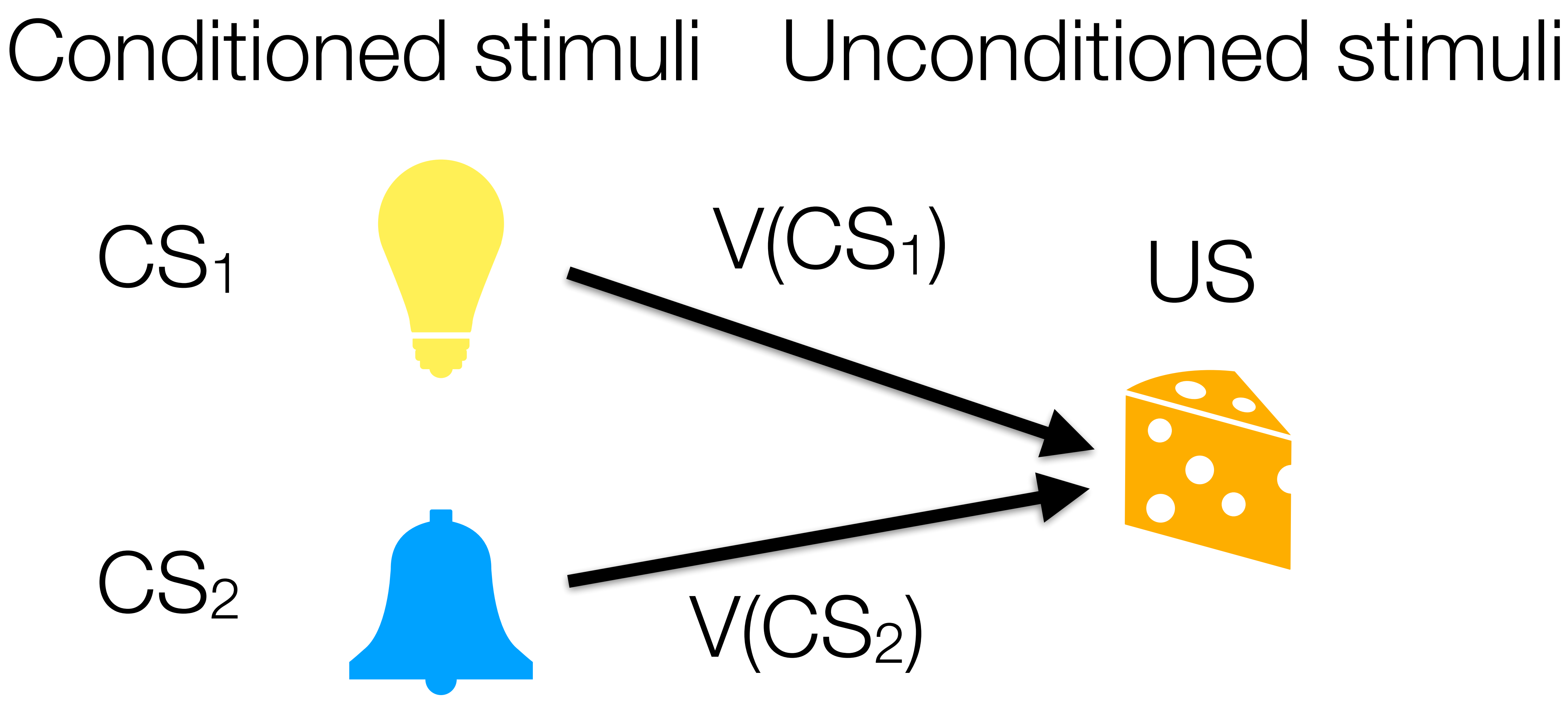
learning rate

Observed outcome

Predicted outcome

δ

Reward prediction error (RPE)



The delta-rule of learning:

- Learning occurs only when events violate expectations ($\delta \neq 0$)
- The magnitude of the error corresponds to how much we update our beliefs

From Rescorla-Wagner to Q-learning

Rescorla-Wagner model

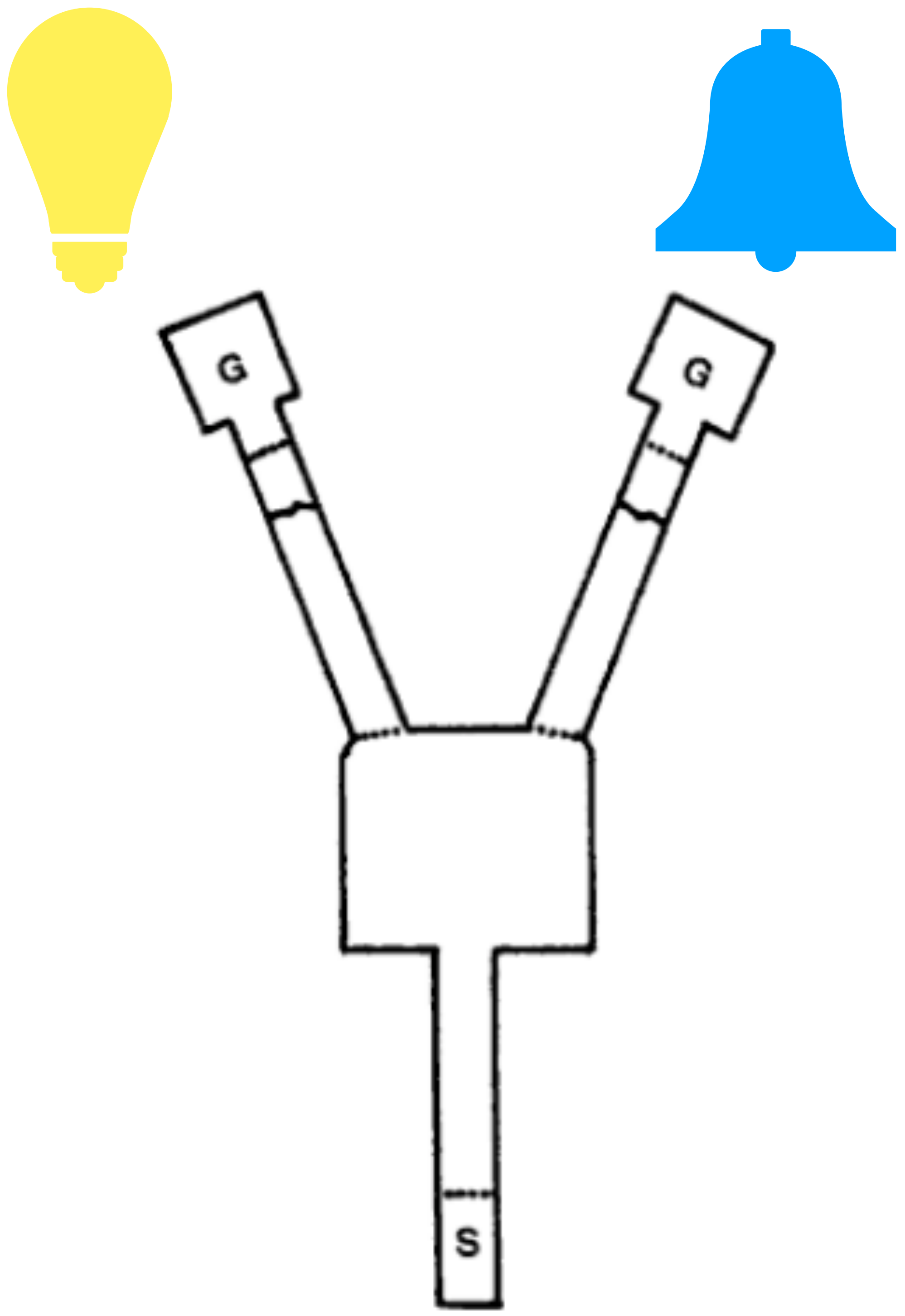
(Bush & Mosteller, 1951; Rescorla & Wagner, 1972)

$$V_{new}(CS_i) = V_{old}(CS_i) + \eta \left[\lambda_{US} - \sum_i V_{old}(CS_i) \right]$$

Q-learning

(Watkins, 1989)

Y-maze example



From Rescorla-Wagner to Q-learning

Rescorla-Wagner model

(Bush & Mosteller, 1951; Rescorla & Wagner, 1972)

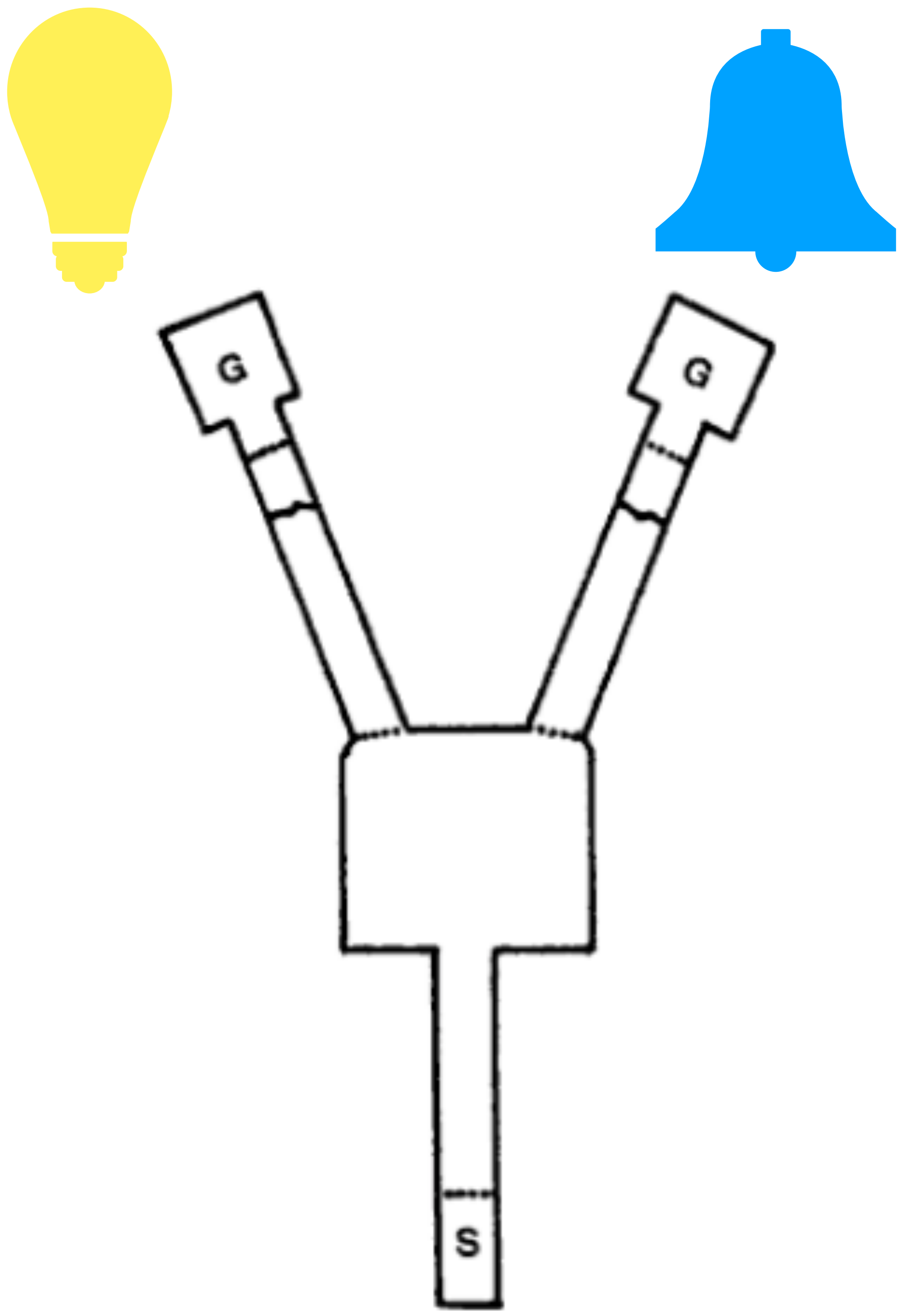
$$V_{new}(CS_i) = V_{old}(CS_i) + \eta \left[\lambda_{US} - \sum_i V_{old}(CS_i) \right]$$

Q-learning

(Watkins, 1989)

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \eta \left[r - Q(s_t, a_t) \right]$$

Y-maze example



From Rescorla-Wagner to Q-learning

Rescorla-Wagner model

(Bush & Mosteller, 1951; Rescorla & Wagner, 1972)

$$V_{new}(CS_i) = V_{old}(CS_i) + \eta \left[\lambda_{US} - \sum_i V_{old}(CS_i) \right]$$

Q-learning

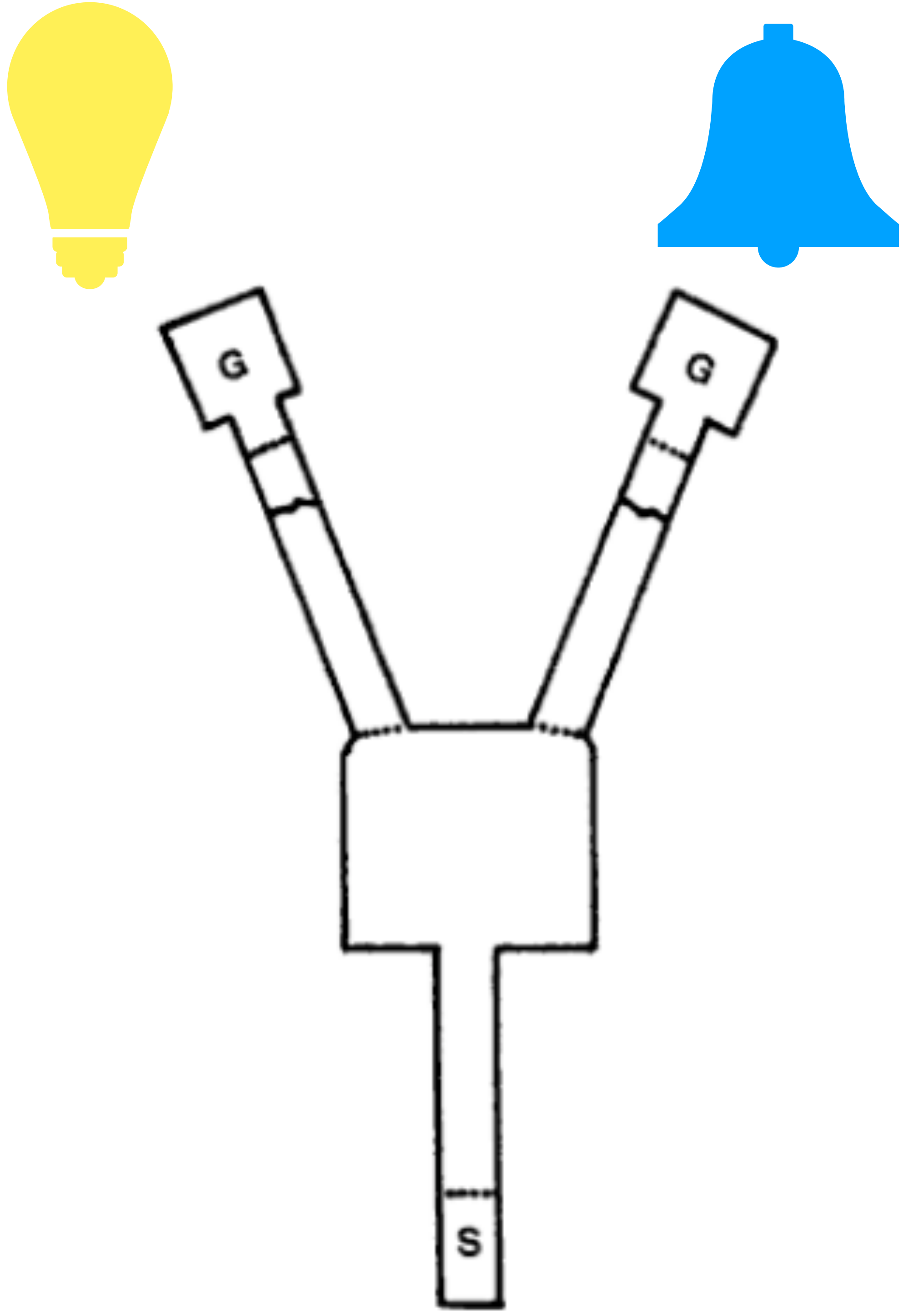
(Watkins, 1989)

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \eta \left[r - Q(s_t, a_t) \right]$$

Observed reward

Predicted reward

Y-maze example



From Rescorla-Wagner to Q-learning

Rescorla-Wagner model
(Bush & Mosteller, 1951; Rescorla & Wagner, 1972)

$$V_{new}(CS_i) = V_{old}(CS_i) + \eta \left[\lambda_{US} - \sum_i V_{old}(CS_i) \right]$$

Q-learning
(Watkins, 1989)

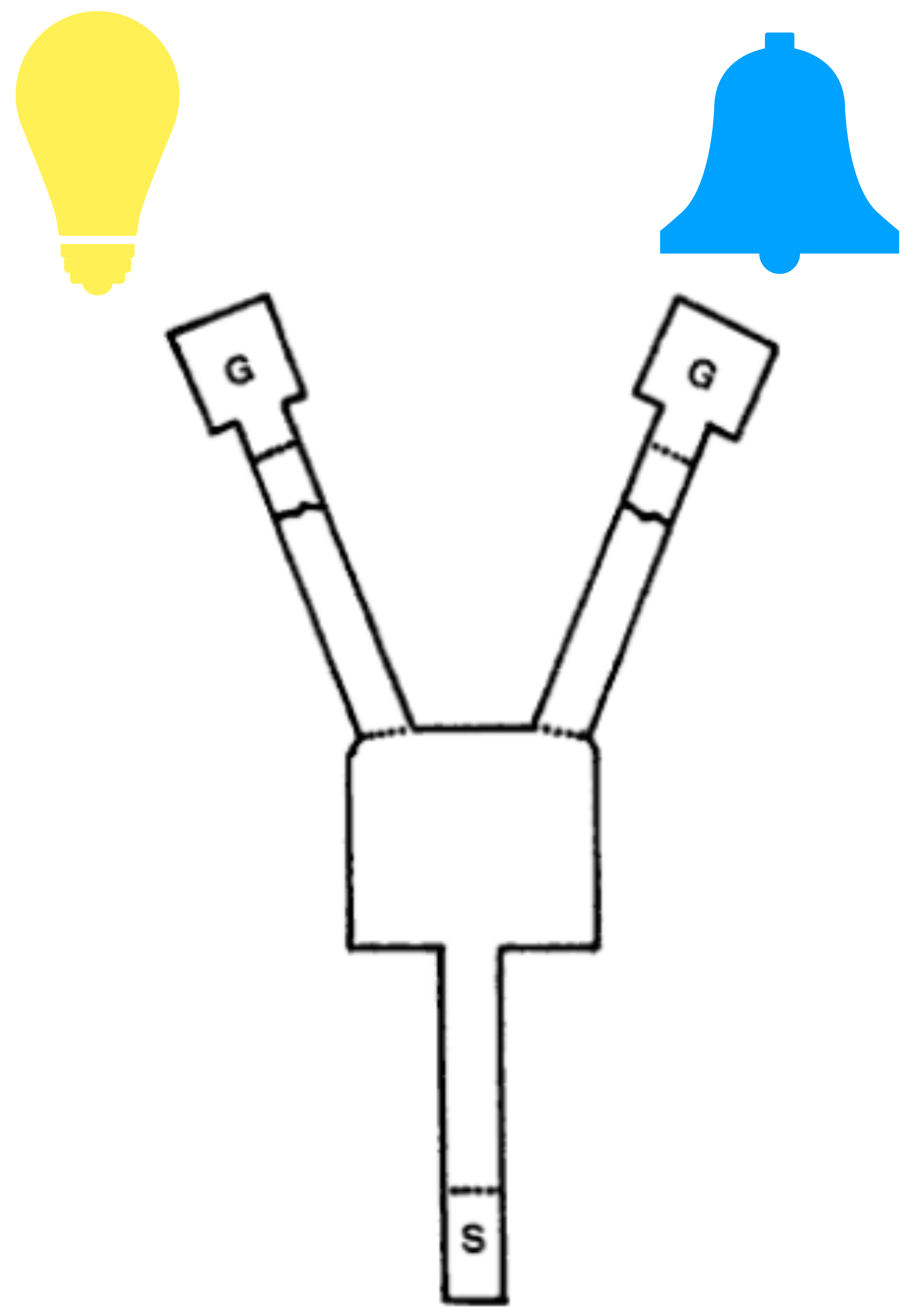
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \eta \left[r - Q(s_t, a_t) \right]$$

learning rate

Observed reward

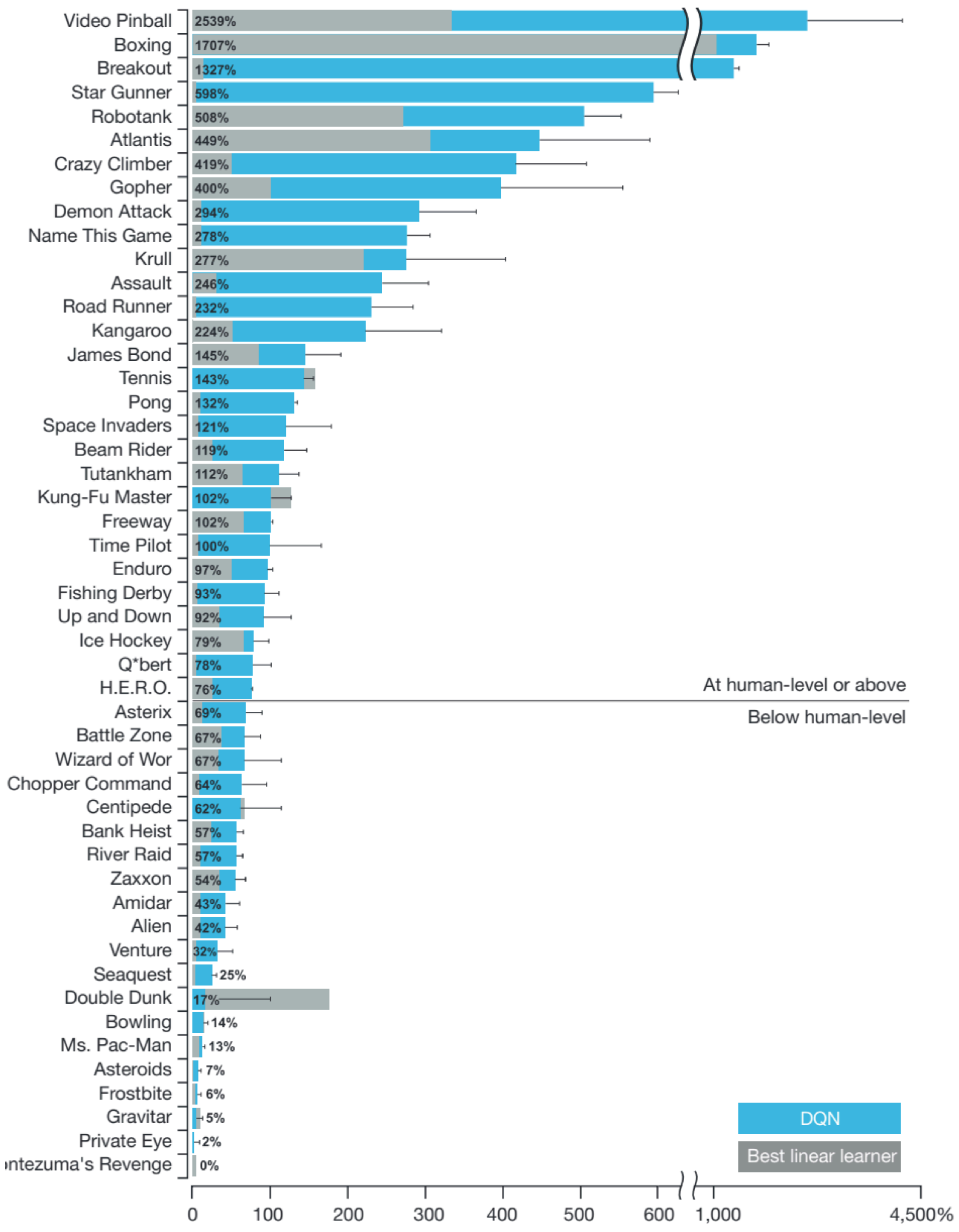
Predicted reward

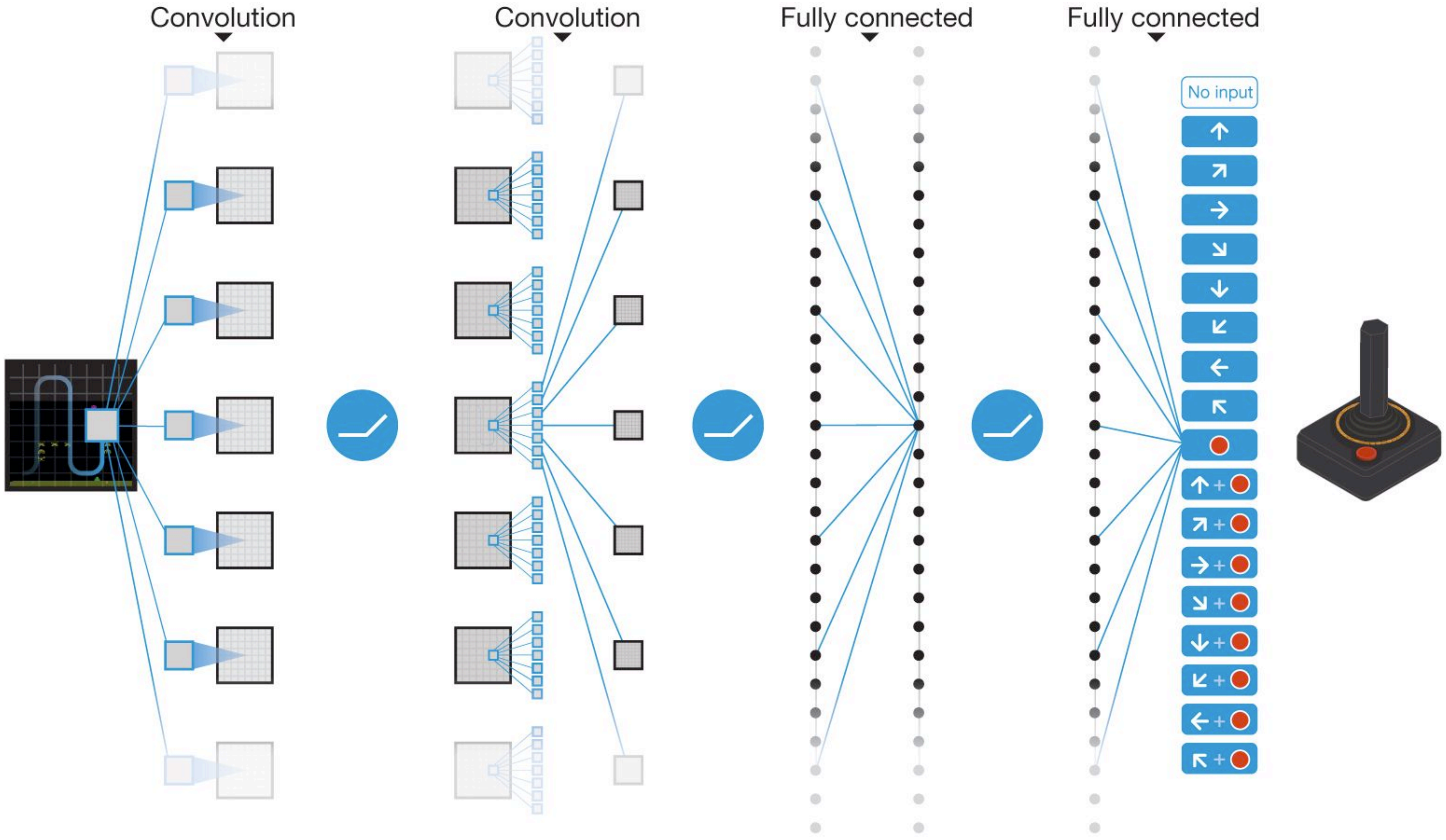
Y-maze example



Mnih et al,. (Nature, 2015)

Deep Q-Learning can play atari games with human-level control





Temporal-Difference (TD) learning

(Sutton & Barton, 1990)

Solving the credit assignment problem (Minsky, 1963):

- Which actions are responsible for (future) rewards?

Temporal-Difference (TD) learning

(Sutton & Barton, 1990)

Solving the credit assignment problem (Minsky, 1963):

- Which actions are responsible for (future) rewards?

Augment reward expectations by the discounted value of the next state

$$V(s) \leftarrow V(s) + \eta \left(r + \underline{\gamma V(s')} - V(s) \right)$$

Temporal-Difference (TD) learning

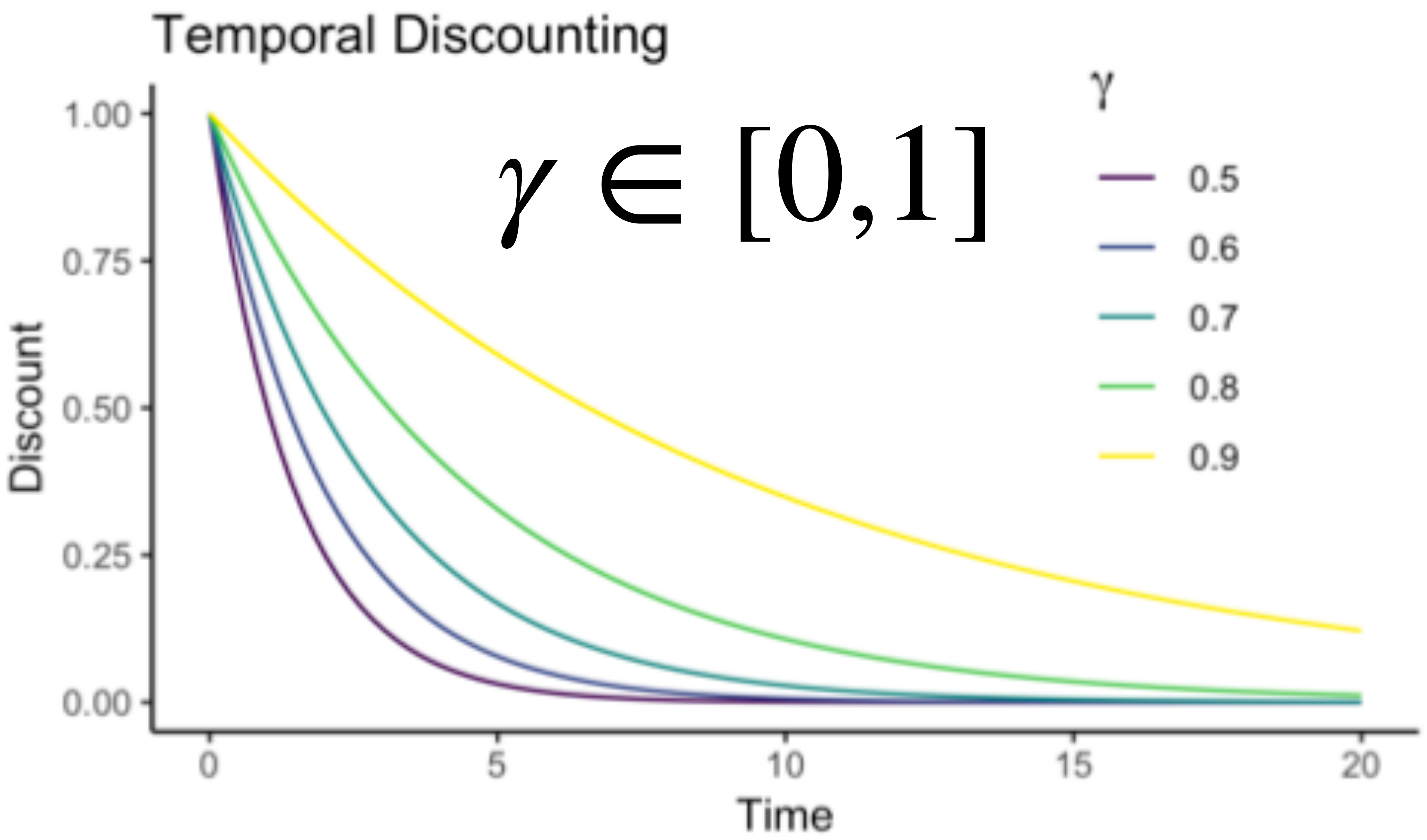
(Sutton & Barton, 1990)

Solving the credit assignment problem (Minsky, 1963):

- Which actions are responsible for (future) rewards?

Augment reward expectations by the discounted value of the next state

$$V(s) \leftarrow V(s) + \eta \left(r + \underline{\gamma V(s')} - V(s) \right)$$



Temporal-Difference (TD) learning

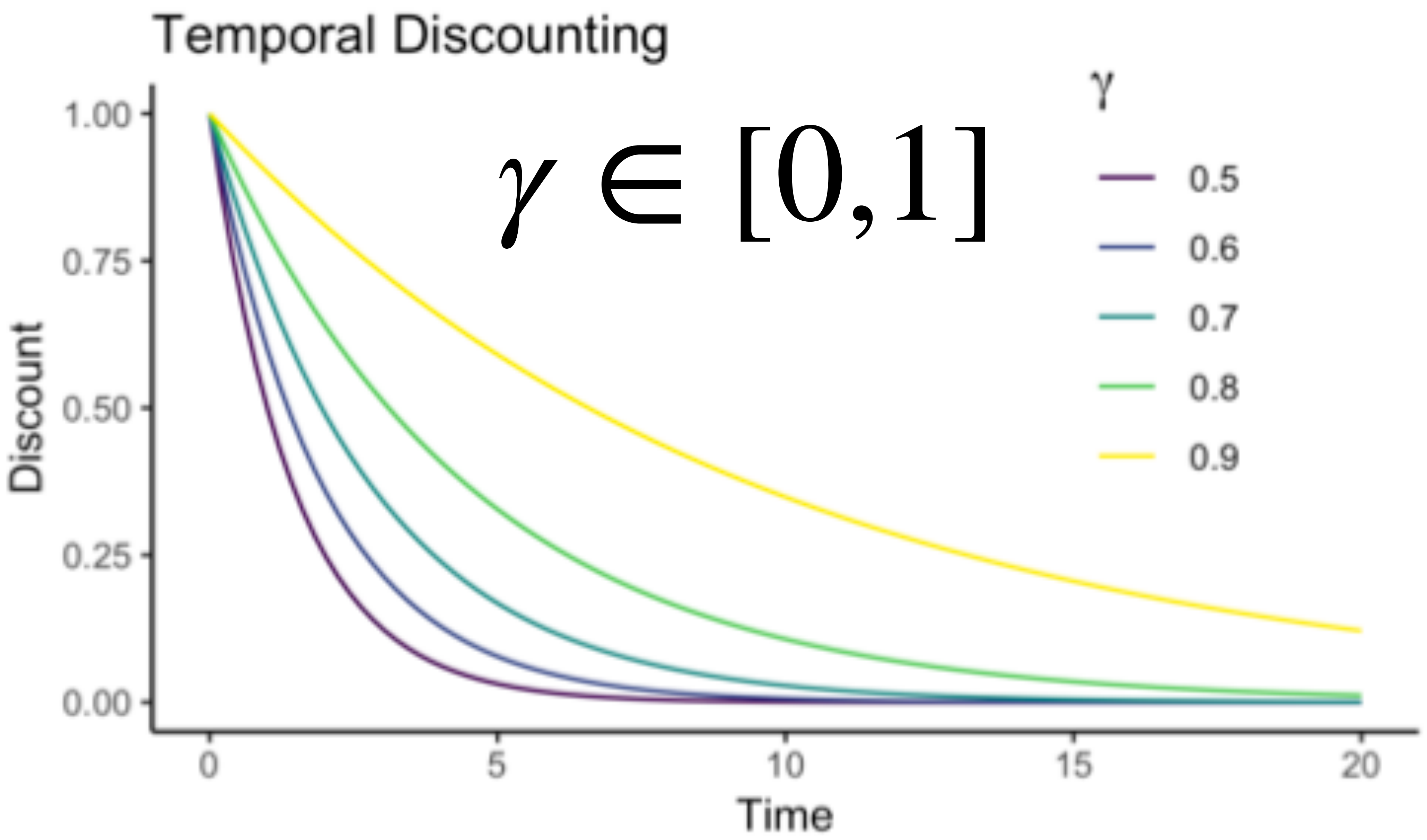
(Sutton & Barton, 1990)

Solving the credit assignment problem (Minsky, 1963):

- Which actions are responsible for (future) rewards?

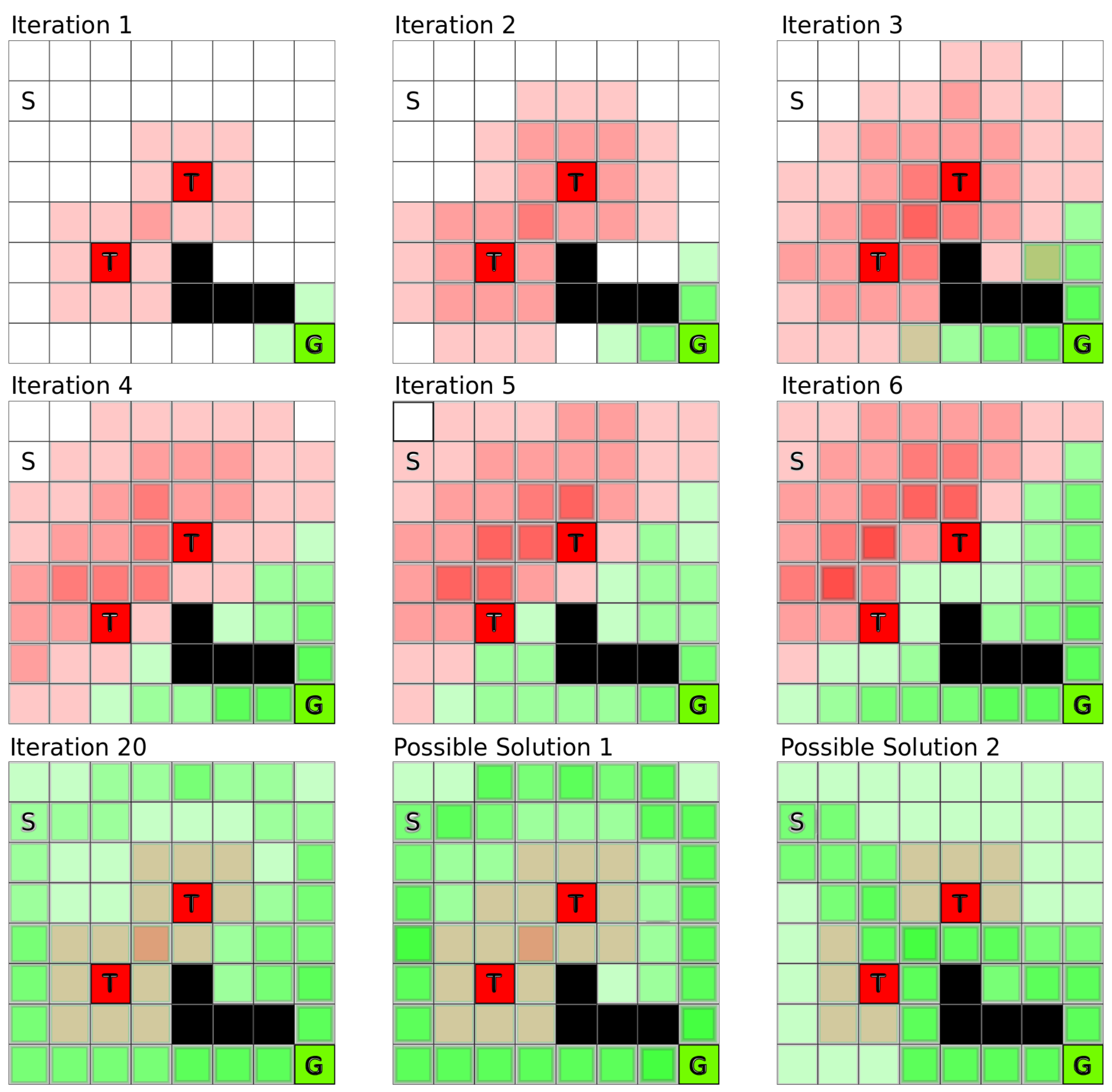
Augment reward expectations by the discounted value of the next state

$$V(s) \leftarrow V(s) + \eta \left(r + \underline{\gamma V(s')} - V(s) \right)$$



$$\gamma \in [0,1]$$

TD backups



Temporal-Difference (TD) learning

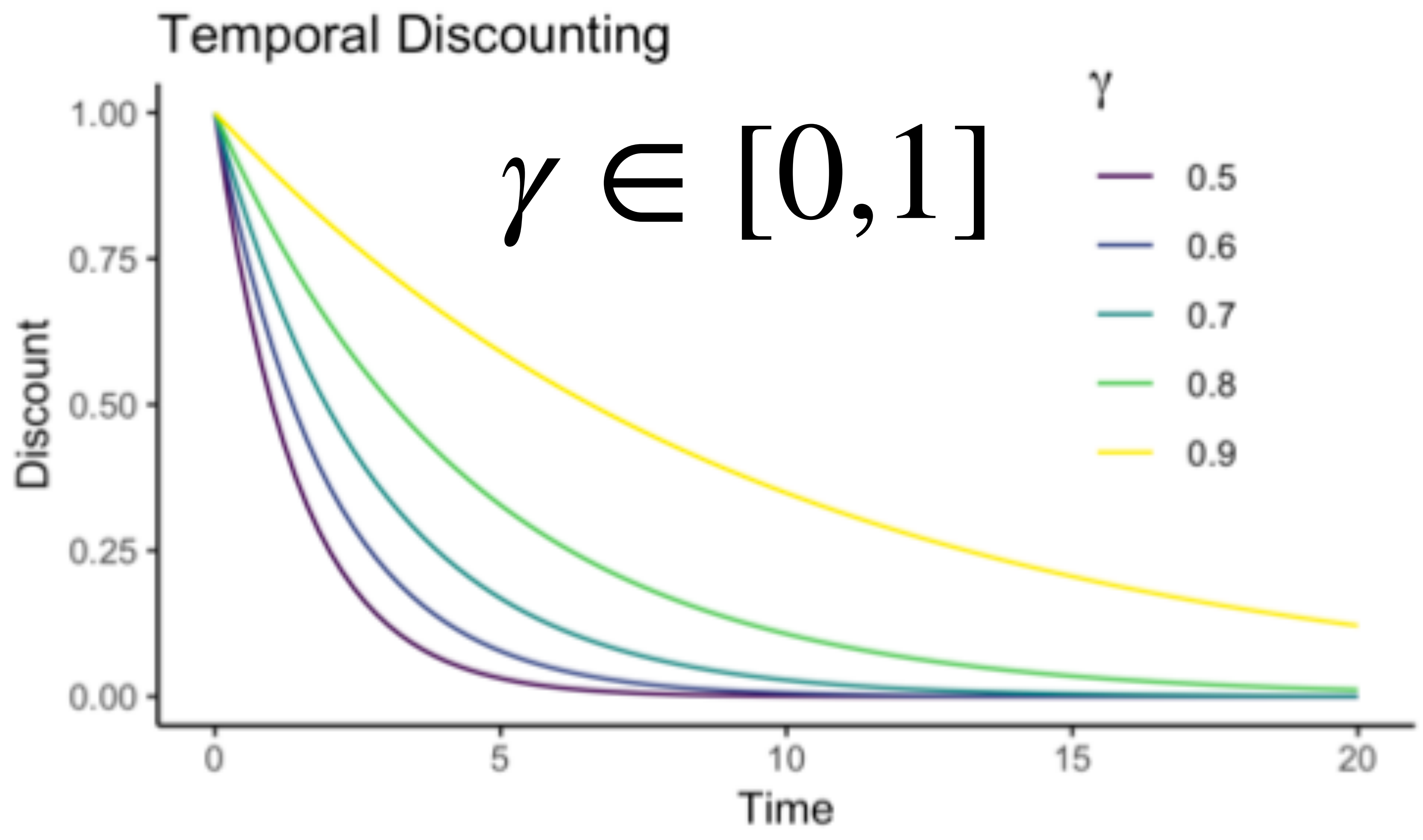
(Sutton & Barton, 1990)

Solving the credit assignment problem (Minsky, 1963):

- Which actions are responsible for (future) rewards?

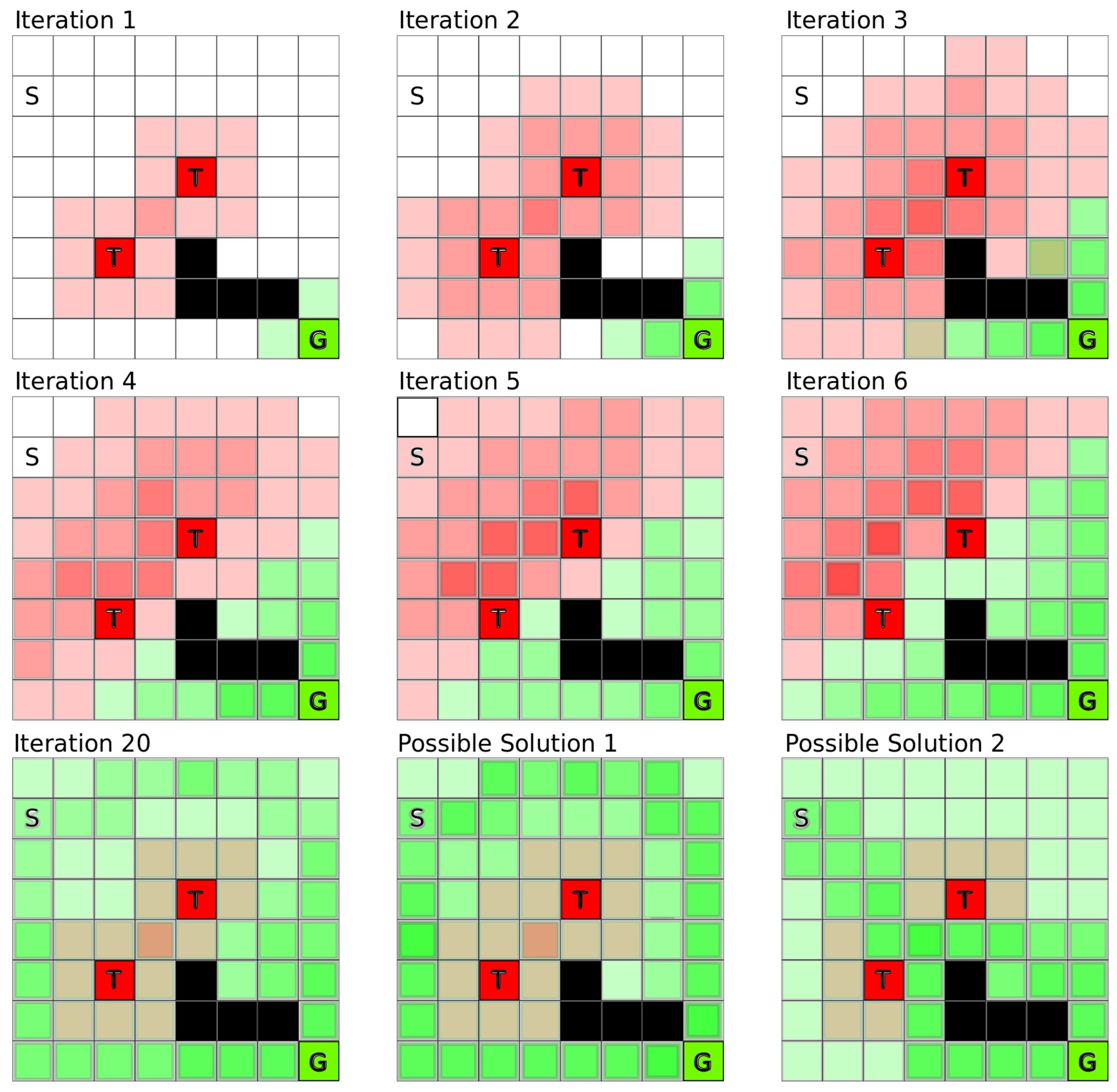
Augment reward expectations by the discounted value of the next state

$$V(s) \leftarrow V(s) + \eta \left(r + \underline{\gamma V(s')} - V(s) \right)$$

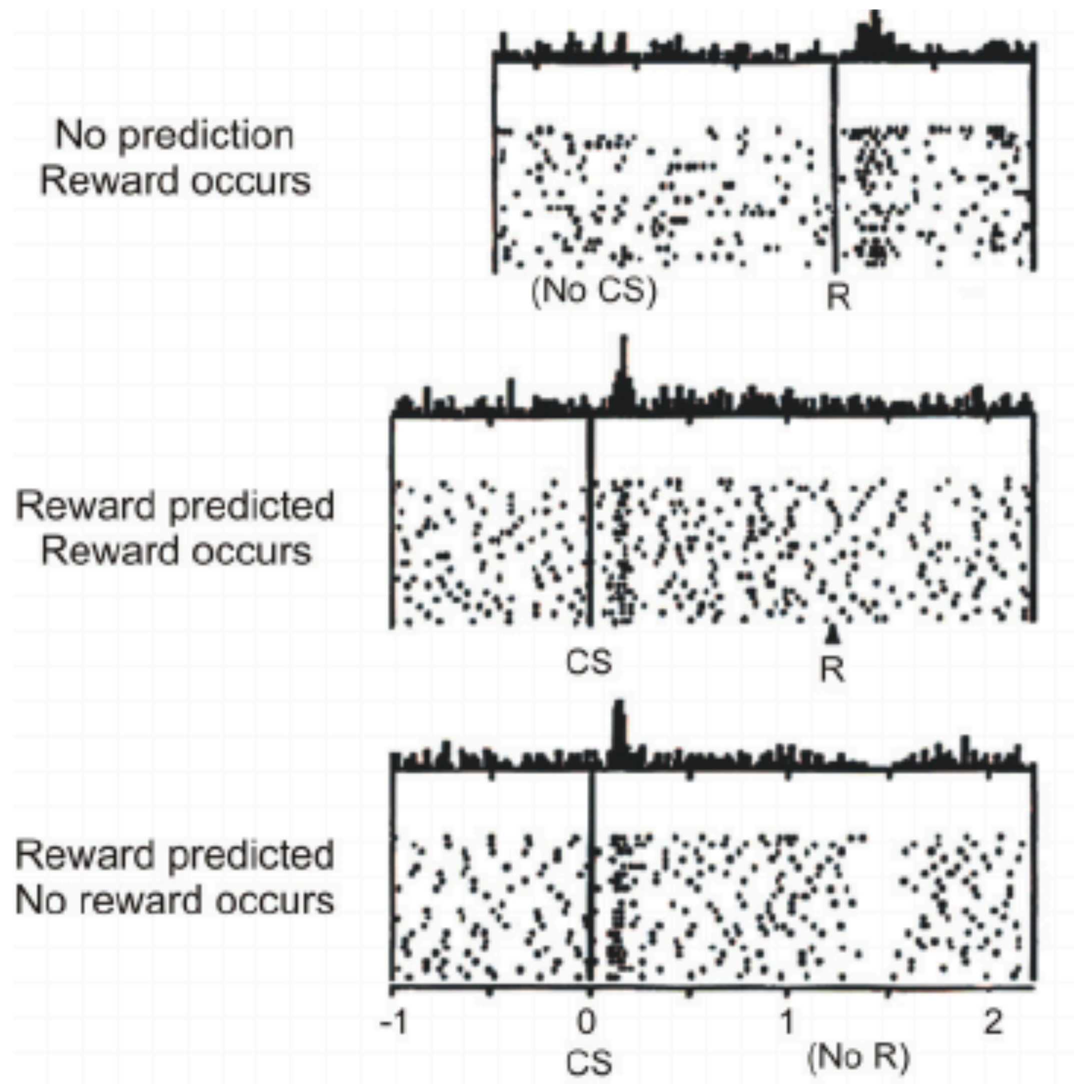


$$\gamma \in [0,1]$$

TD backups



Dopamine Reward Prediction Signal



Schultz et al. (1997)

The RL Problem

The RL Problem

Select a policy π^* that maximizes expected rewards

The RL Problem

Select a policy π^* that maximizes expected rewards

Not just immediate rewards, but discounted future returns

- Value function under some policy π:

$$V_\pi(s) = \mathbb{E}_{\tau \sim \pi}[\sum_{t \in \tau} \gamma^t R_{t+1} \,|\, s_0 = s]$$

The RL Problem

Select a policy π^* that maximizes expected rewards

Not just immediate rewards, but discounted future returns

- Value function under some policy π:

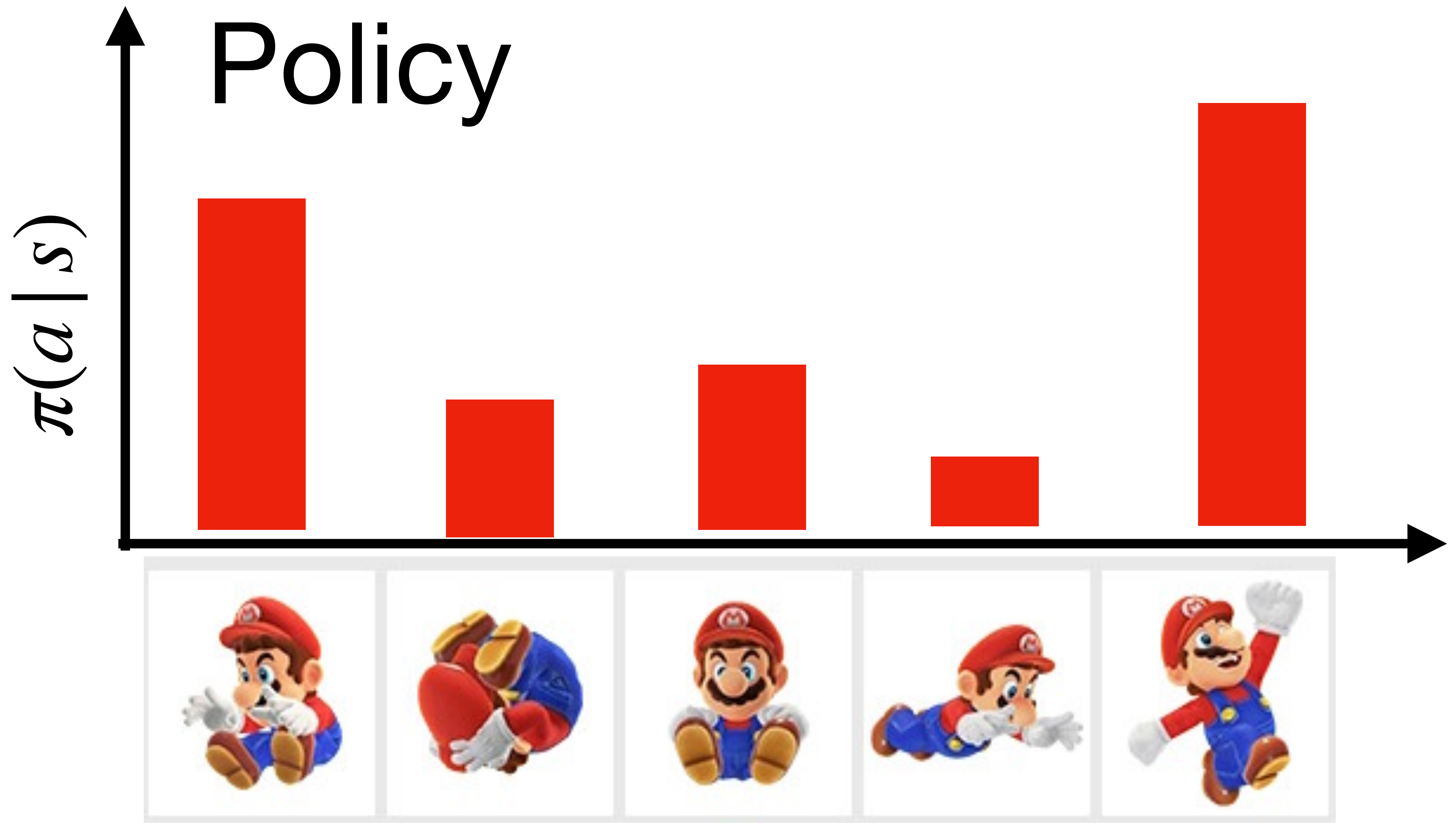
$$V_\pi(s) = \mathbb{E}_{\tau \sim \pi}[\sum_{t \in \tau} \gamma^t R_{t+1} \,|\, s_0 = s]$$

- Let's unpack that a bit:

$$V_\pi(s) = \sum_a \pi(a \,|\, s) \sum_{s'} P(s' \,|\, s, a)\big[R(s', a) + \gamma V_\pi(s')\big]$$

- The expectation $\mathbb{E}_{\tau \sim \pi}$ can be rewritten in terms of the policy and state transitions
- The sum can be written recursively as immediate reward + discounted future reward

The RL Problem



Select a policy π^* that maximizes expected rewards

Not just immediate rewards, but discounted future returns

- Value function under some policy π:

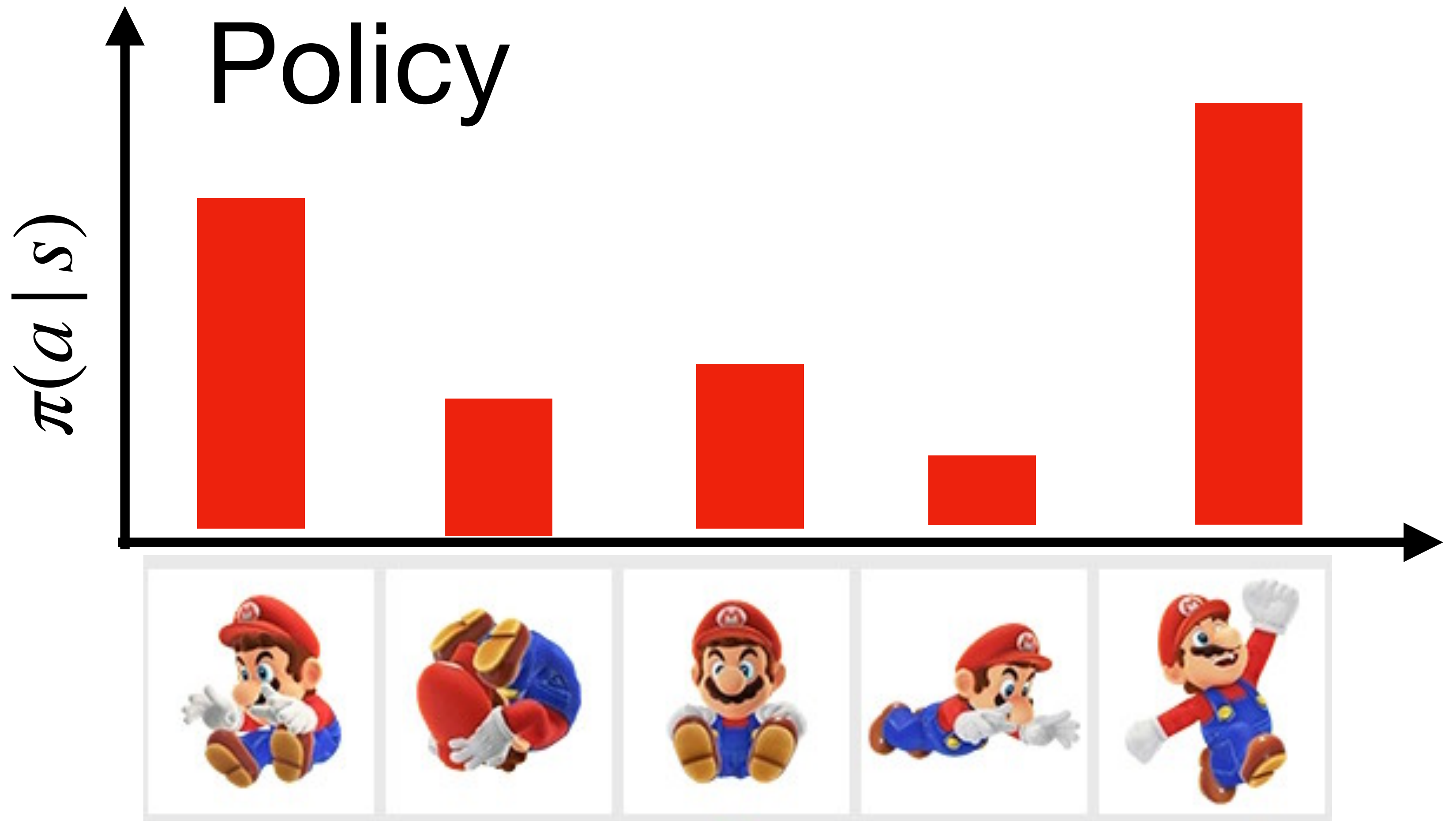
$$V_\pi(s) = \mathbb{E}_{\tau\sim\pi}[\sum_{t\in\tau} \gamma^t R_{t+1} \mid s_0 = s]$$

- Let's unpack that a bit:

$$V_\pi(s) = \sum_{a} \pi(a \mid s) \sum_{s'} P(s' \mid s, a)\left[R(s', a) + \gamma V_\pi(s')\right]$$

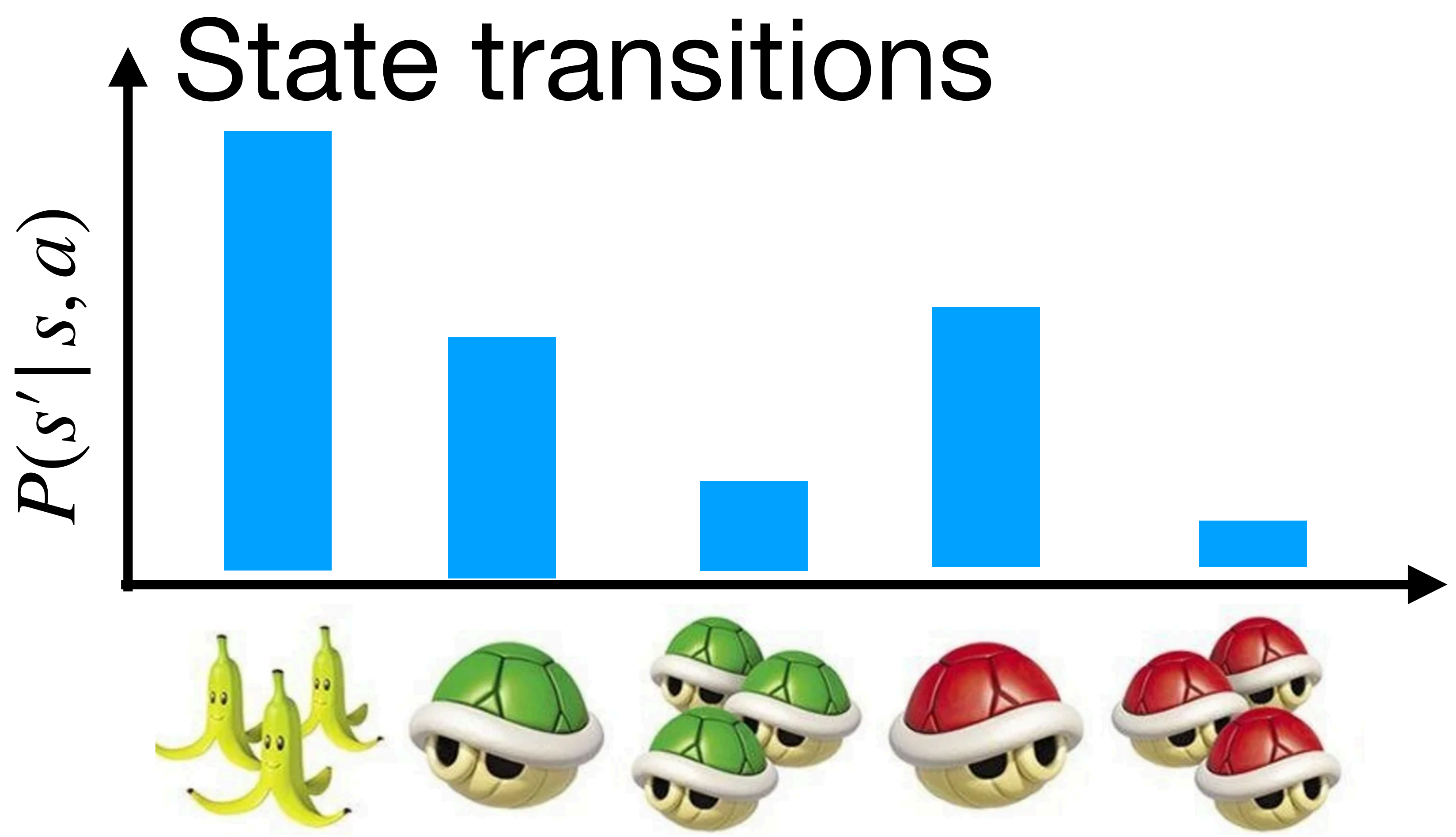
- The expectation $\mathbb{E}_{\tau\sim\pi}$ can be rewritten in terms of the policy and state transitions
- The sum can be written recursively as immediate reward + discounted future reward

The RL Problem



Select a policy π^* that maximizes expected rewards

Not just immediate rewards, but discounted future returns



- Value function under some policy π:

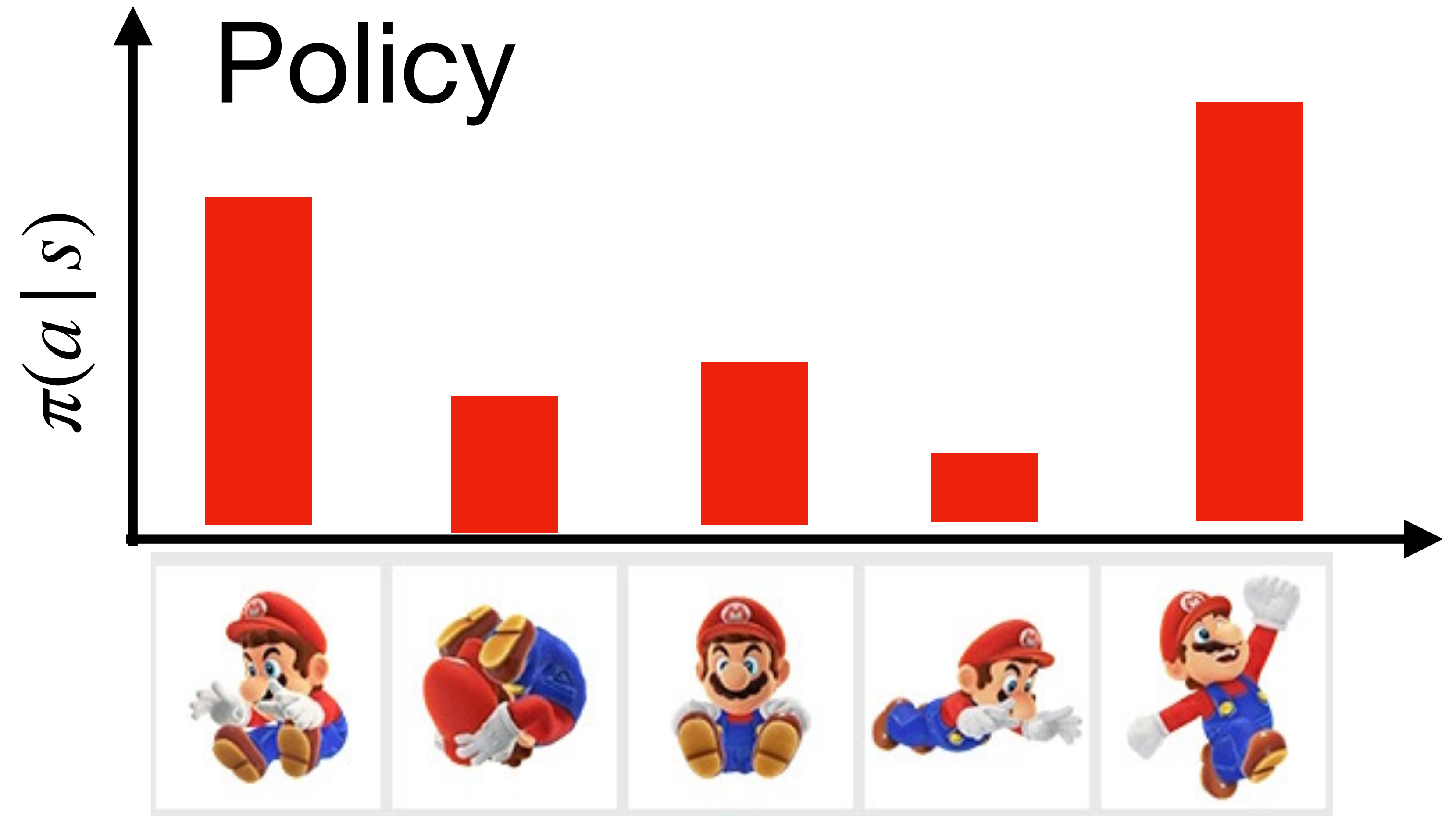
$$V_\pi(s) = \mathbb{E}_{\tau\sim\pi}[\sum_{t\in\tau} \gamma^t R_{t+1} \mid s_0 = s]$$

- Let's unpack that a bit:

$$V_\pi(s) = \sum_a \pi(a \mid s) \sum_{s'} P(s' \mid s, a)\left[R(s', a) + \gamma V_\pi(s')\right]$$

- The expectation $\mathbb{E}_{\tau\sim\pi}$ can be rewritten in terms of the policy and state transitions
- The sum can be written recursively as immediate reward + discounted future reward

The RL Problem



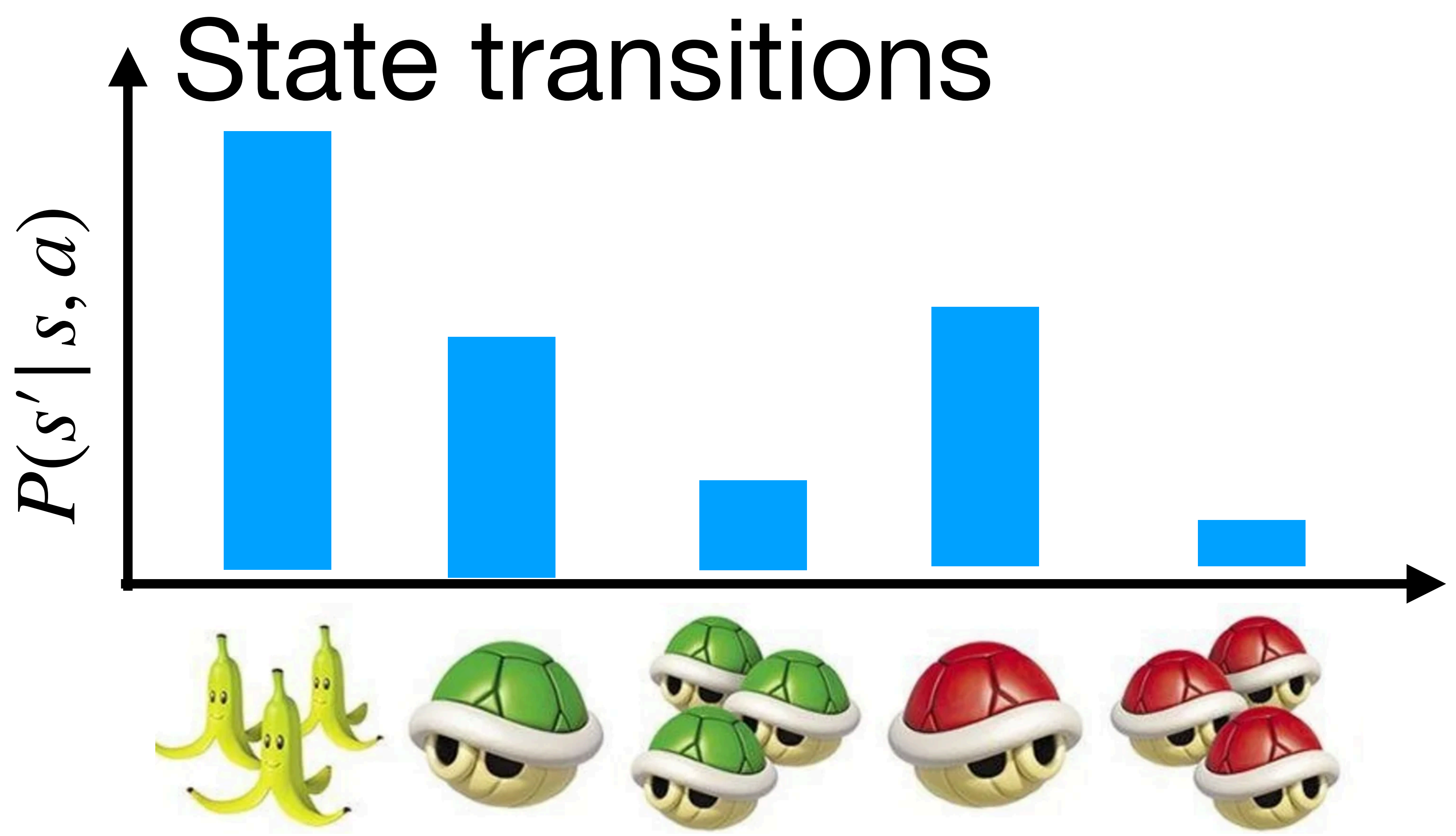
Select a policy π^* that maximizes expected rewards

Not just immediate rewards, but discounted future returns

- Value function under some policy π:

$$V_\pi(s) = \mathbb{E}_{\tau\sim\pi}[\sum_{t\in\tau} \gamma^t R_{t+1} \mid s_0 = s]$$

- Let's unpack that a bit:



$$V_\pi(s) = \sum_a \pi(a \mid s) \sum_{s'} P(s' \mid s, a) \left[R(s', a) + \gamma V_\pi(s')\right]$$

- The expectation $\mathbb{E}_{\tau\sim\pi}$ can be rewritten in terms of the policy and state transitions
- The sum can be written recursively as immediate reward + discounted future reward

Optimal policies via Bellman Equations

- This recursive formulation of the value function is known as the Bellman equation

$$V_\pi(s) = \sum_a \pi(a|s) \sum_{s'} P(s'|s,a)\left[R(s',a) + \gamma V_\pi(s')\right]$$

  - This allows us to break the optimization problem into series of simpler sub-problems
  - if each sub-problem is solved optimally, the overall problem will also be optimal
- Theoretically optimal solution:
  - We first define an optimal value function by assuming value-maximizing actions:

$$V_*(s) = \arg\max_a \sum_{s'} P(s'|s,a)\left[R(s,a) + \gamma V_*(s')\right]$$

- We then (theoretically) arrive at an optimal policy by selecting actions that maximize value:

$$\pi_* = \arg\max_a V_*(s)$$

Optimal policies via Bellman Equations

- This recursive formulation of the value function is known as the Bellman equation

$$V_\pi(s) = \sum_a \pi(a \mid s) \sum_{s'} P(s' \mid s, a)\left[R(s', a) + \gamma V_\pi(s')\right]$$

  - This allows us to break the optimization problem into series of simpler sub-problems
  - if each sub-problem is solved optimally, the overall problem will also be optimal
- Theoretically optimal solution:
  - We first define an optimal value function by assuming value-maximizing actions:

$$V_*(s) = \arg\max_a \sum_{s'} P(s' \mid s, a)\left[R(s, a) + \gamma V_*(s')\right]$$

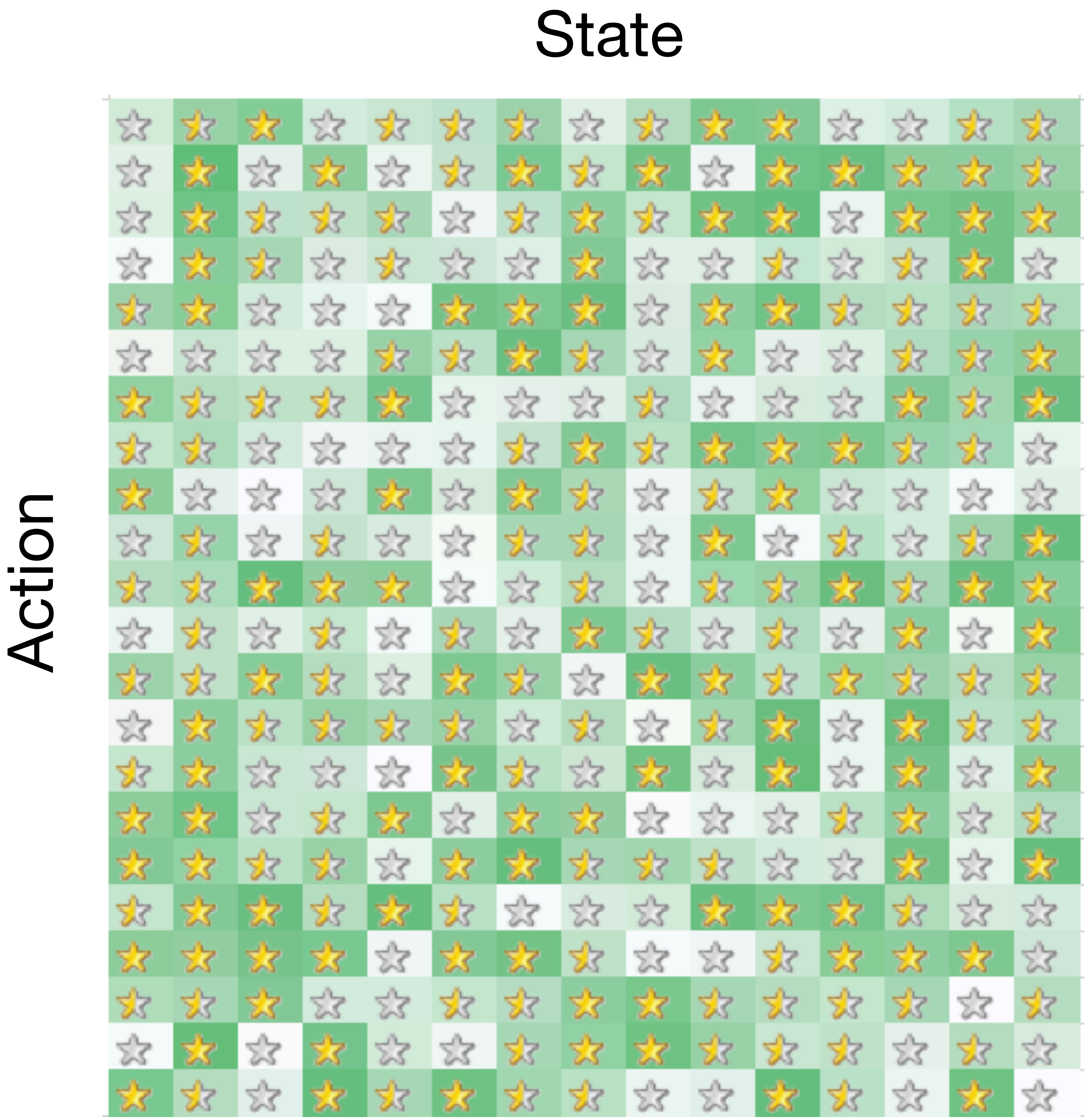
- We then (theoretically) arrive at an optimal policy by selecting actions that maximize value:

$$\pi_* = \arg\max_a V_*(s)$$

* In practice, optimal solutions are often unobtainable

[If time] Tabular methods for finding optimal policies

- Based on methods from Dynamic programming (Bellman, 1957), Tabular methods were first proposed as solutions for RL problems by Minsky (1961)
- Think of a giant lookup table, where we store a value representation for each combination of state+action
- **Value iteration** and **policy iteration** are examples
- Caveat: solutions require repeat visits to each state, which is infeasible in most real-world problems



Value iteration

Iteratively visit all states and update the value function until a “good enough” solution has been reached.

1. Initialize the value function as $V_0(s) = 0$ for all states

2. For all s in $\mathcal{S}$:

$$V_{k+1}(s) = \max_{a \in A} \sum_{s'} P(s' \mid s, a)[R(s, a) + \gamma V_k(s')]$$

until $\max_{s \in \mathcal{S}} \mid V_k(s) - V_{k-1}(s) \mid < \theta$ Bellman residual

Value iteration

Iteratively visit all states and update the value function until a "good enough" solution has been reached.

1. Initialize the value function as $V_0(s) = 0$ for all states

2. For all s in $\mathcal{S}$:

$$V_{k+1}(s) = \max_{a \in A} \sum_{s'} P(s' \mid s, a)[R(s, a) + \gamma V_k(s')]$$

until $\max_{s \in \mathcal{S}} |V_k(s) - V_{k-1}(s)| < \theta$ Bellman residual

V_k converges on V_* as $k \to \infty$, and perhaps sooner, but with many costly sweeps through the state space

Policy iteration

Alternate between evaluating a policy and then improving the policy.

Start with π_0 (typically a random policy), and then iterate for all $s \in \mathcal{S}$ in each step

- **Policy Evaluation**

$$V_{\pi_k}(s) = \mathbb{E}_{\pi_k}\Big[R(s', a) + \gamma V_{\pi_k}(s')\Big]$$

- **Policy Improvement**

$$\pi_{k+1} = \arg\max_a \sum_{s'} P(s' \mid s, a)\Big[R(s, a) + \gamma V_{\pi_k}\Big]$$

Policy iteration

Alternate between evaluating a policy and then improving the policy.

Start with π_0 (typically a random policy), and then iterate for all $s \in \mathcal{S}$ in each step

- **Policy Evaluation**

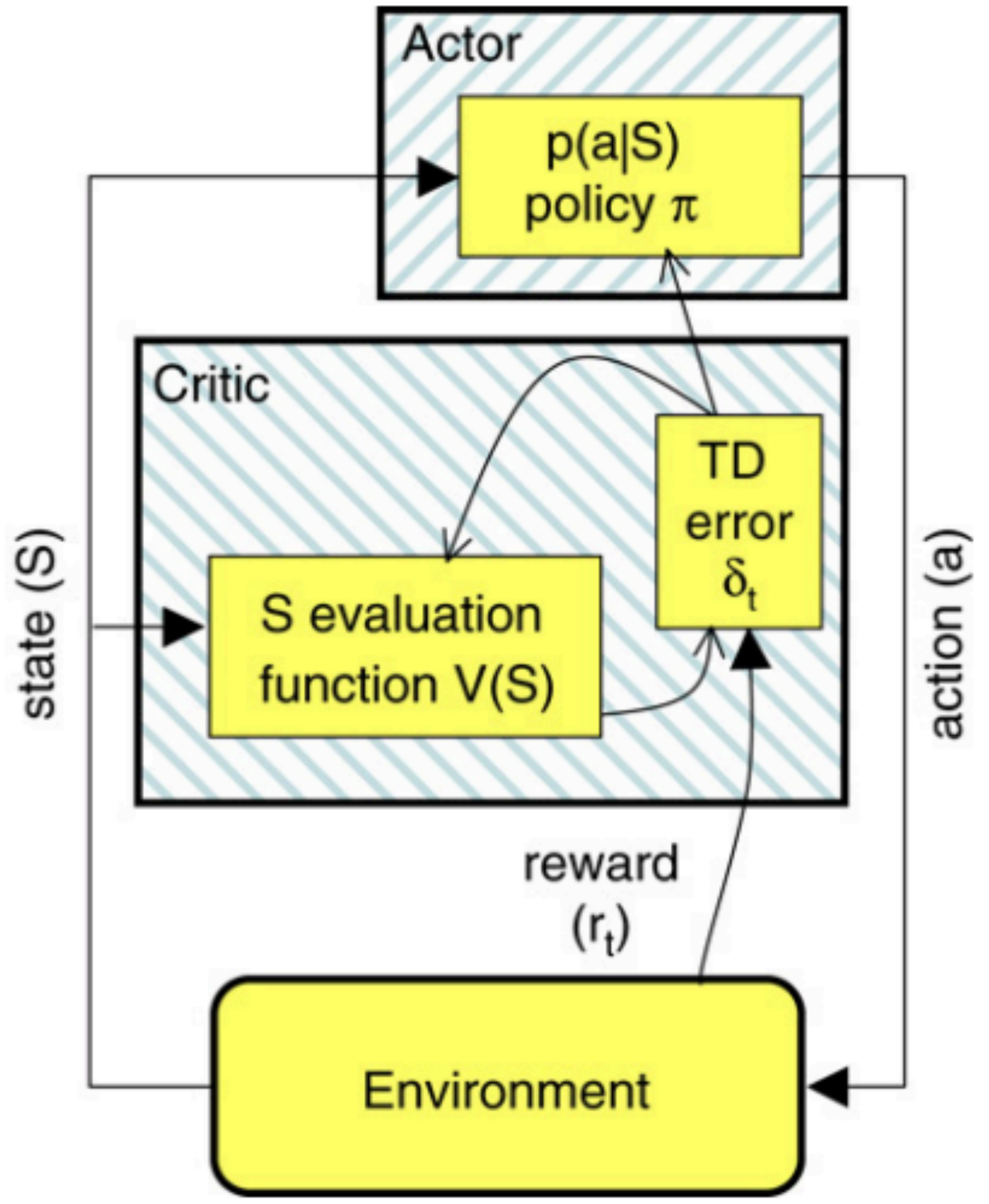
$$V_{\pi_k}(s) = \mathbb{E}_{\pi_k}\left[R(s', a) + \gamma V_{\pi_k}(s')\right]$$

- **Policy Improvement**

$$\pi_{k+1} = \arg\max_a \sum_{s'} P(s' \mid s, a)\left[R(s, a) + \gamma V_{\pi_k}\right]$$

Policy can converge faster than value function, but still requires visiting all states multiple times and lacks convergence guarantees

Actor Critic



We've already defined value updates in terms of RPE

$$V(s) \leftarrow V(s) + \eta\delta_t$$

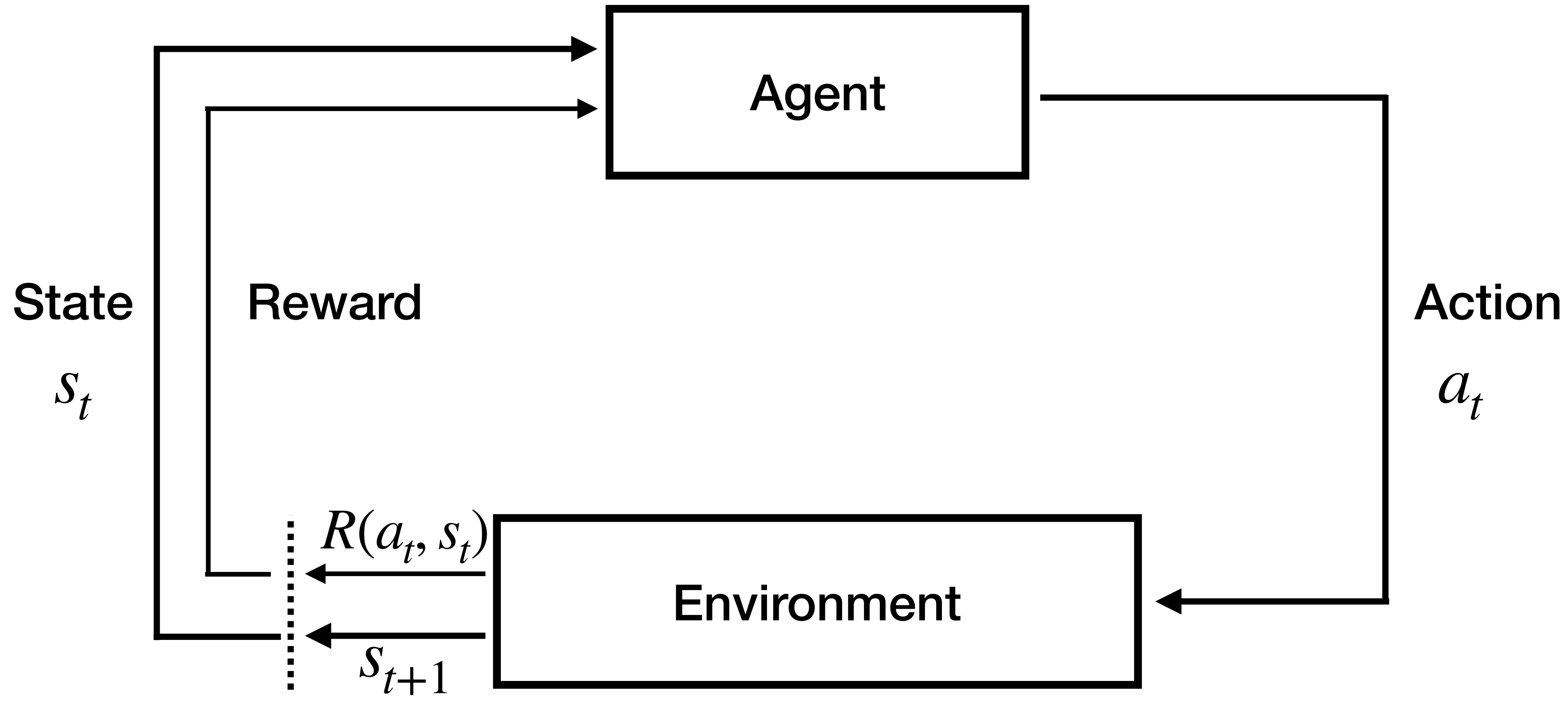
We can use a similar learning rule to update a policy

$$\pi(s, a) \leftarrow \pi(s, a) + \eta_\pi\delta_t$$

Policy is learned gradually rather than an argmax

Similar to modern policy-gradient methods used in many Deep RL contexts

RL summary



- *Normative* framework for learning an optimal policy π^* in arbitrarily complex environments
  - With modern bells and whistles, is able to beat human-experts in a variety of domains
- Also provides a *descriptive* model of human learning
  - TD-learning prediction error tracks dopamine signals in the brain (more on this next week)
  - Value representations and policies seem to capture psychologically relevant representations
  - But where is the map? Where is the model of the environment?

5 minute break

Goals and habits (Dolan & Dayan 2013)

Goals and habits (Dolan & Dayan 2013)

Goal-directed actions

Goals and habits (Dolan & Dayan 2013)

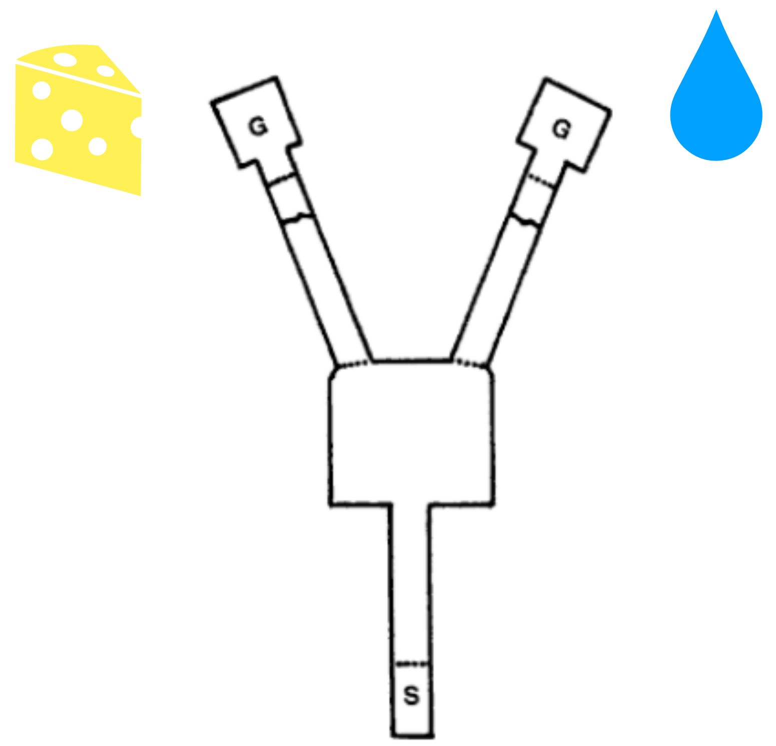
Goal-directed actions

1. Knowledge of how actions map to consequences
   Reward-Outcome (R-O) control

Goals and habits (Dolan & Dayan 2013)

Goal-directed actions

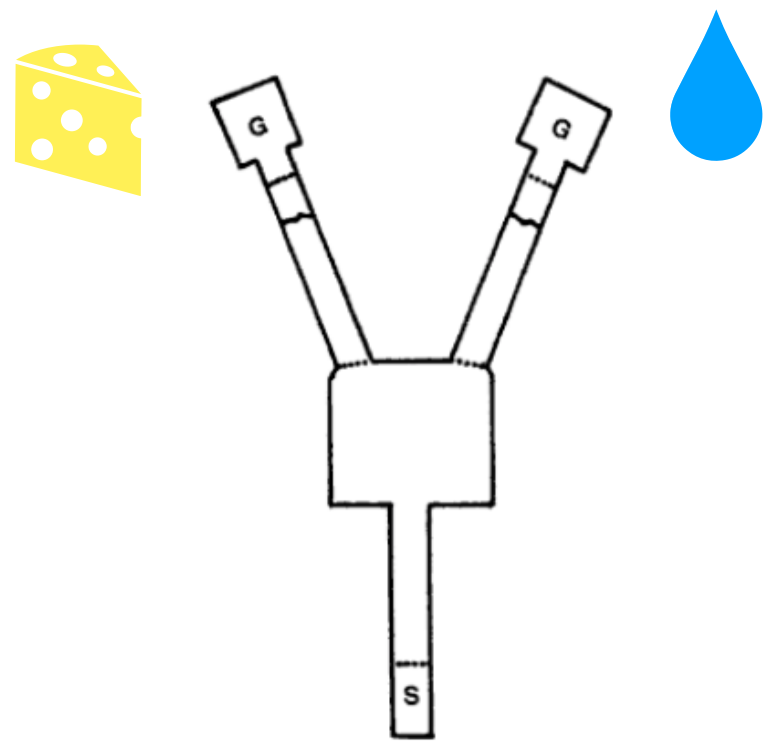
1. Knowledge of how actions map to consequences
   Reward-Outcome (R-O) control
2. Outcomes should be motivationally relevant
   e,.g., food when hungry, water when thirsty



Goals and habits (Dolan & Dayan 2013)

Goal-directed actions

1. Knowledge of how actions map to consequences
   Reward-Outcome (R-O) control
2. Outcomes should be motivationally relevant
   e,.g., food when hungry, water when thirsty

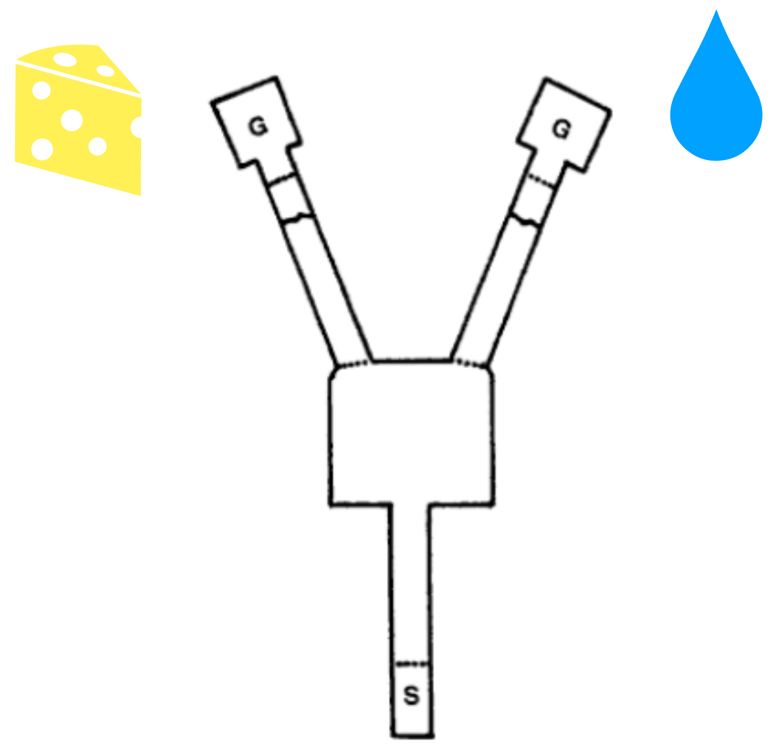


Habitual actions

Goals and habits (Dolan & Dayan 2013)

Goal-directed actions

1. Knowledge of how actions map to consequences
   Reward-Outcome (R-O) control
2. Outcomes should be motivationally relevant
   e,.g., food when hungry, water when thirsty



Habitual actions

- Instrumental responding, even when actions are not motivationally relevant

Model-free RL ⟷ Model-based RL

Model-free RL

- Habit
- Cheap
- $Q(s, a)$
- Myopically selecting actions that have been associated with reward

Model-based RL

- Goal-directed
- Computationally costly
- $P(s', r \mid s, a)$
- Planning and seeking of long term outcomes

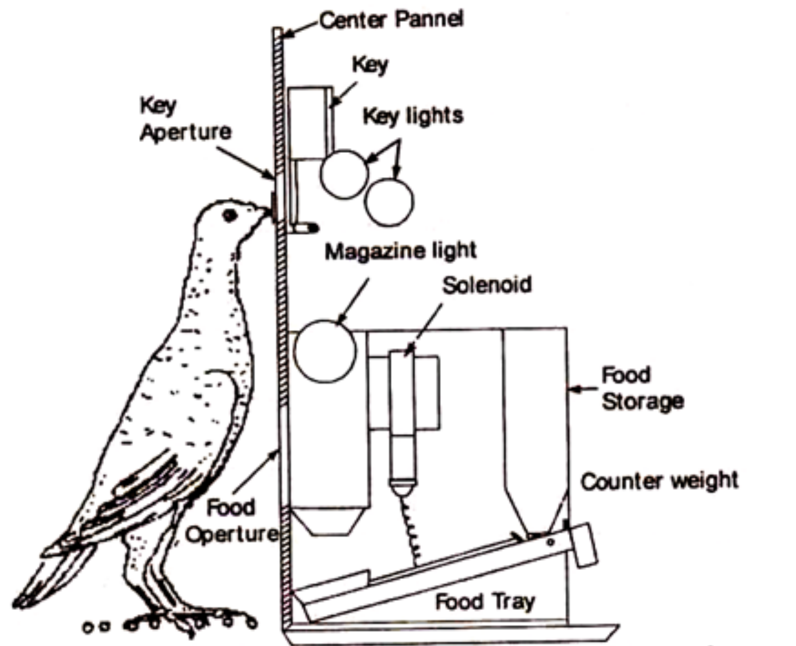
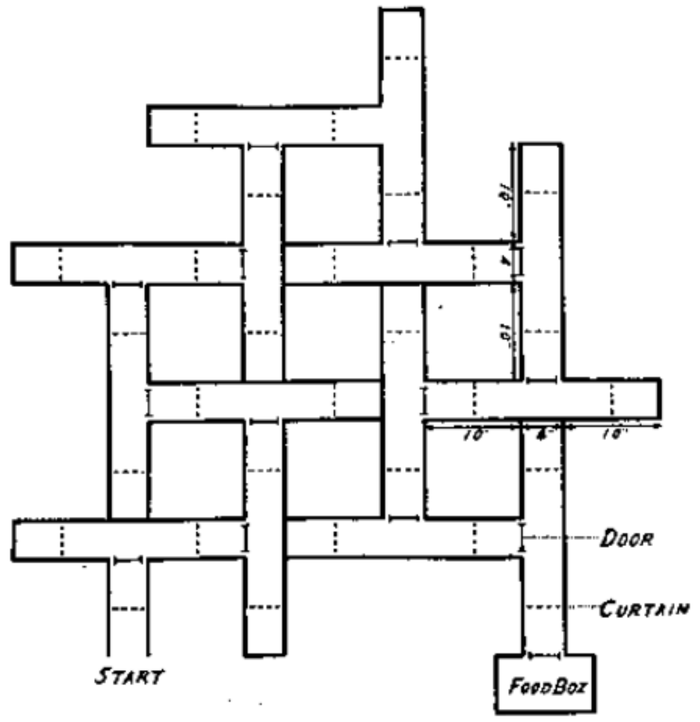
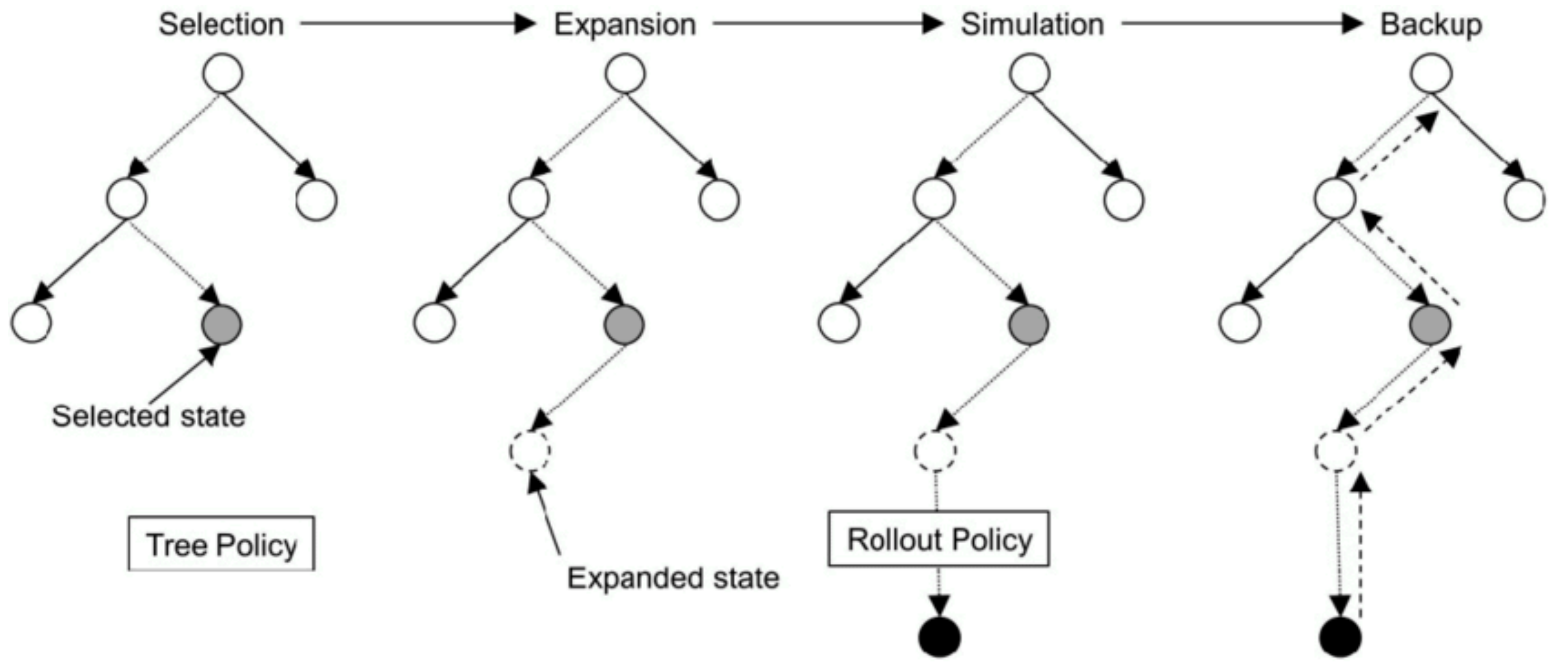


Illustration. Skinner box as adapted for the pigeon.

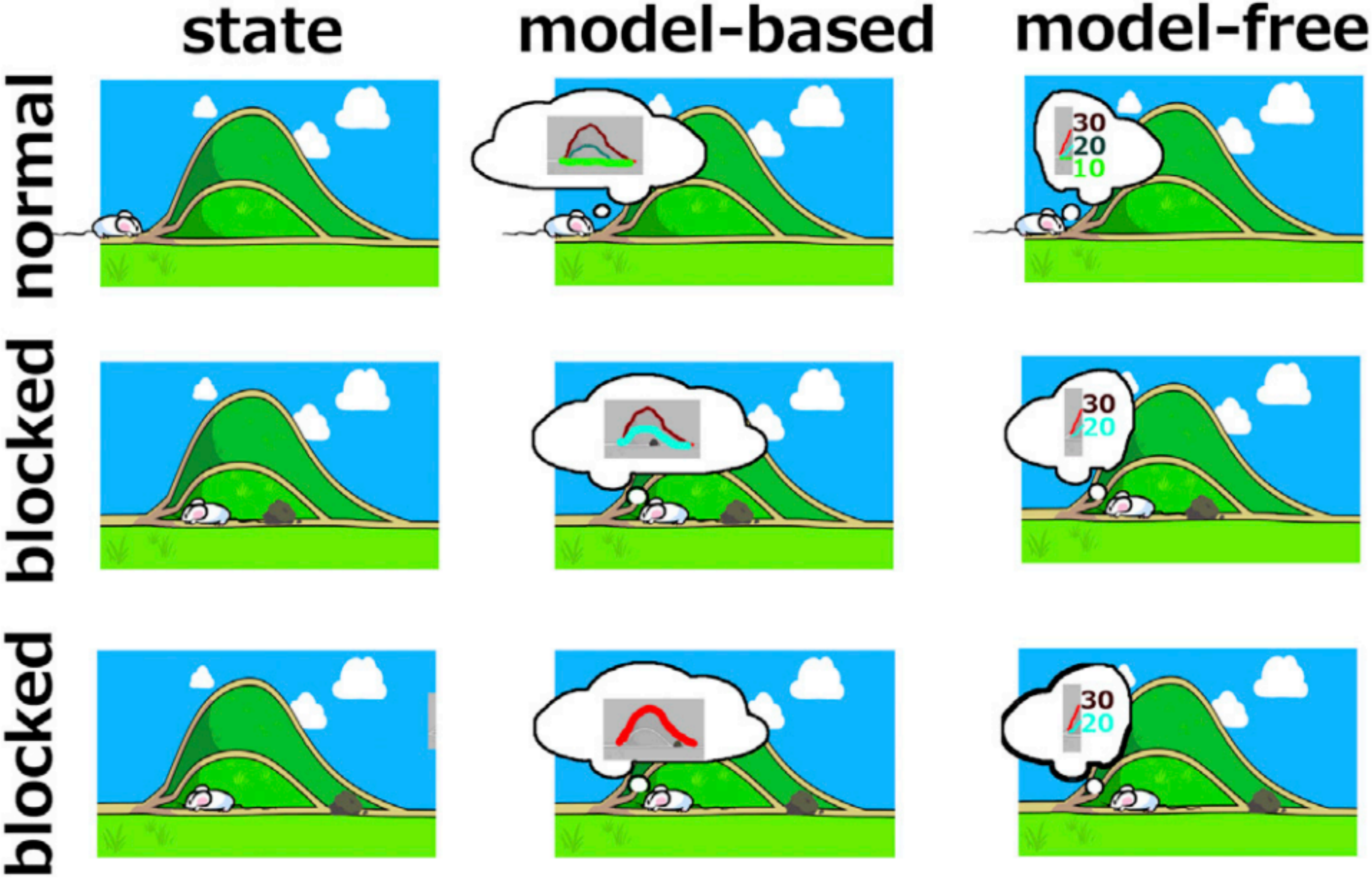


Plan of maze
14-Unit T-Alley Maze
Fig. 1
(From M. H. Elliott, The effect of change of reward on the maze performance of rats. *Univ. Calif. Publ. Psychol.*, 1928, 4, p. 20.)

Monte carlo tree search



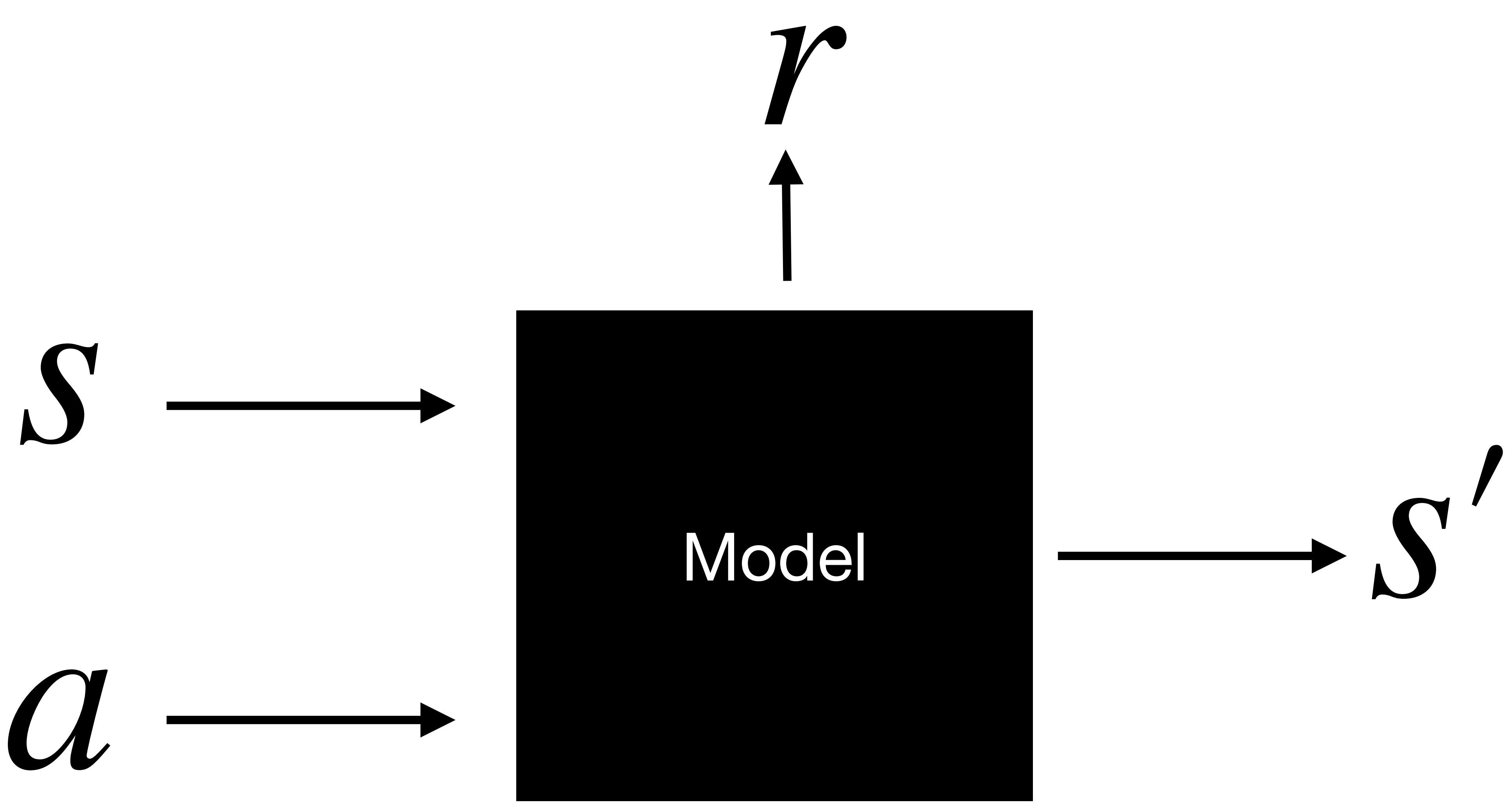
Duarte et al,. (2020)


state
model-based
model-free
normal
blocked
blocked
30
20
10
30
20
30
20

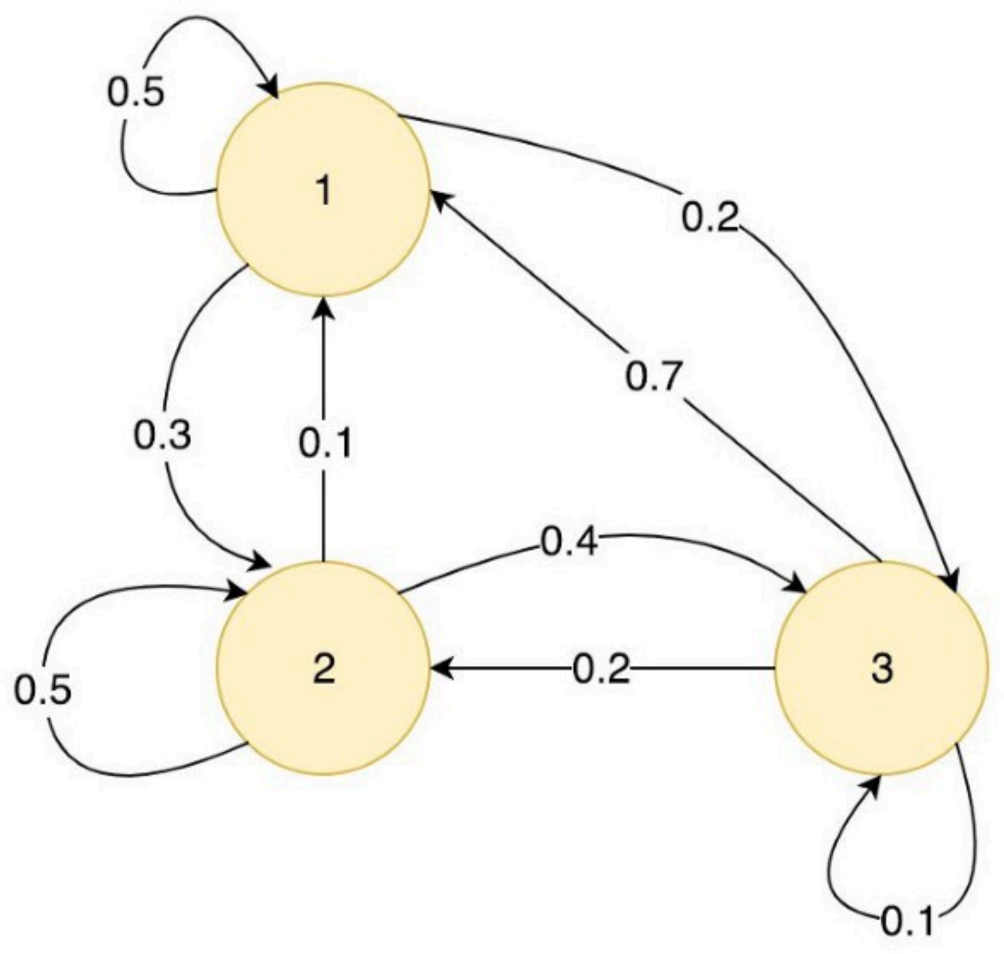
What is the model in model-based RL?

Ingredients:

- Transition model T
- Reward function R
- State space $s \in \mathcal{S}$
- Action space $a \in \mathcal{A}$

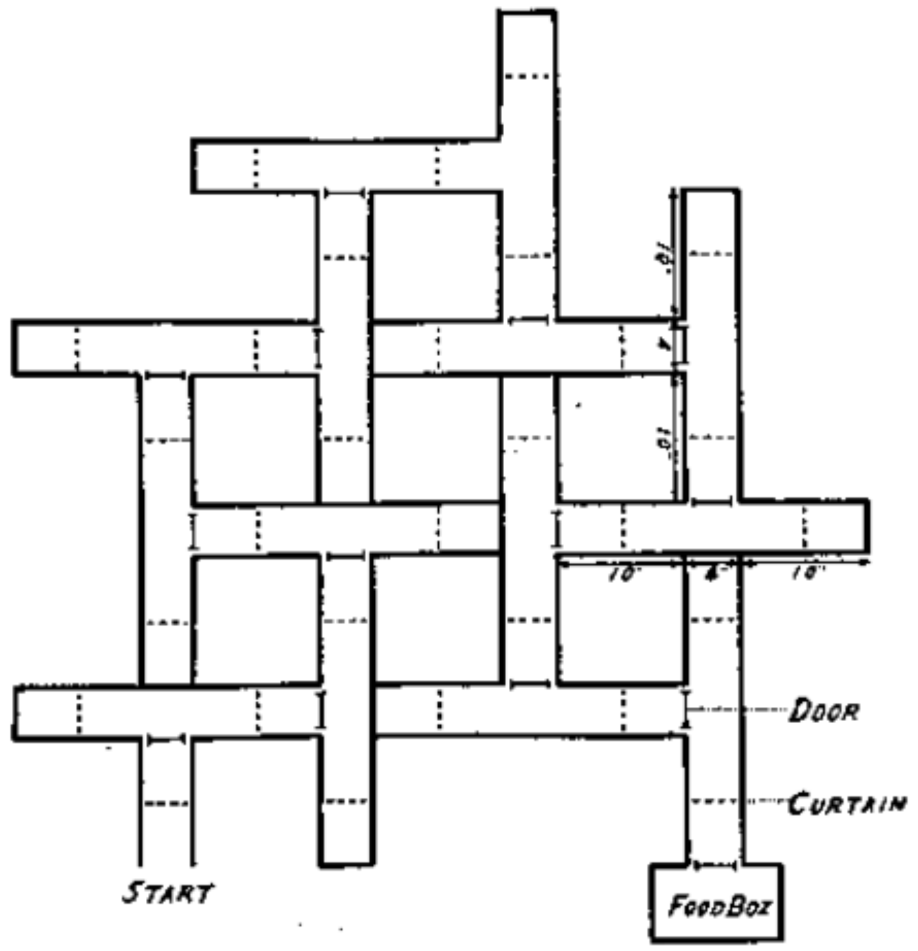


MDP



Transition Matrix

$$\begin{array}{cc} & \begin{array}{ccc} S_1 & S_2 & S_3 \end{array} \\ \begin{array}{c} S_1 \\ S_2 \\ S_3 \end{array} & \begin{bmatrix} 0.5 & 0.1 & 0.7 \\ 0.3 & 0.5 & 0.2 \\ 0.2 & 0.4 & 0.1 \end{bmatrix} \end{array}$$

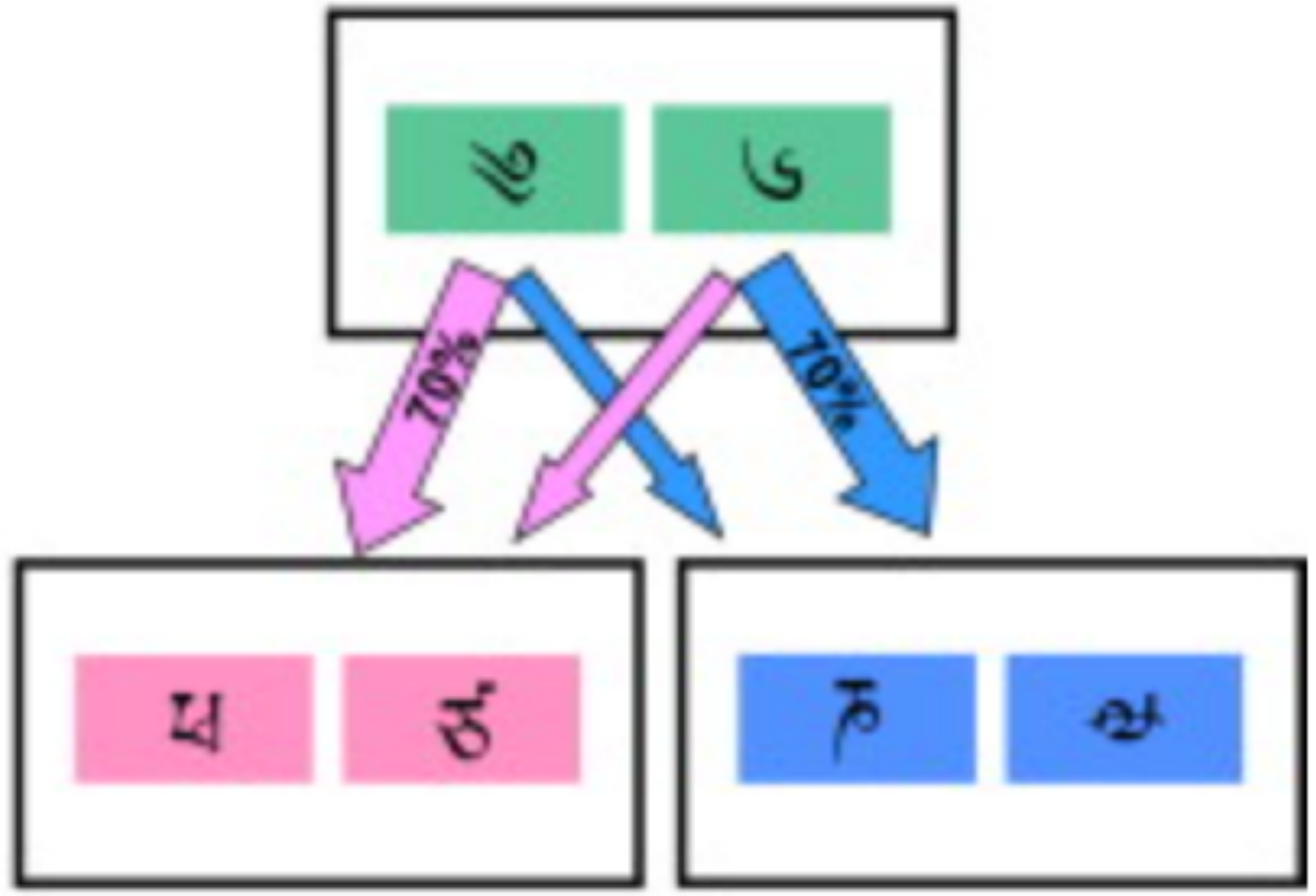


Plan of maze
14-Unit T-Alley Maze
FIG. 1
(From M. H. Elliott, The effect of change of reward on the maze performance of rats. *Univ. Calif. Publ. Psychol.*, 1928, 4, p. 20.)

Two-step task

- Two-stage decision-making task used to distinguish model-free vs. model-based learning

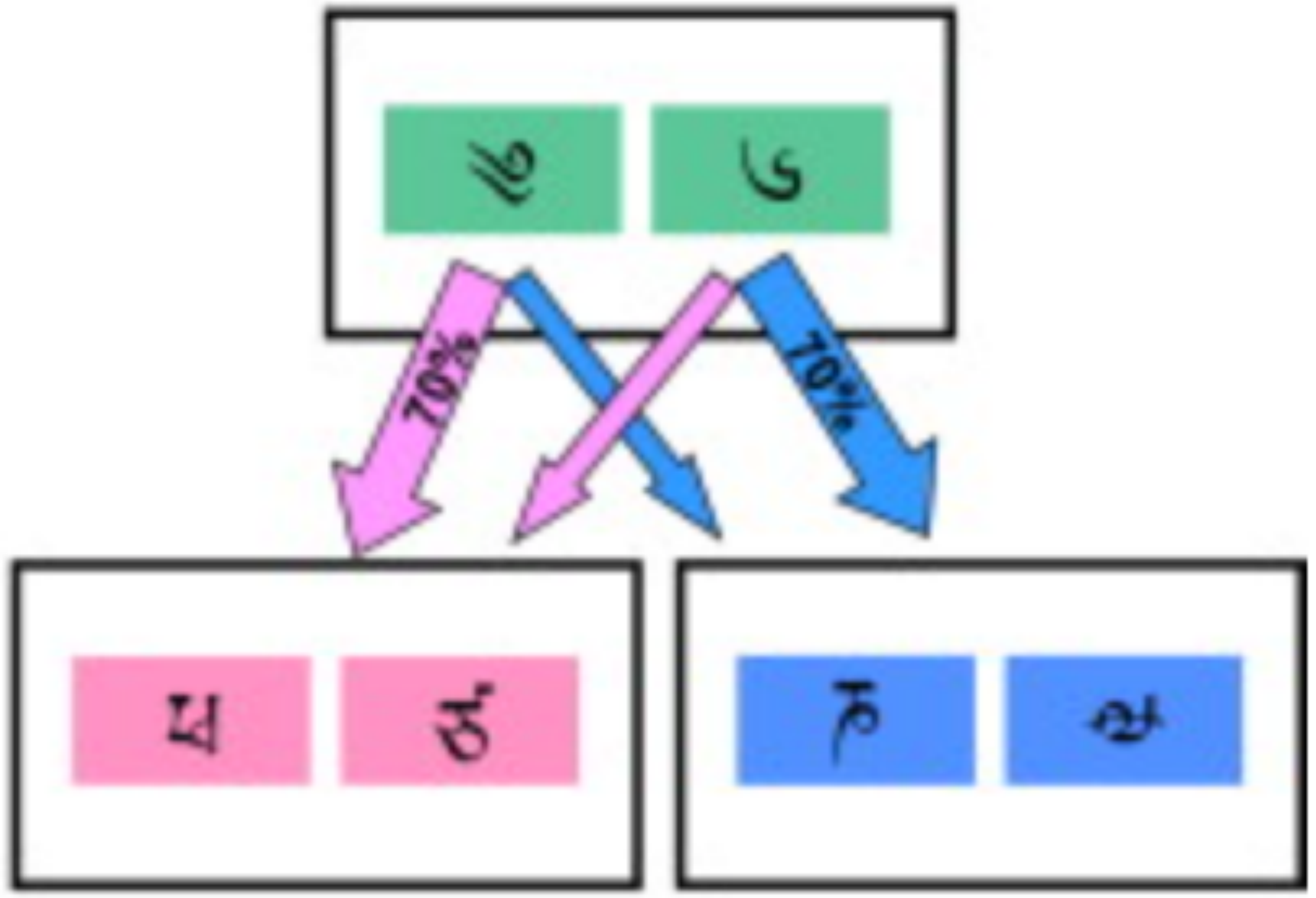
- Rewards of second-stage options changed slowly following a random walk



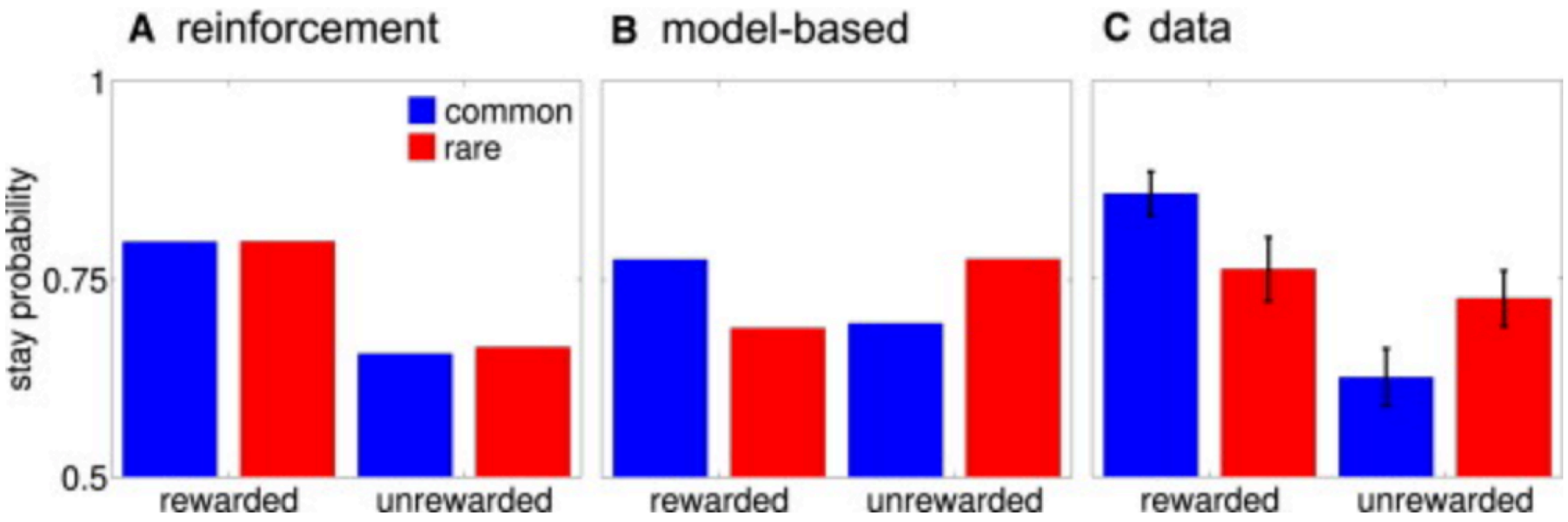
Daw et al., (2011)

Two-step task

- Two-stage decision-making task used to distinguish model-free vs. model-based learning
- Rewards of second-stage options changed slowly following a random walk



- (model-free) RL predictions depend solely on reward
- Model-based RL uses the transition structure
- Data suggests a mixture of both



Wu, Veléz, & Cushman (2022)

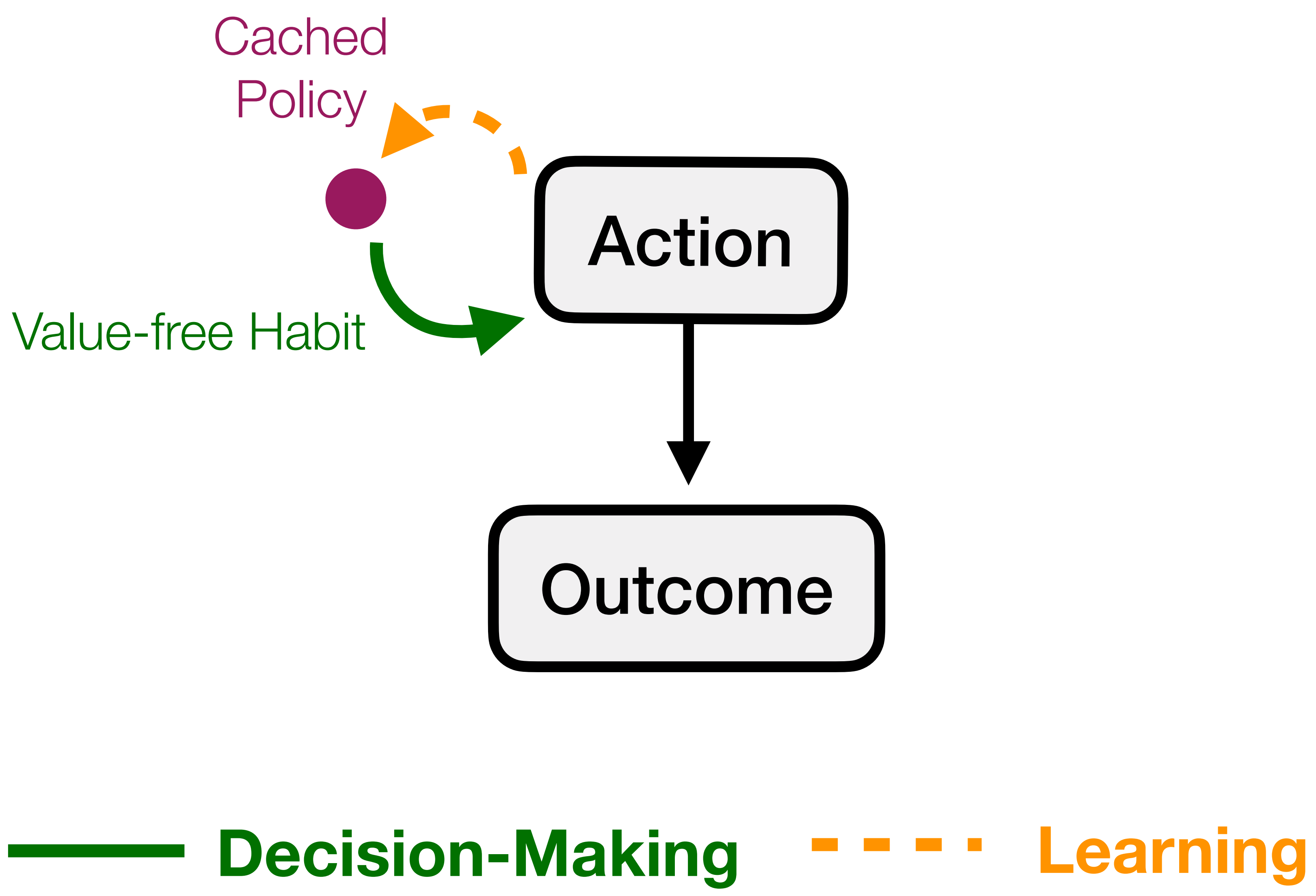
Model-based planning can inform model-free learning

Hierarchy of learning:

Model-based planning can inform model-free learning

Hierarchy of learning:

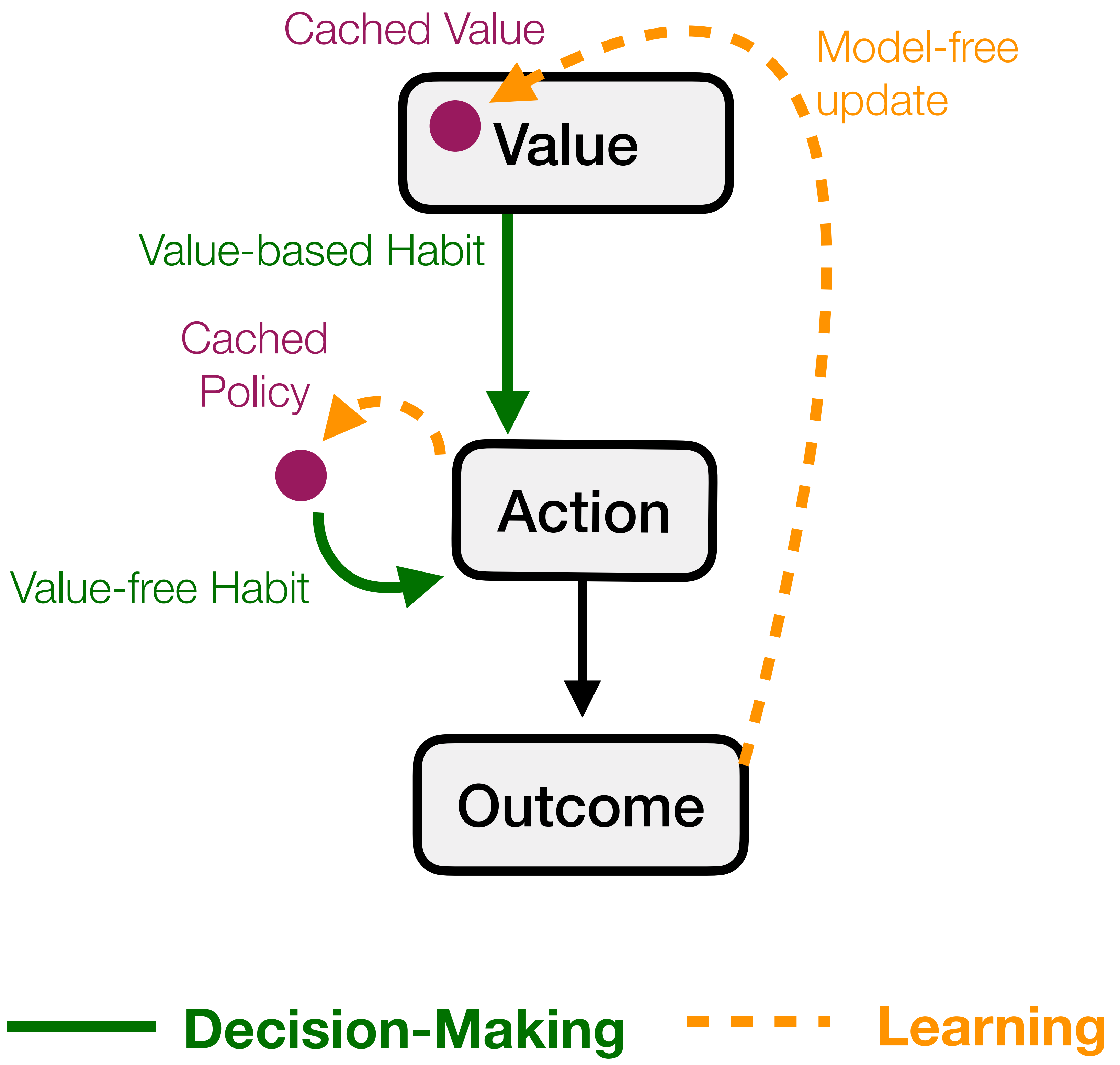
- **Value-free habit**: deploy a *cached policy* by repeating actions performed in the past
  (Thorndike, 1932; Cushman & Morris, 2015; Daw et al., 2005; Gershman, 2020)



Model-based planning can inform model-free learning

Hierarchy of learning:

- **Value-free habit**: deploy a *cached policy* by repeating actions performed in the past
  (Thorndike, 1932; Cushman & Morris, 2015; Daw et al., 2005; Gershman, 2020)
- **Value-based habit**: use a *cached action value* for more flexibility
  (Botvinick & Weinstein, 2014; Keramati et al., 2016; Maisto et al., 2019)



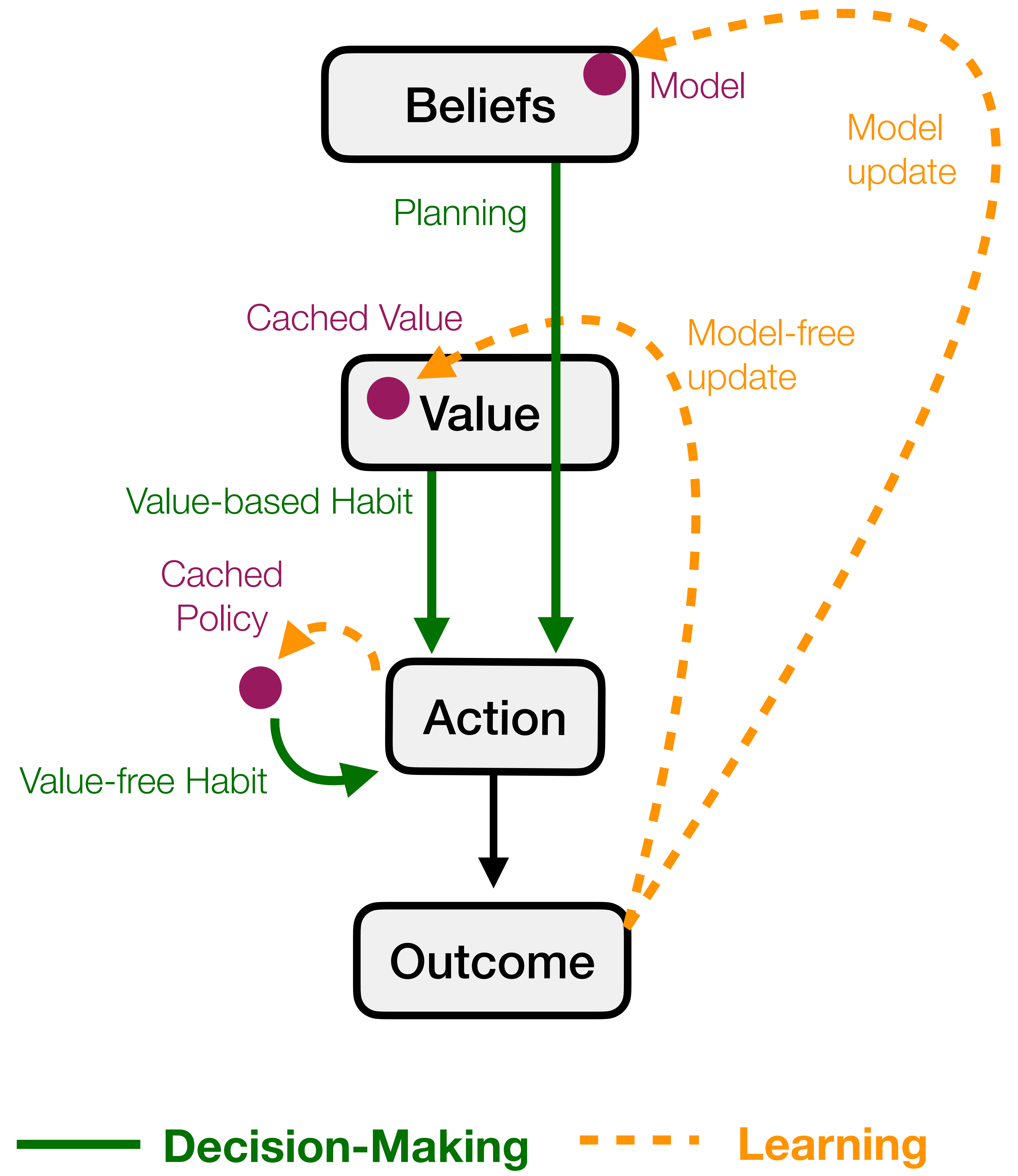
Model-based planning can inform model-free learning

Hierarchy of learning:

- **Value-free habit**: deploy a *cached policy* by repeating actions performed in the past
(Thorndike, 1932; Cushman & Morris, 2015; Daw et al., 2005; Gershman, 2020)

- **Value-based habit**: use a *cached action value* for more flexibility
(Botvinick & Weinstein, 2014; Keramati et al., 2016; Maisto et al., 2019)

- **Model-based planning:** Select actions expected to produced the best outcomes based on our model of the world
(K. J. Miller et al., 2017; Vikbladh et al., 2019)

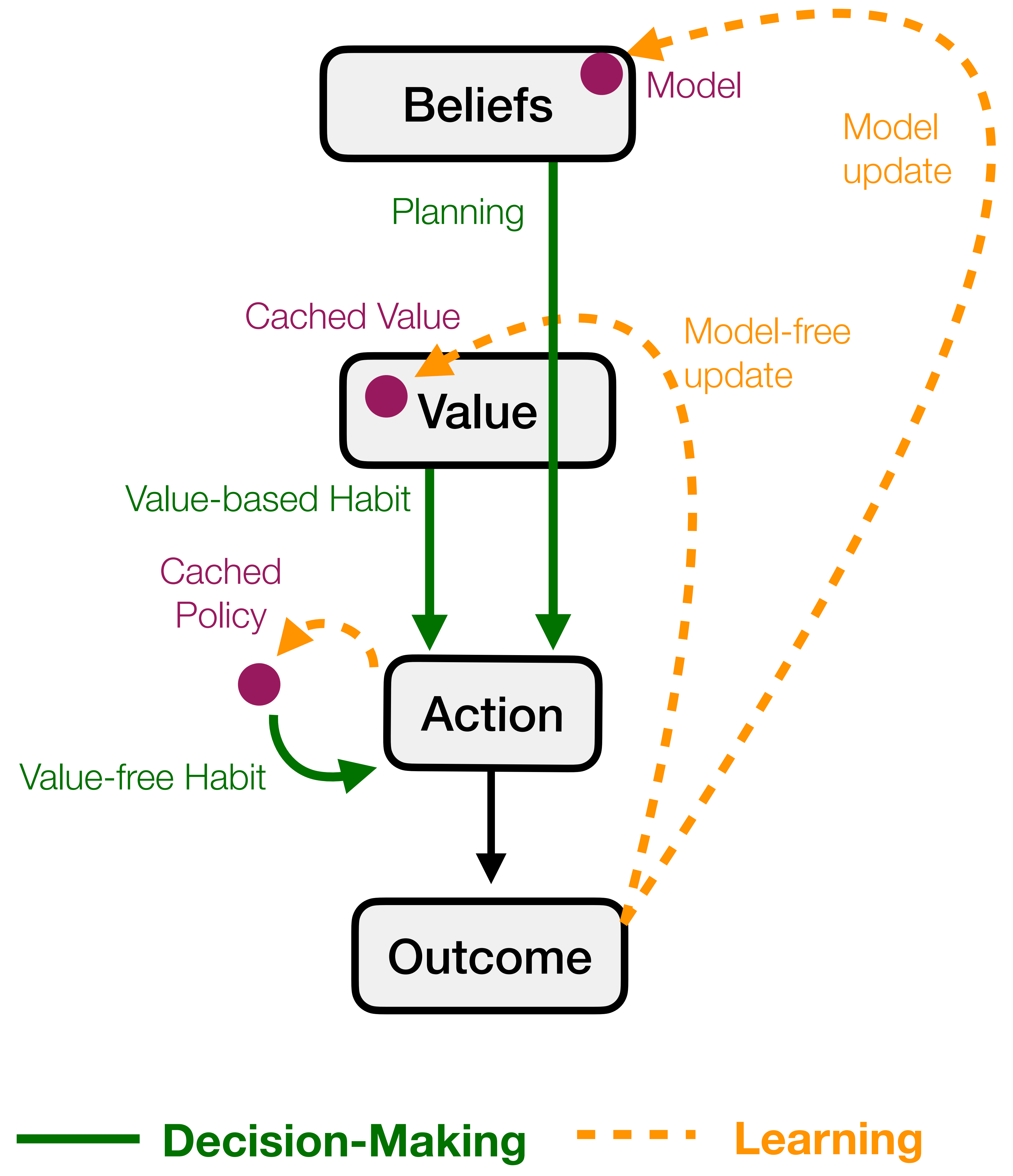


Model-based planning can inform model-free learning

Hierarchy of learning:

- **Value-free habit**: deploy a *cached policy* by repeating actions performed in the past
  (Thorndike, 1932; Cushman & Morris, 2015; Daw et al., 2005; Gershman, 2020)
- **Value-based habit**: use a *cached action value* for more flexibility
  (Botvinick & Weinstein, 2014; Keramati et al., 2016; Maisto et al., 2019)
- **Model-based planning:** Select actions expected to produced the best outcomes based on our model of the world
  (K. J. Miller et al., 2017; Vikbladh et al., 2019)

Model-based planning builds better habits!



Simulating experiences with DYNA

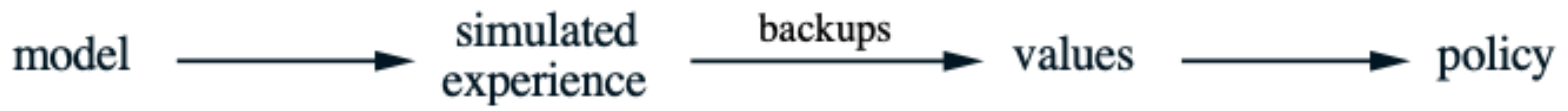
Simulating experiences with DYNA

- Models of the environment can be used for planning

Simulating experiences with DYNA

- Models of the environment can be used for planning

- … but they can also be used to simulate experiences, to learn better values and policies

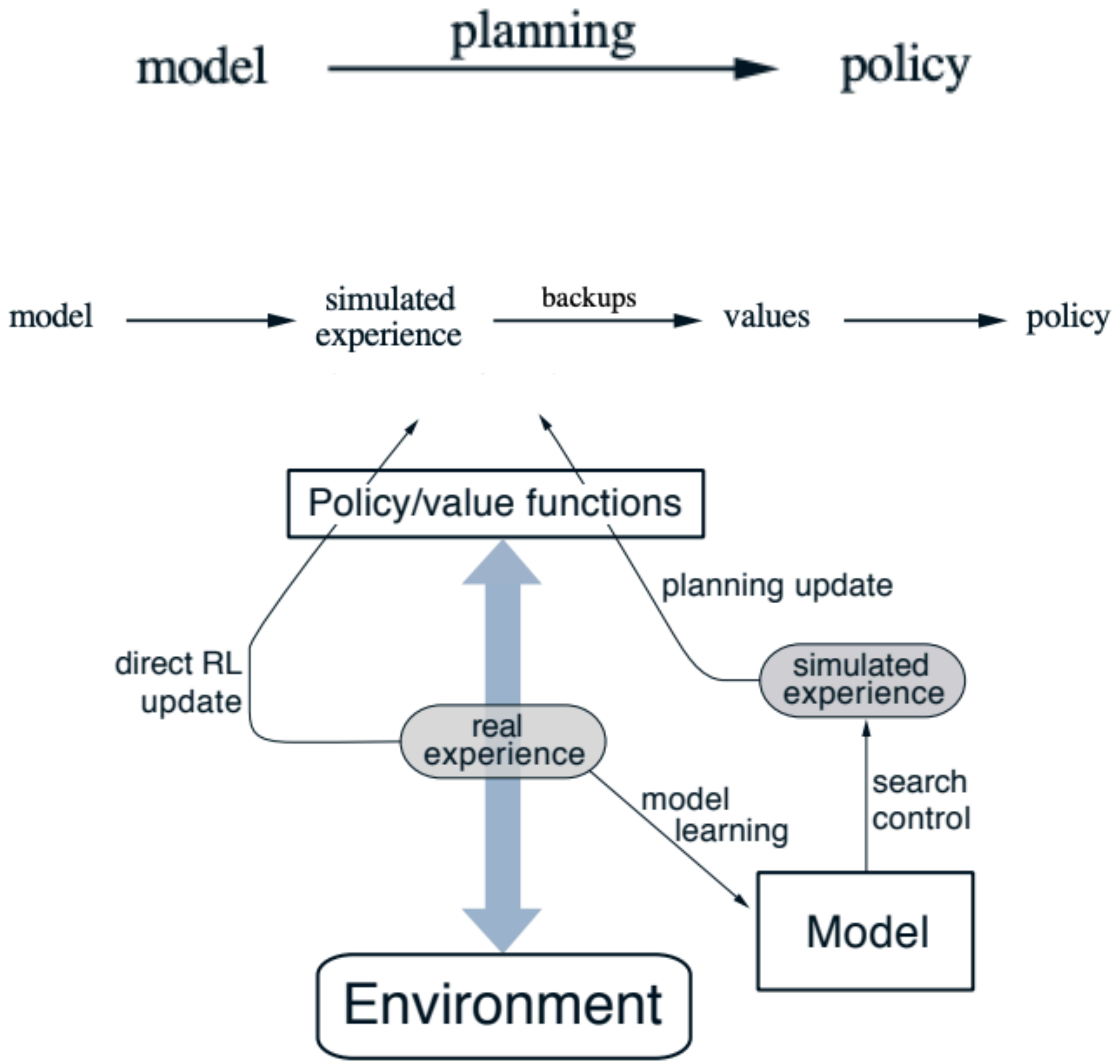


Simulating experiences with DYNA

- Models of the environment can be used for planning

- … but they can also be used to simulate experiences, to learn better values and policies

- DYNA uses simulated experiences to update our policy/value functions, just like real experiences



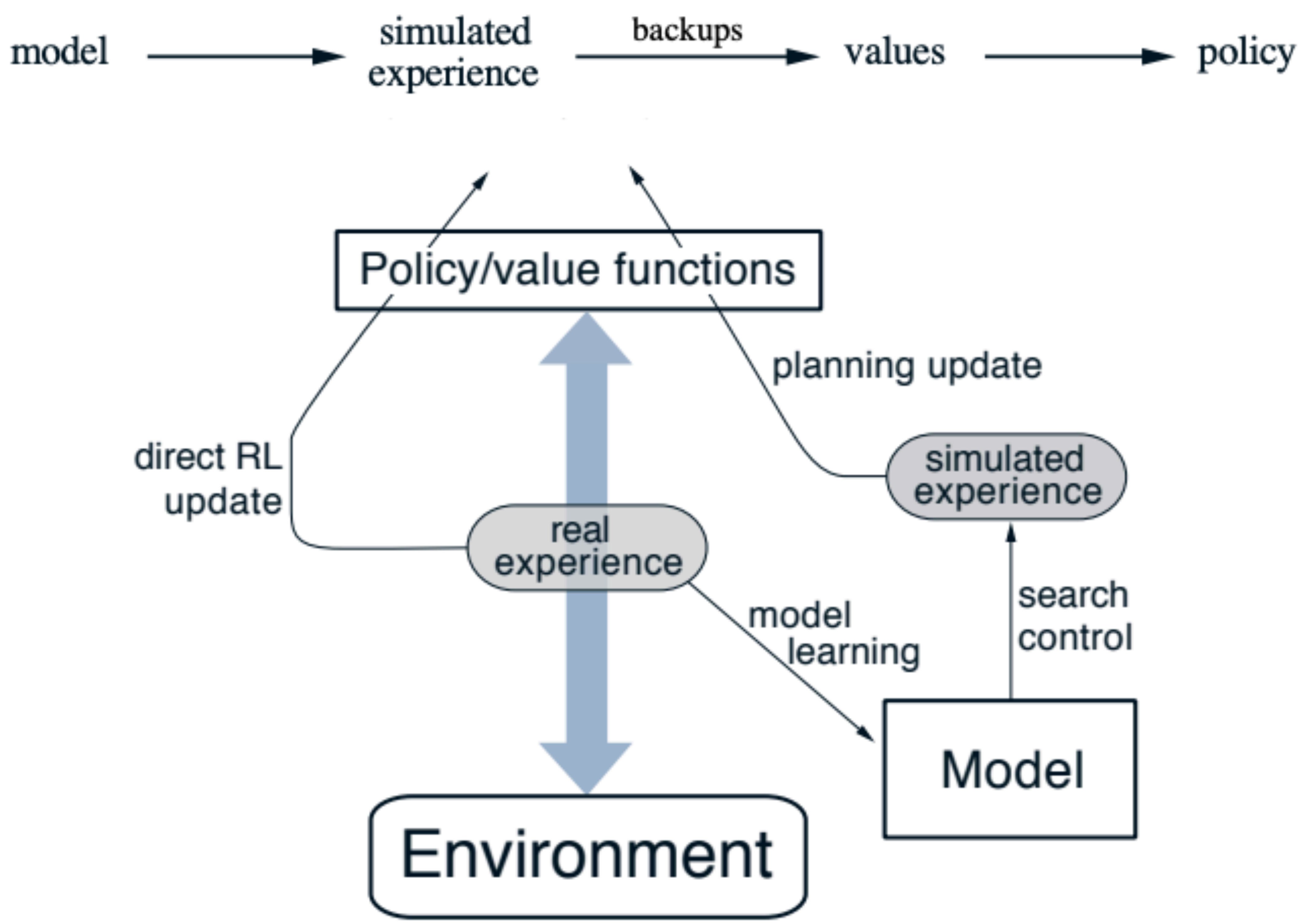
Simulating experiences with DYNA

- Models of the environment can be used for planning

- … but they can also be used to simulate experiences, to learn better values and policies

- DYNA uses simulated experiences to update our policy/value functions, just like real experiences

- These simulations can be controlled to various degrees (e.g., prioritized sweeps)

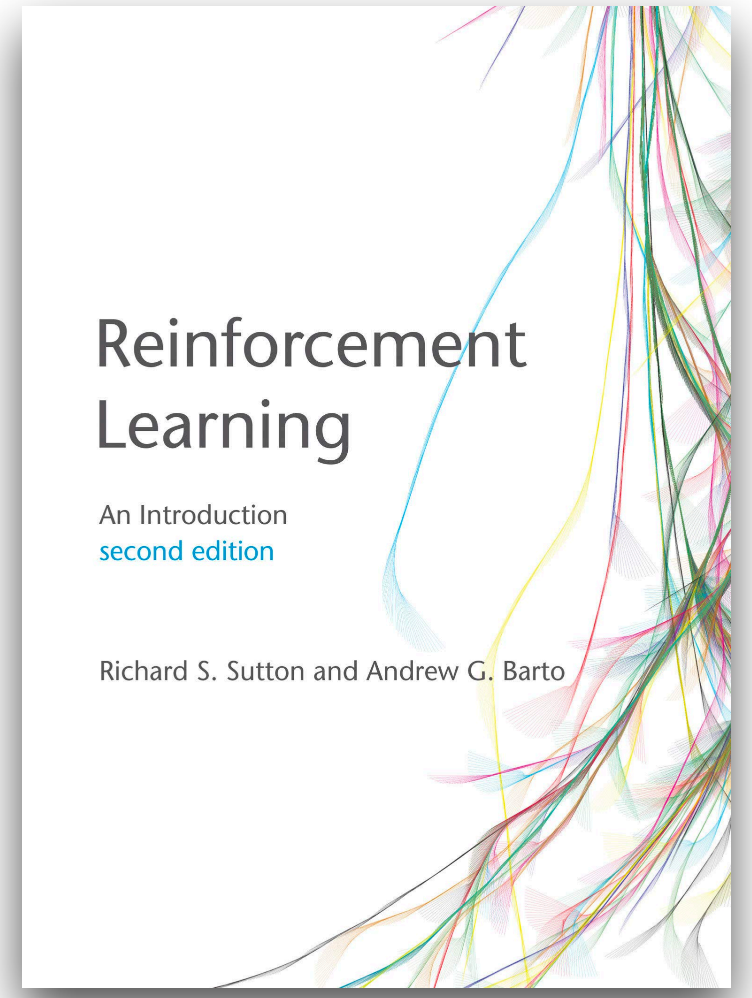


Model-free vs. Model-based summary

- Computationally cheap to use model-free learning
  - Maps onto habits and S-R learning
- Costly but potentially more impactful to use model-based learning
  - Maps onto goal-directed and S-S learning
- Not one or the other, but rather a mixture of both
- Model-based learning can help train model-free value functions and policies
  - Through experience and through simulation (e.g., DYNA)
- Still an open question how model-based representations are learned

Further study

Sutton & Barto book (free PDF link)



Great course and python code notebooks by Philipp!

https://github.com/schwartenbeckph/RL-Course

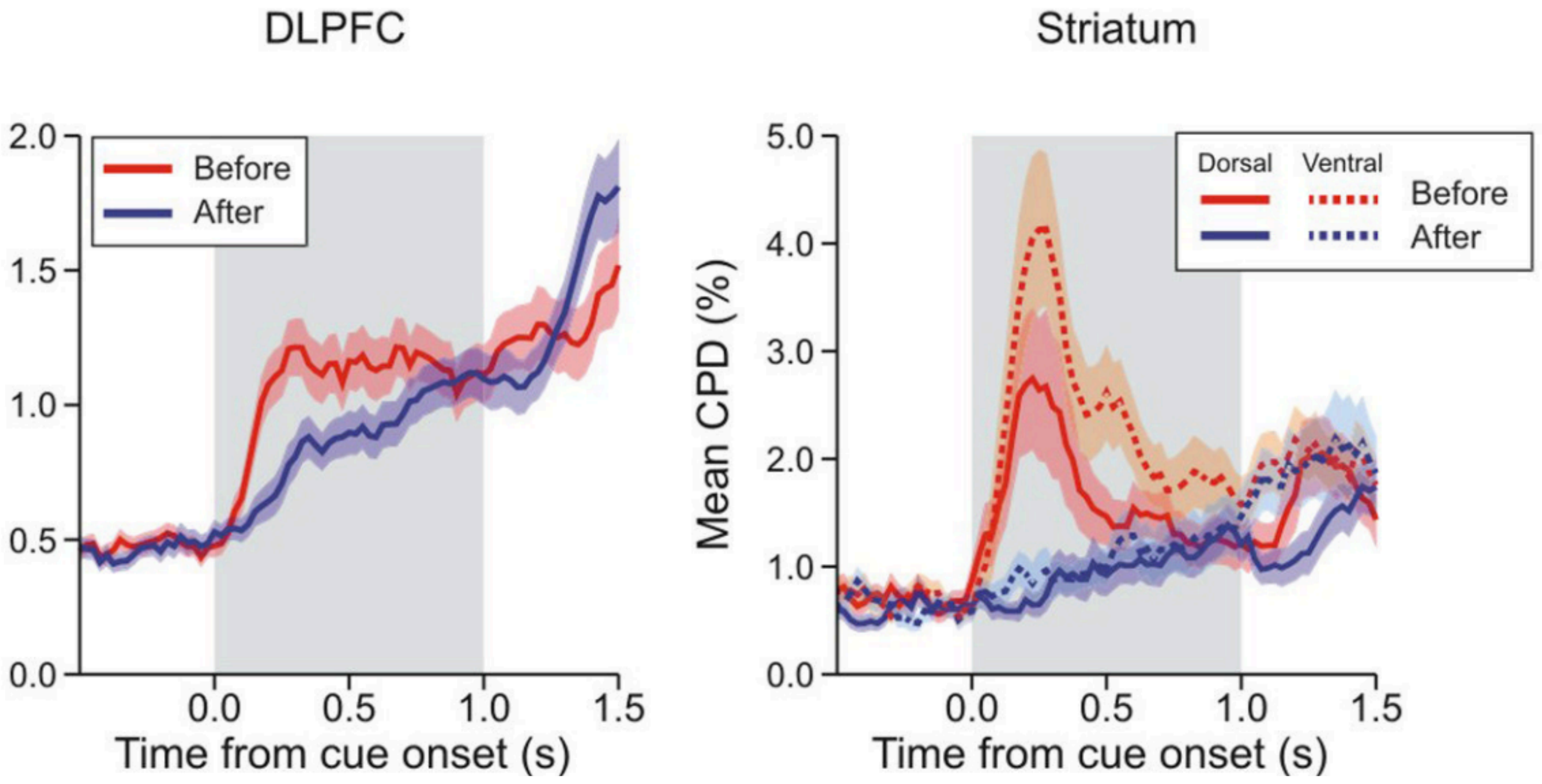
R code notebooks for using RL models (with a focus on social learning)

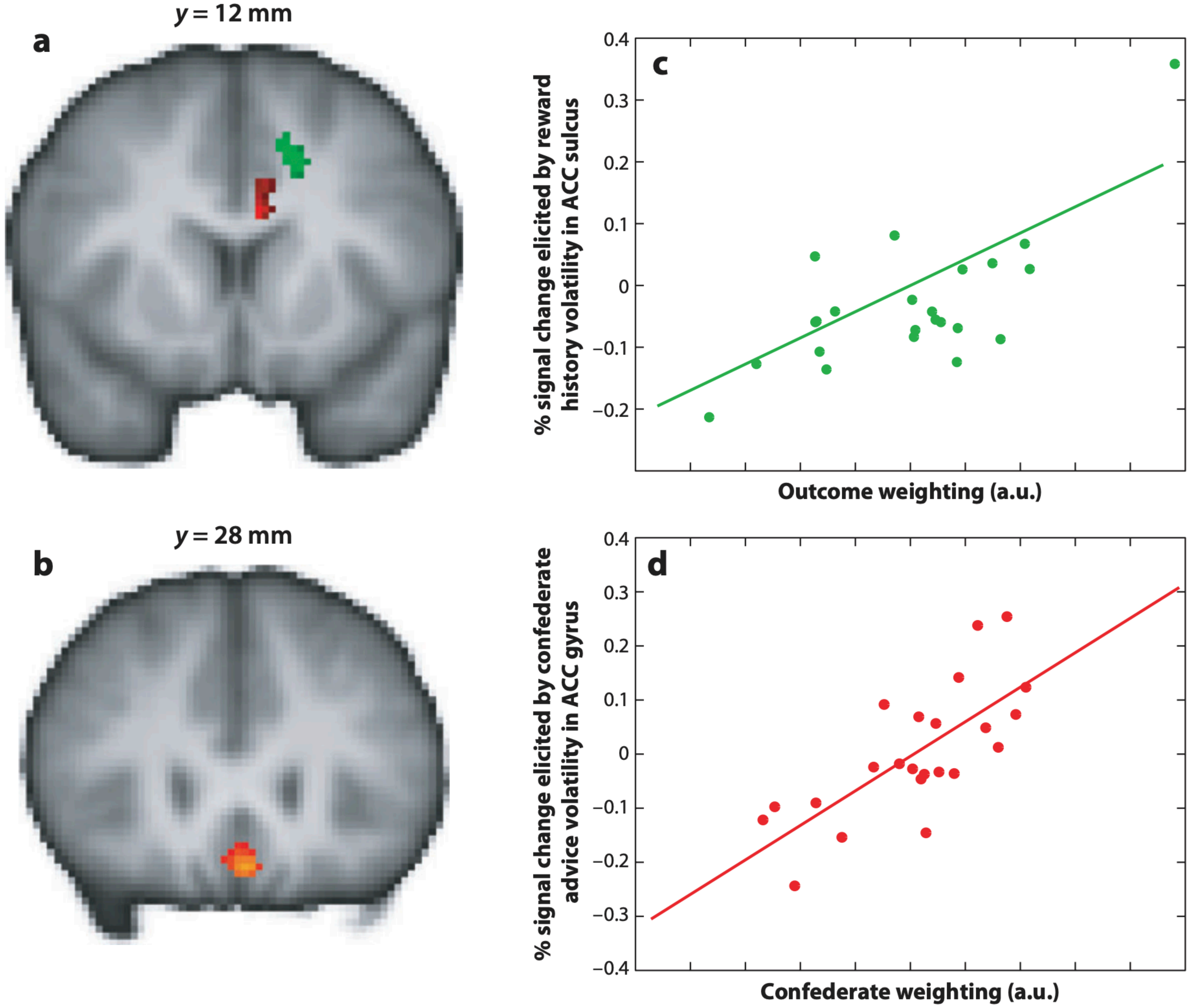
https://cosmos-konstanz.github.io/materials/

Next week

Neural Basis of Reinforcement Learning and Decision Making

Daeyeol Lee,[1,2] Hyojung Seo,[1] and Min Whan Jung[3]





Discussion questions

- How important are optimal policies and optimal value functions? People seem to use “good enough” solutions, so how are those computed?

- If model-based learning influences model-free representations, is the reverse also true? Do model-free characteristics also influence model-based learning?

- Could only partial use of model-based RL in the 2-step task be showing cognitive constraints on fully leveraging model-based representations? Are there other contexts where we can be more or less model-based?