Introduction to Reinforcement Learning

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

David Silver:

(the leader of the AlphaGo team)

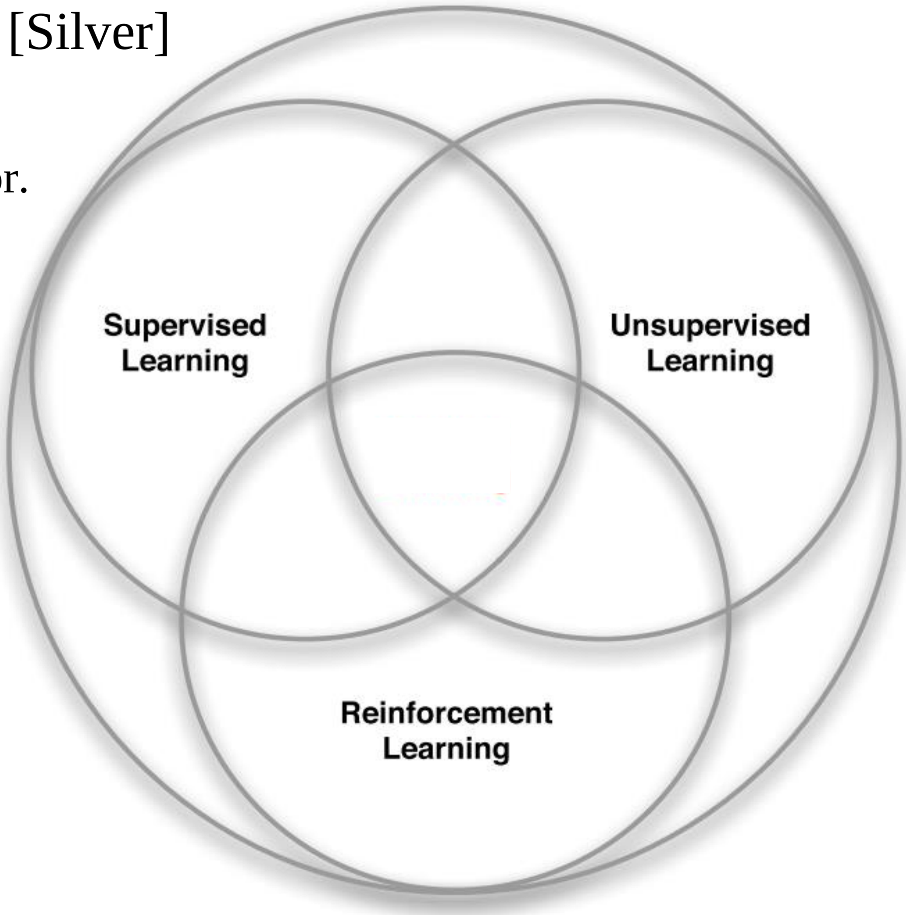
"DL+RL = AI"

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?

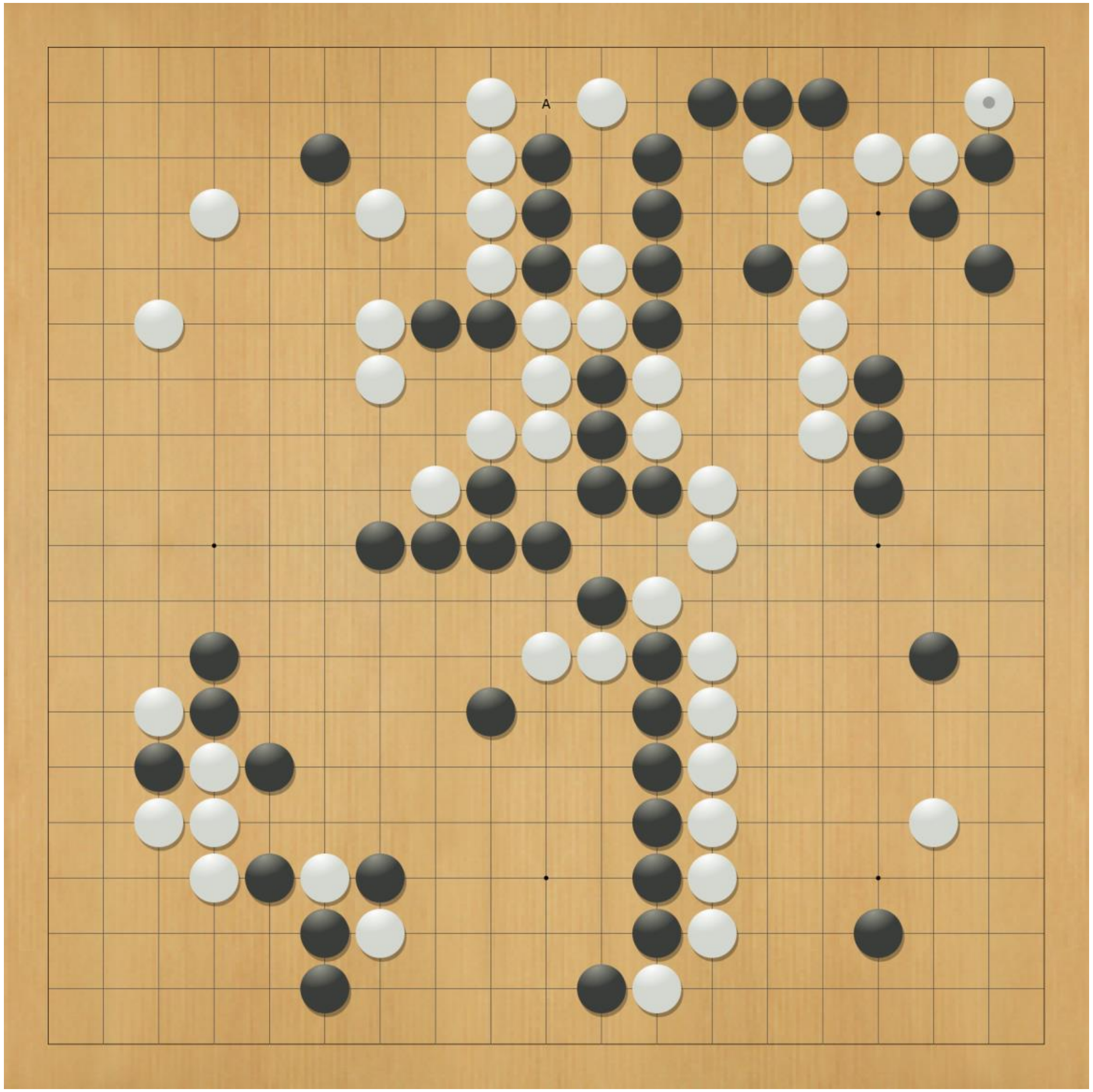
- Characteristics:
  - No supervisor, only a reward signal
  - Feedback is delayed, not instantaneous
  - Time really matters
  - Agent's actions affect the subsequent data
- UL vs. RL:
  - RL is learning from interaction.
  - RL does not rely on examples of correct behavior.
  - RL is trying to maximize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

Successful Examples

- In AI, it has been used to defeat human champions at games of skill.
  - Backgammon (Tesauro, 1994).
  - Connect6/2048/Threes! (Wu et al., 2015). Reach the top levels.
  - Go programs, used in the past 10 years. (Monte-Carlo Tree Search)
  - AlphaGo, using deep reinforcement learning (2016)
- In robotics, fly stunt maneuvers in robot-controlled helicopters (Abbeel et al.) and make a humanoid robot walk.
- In economics, manage an investment portfolio (Choi et al.).
- In neuroscience, model the human brain (Schultz et al.);
- In psychology, predict animal behavior (Sutton and Barto).
- In systems, control a power station
- In engineering, it has been used to allocate bandwidth to mobile phones and to manage complex power systems (Ernst et al.).

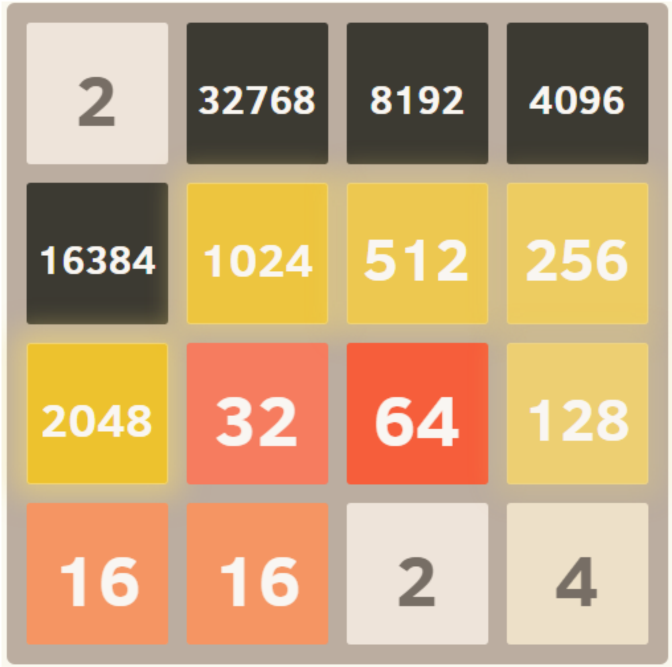
Board Game: Go

- Game 1: AlphaGo vs. 李世石



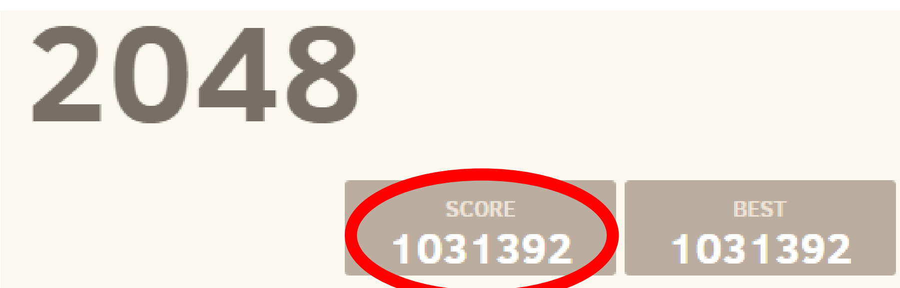
I-Chen Wu

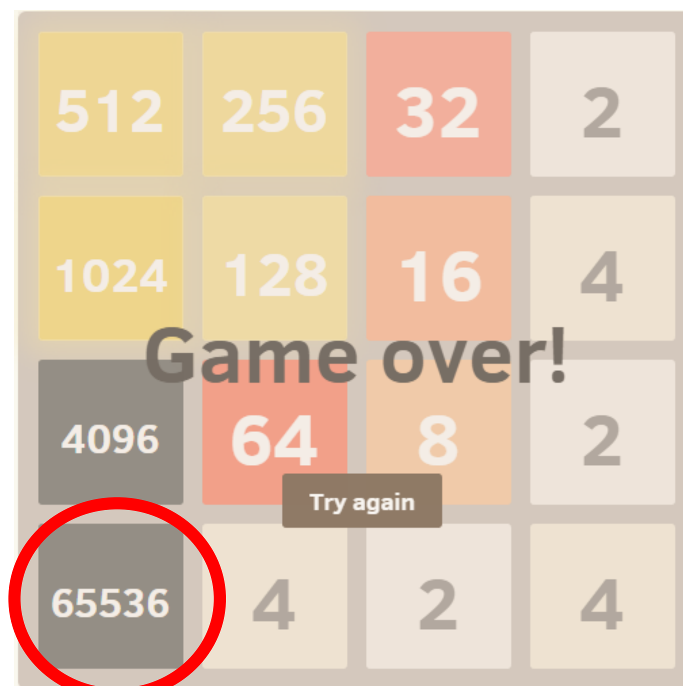
Stochastic Game: 2048 (lab)



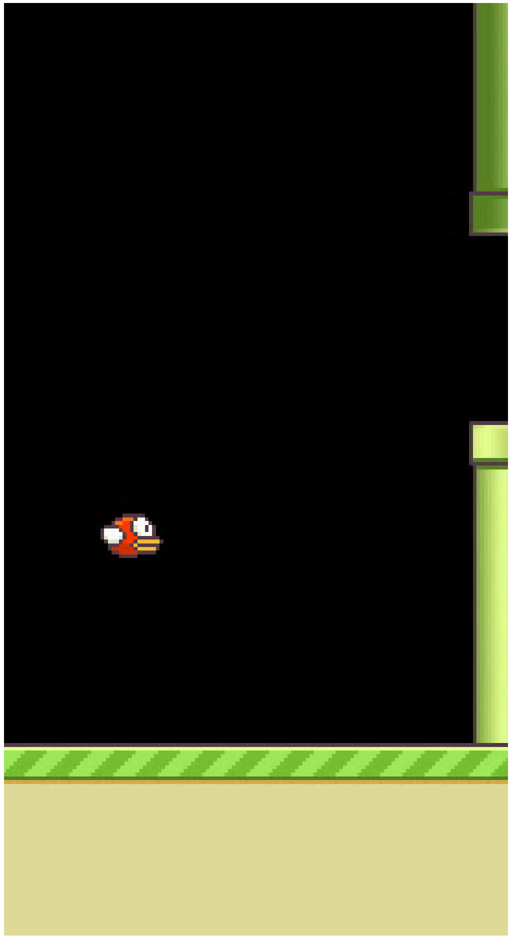
The First Game Reaching 65536 in the World (in 10,000 Trials)

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

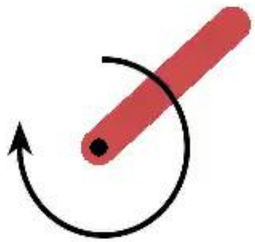




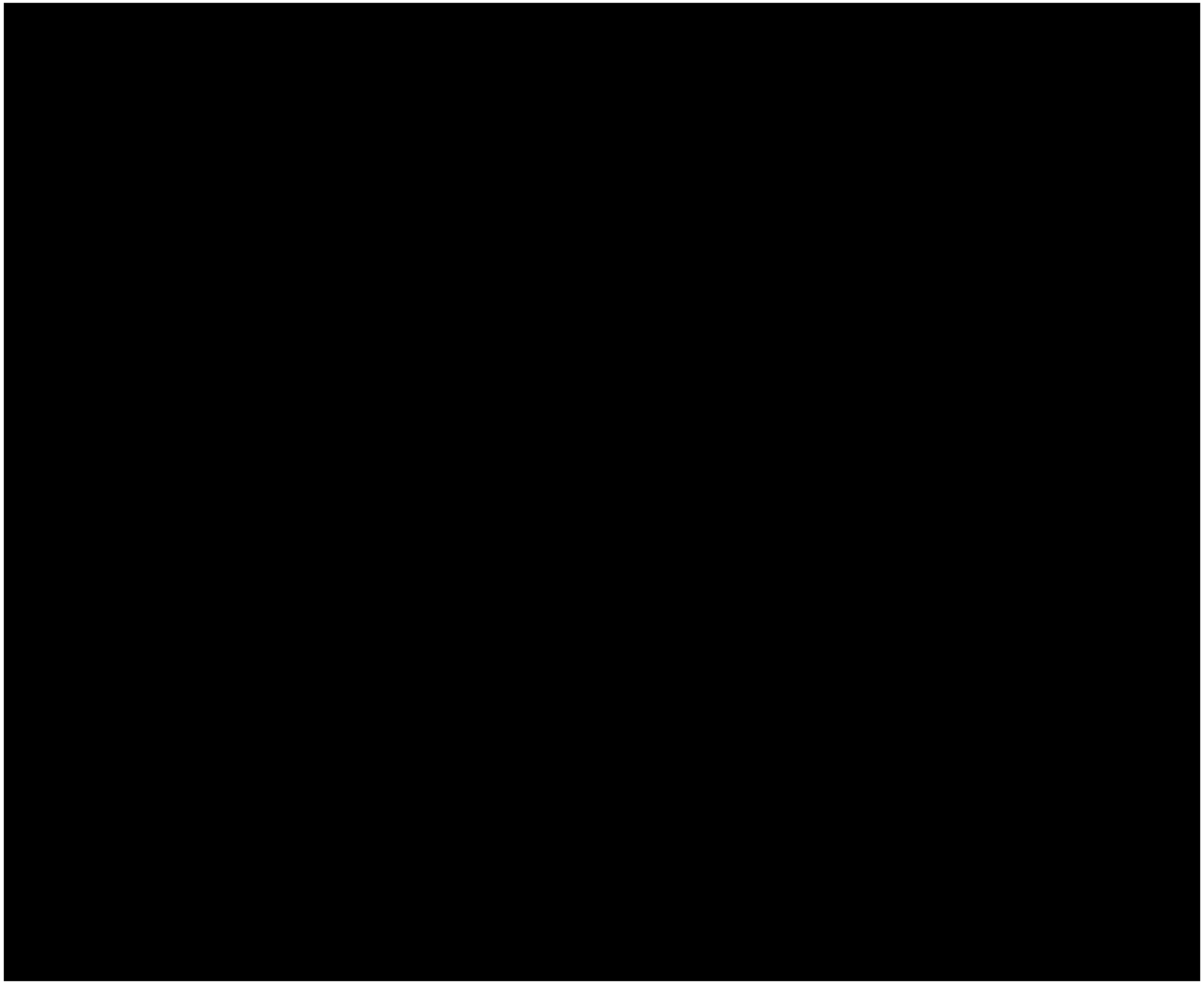
Video Games: Flappy Bird (lab)



Open AI: Gym (lab)



Demo



Another Demo

[Deisenroth et al, 2011] Learning to Control a Low-Cost Manipulator using Data-Efficient Reinforcement Learning

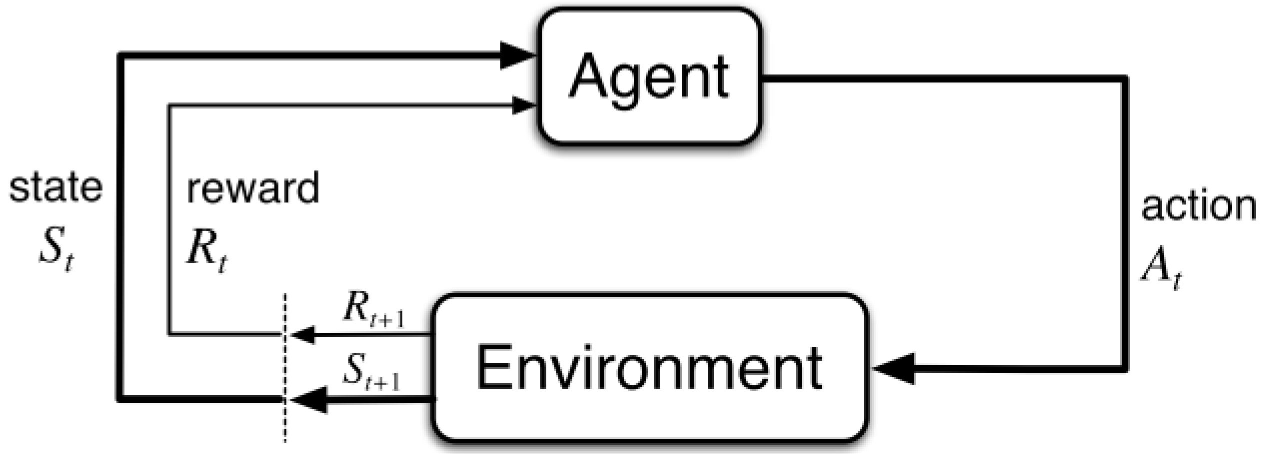
Nvidia Autonomous Car Video

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

- Agent: The learner and decision-maker.
- Environment: The thing it interacts with, comprising everything outside the agent.
- State: whatever information is available to the agent.
- Reward: single numbers.



States and Actions in the Framework

Environment: reaction

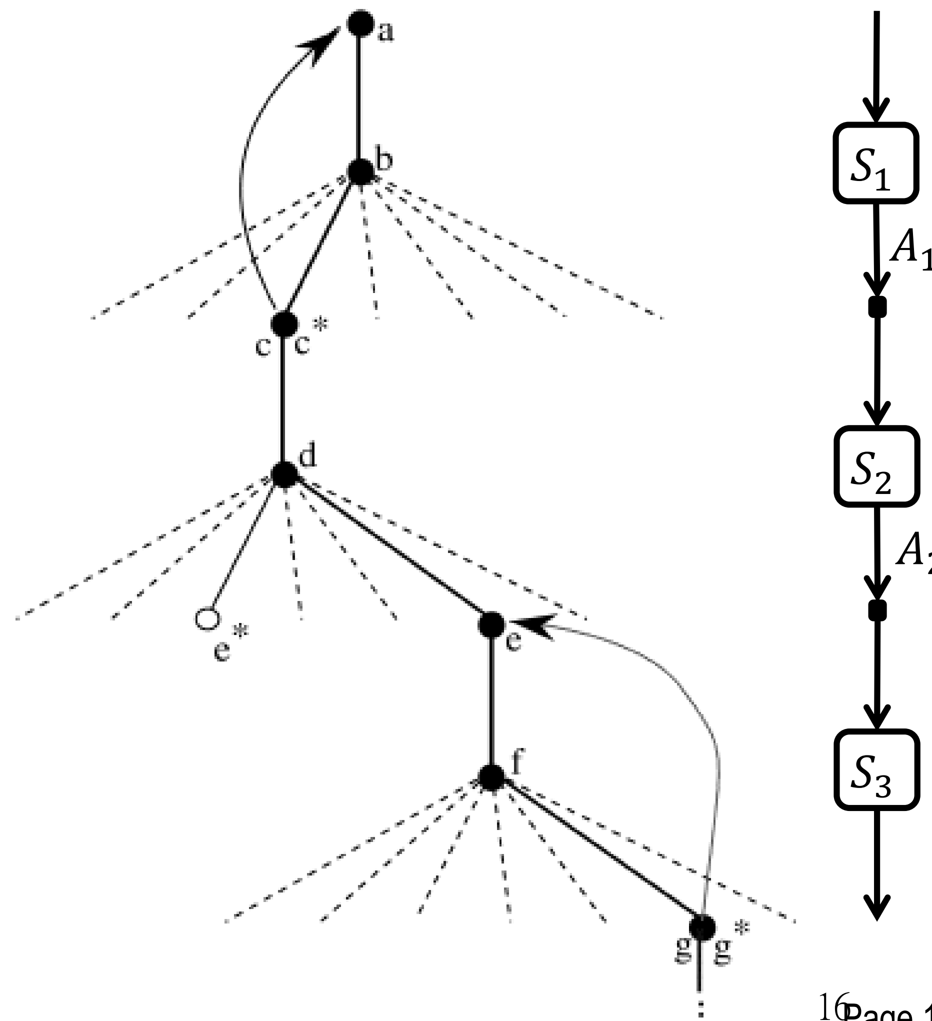
Agent: action

Environment: reaction

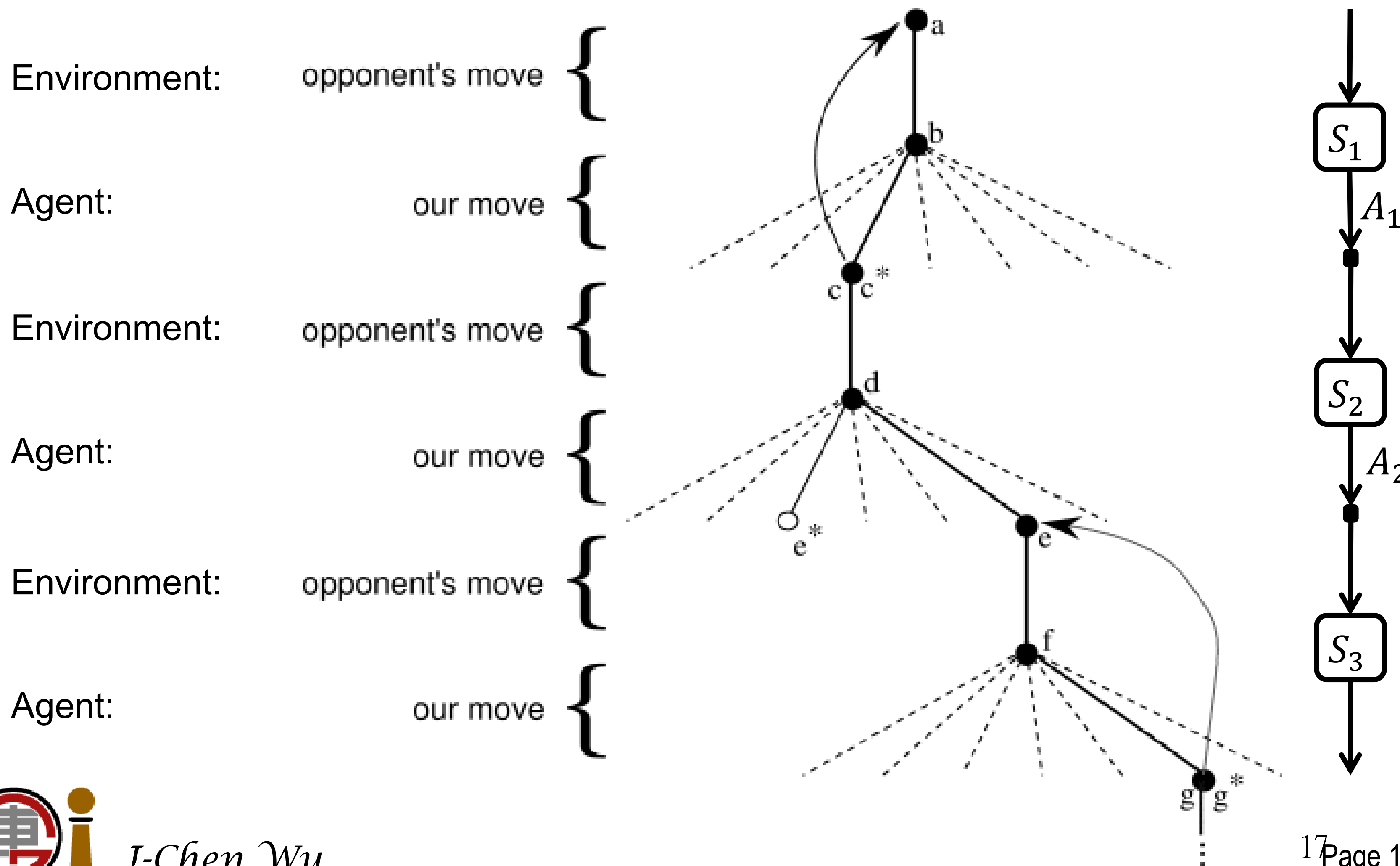
Agent: action

Environment: reaction

Agent: action



Go


Environment:
opponent's move
Agent:
our move
Environment:
opponent's move
Agent:
our move
Environment:
opponent's move
Agent:
our move
a
b
c
c*
d
e*
e
f
g
g*
S_1
A_1
S_2
A_2
S_3

2048

Environment: Tile generation

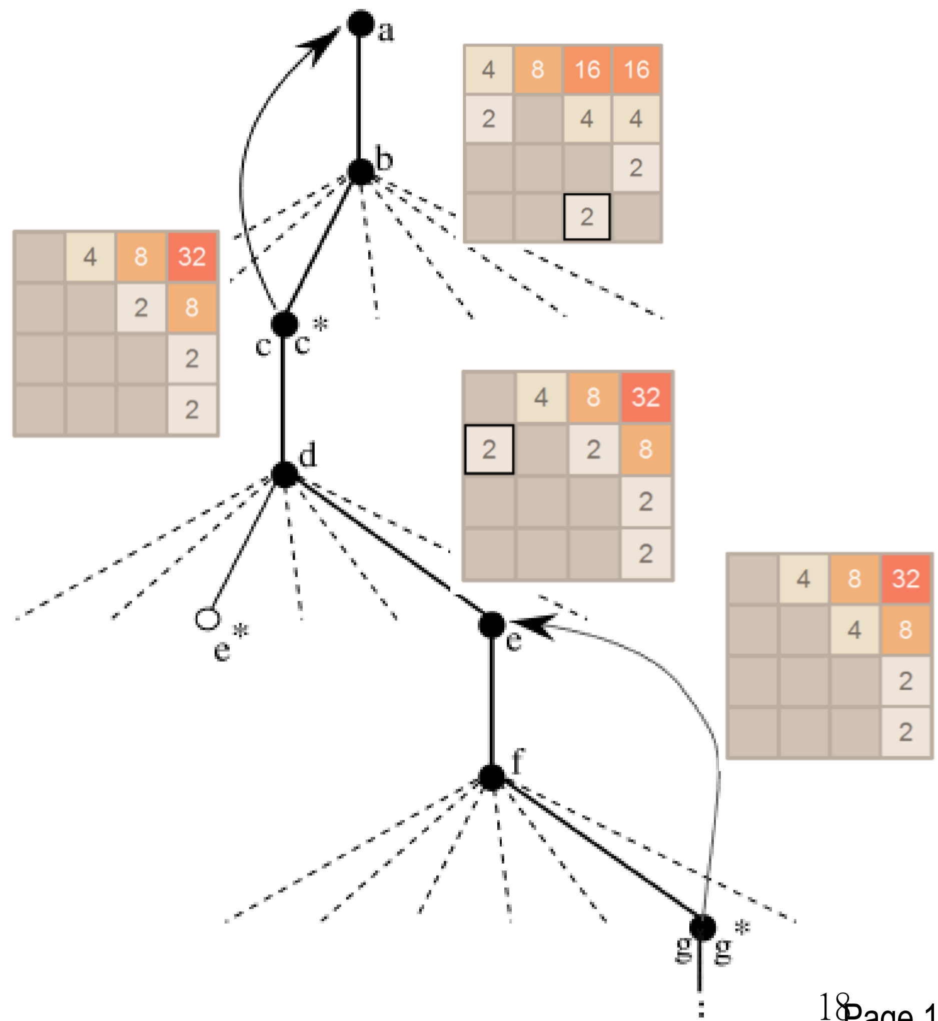
Agent: our move

Environment: Tile generation

Agent: our move

Environment: Tile generation

Agent: our move



Robot

Environment: Dynamics

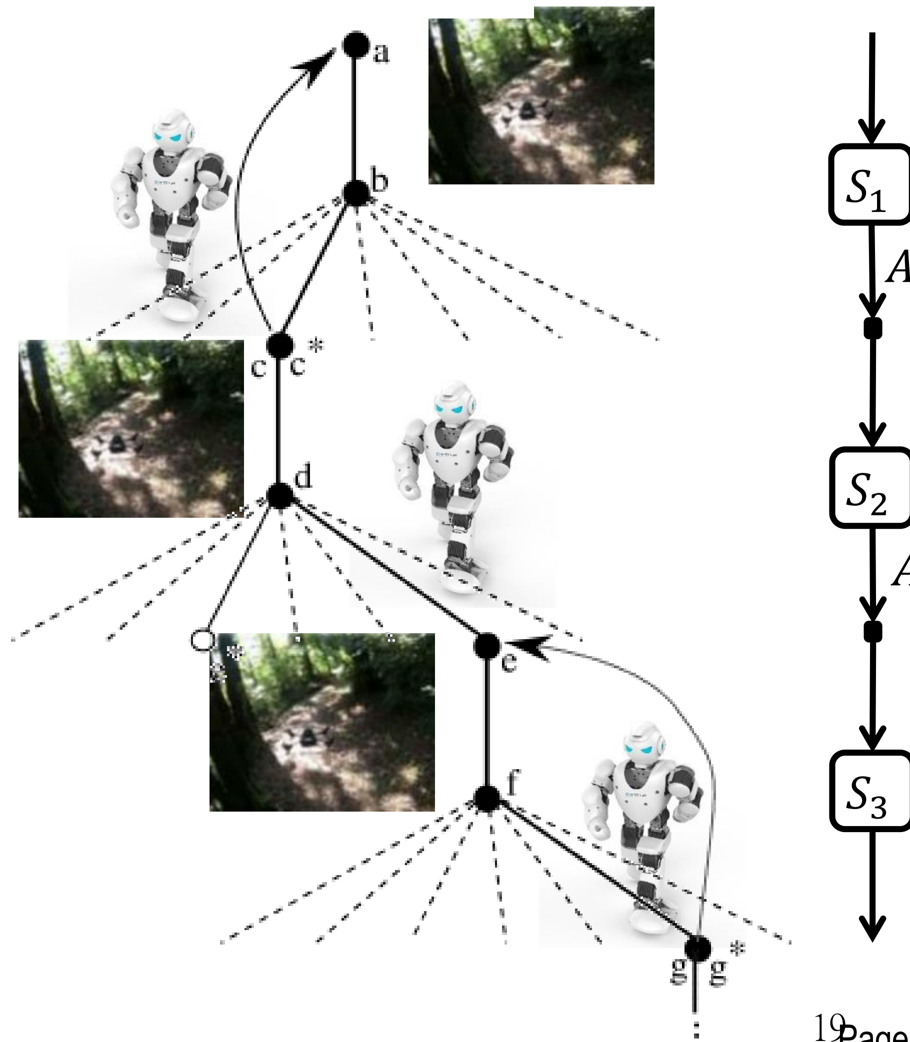
Agent: Navigate

Environment: Dynamics

Agent: Navigate

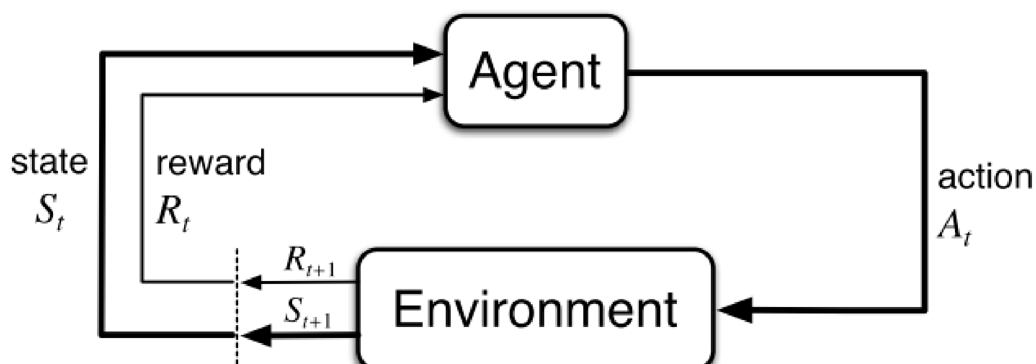
Environment: Dynamics

Agent: Navigate

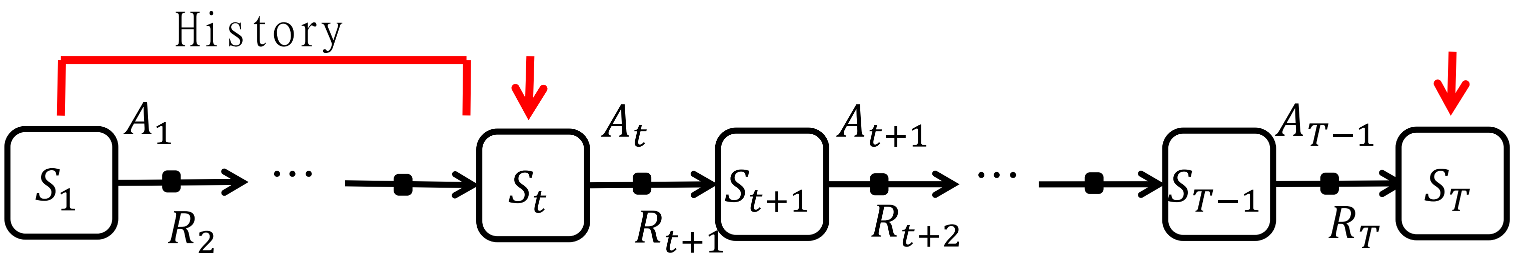


Markov Decision Processes (MDP)

- A 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), $\mathcal{P}_{ss'}^{a} = \mathbb{P}[S_{t+1} = s' | S_t = s, A_t = a]$
  - $\mathcal{R}$ is a reward function, $\mathcal{R}_s^a = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$
  - 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

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

Environment State vs. Agent State

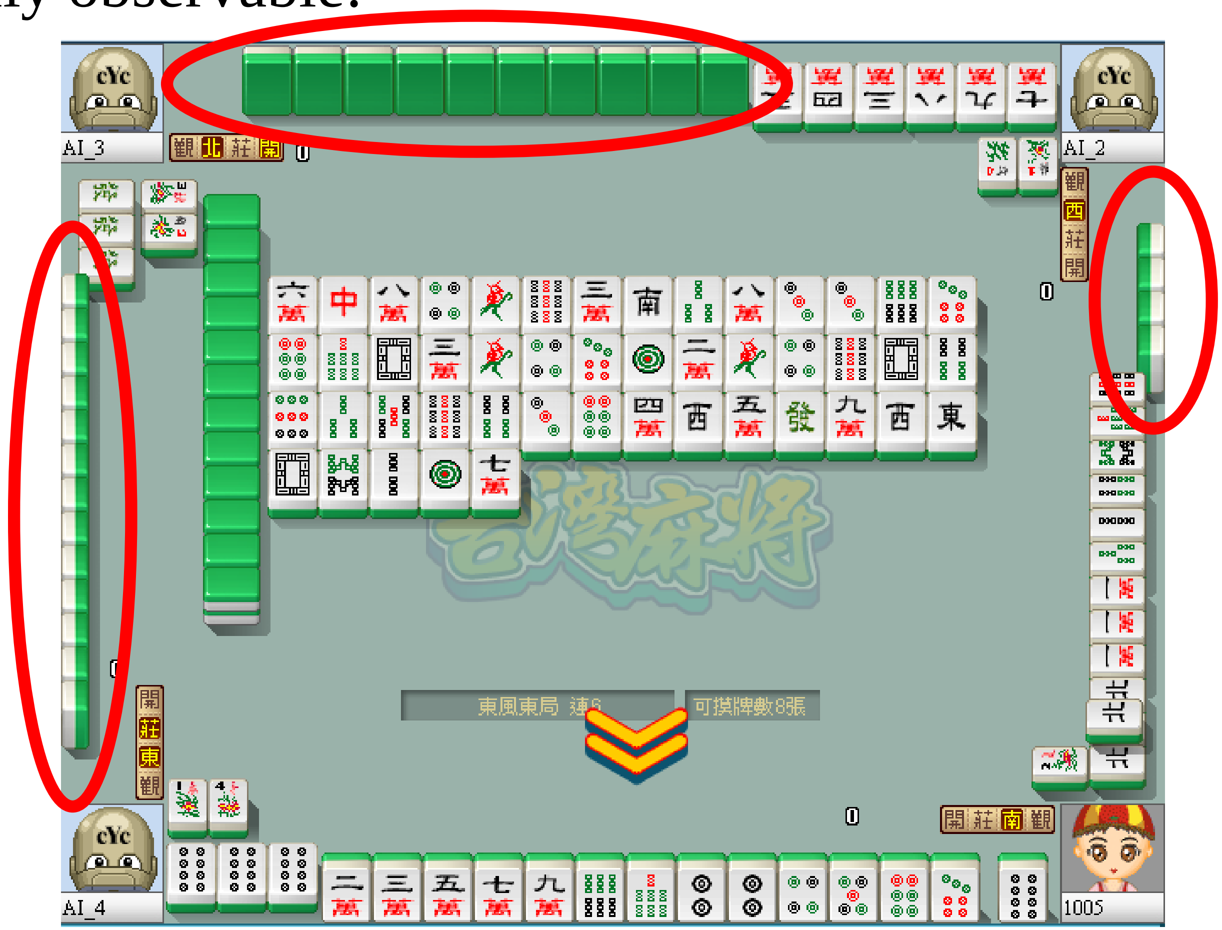
- The environment state S_t^e:
  - the environment's private representation
    - i.e. whatever data the environment uses to pick the next observation/reward
  - The environment state is not necessarily visible to the agent
    - Even if S_t^e is visible, it may contain irrelevant information
- The agent 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:

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

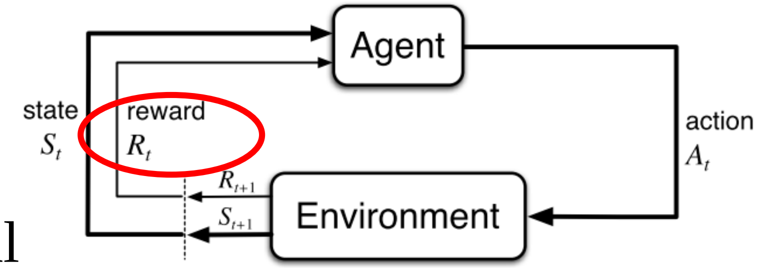
- Partially Observable: (not discussed here)
  - When $S_t^a \neq S_t^e$

Example: Mahjong

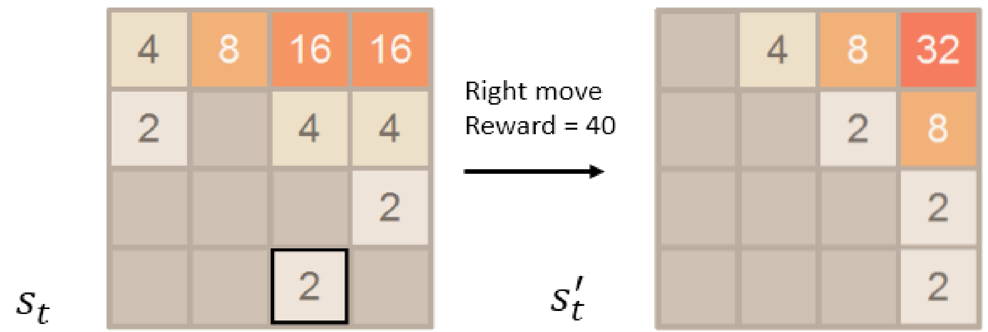
- Partially observable:



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)



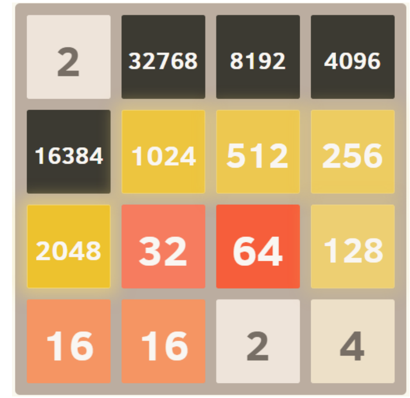
Definition (Reward Hypothesis)

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

Rewards for Previous Examples?

- In AI, it has been used to defeat human champions at games of skill.
  - Backgammon (Tesauro, 1994).
  - Connect6/2048/Threes! (Wu et al., 2015). Reach the top levels.
  - Go programs, used in the past 10 years. (Monte-Carlo Tree Search)
  - AlphaGo, using deep reinforcement learning (2016)
- In robotics, fly stunt maneuvers in robot-controlled helicopters (Abbeel et al.) and make a humanoid robot walk.
- In economics, manage an investment portfolio (Choi et al.).
- In neuroscience, model the human brain (Schultz et al.);
- In psychology, predict animal behavior (Sutton and Barto).
- In systems, control a power station
- In engineering, it has been used to allocate bandwidth to mobile phones and to manage complex power systems (Ernst et al.).

Sequential Decision Making

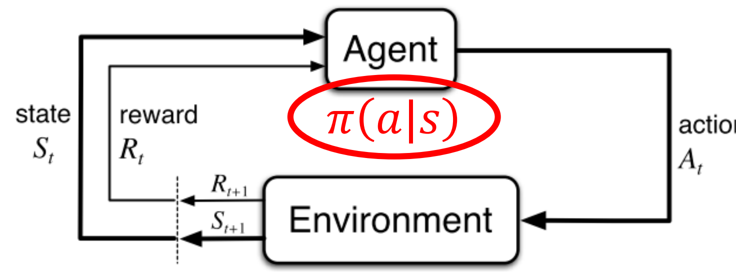


- Goal:
  - Select actions to maximize total future reward
- Notes:
  - Actions may have long term consequences
  - Reward may be delayed
  - It may be better to sacrifice immediate reward to gain more long-term reward
- Examples:
  - In 2048, establish a sequence of $(2^t, 2^{t-1}, 2^{t-2}, \ldots)$
  - 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.

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.
- Types of value functions under policy π:
  - State value function: the expected return from s.

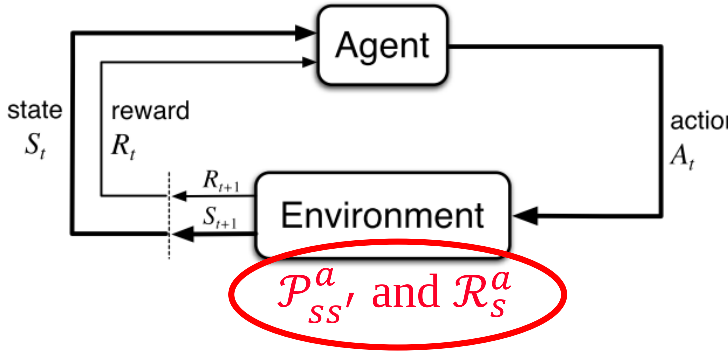
$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots \mid S_t = s] \\ &= \mathbb{E}_\pi[G_t \mid S_t = s] \end{aligned}$$

    - Return $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots$
  - 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, $\mathcal{P}_{ss'}^{a} = \mathbb{P}[S_{t+1} = s' | S_t = s, A_t = a]$
    - predicts the next state
  - $\mathcal{R}$ is a reward function, $\mathcal{R}_{s}^{a} = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$
    - 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)

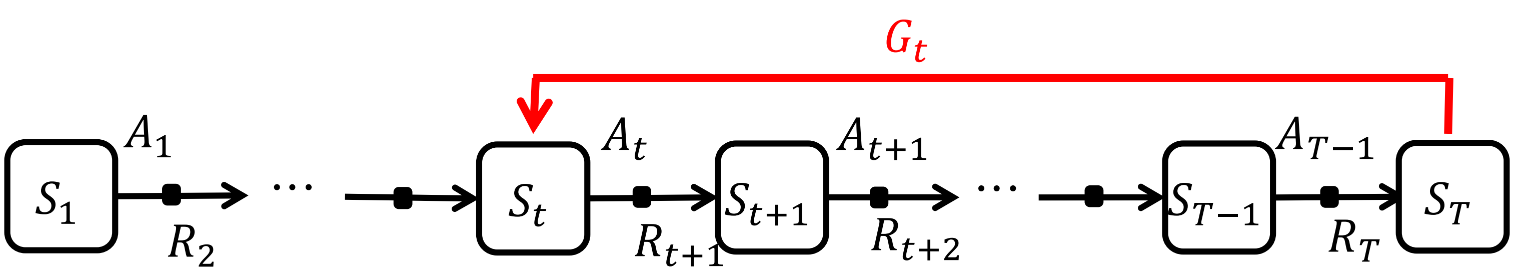
- Model Free
  - Policy and/or Value Function
  - No Model
- Model Based
  - Policy and/or Value Function
  - Model

Model-free Reinforcement Learning

- Temporal Difference (TD) Learning
  - TD methods learn directly from episodes of experience
  - TD is model-free: no knowledge of MDP transitions / rewards
  - TD learns from incomplete episodes, by bootstrapping
  - TD updates a guess towards a guess
- Monte-Carlo (MC) Learning
  - MC methods learn directly from episodes of experience
  - MC is model-free: no knowledge of MDP transitions / rewards
  - MC learns from complete episodes: no bootstrapping
  - MC uses the simplest possible idea: value = mean return
  - Caveat: can only apply MC to episodic MDPs
    - All episodes must terminate
  - Monte-Carlo Tree Search (MCTS) is a successful one based on MC learning.

Monte-Carlo Learning

- Incremental Monte-Carlo
  - Update value $V(S_t)$ toward actual return G_t
    $V(S_t) \leftarrow V(S_t) + \alpha(G_t - V(S_t))$
  - α: learning rate, or called step size.
- Unbiased, but high variance.



Temporal-Difference Learning

- Simplest temporal-difference learning algorithm: TD(0)
  - Update value $V(S_t)$ toward estimated return $R_{t+1} + \gamma V(S_{t+1})$
    $V(S_t) \leftarrow V(S_t) + \alpha(R_{t+1} + \gamma V(S_{t+1}) - V(S_t))$
  - TD target: $R_{t+1} + \gamma V(S_{t+1})$
  - TD error: $R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$
  - α: learning rate, or called step size.
- Biased, but lower variance