Lecture 11: Other RL Topics

Chonggang Wang

Outline

- Exploration & Exploitation
  - Multi-Armed Bandits
- Federated Reinforcement Learning
- Multi-Agent Reinforcement Learning
  - DeepNash - Mastering Stratego, the classic game of imperfect information

* materials are modified from David Silver's RL lecture notes

Exploration and Exploitation Dilemma

- Online decision-making involves a fundamental choice:

  Exploitation Make the best decision given current information

  Exploration Gather more information
- The best long-term strategy may involve short-term sacrifices
- Gather enough information to make the best overall decisions

Examples

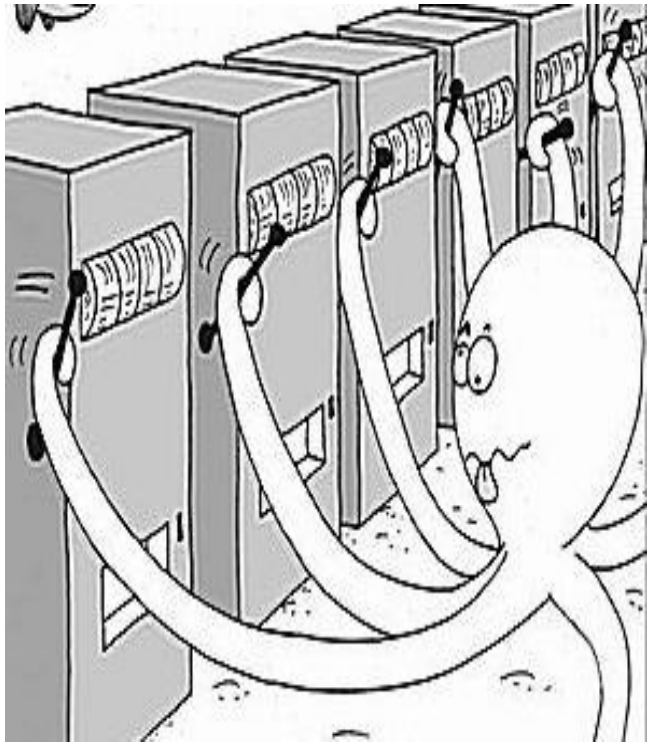
- Restaurant Selection
  - Exploitation Go to your favourite restaurant
  - Exploration Try a new restaurant
- Online Banner Advertisements
  - Exploitation Show the most successful advert
  - Exploration Show a different advert
- Oil Drilling
  - Exploitation Drill at the best known location
  - Exploration Drill at a new location
- Game Playing
  - Exploitation Play the move you believe is best
  - Exploration Play an experimental move

Principles

- Naïve Exploration
  - Add noise to greedy policy
- Optimistic Initialization
  - Assume the best until proven otherwise
- Optimistic in the face of uncertainty
  - Prefer actions with uncertain values
- Probability matching
  - Select actions according to probability they are best
- Information state search
  - Lookahead search incorporate value of information

Multi-Armed Bandits

- A multi-armed bandit is a tuple $\langle \mathcal{A}, \mathcal{R} \rangle$
- $\mathcal{A}$ is a known set of m actions (or "arms")
- $\mathcal{R}^a(r) = \mathbb{P}[r|a]$ is an unknown probability distribution over rewards
- At each step t the agent selects an action $a_t \in \mathcal{A}$
- The environment generates a reward $r_t \sim \mathcal{R}^{a_t}$
- The goal is to maximise cumulative reward $\sum_{\tau=1}^{t} r_\tau$



Regret

- The *action-value* is the mean reward for action a,

$$Q(a) = \mathbb{E}[r|a]$$

- The *optimal value* V^* is

$$V^* = Q(a^*) = \max_{a \in \mathcal{A}} Q(a)$$

- The *regret* is the opportunity loss for one step

$$l_t = \mathbb{E}[V^* - Q(a_t)]$$

- The *total regret* is the total opportunity loss

$$L_t = \mathbb{E}\left[\sum_{\tau=1}^{t} V^* - Q(a_\tau)\right]$$

- Maximise cumulative reward $\equiv$ minimise total regret

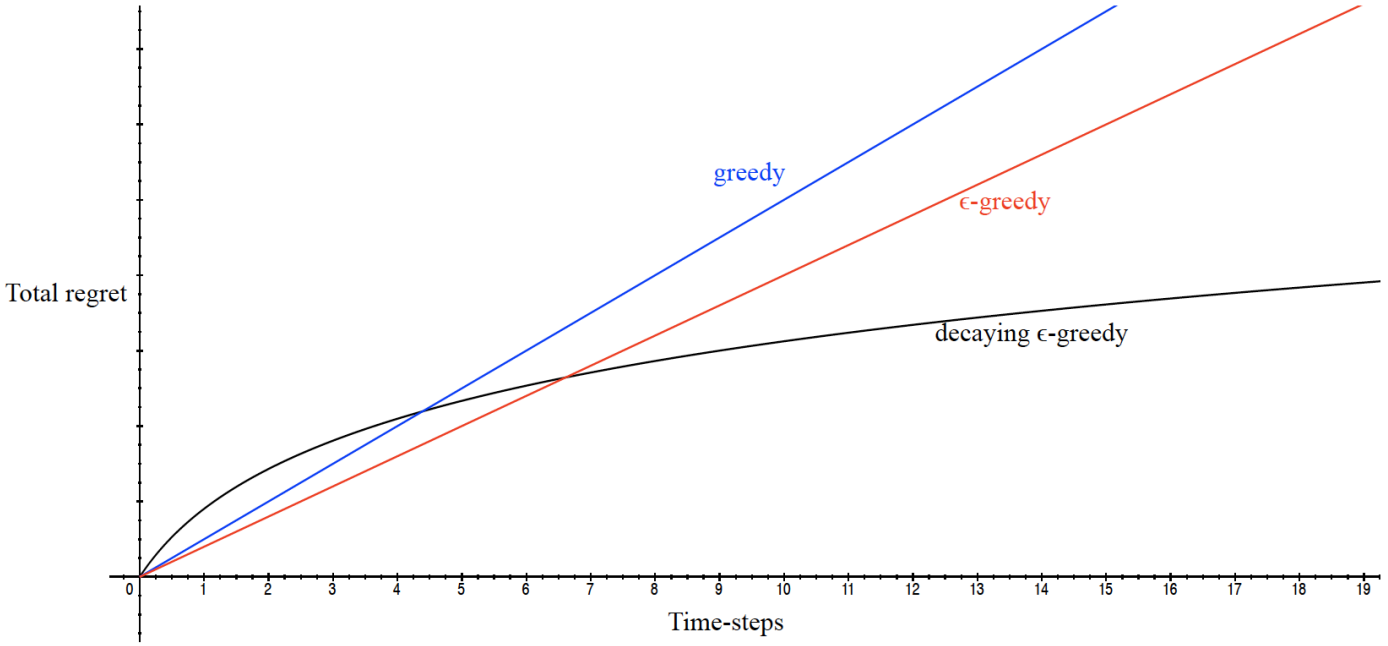
Counting Regret

- The *count* $N_t(a)$ is expected number of selections for action a
- The *gap* Δ_a is the difference in value between action a and optimal action a^*, $\Delta_a = V^* - Q(a)$
- Regret is a function of gaps and the counts

$$\begin{aligned} L_t &= \mathbb{E}\left[\sum_{\tau=1}^{t} V^* - Q(a_\tau)\right] \\ &= \sum_{a \in \mathcal{A}} \mathbb{E}\left[N_t(a)\right] \left(V^* - Q(a)\right) \\ &= \sum_{a \in \mathcal{A}} \mathbb{E}\left[N_t(a)\right] \Delta_a \end{aligned}$$

- A good algorithm ensures small counts for large gaps
- Problem: gaps are not known!

Linear or Sublinear Regret



- If an algorithm forever explores it will have linear total regret
- If an algorithm never explores it will have linear total regret
- Is it possible to achieve sublinear total regret?

Outline

- Introduction
- Multi-Armed Bandits
  - Naïve methods
  - Optimistic in the fact of uncertainty
  - Information space
- MDPs

Greedy Algorithm

- We consider algorithms that estimate $\hat{Q}_t(a) \approx Q(a)$
- Estimate the value of each action by Monte-Carlo evaluation

$$\hat{Q}_t(a) = \frac{1}{N_t(a)} \sum_{t=1}^{T} r_t \mathbf{1}(a_t = a)$$

Sample average

- The *greedy* algorithm selects action with highest value

$$a_t^* = \underset{a \in \mathcal{A}}{\text{argmax}}\ \hat{Q}_t(a)$$

- Greedy can lock onto a suboptimal action forever
- $\Rightarrow$ Greedy has linear total regret

ϵ-Greedy Algorithm

- The ϵ-*greedy* algorithm continues to explore forever
  - With probability $1 - \epsilon$ select $a = \underset{a \in \mathcal{A}}{\text{argmax}}\ \hat{Q}(a)$
  - With probability ϵ select a random action
- Constant ϵ ensures minimum regret

$$l_t \geq \frac{\epsilon}{\mathcal{A}} \sum_{a \in \mathcal{A}} \Delta_a$$

- $\Rightarrow$ ϵ-greedy has linear total regret

Decaying ϵ_t-Greedy Algorithm

- Pick a decay schedule for $\epsilon_1, \epsilon_2, \ldots$
- Consider the following schedule

$$c > 0$$

$$d = \min_{a|\Delta_a > 0} \Delta_i$$

Smallest non-zero gap

$$\epsilon_t = \min \left\{ 1, \frac{c|\mathcal{A}|}{d^2 t} \right\}$$

- Decaying ϵ_t-greedy has *logarithmic* asymptotic total regret!
- Unfortunately, schedule requires advance knowledge of gaps
- Goal: find an algorithm with sublinear regret for any multi-armed bandit (without knowledge of $\mathcal{R}$)

Optimistic Initialization

- Simple and practical idea: initialise $Q(a)$ to high value
- Update action value by incremental Monte-Carlo evaluation
- Starting with $N(a) > 0$

$$\hat{Q}_t(a_t) = \hat{Q}_{t-1} + \frac{1}{N_t(a_t)}(r_t - \hat{Q}_{t-1})$$

- Encourages systematic exploration early on
- But can still lock onto suboptimal action
- $\Rightarrow$ greedy $+$ optimistic initialisation has linear total regret
- $\Rightarrow$ ϵ-greedy $+$ optimistic initialisation has linear total regret

Lower Bound on Regret

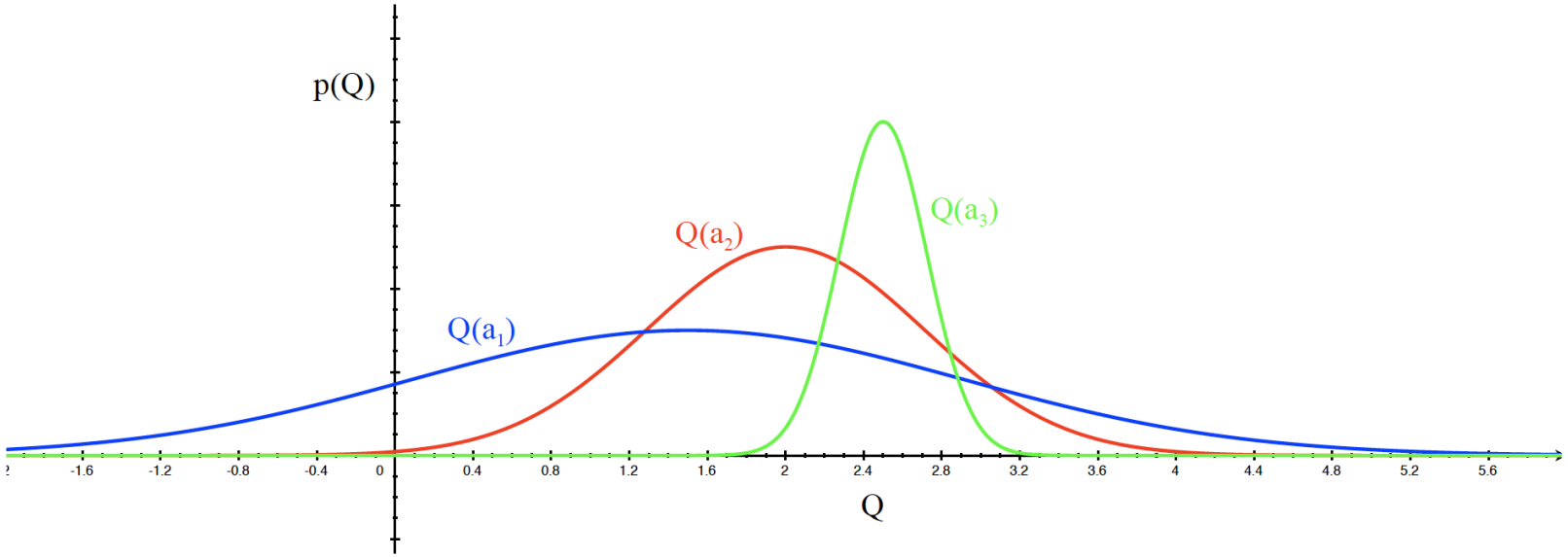
- The performance of any algorithm is determined by similarity between optimal arm and other arms
- Hard problems have similar-looking arms with different means
- This is described formally by the gap Δ_a and the similarity in distributions $KL(\mathcal{R}^a||\mathcal{R}^a*)$

Theorem (Lai and Robbins)

Asymptotic total regret is at least logarithmic in number of steps

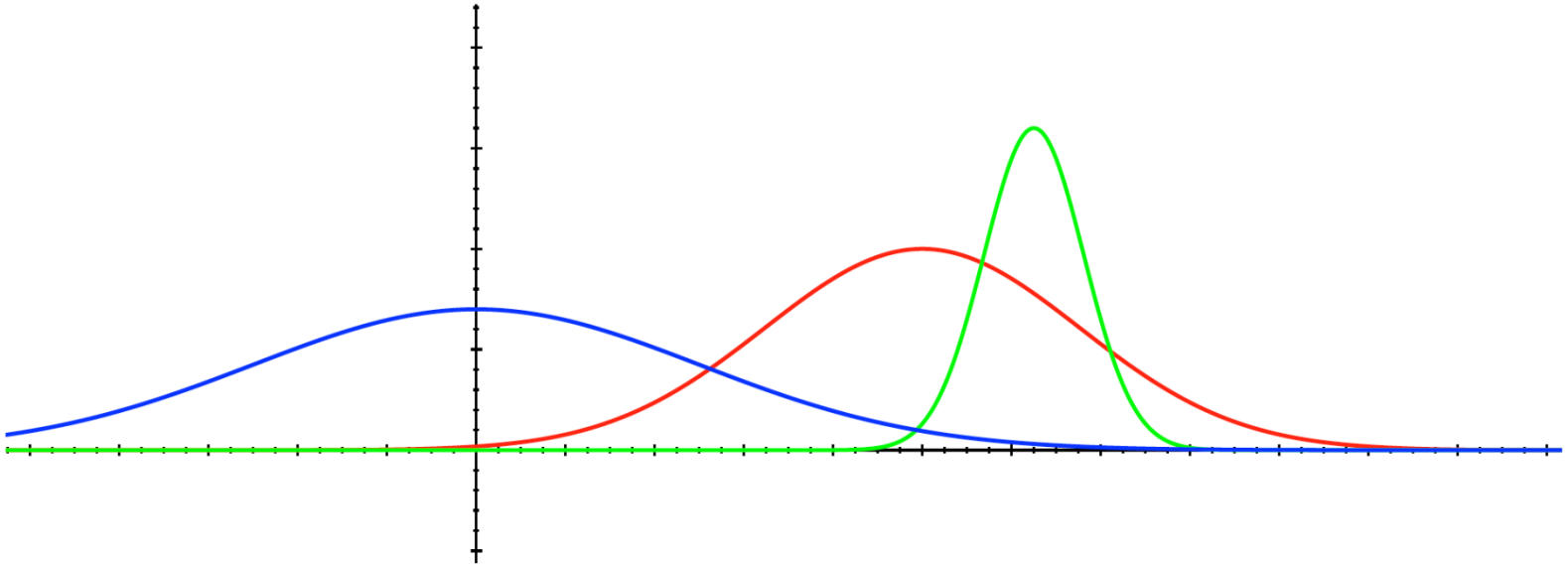
$$\lim_{t\to\infty} L_t \geq \log t \sum_{a|\Delta_a>0} \frac{\Delta_a}{KL(\mathcal{R}^a||\mathcal{R}^{a^*})}$$

Optimistic in the Face of Uncertainty



- Which action should we pick?
- The more uncertain we are about an action-value
- The more important it is to explore that action
- It could turn out to be the best action

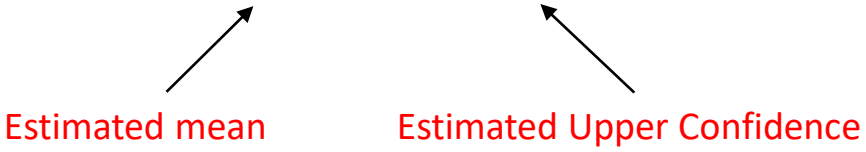
Optimistic in the Face of Uncertainty



- After picking blue action
- We are less uncertain about the value
- And more likely to pick another action
- Until we home in on best action

Upper Confidence Bounds

- Estimate an upper confidence $\hat{U}_t(a)$ for each action value
- Such that $Q(a) \leq \hat{Q}_t(a) + \hat{U}_t(a)$ with high probability



- This depends on the number of times $N(a)$ has been selected
  - Small $N_t(a) \Rightarrow$ large $\hat{U}_t(a)$ (estimated value is uncertain)
  - Large $N_t(a) \Rightarrow$ small $\hat{U}_t(a)$ (estimated value is accurate)
- Select action maximising Upper Confidence Bound (UCB)

$$a_t = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\ \hat{Q}_t(a) + \hat{U}_t(a)$$

Hoeffding's Inequality

Theorem (Hoeffding's Inequality)

Let $X_1, ..., X_t$ be i.i.d. random variables in [0,1], and let $\overline{X}_t = \frac{1}{\tau}\sum_{\tau=1}^{t} X_\tau$ be the sample mean. Then

$$\mathbb{P}\left[\mathbb{E}\left[X\right] > \overline{X}_t + u\right] \leq e^{-2tu^2}$$

- We will apply Hoeffding's Inequality to rewards of the bandit conditioned on selecting action a

$$\mathbb{P}\left[Q(a) > \hat{Q}_t(a) + U_t(a)\right] \leq e^{-2N_t(a)U_t(a)^2}$$

Calculating Upper Confidence Bounds

- Pick a probability p that true value exceeds UCB
- Now solve for $U_t(a)$

$$e^{-2N_t(a)U_t(a)^2} = p$$

$$U_t(a) = \sqrt{\frac{-\log p}{2N_t(a)}}$$

- Reduce p as we observe more rewards, e.g. $p = t^{-4}$
- Ensures we select optimal action as $t \to \infty$

$$U_t(a) = \sqrt{\frac{2 \log t}{N_t(a)}}$$

UCB1 Algorithm

- This leads to the UCB1 algorithm

$$a_t = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\, Q(a) + \sqrt{\frac{2 \log t}{N_t(a)}}$$

Theorem

The UCB algorithm achieves logarithmic asymptotic total regret

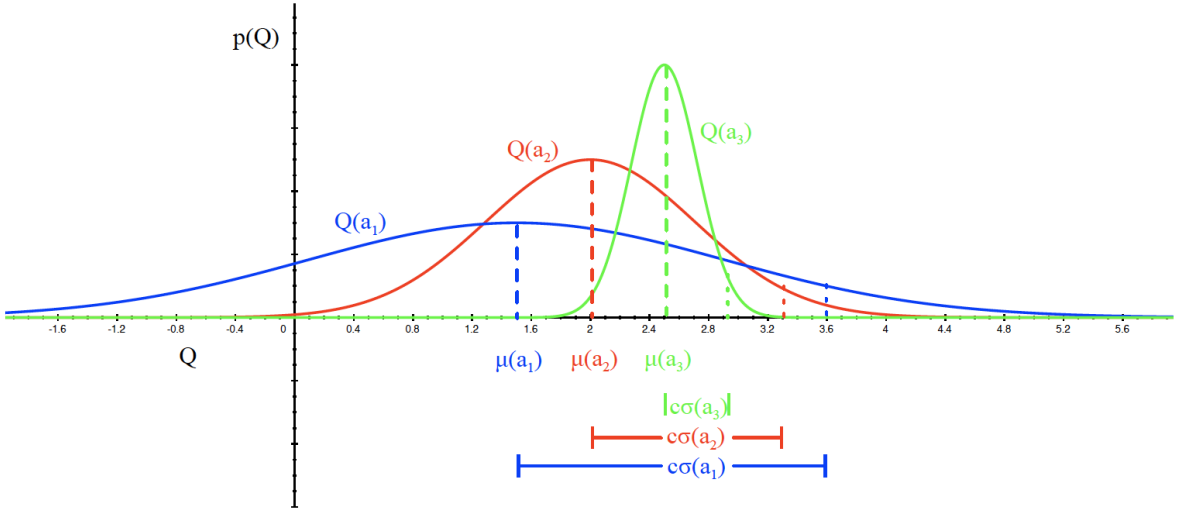
$$\lim_{t \to \infty} L_t \leq 8 \log t \sum_{a | \Delta_a > 0} \Delta_a$$

Bayesian Bandits

- So far we have made no assumptions about the reward distribution $\mathcal{R}$
  - Except bounds on rewards
- Bayesian bandits exploit prior knowledge of rewards, $p[\mathcal{R}]$
- They compute posterior distribution of rewards $p[\mathcal{R} \mid h_t]$
  - where $h_t = a_1, r_1, ..., a_{t-1}, r_{t-1}$ is the history
- Use posterior to guide exploration
  - Upper confidence bounds (Bayesian UCB)
  - Probability matching (Thompson sampling)
- Better performance if prior knowledge is accurate

Bayesian UCB Example: Independent Gaussians

- Assume reward distribution is Gaussian, $\mathcal{R}_a(r) = \mathcal{N}(r; \mu_a, \sigma_a^2)$



- Compute Gaussian posterior over μ_a and σ_a^2 (by Bayes law)

$$p\left[\mu_a, \sigma_a^2 \mid h_t\right] \propto p\left[\mu_a, \sigma_a^2\right] \prod_{t \mid a_t = a} \mathcal{N}(r_t; \mu_a, \sigma_a^2)$$

- Pick action that maximises standard deviation of $Q(a)$

$$a_t = \underset{a \in \mathcal{A}}{\text{argmax}}\ \mu_a + c\sigma_a / \sqrt{N(a)}$$

Probability Matching

- Probability matching selects action a according to probability that a is the optimal action

$$\pi(a \mid h_t) = \mathbb{P}\left[Q(a) > Q(a'), \forall a' \neq a \mid h_t\right]$$

- Probability matching is optimistic in the face of uncertainty
  - Uncertain actions have higher probability of being max
- Can be difficult to compute analytically from posterior

Thompson Sampling

- Thompson sampling implements probability matching

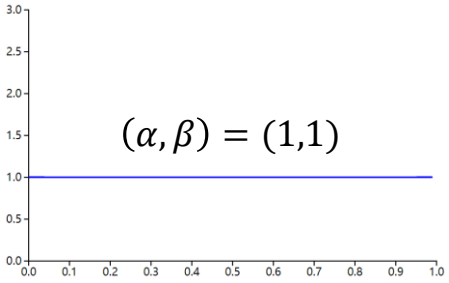
$$\pi(a \mid h_t) = \mathbb{P}\left[Q(a) > Q(a'), \forall a' \neq a \mid h_t\right]$$
$$= \mathbb{E}_{\mathcal{R}|h_t}\left[\mathbf{1}(a = \underset{a \in \mathcal{A}}{\text{argmax}}\ Q(a))\right]$$

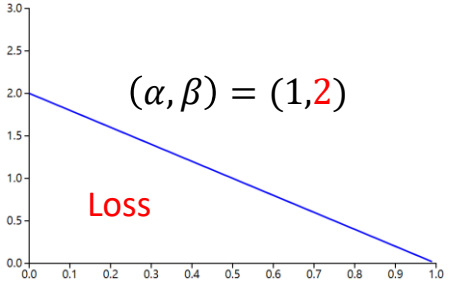
- Use Bayes law to compute posterior distribution $p\left[\mathcal{R} \mid h_t\right]$
- Sample a reward distribution $\mathcal{R}$ from posterior
- Compute action-value function $Q(a) = \mathbb{E}\left[\mathcal{R}_a\right]$
- Select action maximising value on sample, $a_t = \underset{a \in \mathcal{A}}{\text{argmax}}\ Q(a)$
- Thompson sampling achieves Lai and Robbins lower bound!

Beta Distribution

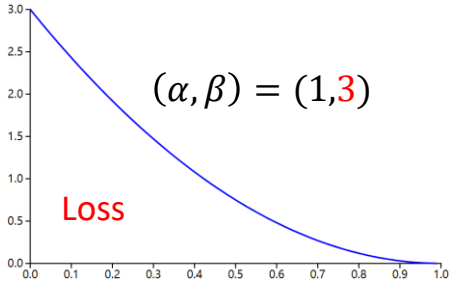
$\theta \sim \text{Beta}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ $(0 \leq \theta \leq 1)$: The probability of winning a game

$$\boldsymbol{f(\theta) = C * \theta^{\alpha-1}(1-\theta)^{\beta-1}}$$

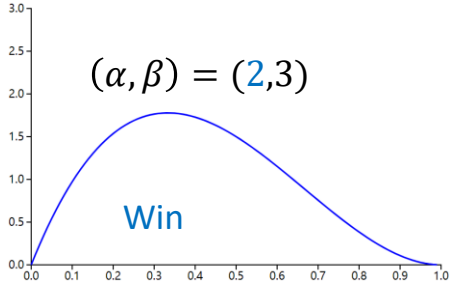




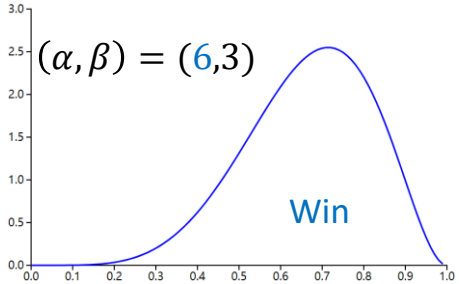
Game #1: Loss



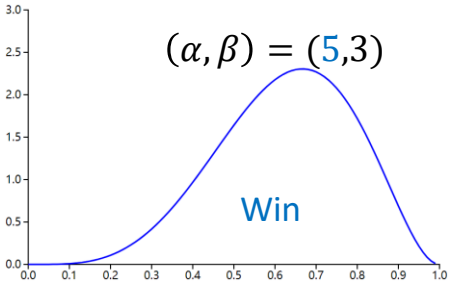
Game #2: Loss



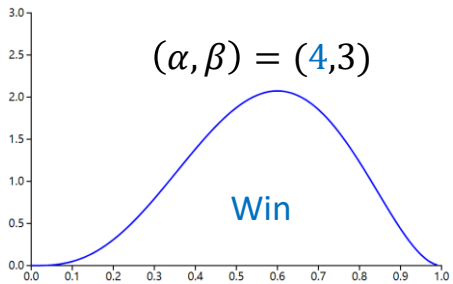
Game #3: Win



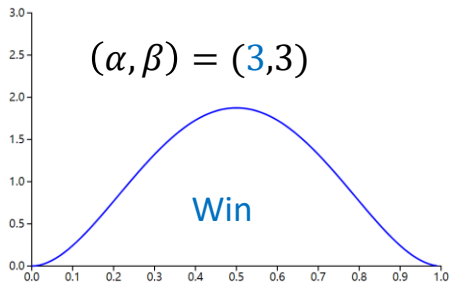
Game #7: Win



Game #6: Win



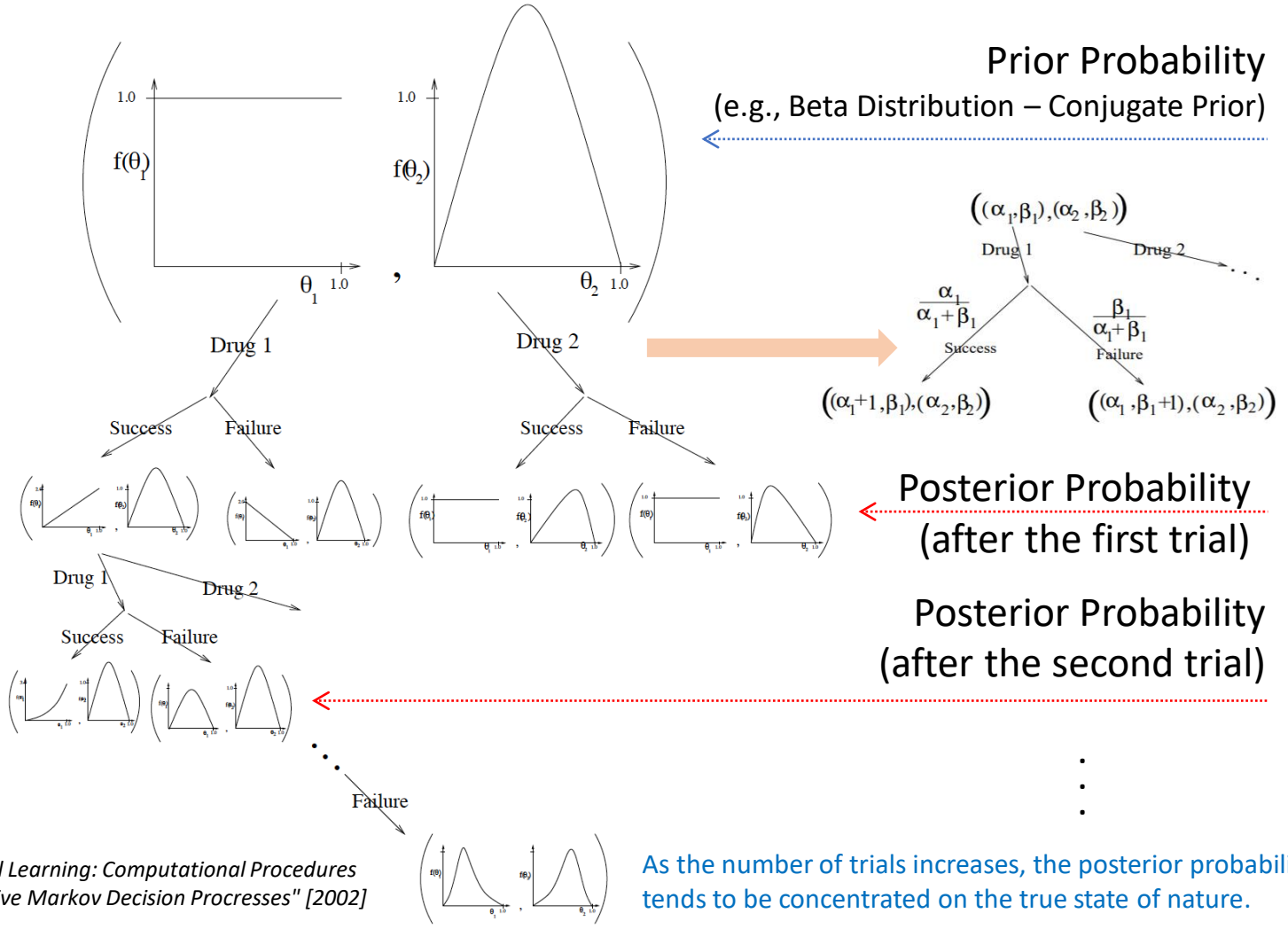
Game #5: Win



Game #4: Win

As the number of trials increases, the posterior probability tends to be concentrated on the true state of nature.

Bayes-Adaptive Bernoulli Bandits



Source: "Optimal Learning: Computational Procedures for Bayes-Adaptive Markov Decision Procresses" [2002]

As the number of trials increases, the posterior probability tends to be concentrated on the true state of nature.

Exploration/Exploitation Principles to MDPs

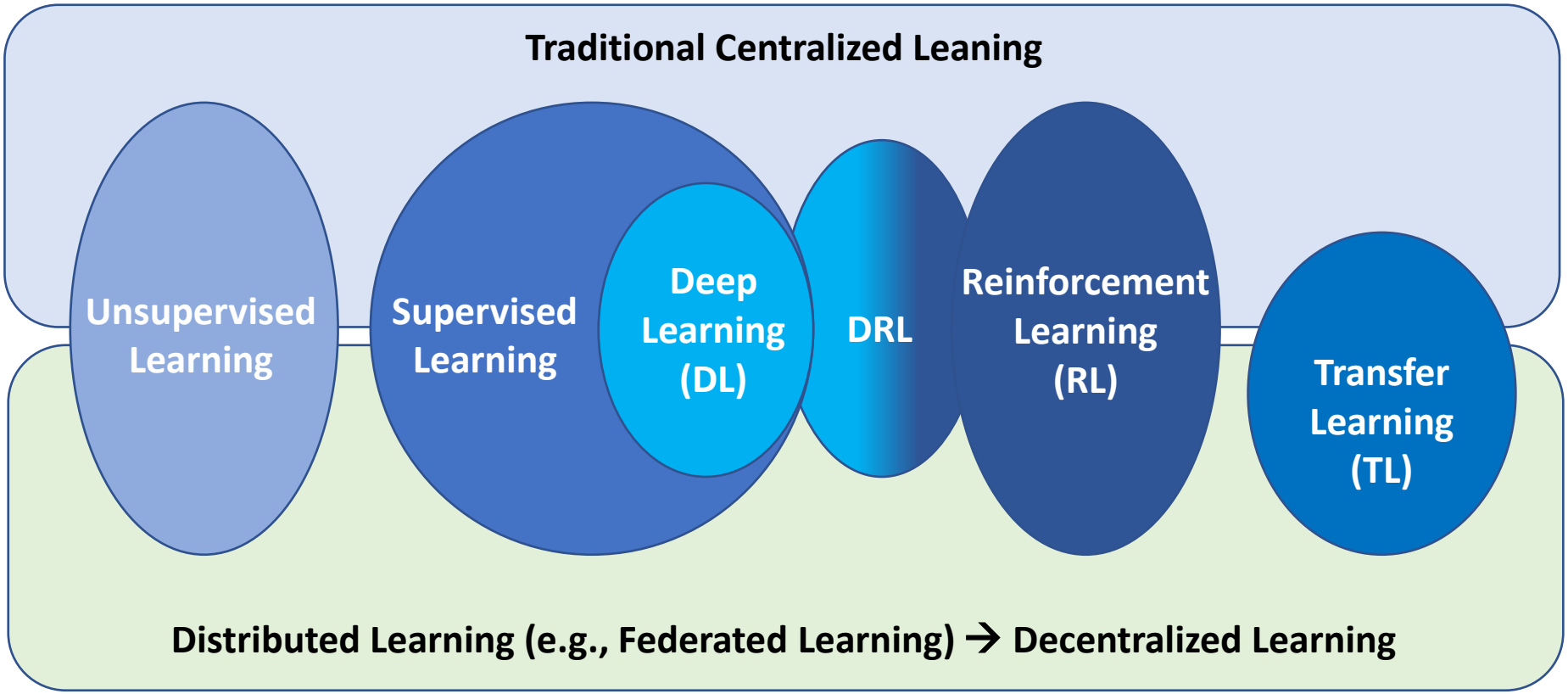
The same principles for exploration/exploitation apply to MDPs.

- Naïve Exploration
  - Add noise to greedy policy
- Optimistic Initialization
  - Assume the best until proven otherwise
- Optimistic in the face of uncertainty
  - Prefer actions with uncertain values
- Probability matching
  - Select actions according to probability they are best
- Information state search
  - Lookahead search incorporate value of information

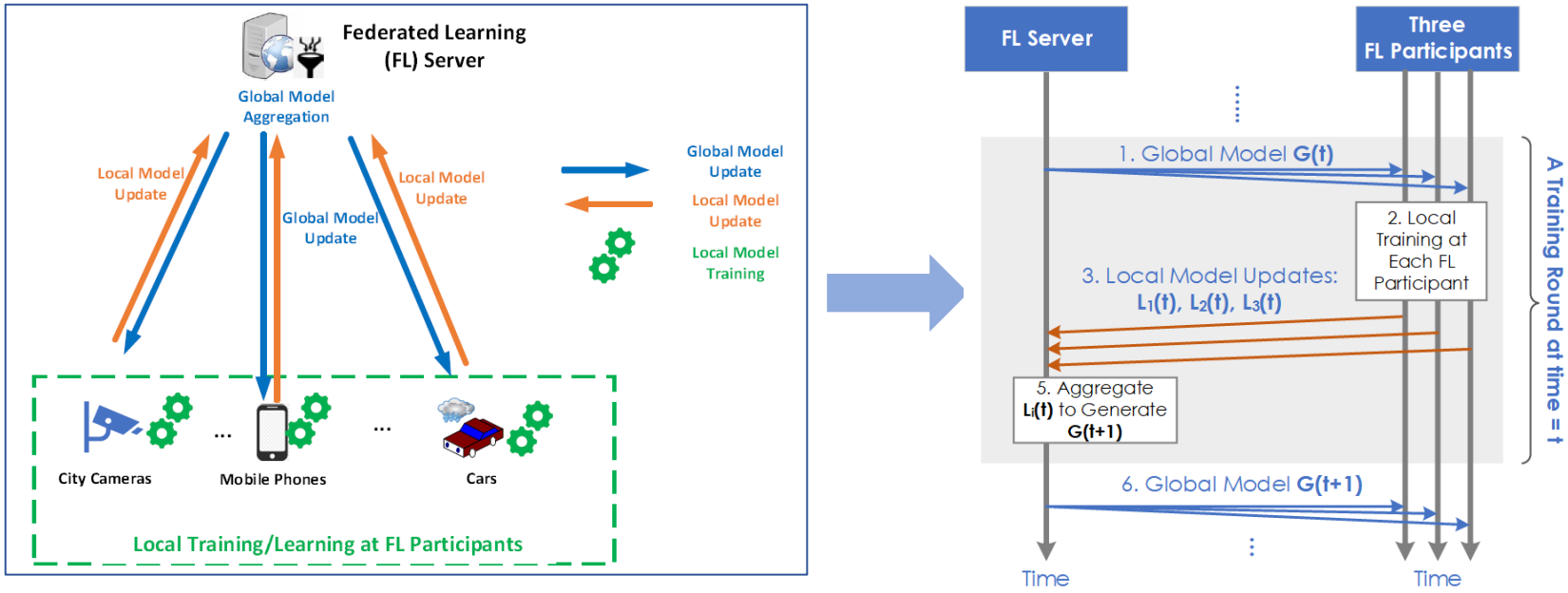
Outline

- Exploration & Exploitation
  - Multi-Armed Bandits
- Federated Reinforcement Learning
- Multi-Agent Reinforcement Learning
  - DeepNash - Mastering Stratego, the classic game of imperfect information

Artificial Intelligence (AI)

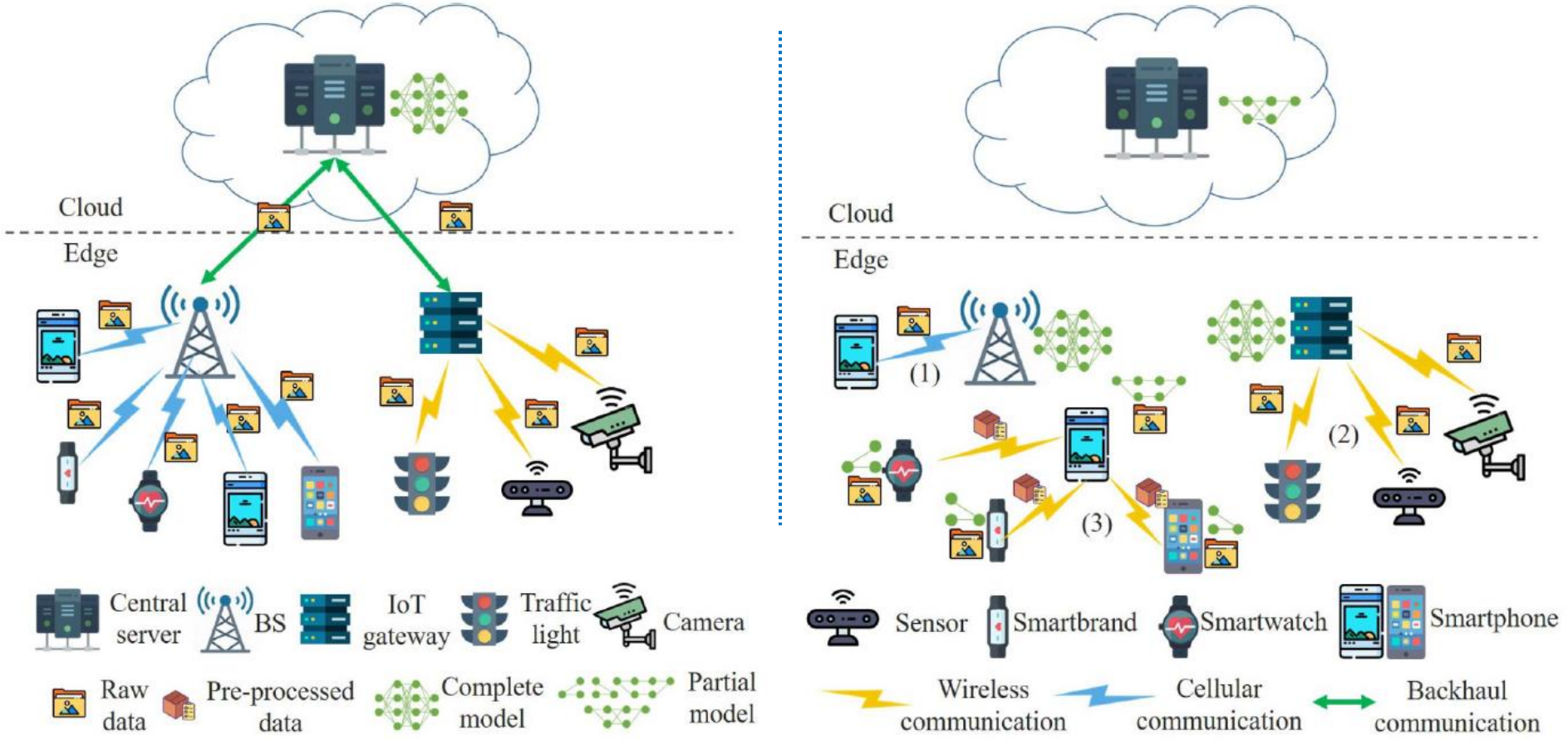


Federated Learning Workflow



- **Advantages:** No need for centralized training data collection and centralized, data privacy, edge intelligence, learning quality
- **Disadvantages:** New threats and attacks to distributed participants (e.g., **backdoor attack**), communications overhead, varying data quality, selection of FL participant

Edge/Pervasive AI



Centralized Intelligence **Edge Intelligence**

Image Source: "Edge Intelligence: Empowering Intelligence to the Edge of Network (2021)"

Federated Learning Workflow – Fully Distributed

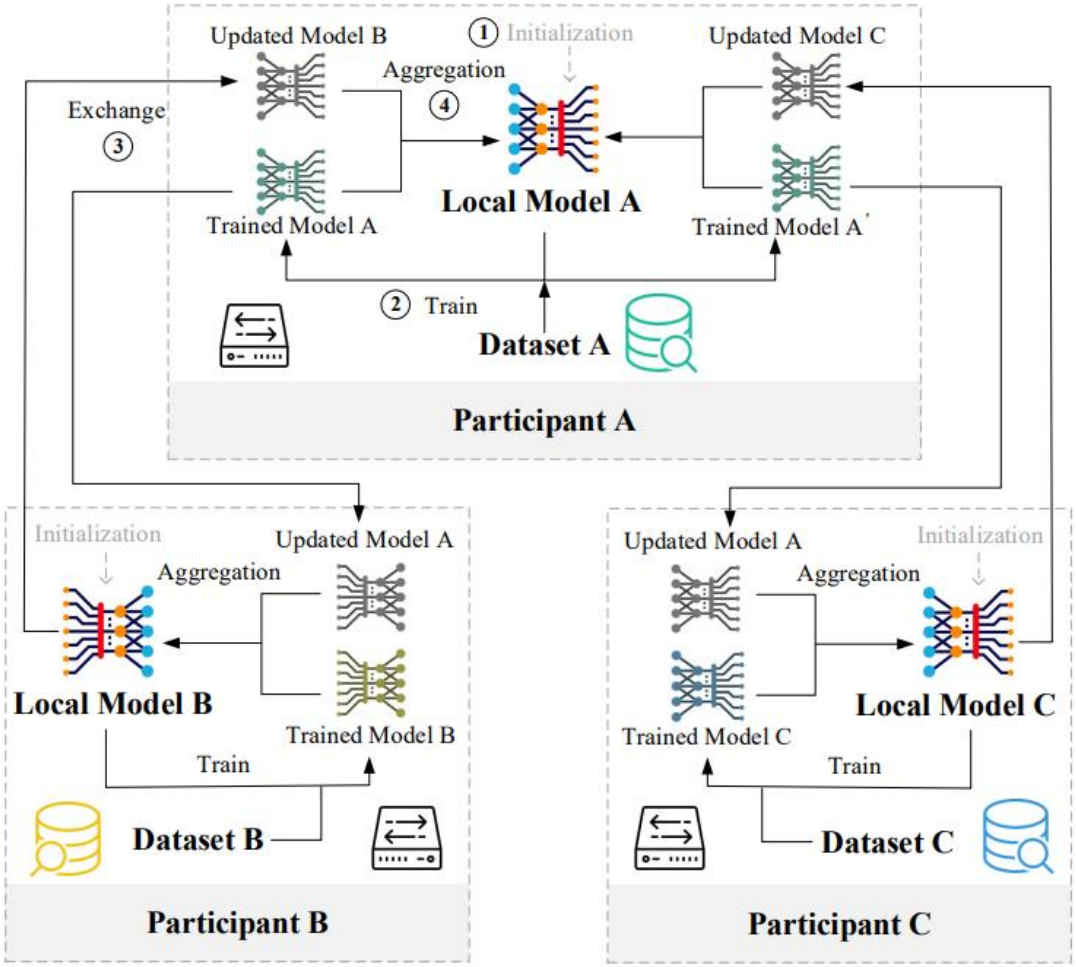


Image Source: "Federated Reinforcement Learning: Techniques, Applications, and Open Challenges" (2021)

Federated Reinforcement Learning

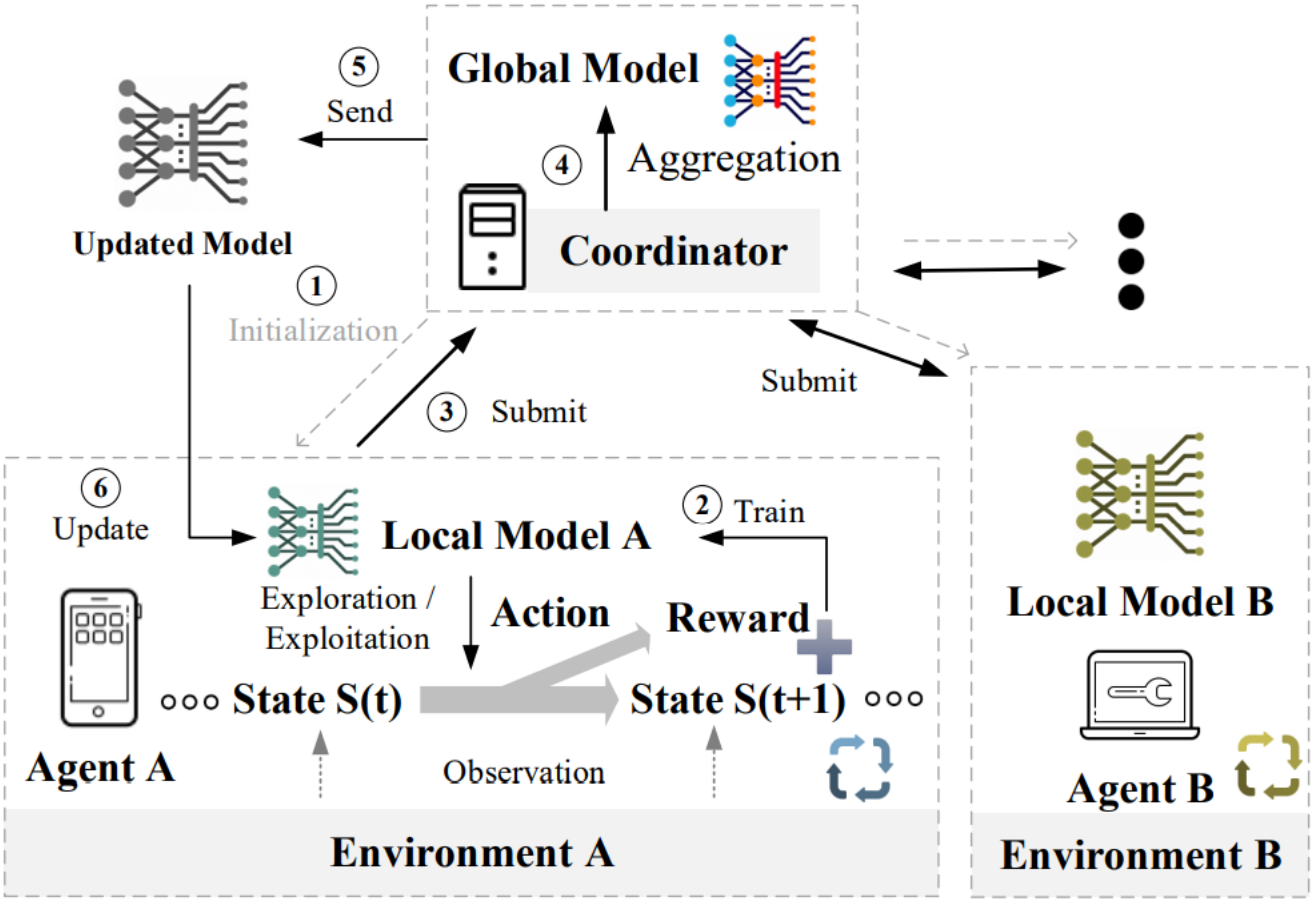


Image Source: "Federated Reinforcement Learning: Techniques, Applications, and Open Challenges" (2021)

Federated Reinforcement Learning

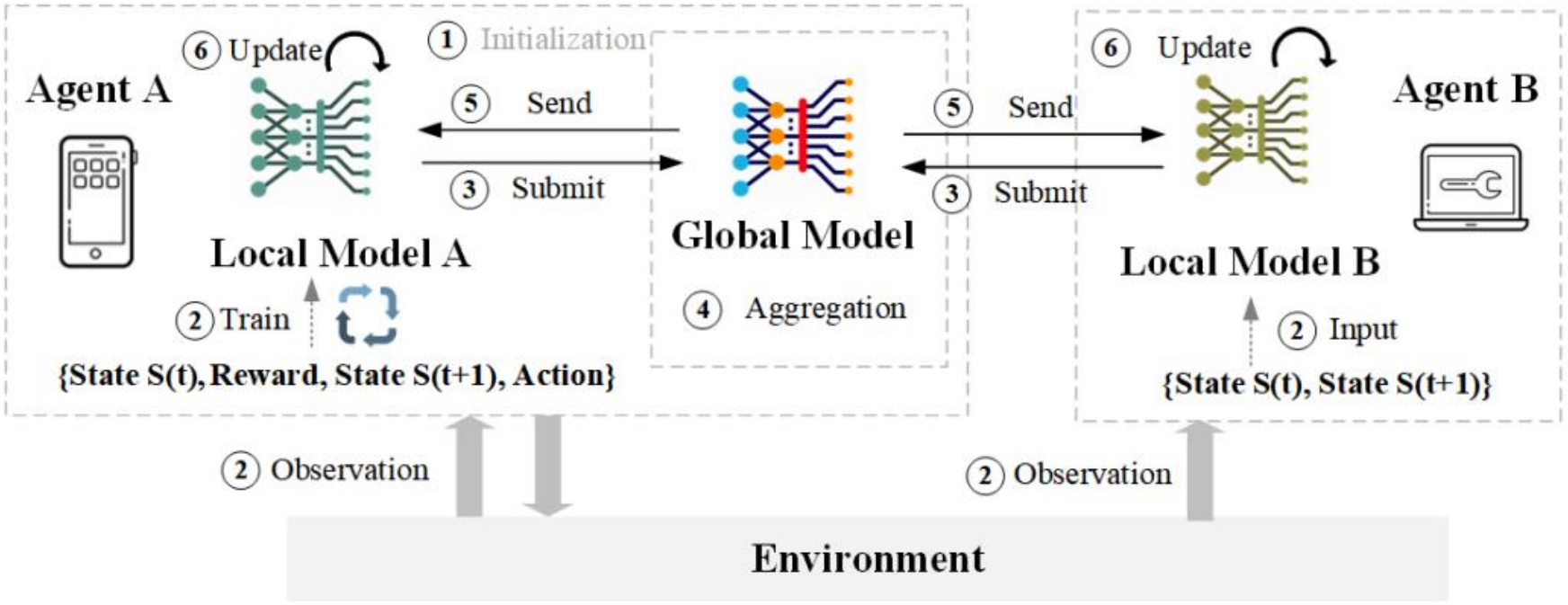


Image Source: "Federated Reinforcement Learning: Techniques, Applications, and Open Challenges" (2021)

Outline

- Exploration & Exploitation
  - Multi-Armed Bandits
- Federated Reinforcement Learning
- Multi-Agent Reinforcement Learning
  - DeepNash - Mastering Stratego, the classic game of imperfect information

Multi-Agent Reinforcement Learning (MARL)

- MARL Categories
  - Cooperative – Agents have the same goal
  - Competitive – Two agents compete with each other
  - Hybrid – Some agents have the same goal and compete with other agents
- MARL Architecture

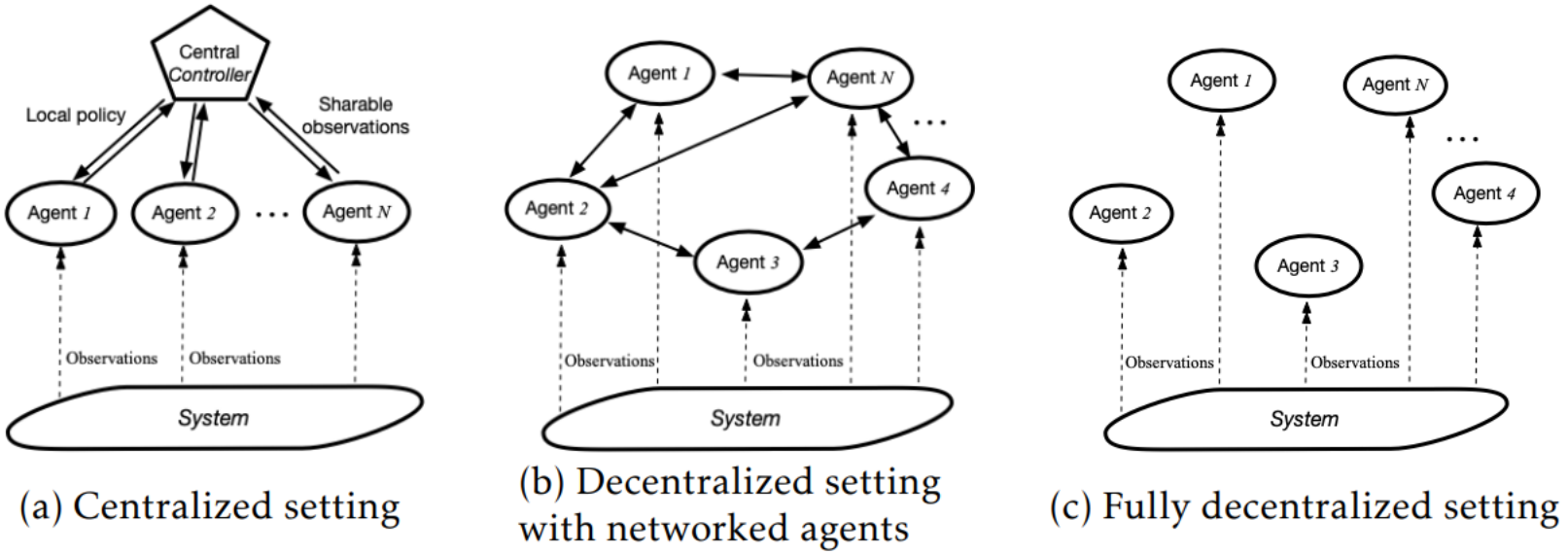


Image Source: "Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms" (2021)

DeepNash – Mastering Stratego

- Published in Science on Dec 1, 2022 (https://www.science.org/doi/10.1126/science.add4679)
- Marshal (10), General (9), Colonels (8), Majors (7), Captains (6), Lieutenants (5), Sergeants (4), **Miners (3), Scouts (2), Spy (s), Bombs (B), Flags (F)**

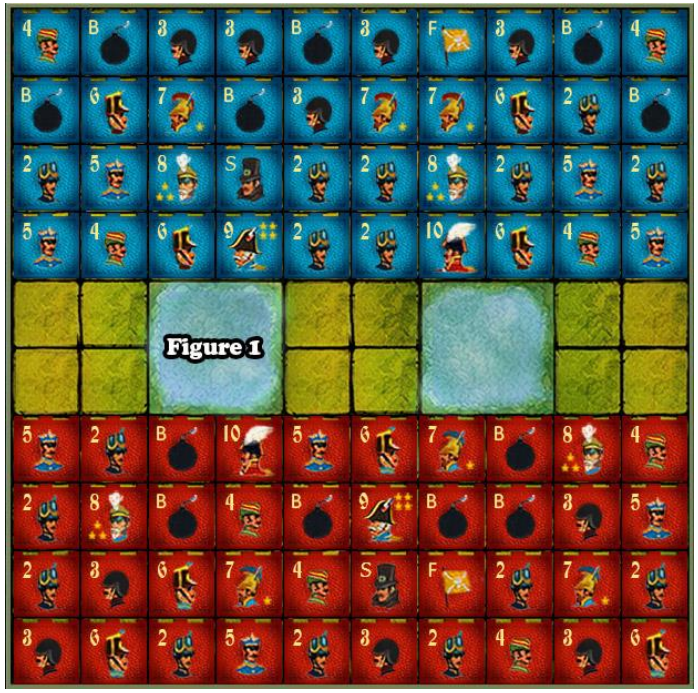


Image Source: https://www.ultraboardgames.com/stratego/game-rules.php

Chinese Game JunQi - Similar to Stratego

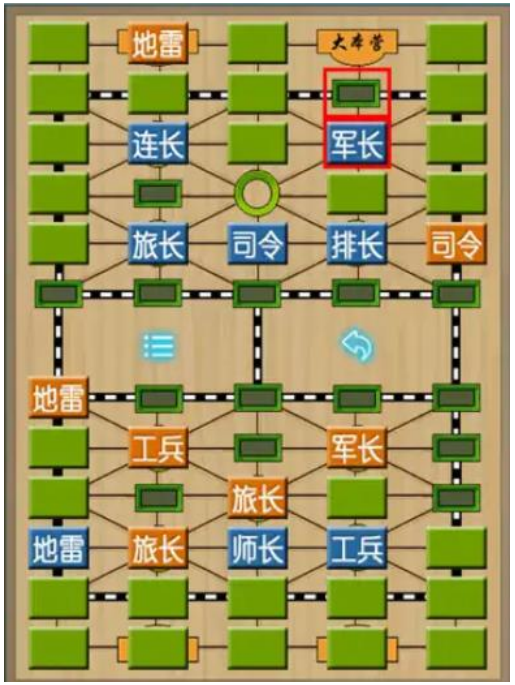


Image Source: AppStore

DeepNash – Mastering Stratego

- Published in Science on Dec 1, 2022 (https://www.science.org/doi/10.1126/science.add4679)
- Abstract – "We introduce DeepNash, an autonomous agent that plays the imperfect information game Stratego at a human expert level. Stratego is one of the few iconic board games that artificial intelligence (AI) has not yet mastered. It is a game characterized by a twin challenge: It requires long-term strategic thinking as in chess, but it also requires dealing with imperfect information as in poker. The technique underpinning DeepNash uses a game-theoretic, model-free deep reinforcement learning method, without search, that learns to master Stratego through self-play from scratch. DeepNash beat existing state-of-the-art AI methods in Stratego and achieved a year-to-date (2022) and all-time top-three ranking on the Gravon games platform, competing with human expert players."
- **Regularised Nash Dynamics (R-NaD)**
- "The present work not only adds to the growing list of games that AI systems can play as well or even better than humans but may also facilitate further applications of reinforcement learning methods in real-world, large-scale multiagent problems that are characterized by imperfect information and thus are currently unsolvable" – From *Science*
- "The value of mastering Stratego goes beyond gaming. In pursuit of our mission of solving intelligence to advance science and benefit humanity, we need to build advanced AI systems that can operate in complex, real-world situations with limited information of other agents and people. Our paper shows how DeepNash can be applied in situations of uncertainty and successfully balance outcomes to help solve complex problems." – From *DeepMind*

Conclusion

- Have covered several principles for exploration and exploitation
  - Naïve methods
  - Optimistic Initialization
  - Upper confidence bounds
  - Probability matching
  - Each principle was developed in bandit setting, but can also apply to MDP setting
- Federated Reinforcement Learning
- Multi-Agent Reinforcement Learning
  - DeepNash for Stratego

References

- Book Chapter 2, “Reinforcement Learning: An Introduction” (2nd Edition) [2018]
- Dissertation Chapters 1 & 2: "Optimal Learning: Computational Procedures for Bayes-Adaptive Markov Decision Procresses" [2002]
- [UCB] “Finite-time Analysis of the Multiarmed Bandit Problem” [2002]
- “Asymptotically Efficient Adaptive Allocation Rules” [1985]
- “Federated Reinforcement Learning: Techniques, Applications, and Open Challenges” [2021]
- “Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms” [2021]
- “Mastering the Game of Stratego with Model-Free Multiagent Reinforcement Learning” [2022]

AI learns how to walk

Congratulations! You have successfully stepped into the AI world. I wish you all an enjoyable and wonderful journey in the future.

Thank You !