Introduction to Reinforcement Learning (RL)

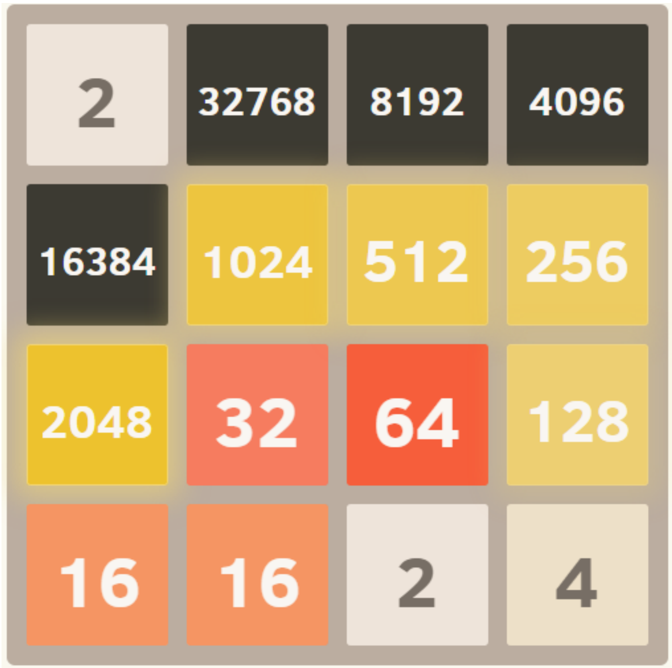
I-Chen Wu

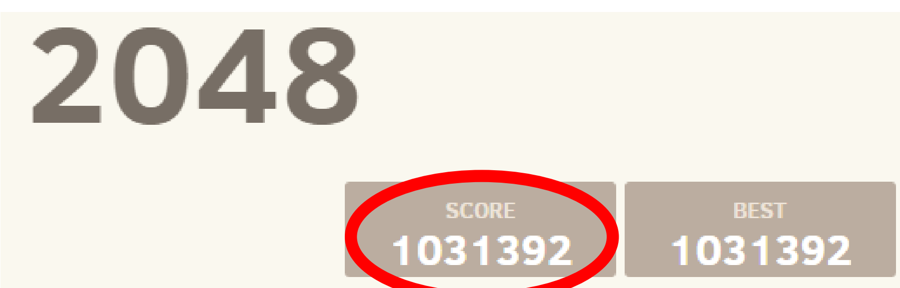
- Sutton, R.S. and Barto, A.G., Reinforcement Learning: An Introduction, MIT Press, Cambridge, MA, 1998.
  - http://webdocs.cs.ualberta.ca/~sutton/book/ebook/the-book.html
  - Bible in this area.
- David Silver, Online Course for Deep Reinforcement Learning.
  - http://www.cs.ucl.ac.uk/staff/D.Silver/web/Teaching.html

Successful RL Examples

- Games: Super-human levels
  - Backgammon (Tesauro, 1994).
  - Connect6/2048/Threes! (CGI, 2022). Reach the top levels.
  - AlphaGo/AlphaZero/Muzero, using deep reinforcement learning (2016)
  - Open AI Five for Dota 2, 2019
  - AlphaStar for StarCraft by DeepMind (in nature), 2019
- Robotics: robot-controlled helicopters and humanoid robot walk (Abbeel et al.).
- Autonomous driving/racing: AWS DeepRacer (Amazon, CGI, 2019-)
- Manufacturing scheduling (CGI, 2022).
- Chip design: a fast graph placement by Google Brain (Nature, 2021)
- Optimizing matrix multiplication: AlphaTensor (2022)
- Chat bot: RLHF in Chat-GPT (OpenAI, 2022)
  - Reinforcement Learning from Human Feedback
- …(Many more successful examples for deep reinforcement learning)

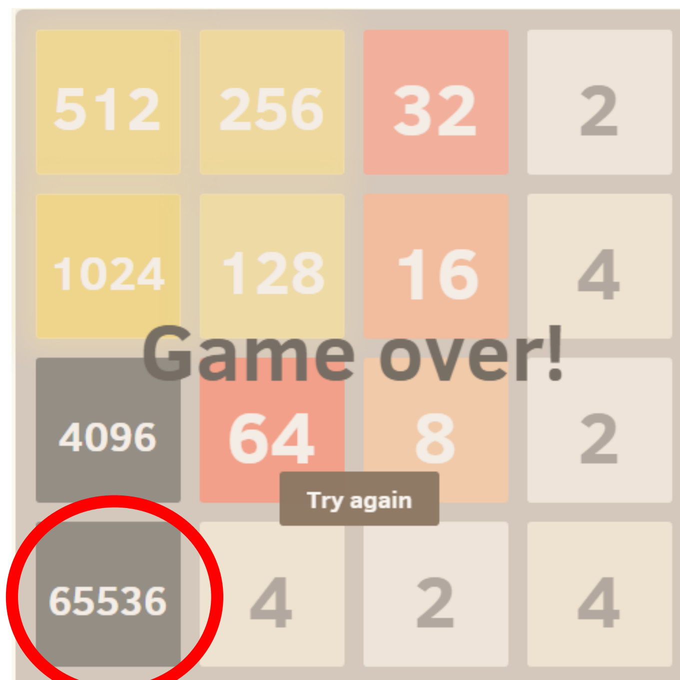
Stochastic Game: 2048





The First Game Reaching 65536 in the World (in 10,000 Trials) in 2015

http://2048.aigames.nctu.edu.tw/replay.php



AlphaGo/AlphaZero

- The Game of Go
  - AlphaGo vs. 李世石: 4:1 (2016)
  - AlphaGo Zero vs. 柯潔: 3:0 (2017)

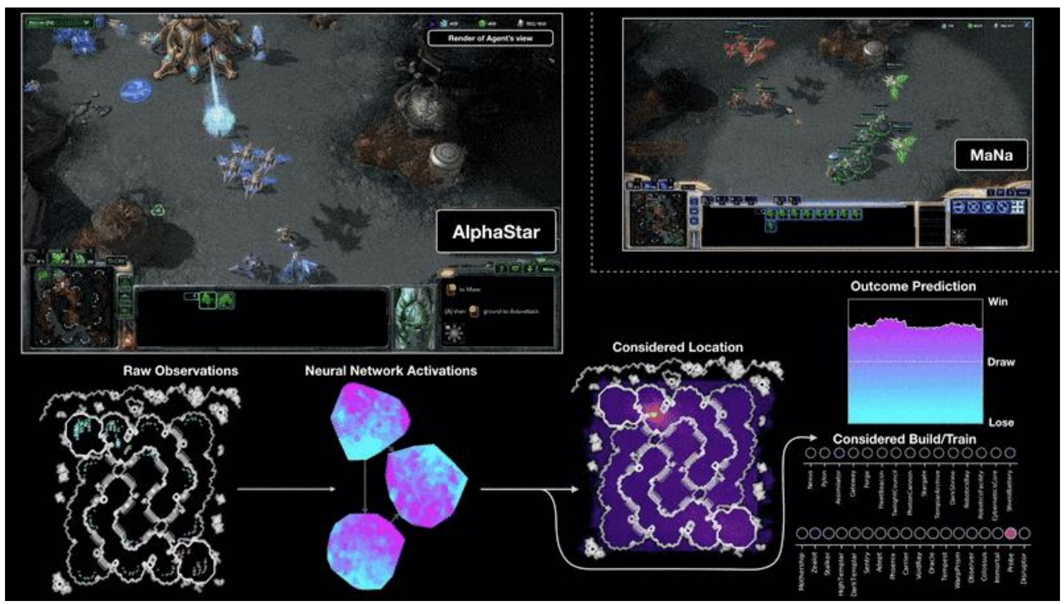
Dota 2: OpenAI Five (OpenAI)

- Dota 2 is played with two teams defending bases in opposite corners. Each team have five players, each controlling a hero unit with unique abilities
- OpenAI Five became the first AI system to defeat the world champions at an esports game (2019)

Source of image: https://technews.tw/2019/04/16/ai-dota-fight-alongside/
Berner, Christopher, et al. "Dota 2 with large scale deep reinforcement learning." *arXiv preprint arXiv:1912.06680* (2019).

StarCraft II: AlphaStar (DeepMind)

- StarCraft is a real-time strategy game in which players balance high-level economic decisions with individual control of hundreds of units
- AlphaStar was rated at Grandmaster level for all three StarCraft races, above 99.8% of officially ranked human players (2019)



https://www.deepmind.com/blog/alphastar-mastering-the-real-time-strategy-game-starcraft-ii

Vinyals, O., Babuschkin, I., Czarnecki, W.M. *et al.* Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* **575**, 350–354 (2019).

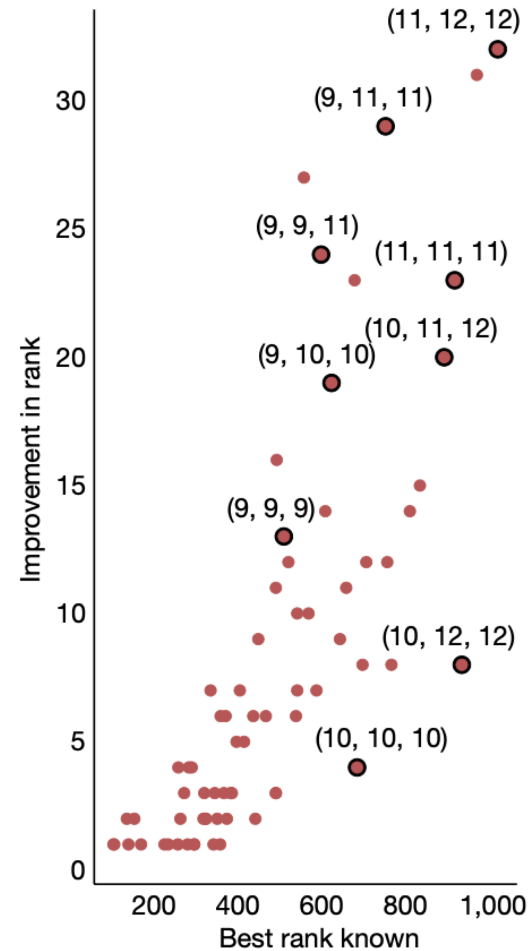
Pluribus Poker (CMU and FAIR)

- Pluribus: stronger than top human professionals in six-player no-limit Texas hold'em poker (2019)

Source of image: https://zhuanlan.zhihu.com/p/73336511
Noam Brown Tuomas Sandholm ,Superhuman AI for multiplayer poker.Science365,885-890(2019).DOI:10.1126/science.aay2400

AlphaTensor (2022)

Size (*n*, *m*, *p*)	Best method known	Best rank known	AlphaTensor rank Modular	AlphaTensor rank Standard
(2, 2, 2)	(Strassen, 1969)[2]	7	7	7
(3, 3, 3)	(Laderman, 1976)[15]	23	23	23
(4, 4, 4)	(Strassen, 1969)[2] (2, 2, 2) ⊗ (2, 2, 2)	49	47	49
(5, 5, 5)	(3, 5, 5) + (2, 5, 5)	98	96	98
(2, 2, 3)	(2, 2, 2) + (2, 2, 1)	11	11	11
(2, 2, 4)	(2, 2, 2) + (2, 2, 2)	14	14	14
(2, 2, 5)	(2, 2, 2) + (2, 2, 3)	18	18	18
(2, 3, 3)	(Hopcroft and Kerr, 1971)[16]	15	15	15
(2, 3, 4)	(Hopcroft and Kerr, 1971)[16]	20	20	20
(2, 3, 5)	(Hopcroft and Kerr, 1971)[16]	25	25	25
(2, 4, 4)	(Hopcroft and Kerr, 1971)[16]	26	26	26
(2, 4, 5)	(Hopcroft and Kerr, 1971)[16]	33	33	33
(2, 5, 5)	(Hopcroft and Kerr, 1971)[16]	40	40	40
(3, 3, 4)	(Smirnov, 2013)[18]	29	29	29
(3, 3, 5)	(Smirnov, 2013)[18]	36	36	36
(3, 4, 4)	(Smirnov, 2013)[18]	38	38	38
(3, 4, 5)	(Smirnov, 2013)[18]	48	47	47
(3, 5, 5)	(Sedoglavic and Smirnov, 2021)[19]	58	58	58
(4, 4, 5)	(4, 4, 2) + (4, 4, 3)	64	63	63
(4, 5, 5)	(2, 5, 5) ⊗ (2, 1, 1)	80	76	76



I-Chen Wu

Alhussein Fawzi, Matej Balog, Aja Huang, Thomas Hubert, Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Francisco J. R. Ruiz, Julian Schrittwieser, Grzegorz Swirszcz, David Silver, Demis Hassabis, and Pushmeet Kohli. Discovering faster matrix multiplication algorithms with reinforcement learning. Nature, 610:47–53, 2022.

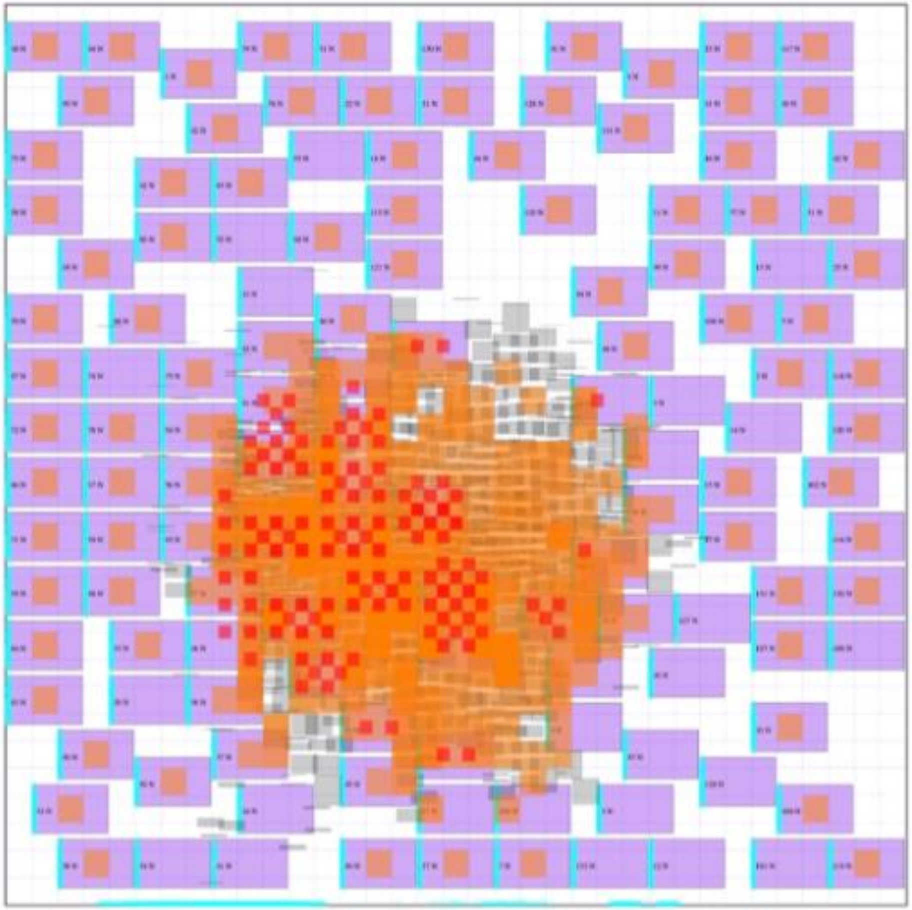
DeepRacer

- AWS DeepRacer (by our lab CGI)
  - 2020 AWS DeepRacer World Championship Cup: 1st + 3rd places
  - 2022 AWS DeepRacer World Championship Cup: 1st + 2nd + 3rd places

Hoang-Giang Cao, I Lee, Bo-Jiun Hsu, Zheng-Yi Lee, Yu-Wei Shih, Hsueh-Cheng Wang, I-Chen Wu, "Image-based Regularization for Action Smoothness in Autonomous Miniature Racing Car with Deep Reinforcement Learning", 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Detroit, October 2023.

Better and Faster Chip Design

- Better and faster for chip design than any human designer.
  - Generate chip floorplans that are comparable or superior to human experts in under six hours,
  - whereas humans take months to produce acceptable floorplans for modern accelerators.



[1] A. Mirhoseini, et al. (by Google brain), A graph placement methodology for fast chip design, Nature, 2021

Reinforcement Learning from Human Feedback (RLHF) for ChatGPT

By OpenAI CEO (2022)

(at 6:56/2:23:56, Sam Altman in Lex Fridman Podcast)

- "… And RLHF is how we take some human feedback,
  - the simplest version of this is show two outputs
  - ask which one is better than the other
  - which one the human raters prefer
  - and then feed that back into the model with RL
  - **that process works remarkably well with in my opinion**
  - **remarkably little data to make the model more useful**
- So, RLHF is how we align the model to what humans want it to do.
  …"

https://youtu.be/L_Guz73e6fw

Sam Altman: OpenAI CEO on GPT-4, ChatGPT, and the Future of AI | Lex Fridman Podcast #367

David Silver:

(the leader of the AlphaGo team)

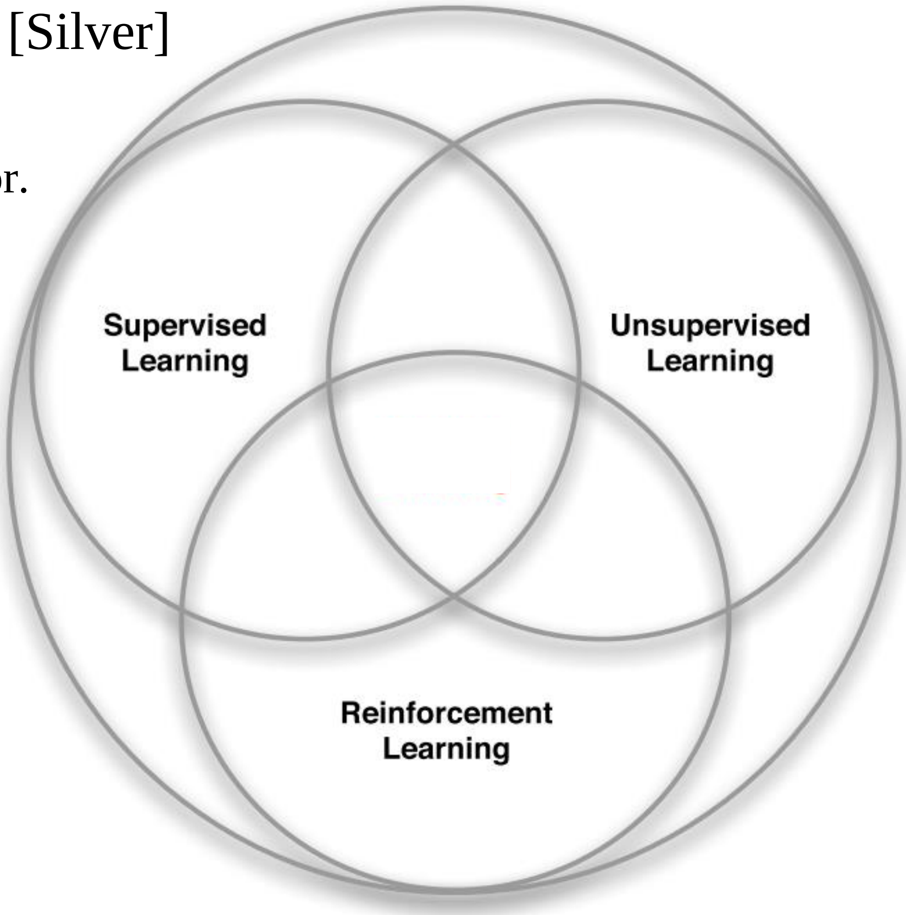
"DL+RL = AI"

Many Faces of Reinforcement Learning

- Computer Science
  - Machine Learning
- Engineering
  - Optimal Control
- Mathematics
  - Operations Research
- Economics
  - Bounded Rationality
- Psychology
  - Classical/Operant Conditioning
- Neuroscience
  - Reward System

Branches of Machine Learning

- Supervised Learning (SL) [Silver]
  - learning from a training set of labeled examples provided by a knowledgeable external supervisor.
- Unsupervised Learning (UL)
  - typically about finding structure hidden in collections of unlabeled data.
- Reinforcement Learning (RL)
  - learning from interaction



What are different from others?

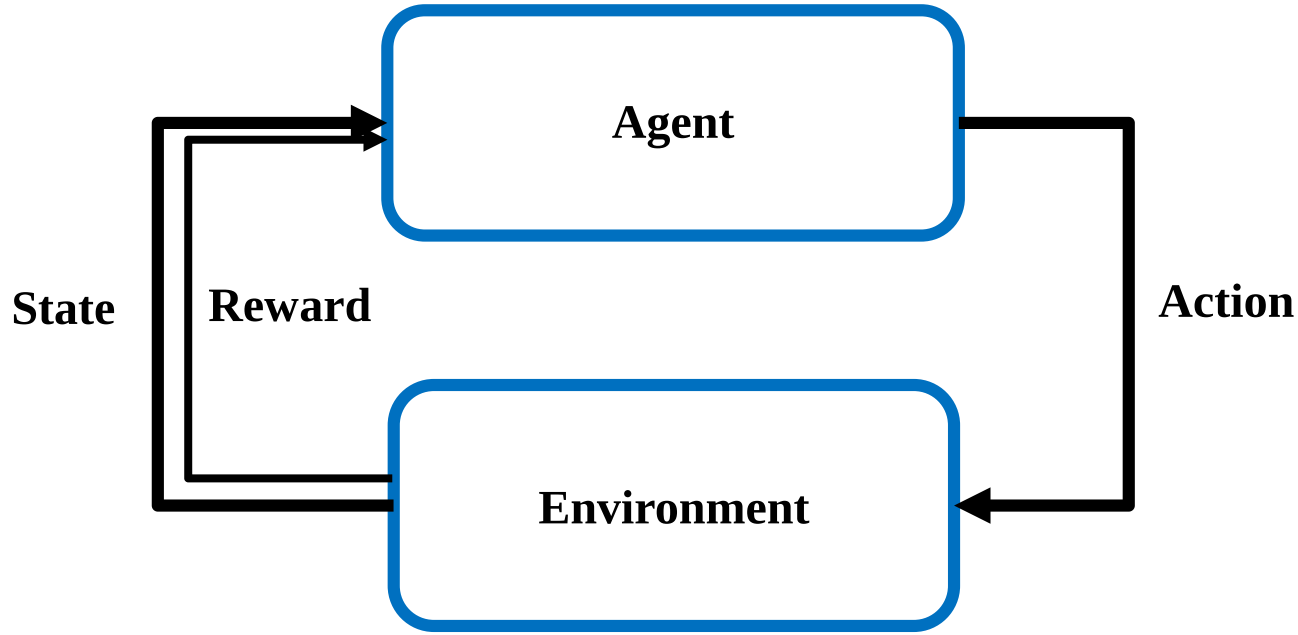
- Characteristics:
  – No supervisor, only a reward signal
  – Feedback is delayed, not instantaneous
  – Time really matters
  – Agent's actions affect the subsequent data and actions
- UL vs. RL:
  – RL is learning from interaction.
  – RL does not rely on examples of correct behavior.
  – RL is trying to maximize a reward signal, instead of trying to find hidden structure.

Reinforcement Learning

- A computational approach to learning from interaction
  - Explore designs for machines that are effective in
    - solving learning problems of scientific or economic interest,
    - evaluating the designs through mathematical analysis or computational experiments.
  - Focus on goal-directed learning from interaction, when compared with other approaches to machine learning.
  - The learner must discover which actions yield the most reward by trying them.
    - Two characteristics: most important distinguishing features of reinforcement learning.
      - trial-and-error search
      - delayed reward

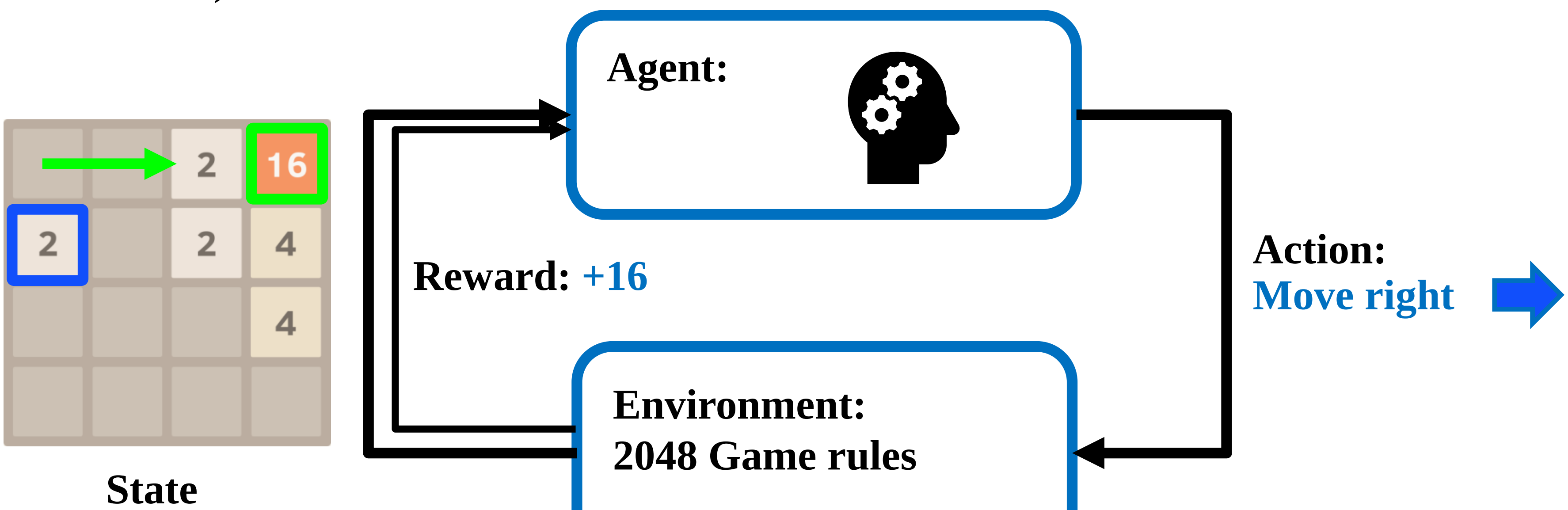
Agent-Environment Interaction Framework

- A kind of AI **computational approach** to learning from interaction
- Agent-Environment Interaction Framework (代理者-環境 互動框架)



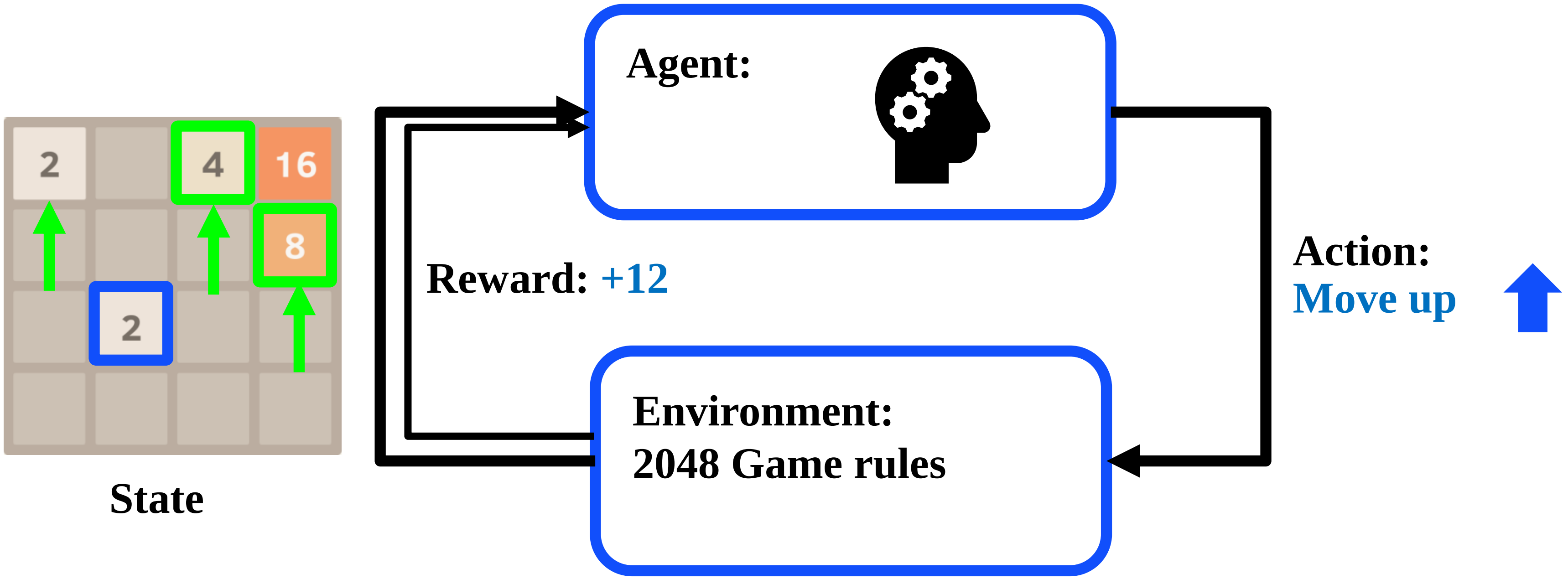
Reinforcement Learning (RL)

- A kind of AI **computational approach** to learning from interaction
- Agent-Environment Interaction Framework (代理者-環境 互動框架)



Reinforcement Learning (RL)

- A kind of AI **computational approach** to learning from interaction
- Agent-Environment Interaction Framework (代理者-環境 互動框架)



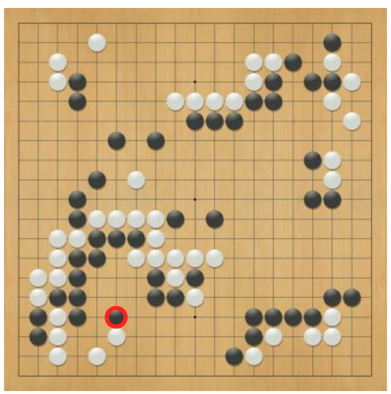
Reinforcement Learning (RL)

- A kind of AI **computational approach** to learning from interaction
- Agent-Environment Interaction Framework (代理者-環境 互動框架)

Agent:
CGI Go Program

Reward: + 0.3

Action:
C2



State

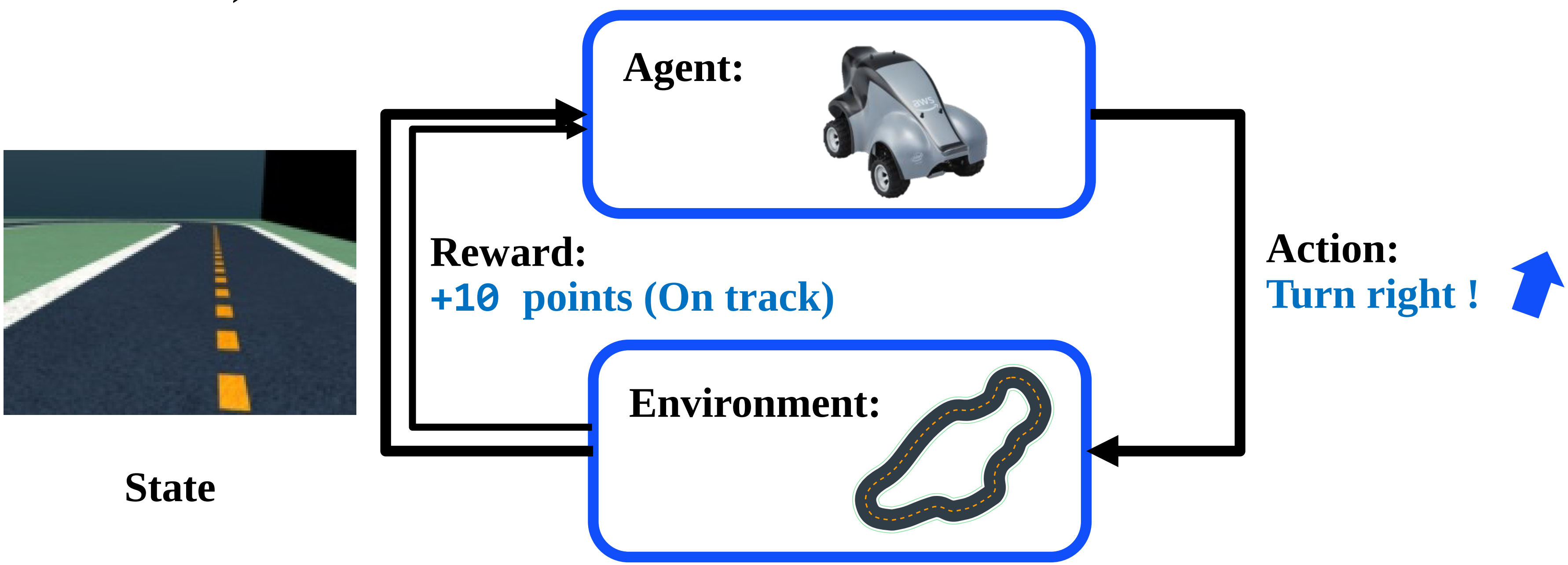
Environment:
Go game rules
對手 :柯潔 (F4)

Reinforcement Learning (RL)

- A kind of AI **computational approach** to learning from interaction
- Agent-Environment Interaction Framework (代理者-環境 互動框架)

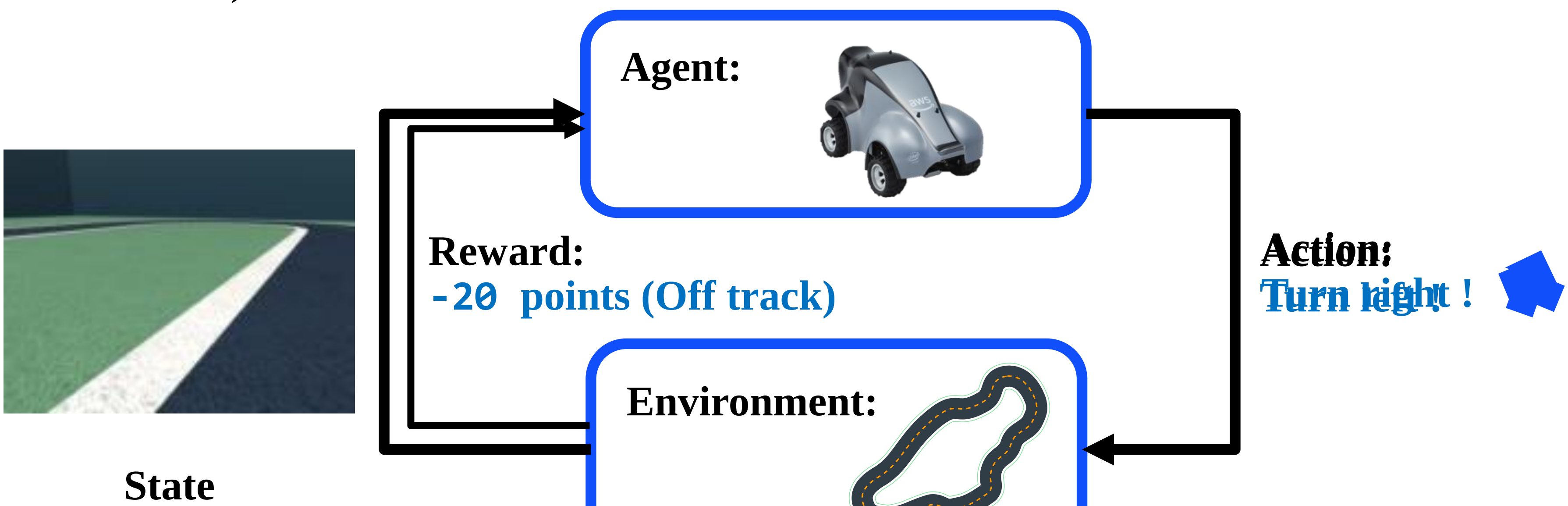
Reinforcement Learning (RL)

- A kind of AI **computational approach** to learning from interaction
- Agent-Environment Interaction Framework (代理者-環境 互動框架)



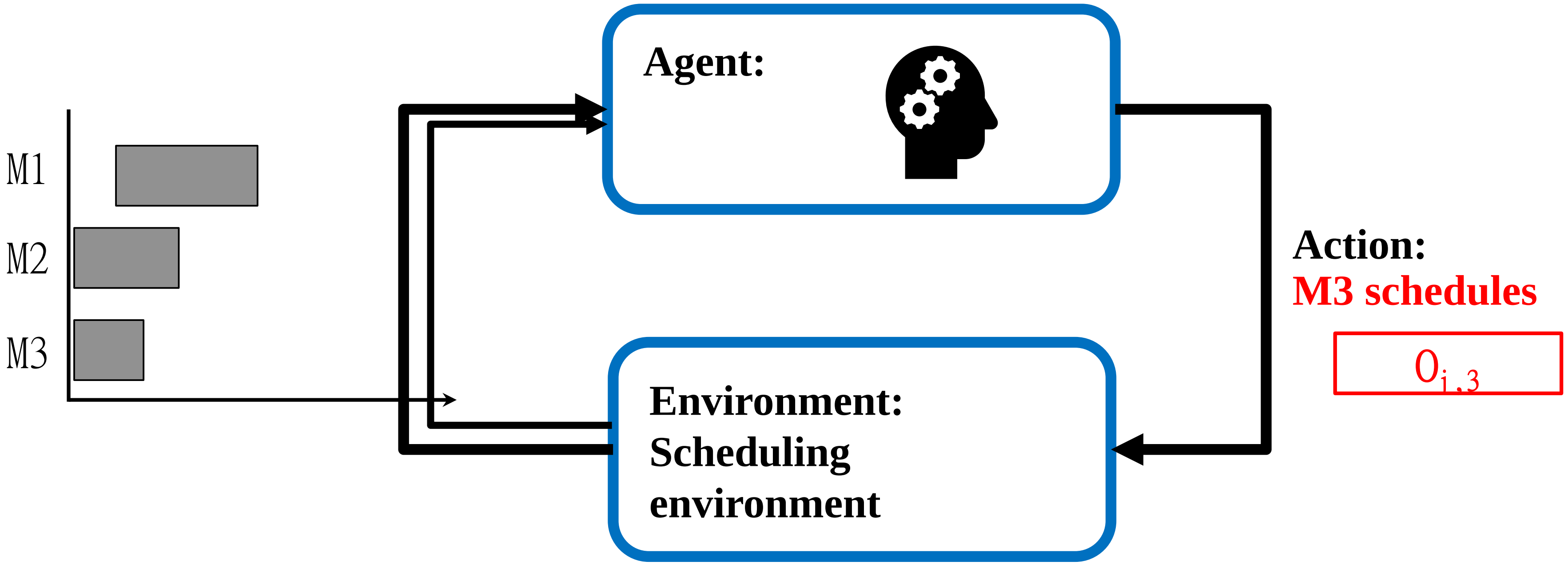
Reinforcement Learning (RL)

- A kind of AI **computational approach** to learning from interaction
- Agent-Environment Interaction Framework (代理者-環境 互動框架)



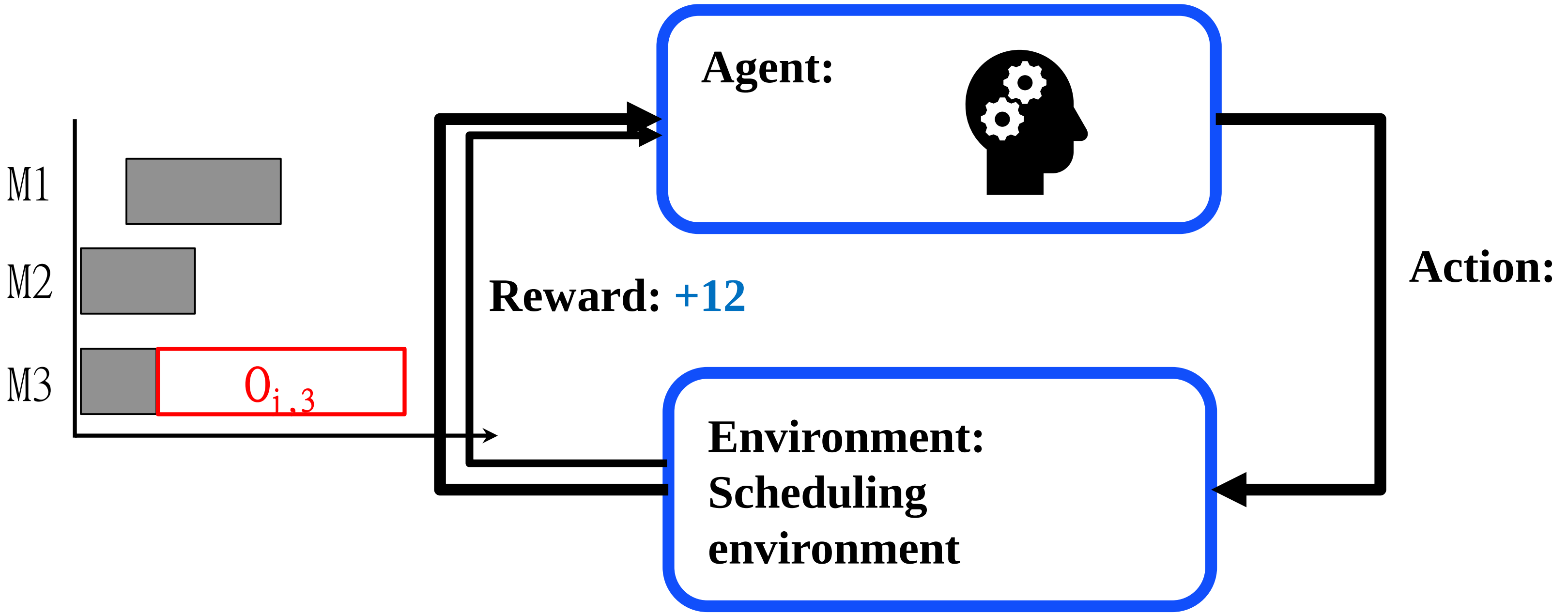
Reinforcement Learning (RL)

- A kind of AI **computational approach** to learn from interaction
- Agent-Environment Interaction Framework



Reinforcement Learning (RL)

- A kind of AI **computational approach** to learn from interaction
- Agent-Environment Interaction Framework



States and Actions in the Framework

Environment: reaction

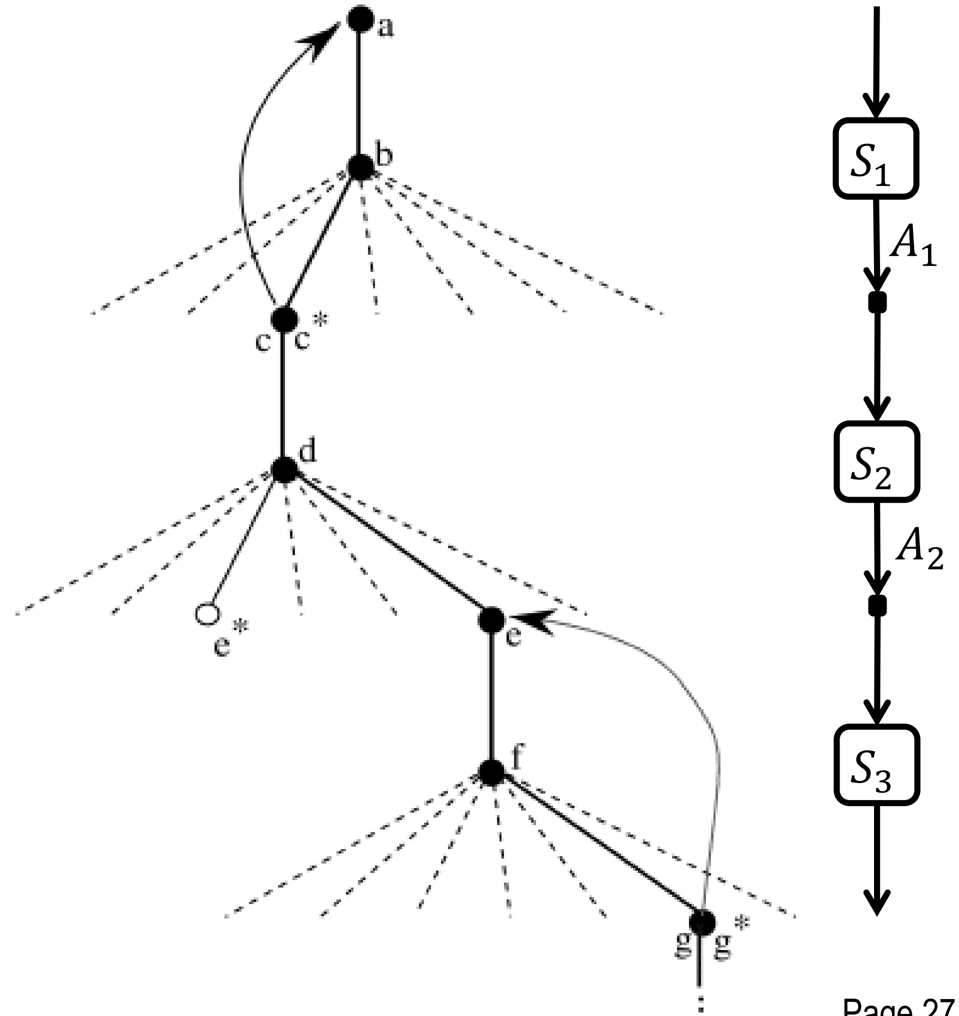
Agent: action

Environment: reaction

Agent: action

Environment: reaction

Agent: action



2048

Environment: Tile generation

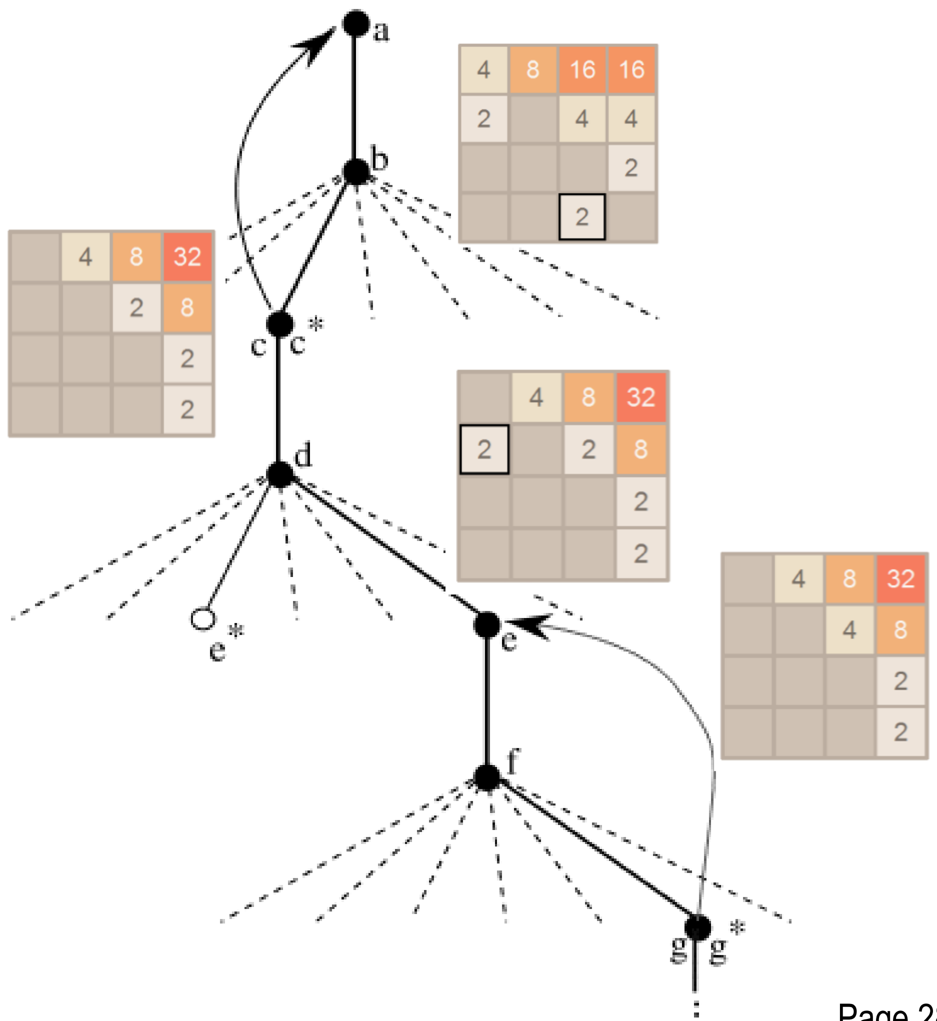
Agent: our move

Environment: Tile generation

Agent: our move

Environment: Tile generation

Agent: our move



Go

Environment: opponent's move

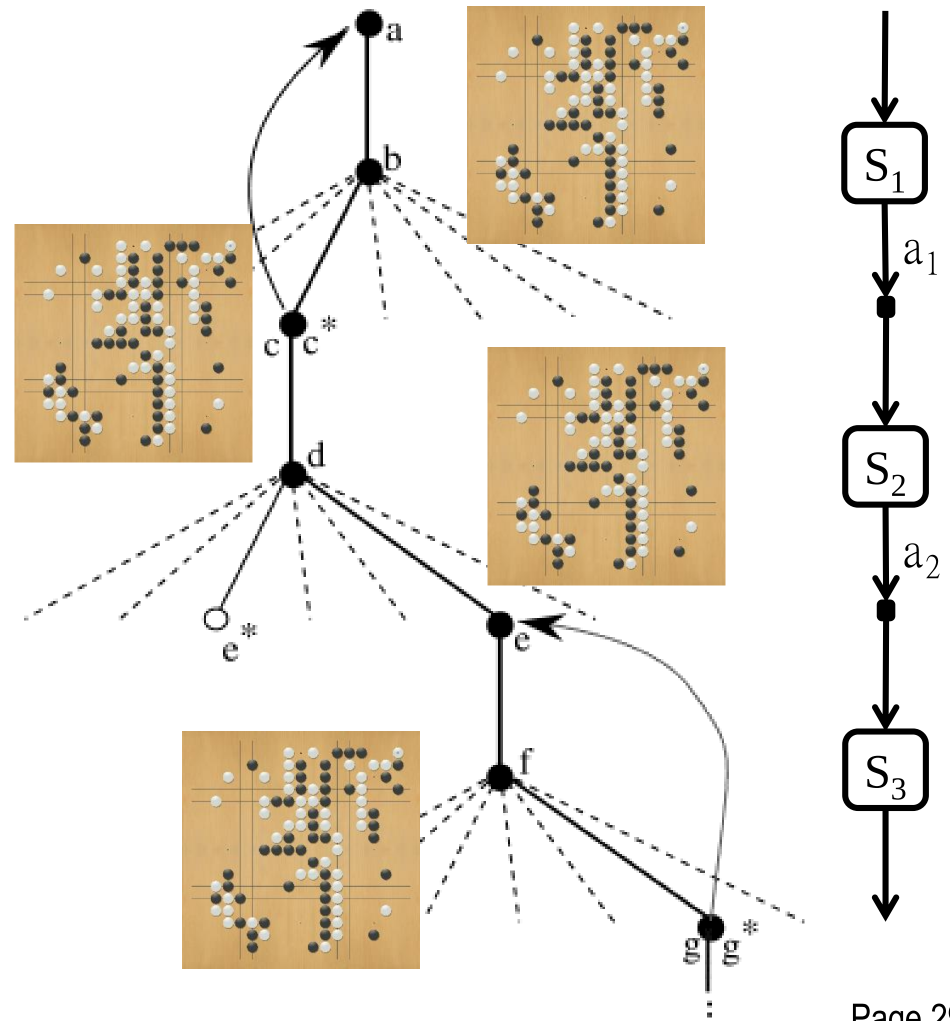
Agent: our move

Environment: opponent's move

Agent: our move

Environment: opponent's move

Agent: our move



Robot

Environment: Dynamics

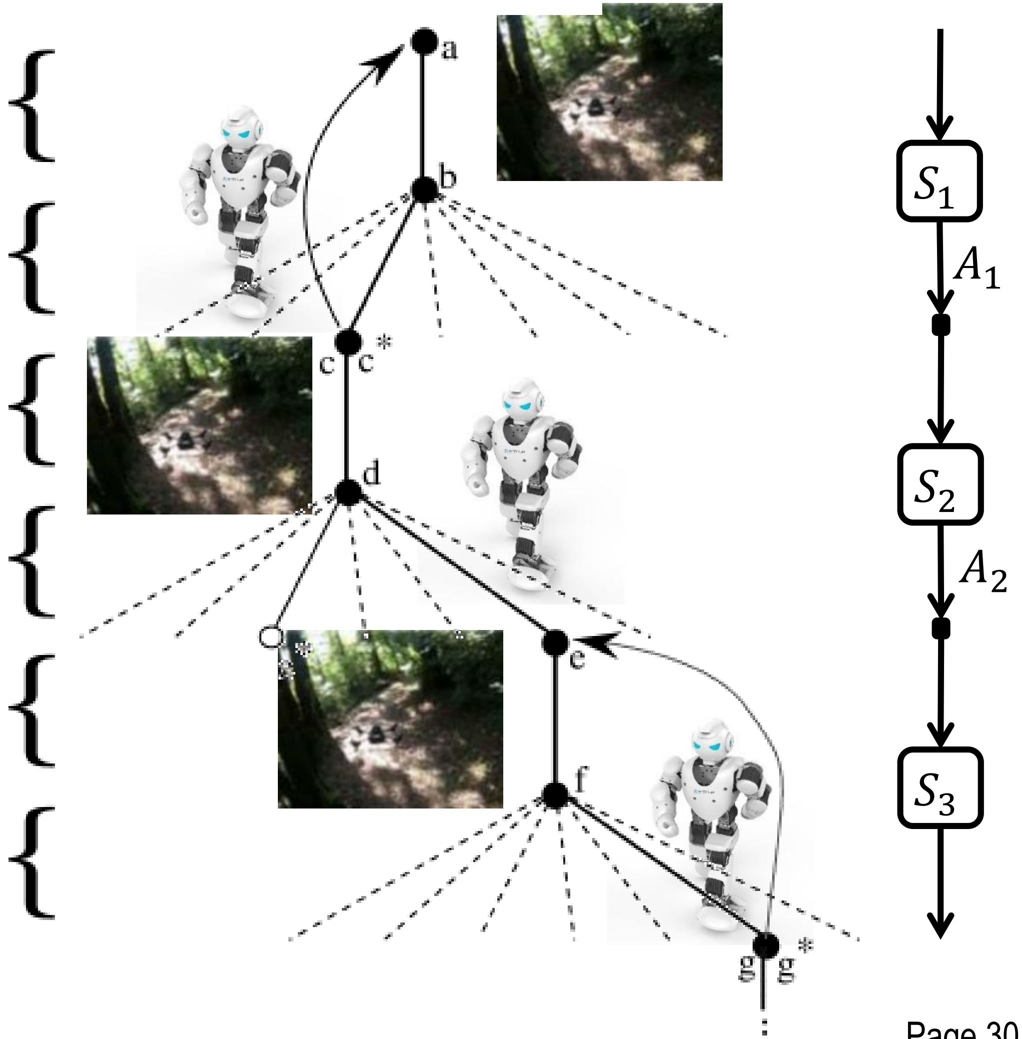
Agent: Navigate

Environment: Dynamics

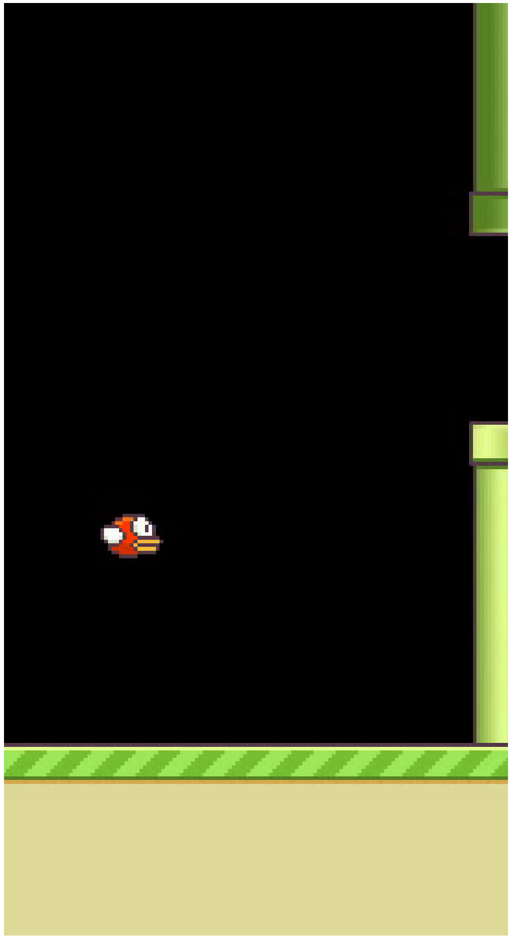
Agent: Navigate

Environment: Dynamics

Agent: Navigate



More Example: Flappy Bird

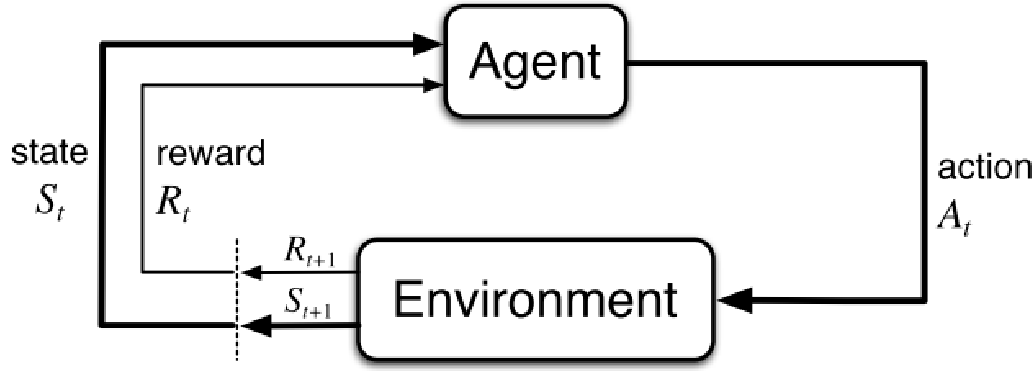


More Toy Example: Pendulum

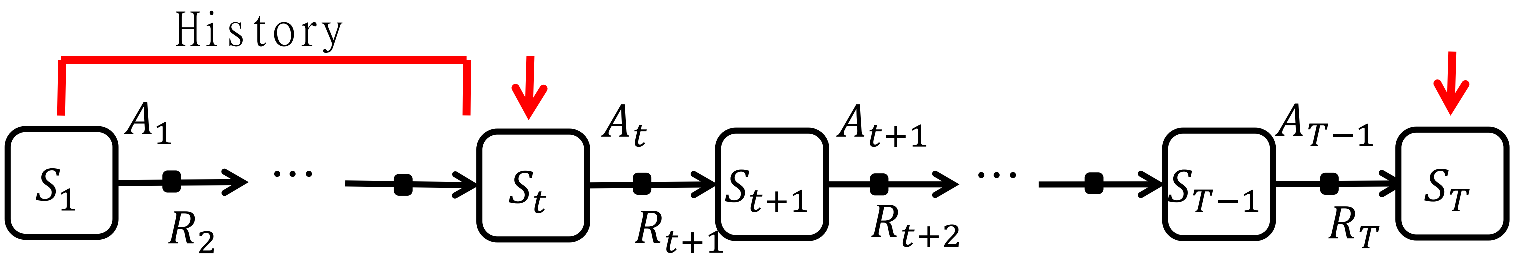
More Example: RL Demo (DDPG)

Markov Decision Processes (MDP)

- A (Finite) Markov Decision Process is a tuple $<\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma>$
  - $\mathcal{S}$ is a (finite) set of states
  - $\mathcal{A}$ is a (finite) set of actions
  - $\mathcal{P}$ is a state transition probability matrix (part of the environment), $\mathcal{P}_{ss'}^{a} = \mathbb{P}[S_{t+1} = s' \mid S_t = s, A_t = a]$
  - $\mathcal{R}$ is a reward function, $\mathcal{R}_{s}^{a} = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a]$
  - γ is a discount factor $\gamma \in [0, 1]$.



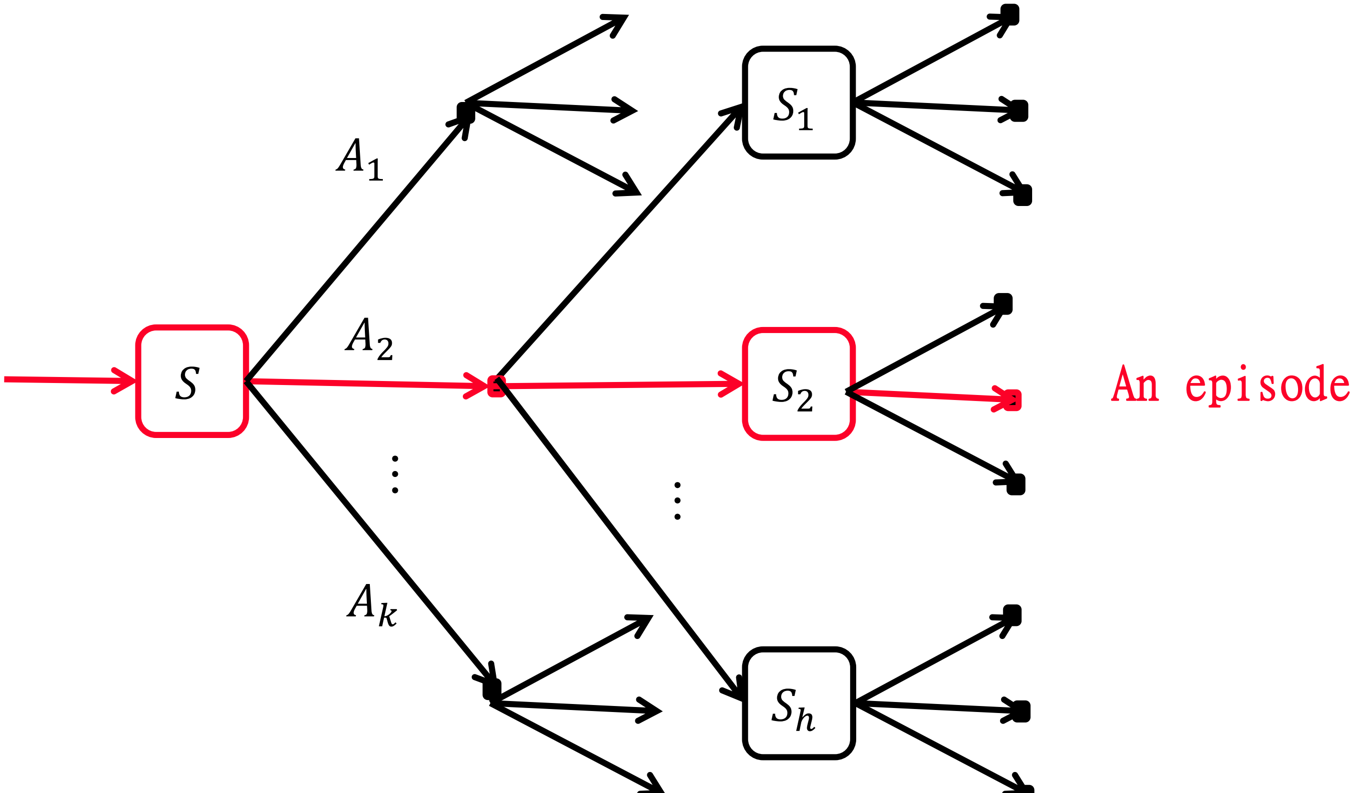
Markov Property



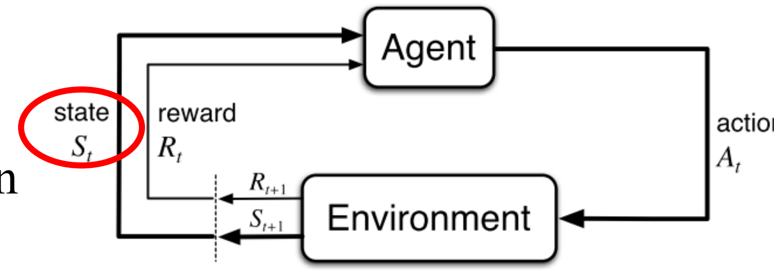
- An episode: (assuming finite and MDP here for simplicity)
  - States: S_i
    - Initial state: S_1
    - Current state: S_t
    - End state: S_T (not necessarily required)
  - Actions: A_i
  - History: $H_t = (S_1, A_1, R_2, S_2, A_2, R_3, S_3, \ldots, R_t)$
- Markov Property:
  - "The future is independent of the past given the present"
  - A state S_t is Markov if and only if

$$\mathbb{P}[S_{t+1} | S_t] = \mathbb{P}[S_{t+1} | S_1, \ldots, S_t]$$

Episode and Space



Environment State vs. Agent State



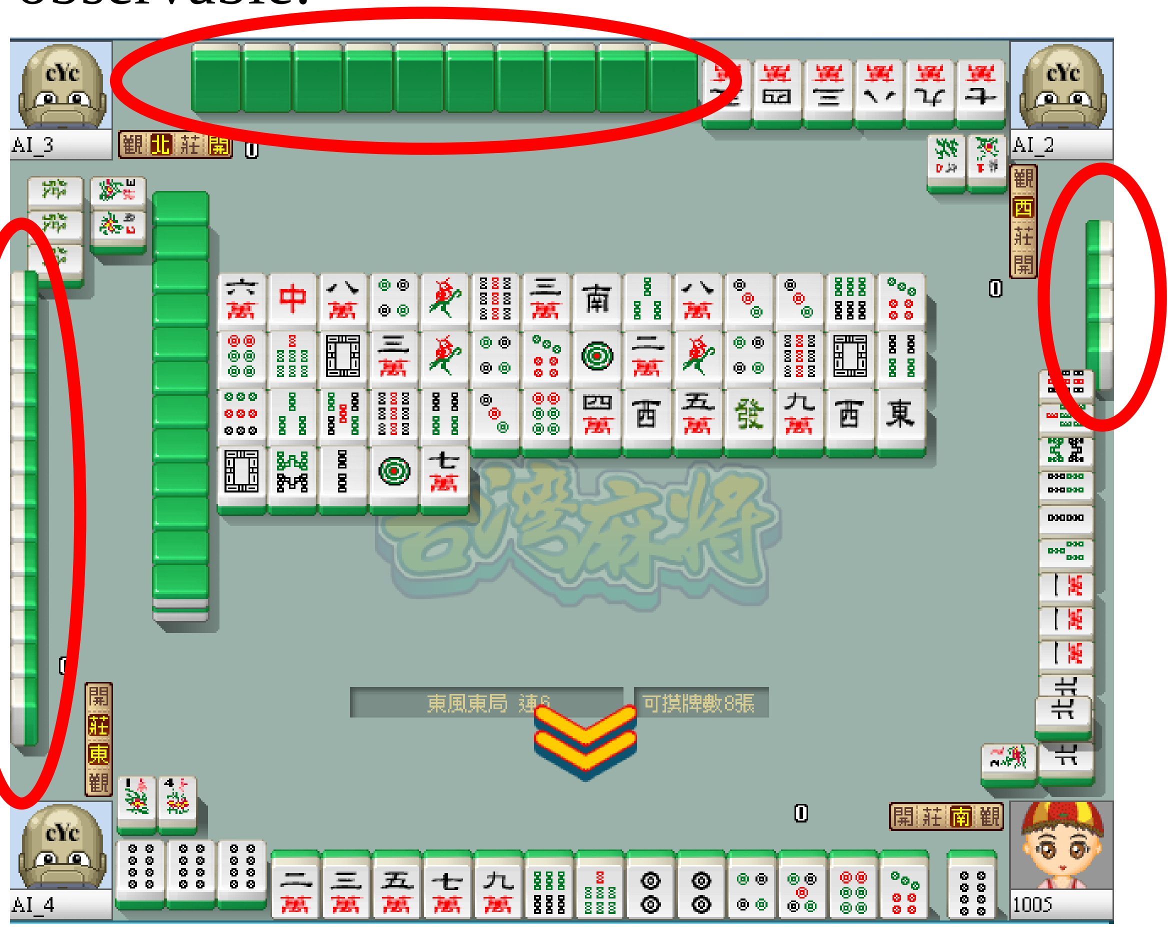
- The environment state S_t^e:
  - the environment's private representation
    - i.e. whatever data the environment uses to pick the next observation/reward
  - The environment state is not necessarily visible to the agent
    - Even if S_t^e is visible, it may contain irrelevant information
- The agent state S_t^a:
  - The agent's internal representation
    - i.e. whatever information the agent uses to pick the next action
    - i.e. it is the information used by reinforcement learning algorithms
  - It can be any function of history:

$$S_t^a = f(H_t)$$

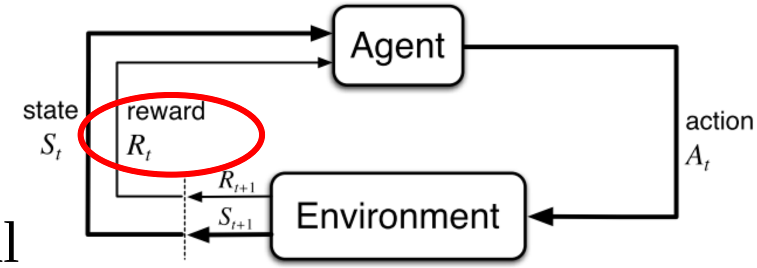
- Partially Observable: (not discussed here)
  - When $S_t^a \neq S_t^e$

Example: Mahjong

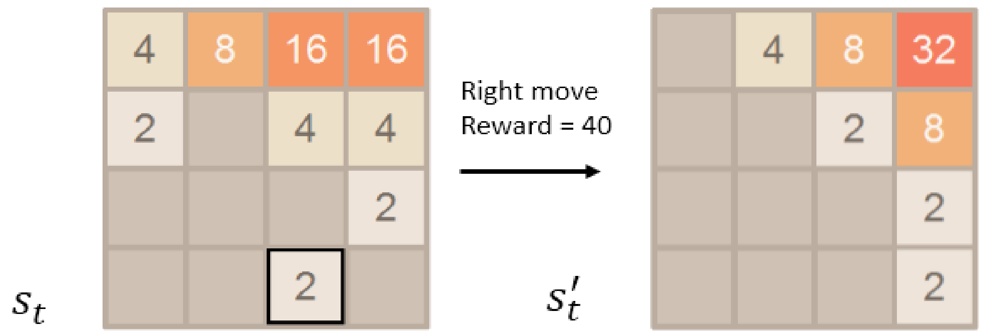
- Partially observable:



Rewards



- A reward R_t is a scalar feedback signal
  - Indicates how well agent is doing at step t
  - The agent's job is to maximize cumulative reward
  - Reinforcement learning is based on the reward hypothesis
  - Example: (2048)

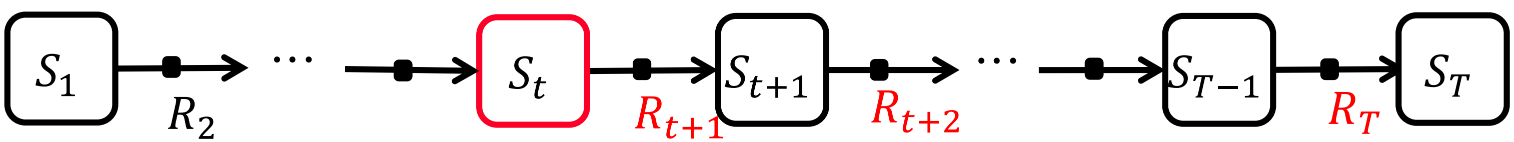


Definition (Reward Hypothesis)

- All goals can be described by the maximization of expected cumulative reward

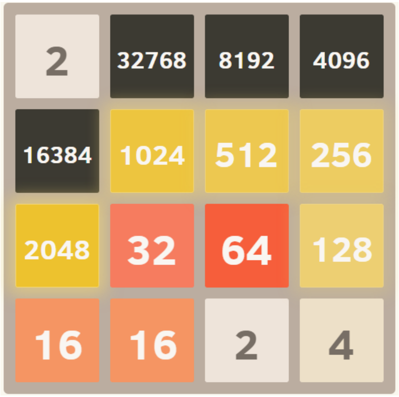
Sequential Decision Making

- Goal:
  - Select actions to maximize total future reward
- Maximize $R_{t+1} + R_{t+2} + \cdots + R_T$
  - assuming time = t.



- Notes:
  - Actions may have long term consequences
  - Reward may be delayed
  - It may be better to sacrifice immediate reward to gain more long-term reward

Sequential Decision Making – Examples



- Examples:
  - In 2048, establish a sequence of $(2^t, 2^{t-1}, 2^{t-2}, \ldots)$
  - In chess, block opponent moves to help winning chances many moves from now.
  - In a financial investment, may take months to mature
  - In robotics, refuel a helicopter to prevent a crash.

Return

Definition

- The return G_t is the total discounted reward from time-step t.

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

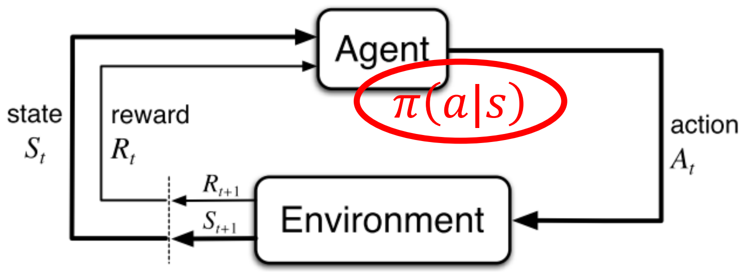
Notes:

- The discount $\gamma \in [0, 1]$ is the present value of future rewards
- The value of receiving reward R is diminishing
  - $\gamma^k R$, after $k+1$ time-steps.
- This values immediate reward above delayed reward.
- Discount:
  - γ close to 0 leads to ”myopic” evaluation
  - γ close to 1 leads to ”far-sighted” evaluation
  - Important for infinite episodes.

Major Components of an RL Agent

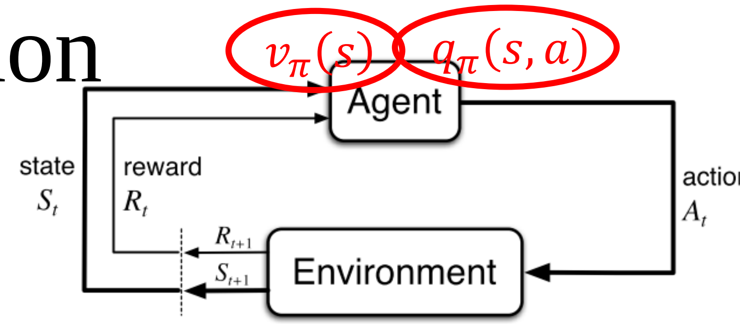
- Value function: how good is each state and/or action
- Policy: agent's behavior function
- Model: agent's representation of the environment

Policy



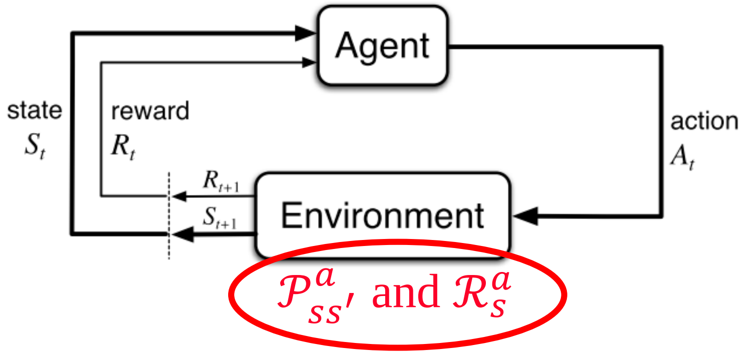
- A policy is the agent's behavior
  - It is a map from state to action,
- Policy types:
  - Deterministic policy: $a = \pi(s_i)$
  - Stochastic policy: $\pi(a|s) = \mathbb{P}[A_t = a|\ S_t = s]$
    - Sometimes, written in $\pi(s, a)$.
- Examples:
  - In 2048: Up/down/left/right
  - In robotics: angle/force/…

Value Function



- A value function is
  a prediction of future reward
  - Used to evaluate the goodness/badness of states
    - therefore to select between actions.
  - Return $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots$
- Types of value functions under policy π:
  - State value function: the expected return from s.
    $$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots \mid S_t = s] \\ &= \mathbb{E}_\pi[G_t \mid S_t = s] \end{aligned}$$
  - Q-Value function: the expected return from s taking action a.
    $$q_\pi(s,a) = \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a]$$
- Examples:
  - In 2048, the expected score from a board S_t.

Model



- A model predicts what the environment will do next
  - $\mathcal{P}$ is a state transition probability matrix,
    $\mathcal{P}^a_{ss'} = \mathbb{P}[S_{t+1} = s' |\ S_t = s, A_t = a]$
    - predicts the next state
  - $\mathcal{R}$ is a reward function,
    $\mathcal{R}^a_s = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$
    - predicts the next (immediate) reward
- Examples:
  - In 2048:
    - After a move, $\mathcal{P}$ is to generate a tile randomly as follows:
      - 2-tile: with probability of 9/10
      - 4-tile: with probability of 1/10

Categorizing RL Agents (Policy & Value)

- Value Based
  - No Policy (Implicit)
  - Value Function
- Policy Based
  - Policy
  - No Value Function (Implicit)
- Actor Critic
  - Policy
  - Value Function

Categorizing RL Agents (Model)

- Model Free
  - Policy and/or Value Function
  - No Model
- Model Based
  - Policy and/or Value Function
  - Model