



21AIE311 – REINFORCEMENT LEARNING

---

# MARKOV DECISION PROCESS

DR. AMUDHA J. AND MR. A. A. NIPPUN KUMAAR  
DEPARTMENT OF CSE  
AMRITA SCHOOL OF COMPUTING, BANGALORE

# PREVIOUSLY

---

- ▶ Recap and Terminologies
- ▶ Markov Decision Process
- ▶ The Agent-Environment Interface
- ▶ Goal and Rewards
- ▶ Returns and Episodes
- ▶ Unified Notion

श्रद्धावान् लभते ज्ञानम्

# LECTURE OVERVIEW

---

- ▶ Recap - Markov Decision Process
- ▶ MDP - Examples
- ▶ Policies and Value Functions
- ▶ Optimal Policies and Optimal Value Functions



# RECAP

---

- ▶ Markov Property: The state can include immediate “sensations,” highly processed sensations, and structures built up over time from sequences of sensations.
- ▶ MDP: If a reinforcement learning task has the Markov Property, it is basically a Markov Decision Process (MDP).
- ▶ MDP framework abstraction - goal-directed learning
- ▶ Three signals pass back and forth between the agent and the environment
  - ▶ The choice made by the agent - Actions
  - ▶ The basis on which the choice is made - States
  - ▶ Agent’s goal - Reward

# RECAP...

---

- ▶ Goals and Rewards: Rewards should be designed to achieve the goal.
- ▶ The reward signal is your way of communicating to the robot what you want it to achieve, not how you want it achieved.
- ▶ Returns: Expected reward over time steps.
- ▶ Episodic Tasks: Interaction breaks naturally into episodes, e.g., plays of a game, trips through a maze.
- ▶ Continual Tasks: Interaction does not have natural episodes. Here terminal state time  $T$  is infinite.
- ▶ Discounted Return

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

# AN EXAMPLE FINITE MDP – RECYCLING ROBOT

- ▶ At each step, robot has to decide whether it should (1) actively search for a can, (2) wait for someone to bring it a can, or (3) go to home base and recharge.
- ▶ Searching is better but runs down the battery; if runs out of power while searching, has to be rescued (which is bad).
- ▶ Decisions made on basis of current energy level: high, low.
- ▶ Reward = number of cans collected

$$S = \{\text{high}, \text{low}\}$$

$$A(\text{high}) = \{\text{search}, \text{wait}\}$$

$$A(\text{low}) = \{\text{search}, \text{wait}, \text{recharge}\}$$

$$R^{\text{search}} = \text{expected no. of cans while searching}$$

$$R^{\text{wait}} = \text{expected no. of cans while waiting}$$

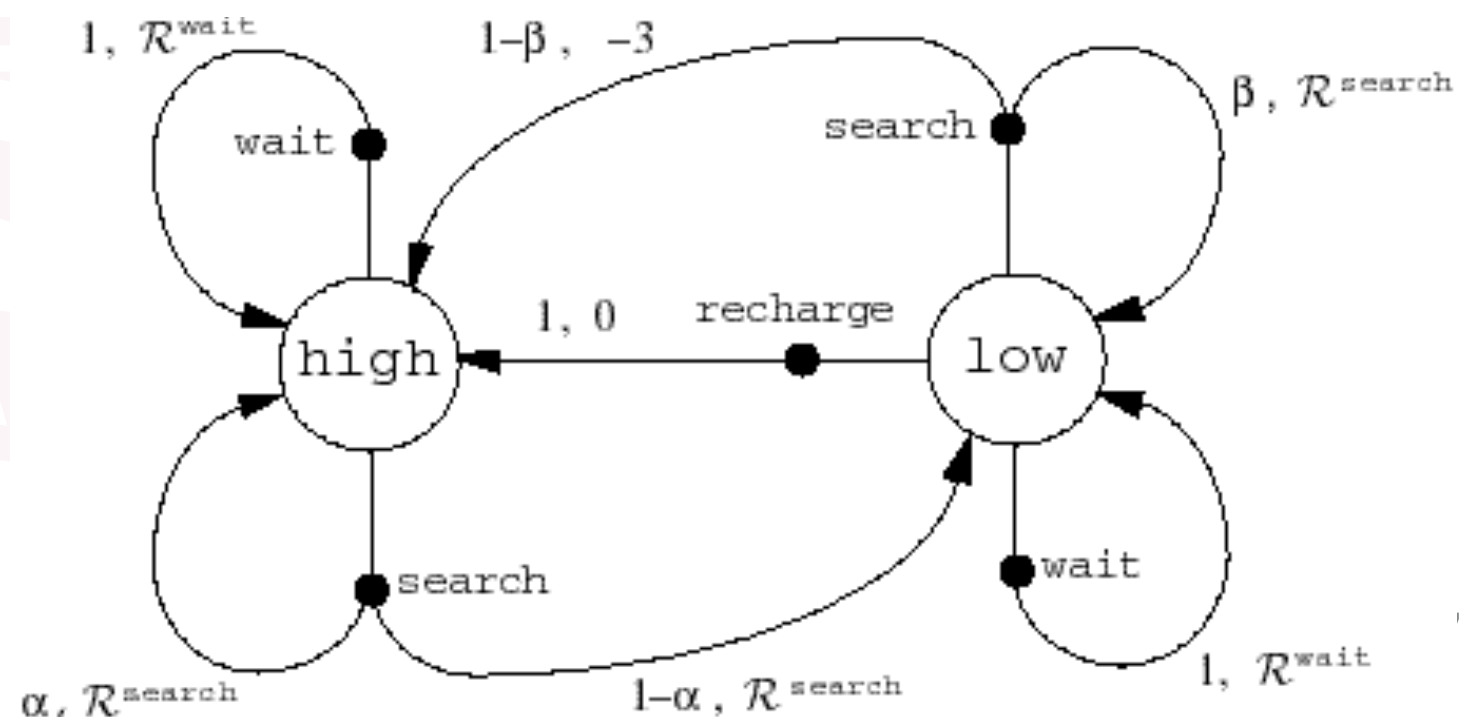
$$R^{\text{search}} > R^{\text{wait}}$$

# RECYCLING ROBOT MDP

**Table 3.1** Transition probabilities and expected rewards for the finite MDP of the recycling robot example.

$s$	$s'$	$a$	$\mathcal{P}_{ss'}^a$	$\mathcal{R}_{ss'}^a$
high	high	search	$\alpha$	$\mathcal{R}^{\text{search}}$
high	low	search	$1 - \alpha$	$\mathcal{R}^{\text{search}}$
low	high	search	$1 - \beta$	$-3$
low	low	search	$\beta$	$\mathcal{R}^{\text{search}}$
high	high	wait	1	$\mathcal{R}^{\text{wait}}$
high	low	wait	0	$\mathcal{R}^{\text{wait}}$
low	high	wait	0	$\mathcal{R}^{\text{wait}}$
low	low	wait	1	$\mathcal{R}^{\text{wait}}$
low	high	recharge	1	0
low	low	recharge	0	0

*Note:* There is a row for each possible combination of current state,  $s$ , next state,  $s'$ , and action possible in the current state,  $a \in \mathcal{A}(s)$ .



# GRID WORLD ENVIRONMENT

- ▶ It is a simple Grid World (GW) Environment.

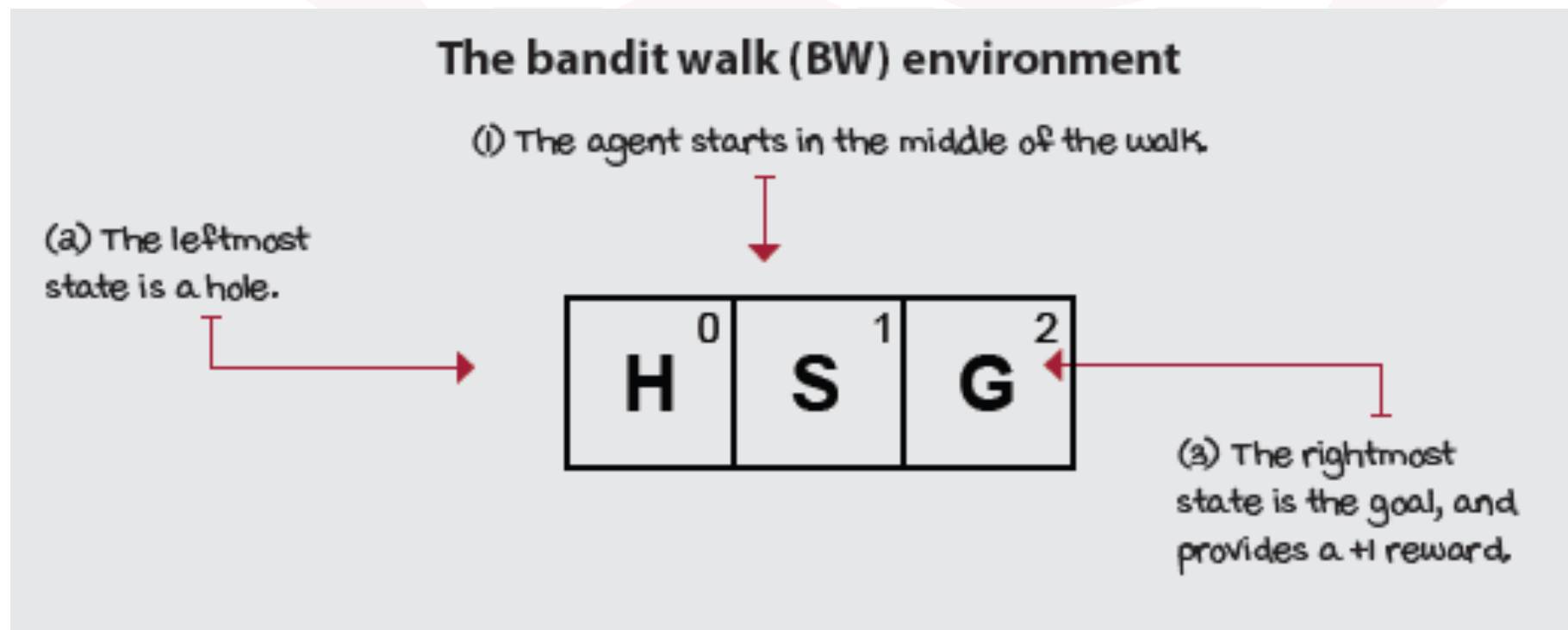
0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

- ▶ A finite set of States (Each cell in the GW) is represented by the cell numbers.
- ▶ All states tend to have a fully observable discrete state and observation spaces (that is, state equals observation).
- ▶ A finite set of Actions (Left, Down, Right, Up) that directly leads to a state change.



# BANDIT WALK (BW)

- ▶ It uses a GW environment.
- ▶ A “walk” is a special case of grid-world environments with a single row.
- ▶ The bandit walk (BW) has three states, but only one non-terminal state.



- ▶ States: Hole(0), Start(1), and Goal(2)
- ▶ Action: Left (0) and Right (1)
- ▶ Reward: +1 in State-2(Goal)
- ▶ A deterministic transition function

## Bandit walk graph



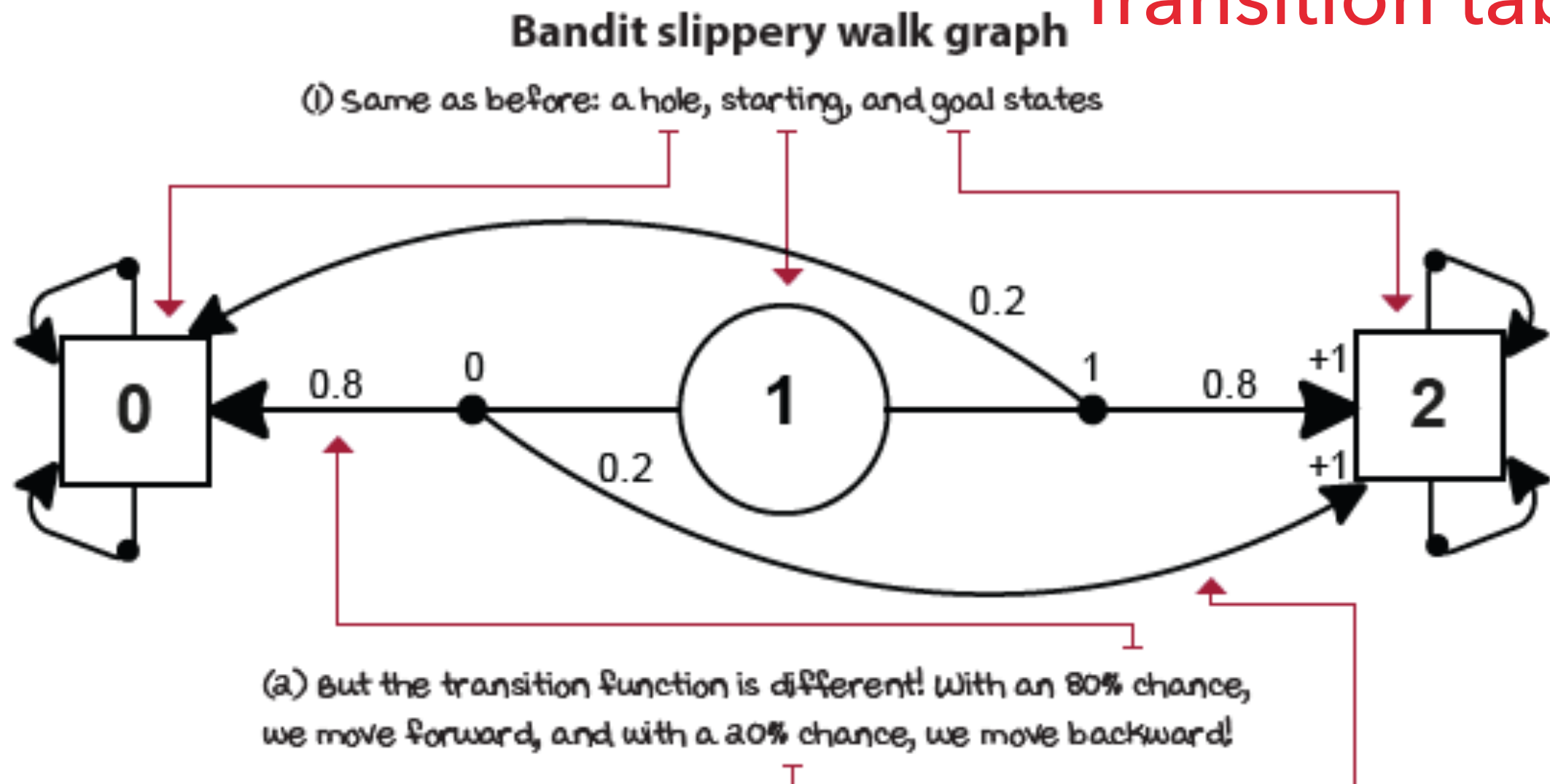
# BANDIT WALK (BW)

State	Action	Next state	Transition probability	Reward signal
0 (Hole)	0 (Left)	0 (Hole)	1.0	0
0 (Hole)	1 (Right)	0 (Hole)	1.0	0
1 (Start)	0 (Left)	0 (Hole)	1.0	0
1 (Start)	1 (Right)	2 (Goal)	1.0	+1
2 (Goal)	0 (Left)	2 (Goal)	1.0	0
2 (Goal)	1 (Right)	2 (Goal)	1.0	0

# BANDIT SLIPPERY WALK (BSW)

- ▶ This a stochastic version of BW
- ▶ Surface of the walk is slippery : Each action there a 20% chance that the agent can go backward.

Transition table ??

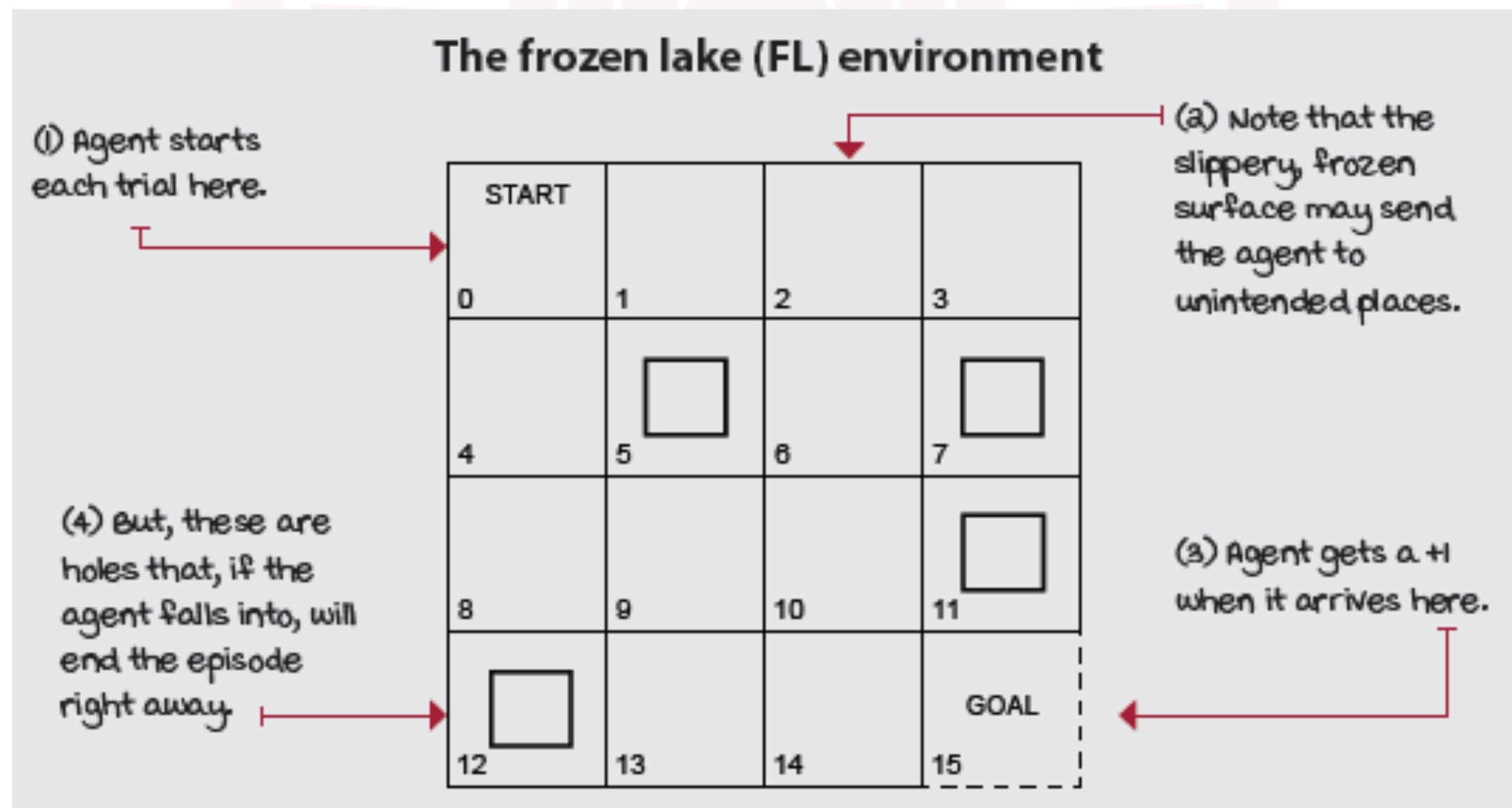


# BANDIT SLIPPERY WALK (BSW)

State	Action	Next state	Transition probability	Reward signal
0 (Hole)	0 (Left)	0 (Hole)	1.0	0
0 (Hole)	1 (Right)	0 (Hole)	1.0	0
1 (Start)	0 (Left)	0 (Hole)	0.8	0
1 (Start)	0 (Left)	2 (Goal)	0.2	+1
1 (Start)	1 (Right)	2 (Goal)	0.8	+1
1 (Start)	1 (Right)	0 (Hole)	0.2	0
2 (Goal)	0 (Left)	2 (Goal)	1.0	0
2 (Goal)	1 (Right)	2 (Goal)	1.0	0

# FROZEN LAKE

- ▶ It is a GW environment.
- ▶ The task in the FL environment is similar to the task in the BW and BSW environments: to go from a start location to a goal location while avoiding falling into holes.
- ▶ The challenge is similar to the BSW, in that the surface of the FL environment is slippery, it's a frozen lake after all. But the environment itself is larger.





# FROZEN LAKE – STATES

- ▶ A state is a unique and self-contained configuration of the problem.
- ▶ The set of all possible states, the state space, is defined as the set  $S$ . The state space can be finite or infinite.
- ▶ State space is different than the set of variables that compose a single state. This set should always be finite.
- ▶ In the end, the state space is a set of sets. The inner set must be of equal size and finite, as it contains the number of variables representing the states.
- ▶ The outer set can be infinite depending on the types of elements of the inner sets.

## State space: A set of sets

### FL state space

[ [0], [1], [2], [3],  
[4], [5], [6], [7],  
[8], [9], [10], [11],  
[12], [13], [14], [15] ]

### Some other state space

[ [0.12, -1.24, 0, -1, 1.44],  
[0.121, -1.24, 0, -1, 1.44],  
[0.1211, -1.24, 0, -1, 1.44],  
... ]

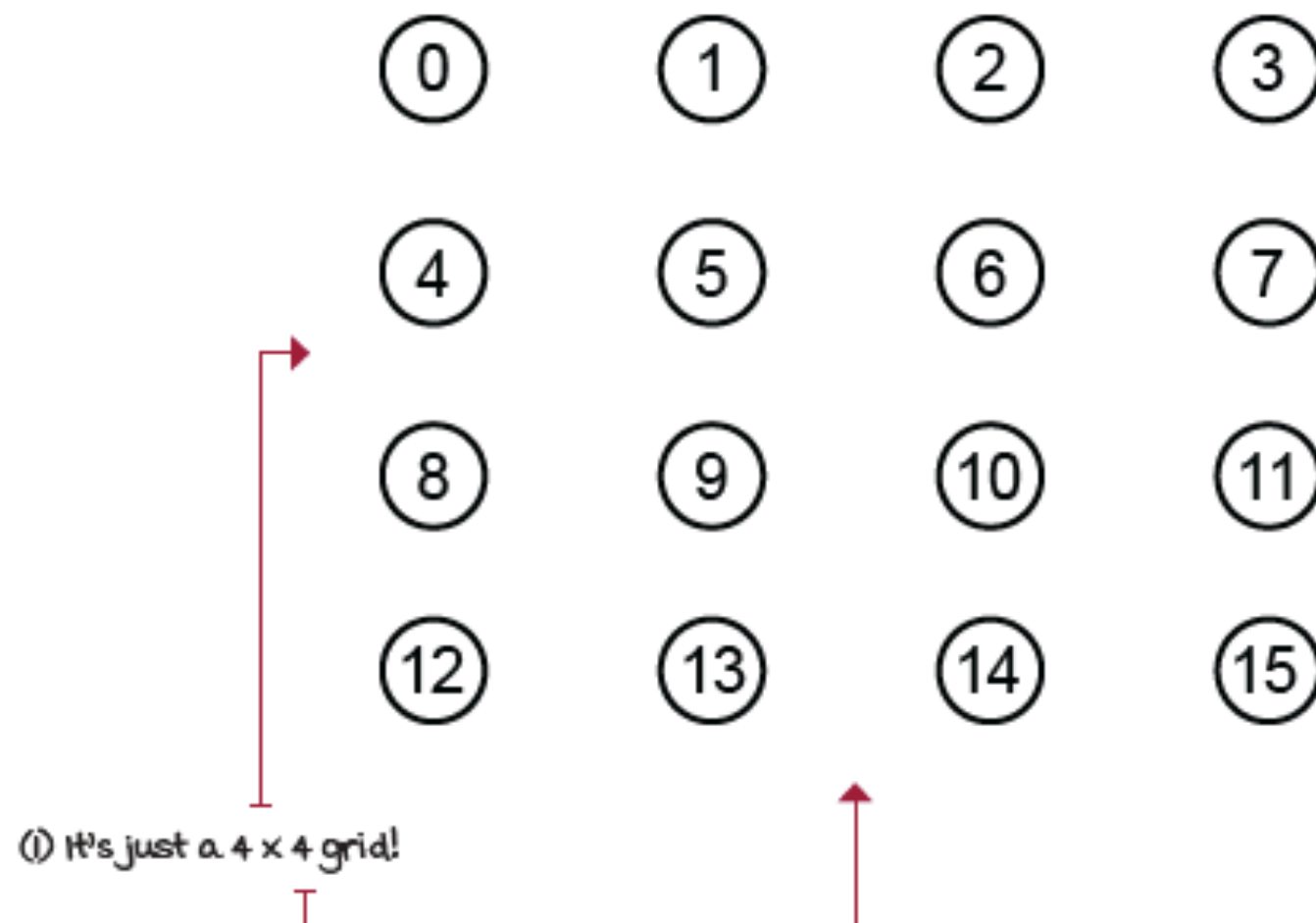
(i) The inner set (the number of variables that compose the states) must be finite.  
The size of the inner set must be a positive integer.

(a) But the outer set may be infinite:  
if any of the inner sets' elements is  
continuous, for instance.

# FROZEN LAKE – STATES

- ▶ For the BW, BSW, and FL environments, the state is composed of a single variable containing the id of the cell where the agent is at any given time.


**States in the FL contain a single variable indicating the id of the cell in which the agent is at any given time step**





# FROZEN LAKE – STATES

- ▶ MDPs are fully observable - Observation and states are same
- ▶ Partially observable Markov decision processes (POMDPs)
  - Observation are the only thing the agent can see instead of the state.
- ▶ BW, BSW and FL are?? **MDP**
- ▶ Are BW, BSW, FL contain Markov Property ?? **Yes**

 **SHOW ME THE MATH**  
The Markov property

(1) The probability of the next state ...

(3) ... will be the same ...

$$P(S_{t+1}|S_t, A_t) = P(S_{t+1}|S_t, A_t, S_{t-1}, A_{t-1}, \dots)$$

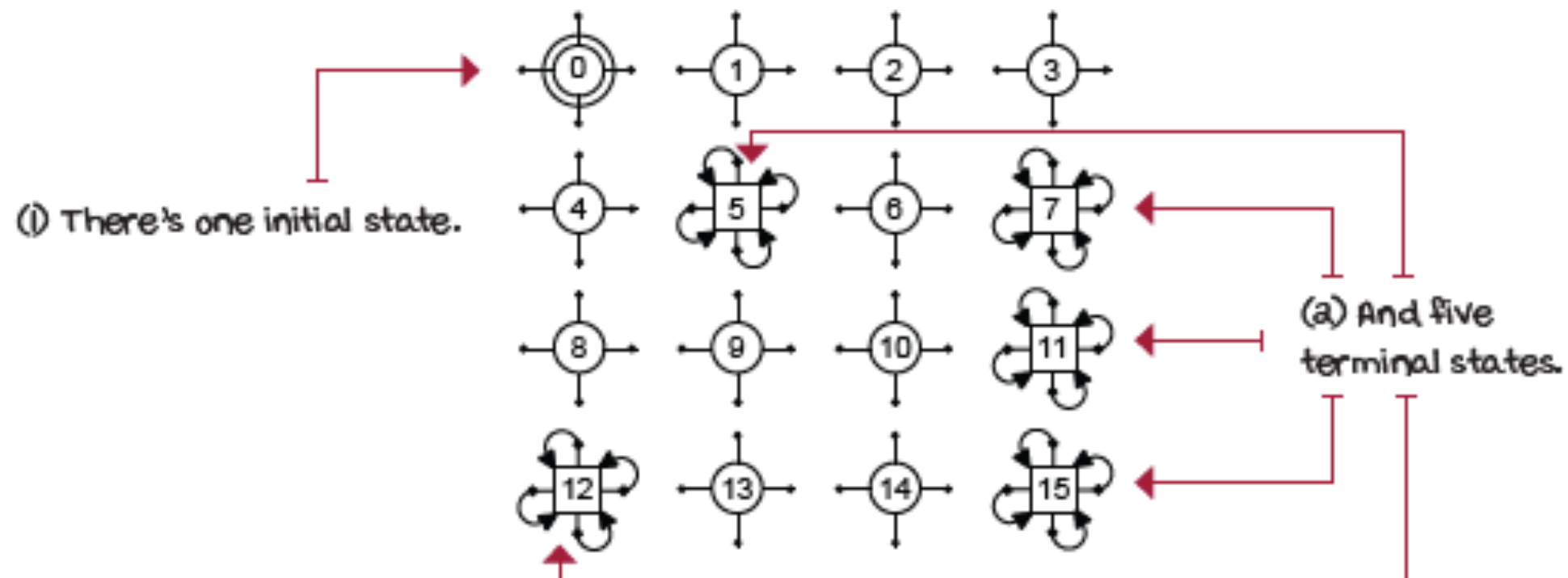
(2) ... given the current state and current action ...

(4) ... as if you give it the entire history of interactions.

# FROZEN LAKE – STATES

- ▶ State Space:  $S^+$
- ▶  $S^i$  - Initial States - Subset of  $S^+$
- ▶  $S$  - Absorption State/Transition State - Subset of  $S^+$
- ▶ A terminal state is a special state: it must have all available actions transitioning, with probability 1, to itself, and these transitions must provide no reward.

**States in the frozen lake environment**



# FROZEN LAKE – ACTIONS

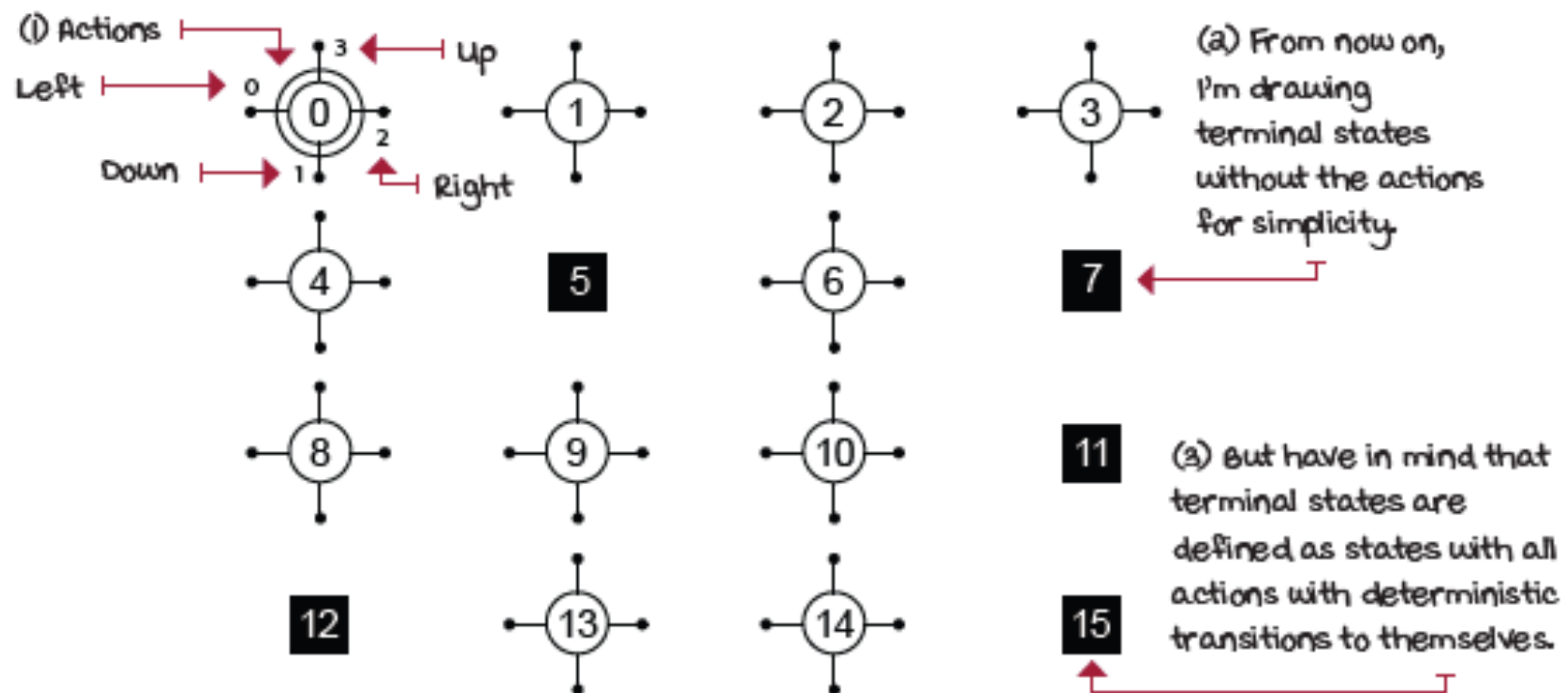
---

- ▶ Action  $A(s)$  - A set of actions that depends on the state.
- ▶ If Needed  $A$  - A set of actions common to all the possible states.
- ▶ Unlike the number of state variables, the number of variables that compose an action may not be constant.
- ▶ Actions can be finite or infinite. The environment makes the set of all available actions known in advance. Agents can select actions either deterministically or stochastically.

# FROZEN LAKE – ACTIONS

- ▶ In the BW, BSW, and FL environments, actions are singletons representing the direction the agent will attempt to move.
- ▶ In FL, there are four available actions in all states: Up, Down, Right, or Left. There's one variable per action, and the size of the action space is four.

The frozen lake environment has four simple move actions



# FROZEN LAKE – TRANSITION FUNCTION

- ▶ It is the consequence of agent actions.
- ▶ The way the environment changes as a response to actions is referred to as the state-transition probabilities, or more simply, the transition function, and is denoted by  $T(s, a, s')$ .



## SHOW ME THE MATH

### The transition function

(1) The transition function is defined...

(2) ... as the probability of transitioning to state  $s'$  at time step  $t$ ...

(3) ... given action  $a$  was selected on state  $s$  in the previous time step  $t-1$ .

$$p(s'|s, a) = P(S_t = s' | S_{t-1} = s, A_{t-1} = a)$$

(4) Given these are probabilities, we expect the sum of the probabilities across all possible next states to sum to 1.

$$\sum_{s' \in S} p(s'|s, a) = 1, \forall s \in S, \forall a \in A(s)$$

(5) That's true for all states  $s$  in the set of states  $S$ , and all actions  $a$  in the set of actions available in state  $s$ .

# FROZEN LAKE – TRANSITION FUNCTION

- ▶ One key assumption of many RL (and DRL) algorithms is that this distribution is stationary.
- ▶ That is, while there may be highly stochastic transitions, the probability distribution may not change during training or evaluation.
- ▶ In the FL environment, we know that there's a 33.3% chance we'll transition to the intended cell (state) and a 66.6% chance we'll transition to orthogonal directions. There's also a chance we'll bounce back to the state we're coming from if it's next to the wall.

	33.3%	
33.3%	S	33.3%
	0%	

Visualize it as the agent facing directly to the intended cell



# FROZEN LAKE – TRANSITION FUNCTION

## The transition function of the frozen lake environment

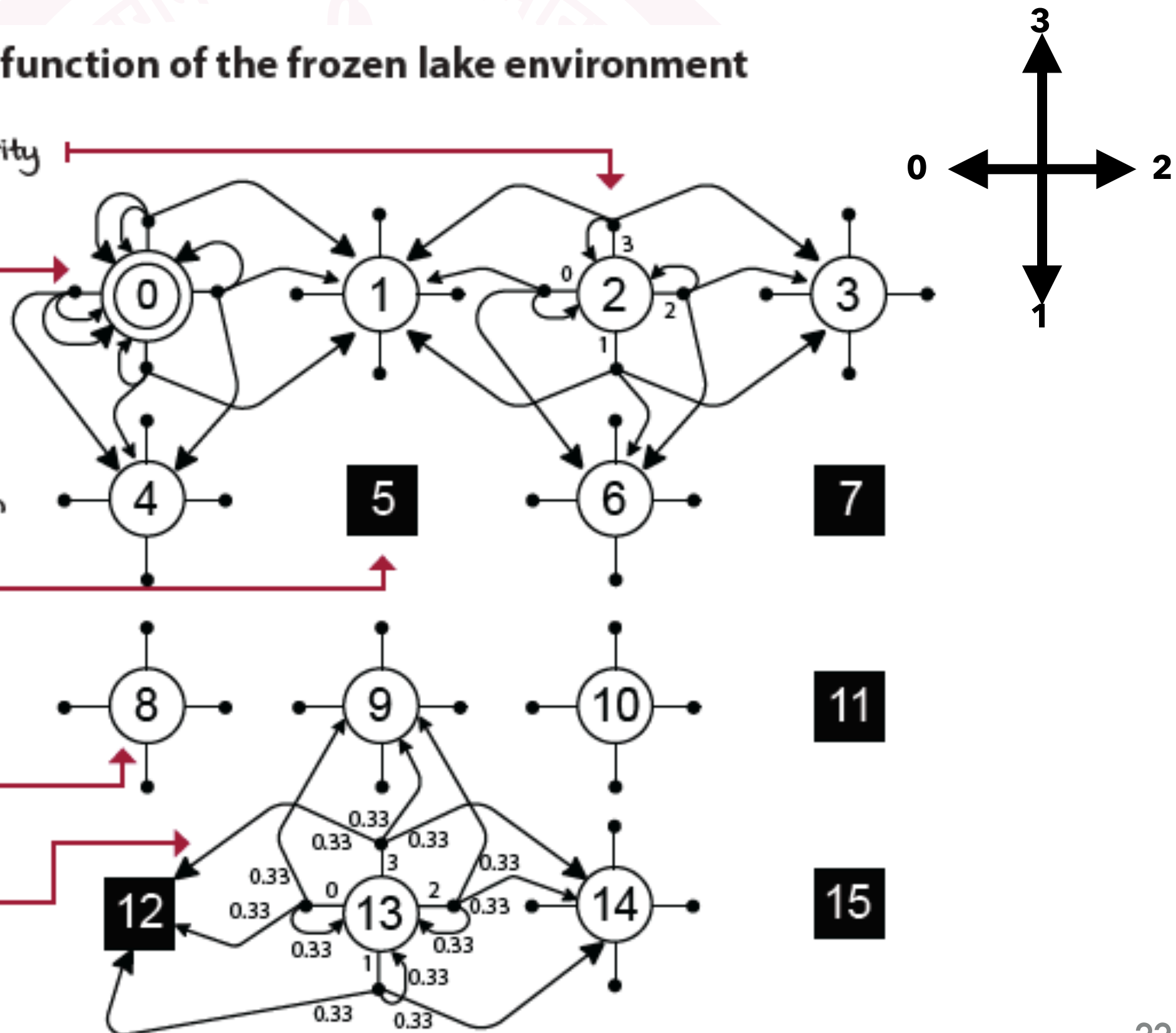
(1) Without probabilities for clarity

(2) Notice that the corner states are special. You bounce back from the horizontal and the vertical walls.

(3) Remember that terminal states have all transitions from all actions looping back to themselves with probability 1.

(4) I'm not drawing all the transitions, of course. This state, for instance, isn't complete.

(5) This environment is highly stochastic!



# FROZEN LAKE – REWARDS

- ▶ The reward function  $R$  maps a transition tuple  $s, a, s'$  to a scalar.
- ▶ The reward function gives a numeric signal of goodness to transitions. When the signal is positive, we can think of the reward as an income or a reward.
- ▶ rewards can also be negative, and we can see these as cost, punishment, or penalty.



## SHOW ME THE MATH

### The reward function

(1) The reward function can be defined as follows.

(2) It can be defined as a function that takes in a state-action pair.

$$r(s, a) = \mathbb{E}[R_t | S_{t-1} = s, A_{t-1} = a]$$

(3) And, it's the expectation of reward at time step  $t$ , given the state-action pair in the previous time step.

(4) But, it can also be defined as a function that takes a full transition tuple  $s, a, s'$ .

(5) And it's also defined as the expectation, but now given that transition tuple.

$$r(s, a, s') = \mathbb{E}[R_t | S_{t-1} = s, A_{t-1} = a, S_t = s']$$

$$R_t \in \mathcal{R} \subset \mathbb{R}$$

(6) The reward at time step  $t$  comes from a set of all rewards  $\mathcal{R}$ , which is a subset of all real numbers.

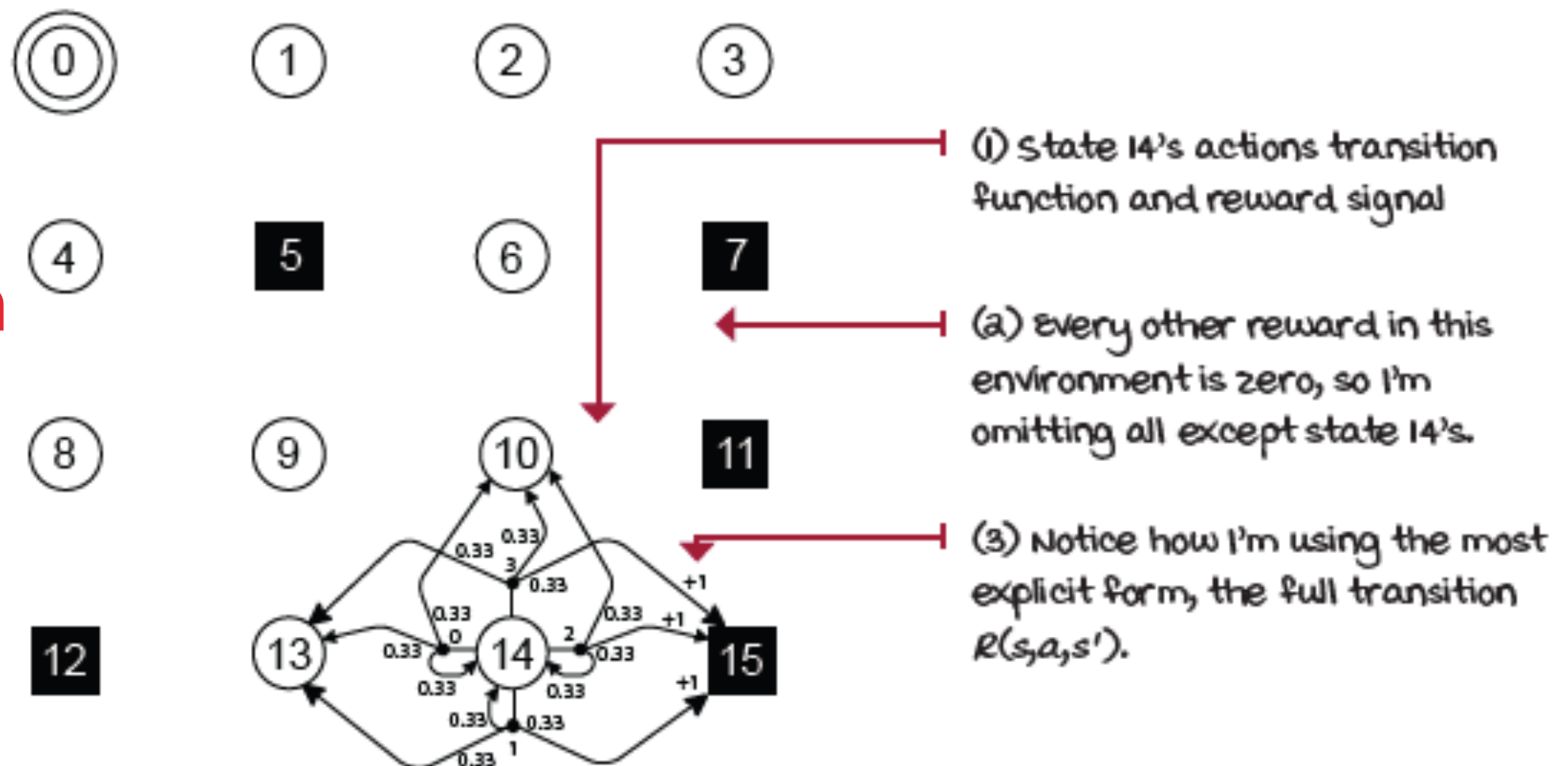


# FROZEN LAKE – REWARDS

- ▶ In the FL environment, the reward function is +1 for landing in state 15, 0 otherwise.
- ▶ There are only three ways to land on 15.
  - ▶ (1) Selecting the Right action in state 14 will transition the agent with 33.3% chance there (33.3% to state 10 and 33.3% back to 14).
  - ▶ (2) selecting the Up
  - ▶ (3) the Down action from state 14 will unintentionally also transition the agent there with 33.3% probability for each action.

## Reward signal for states with non-zero reward transitions

Transition  
table ??



# FROZEN LAKE – TABLE

State	Action	Next state	Transition probability	Reward signal
0	Left	0	0.33	0
0	Left	0	0.33	0
0	Left	4	0.33	0
0	Down	0	0.33	0
0	Down	4	0.33	0
0	Down	1	0.33	0
0	Right	4	0.33	0
0	Right	1	0.33	0
0	Right	0	0.33	0
0	Up	1	0.33	0
0	Up	0	0.33	0
0	Up	0	0.33	0
1	Left	1	0.33	0
1	Left	0	0.33	0
1	Left	5	0.33	0
1	Down	0	0.33	0
1	Down	5	0.33	0
1	Down	2	0.33	0
1	Right	5	0.33	0
1	Right	2	0.33	0
1	Right	1	0.33	0

... and So on

# FROZEN LAKE – TIME STEP

---

- ▶ A time step, also referred to as epoch, cycle, iteration, or even interaction, is a global clock syncing all parties and discretizing time.
- ▶ Episodic and Continual tasks.
- ▶ Episodic and continuing tasks can also be defined from the agent's perspective. We call it the planning horizon.
  - ▶ Finite Horizon - Agent knows the task will terminate in a finite number of time steps. Eg: Frozen Lake
  - ▶ Greedy Horizon - If agent is forced to terminate with 15 time steps in FL
  - ▶ Infinite Horizon - Agent plans for infinite time steps but a manual terminal conditions will be give.
- ▶ FL is an indefinite planning horizon; the agent plans for infinite number of steps, but interactions may stop at any time.

▶

श्रद्धावान् लभते ज्ञानम्

# FROZEN LAKE – DISCOUNTED REWARD

---

- ▶ Because of the possibility of infinite sequences of time steps in infinite horizon tasks, we need a way to discount the value of rewards over time; that is, we need a way to tell the agent that getting +1's is better sooner than later.
- ▶ We commonly use a positive real value less than one to exponentially discount the value of future rewards. The further into the future we receive the reward, the less valuable it is in the present.
- ▶ This number is called the discount factor, or gamma. The discount factor adjusts the importance of rewards over time.

# FROZEN LAKE – DISCOUNTED REWARD



## SHOW ME THE MATH

The discount factor (gamma)

(1) The sum of all rewards obtained during the course of an episode is referred to as the return.

→  $G_t = R_{t+1} + R_{t+2} + R_{t+3} + \dots + R_T$

(a) But we can also use the discount factor this way and obtain the discounted return. The discounted return will downweight rewards that occur later during the episode.

→  $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots + \gamma^{T-1} R_T$

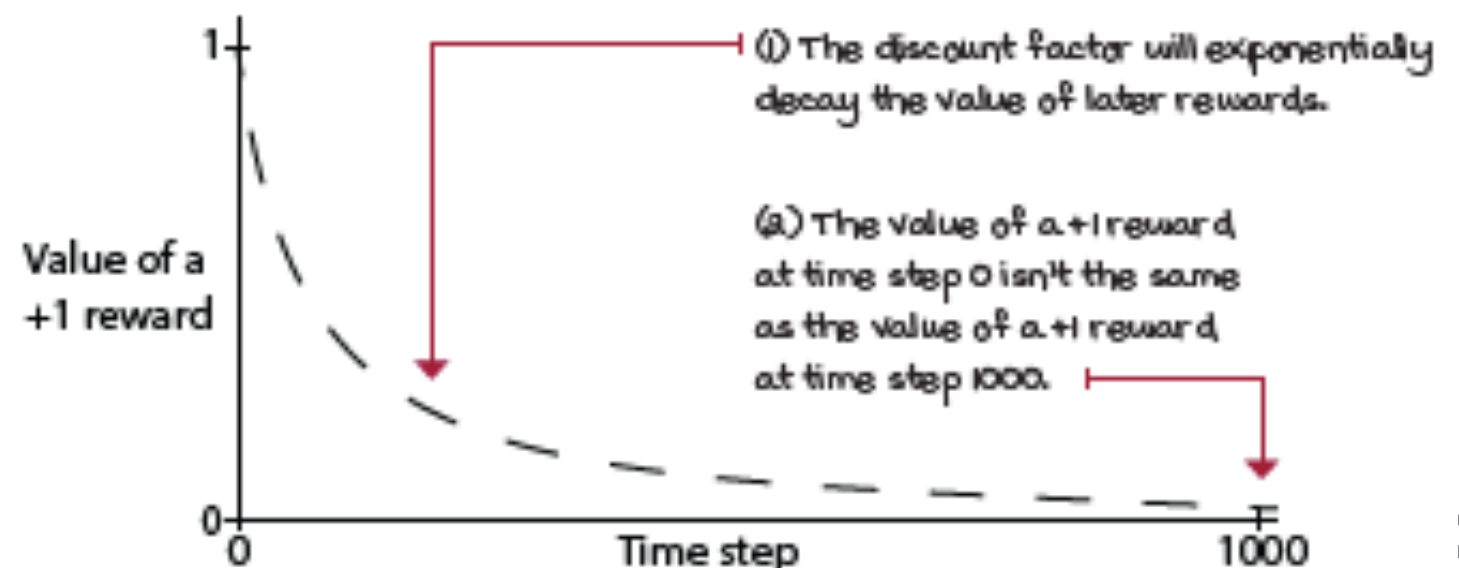
(a) We can simplify the equation and have a more general equation, such as this one.

→  $G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

(4) Finally, take a look at this interesting recursive definition. In the next chapter, we spend time exploiting this form.

→  $G_t = R_{t+1} + \gamma G_{t+1}$

## Effect of discount factor and time on the value of rewards



# VALUE FUNCTIONS

---



श्रद्धावान् लभते ज्ञानम्



**THANK YOU !!!**

श्रद्धावान् लभते ज्ञानम्