Reminders

- HW2 due 2/18
- PRA3 due 2/22
- Course project progress report 1 due 2/27
- Come to OH for course project discussion!

Artificial Intelligence Methods for Social Good

Lecture 10

Basics of Multi-Armed Bandit

17-537 (9-unit) and 17-737 (12-unit)
Instructor: Fei Fang
feifang@cmu.edu

Outline

- Multi-Armed Bandit (MAB) Problems
- Bandit Data-Driven Optimization in Food Rescue
- Markov Decision Process
- Restless Multi-Armed Bandits

Learning Objectives

- Understand the concept of
  - Multi-Armed Bandit (MAB)
  - Zero-regret strategy
  - Upper Confidence Bound (UCB)
  - Probably approximately correct (PAC)
  - Markov Decision Process
  - Restless Multi-Armed Bandits

A Simple Sequential Decision Making Problem

- Multi-Armed Bandit (MAB)
  - K arms (slot machines)
  - Each arm k is associated with a reward distribution R_k (pdf $p_k(r)$), with expected reward μ_k

$$\mu_k = \mathbb{E}_{r\sim R_k}[r] = \int_r rp_k(r)dr$$

  - Gambler does not know R_k, μ_k
  - In each round $t \in \{1 \dots T\}$, gambler chooses one arm I_t, and observe a reward r_t drawn from the distribution R_{I_t}

Multi-Armed Bandit (MAB)

- A special case: Binary MAB
  - Reward is either 0 or 1
  - $\Pr(r = 1) = p_k, \Pr(r = 0) = 1 - p_k, \mu_k = p_k$

	Arm 1	Arm 2	Arm 3
μ_k	0.2	0.3	0.7

Multi-Armed Bandit (MAB)

- Let $\mu^* = \max_k \mu_k$
- Define regret $\rho_T = T\mu^* - \sum_{t=1}^{T} r_t$
- A typical task in MAB: find zero-regret strategy
  - $\lim_{T\to\infty} \frac{\rho_T}{T} = 0$

	Arm 1	Arm 2	Arm 3
μ_k	0.2	0.3	0.7

If always choose arm 1, what is the expected regret?

$\mathbb{E}[\rho_T] =$

Multi-Armed Bandit (MAB)

- Let $\mu^* = \max_k \mu_k$
- Define regret $\rho_T = T\mu^* - \sum_{t=1}^{T} r_t$
- A typical task in MAB: find zero-regret strategy
  - $\lim_{T\to\infty} \frac{\rho_T}{T} = 0$

	Arm 1	Arm 2	Arm 3
μ_k	0.2	0.3	0.7

If always choose arm 1, what is the expected regret?

$$\mathbb{E}[\rho_T] = \mathbb{E}\left[T\mu^* - \sum_{t=1}^{T} \widehat{r_t}\right] = T\mathbb{E}[\mu^*] - \mathbb{E}\left[\sum_{t=1}^{T} \widehat{r_t}\right] = T * 0.7 - T * 0.3 = 0.4T$$

Poll 1

$$\mu^* = \max_k \mu_k$$
$$\rho_T = T\mu^* - \sum_{t=1}^{T} \hat{r}_t$$

- Consider a MAB with 3 arms and the expected reward for arm $k \in \{1..3\}$ is $\mu_k = k/3$. If we randomly choose an arm to pull in each round for T rounds, what would be the expected average regret $\mathbb{E}[\frac{\rho_T}{T}]$?
- A: 1 B: $\frac{1}{2}$ C: $\frac{1}{3}$ D: $\frac{2}{3}$
- E: None of the above
- F: I don't know

	Arm 1	Arm 2	Arm 3
μ_k	1/3	2/3	1

Poll 1

$\mu^* = \max_k \mu_k$
$\rho_T = T\mu^* - \sum_{t=1}^T \hat{r}_t$

- Consider a MAB with 3 arms and the expected reward for arm $k \in \{1..3\}$ is $\mu_k = k/3$. If we randomly choose an arm to pull in each round for T rounds, what would be the expected average regret $\mathbb{E}[\frac{\rho_T}{T}]$?

$$\mathbb{E}\left[\frac{\rho_T}{T}\right] = \mathbb{E}\left[\frac{T\mu^* - \sum_{t=1}^T \hat{r}_t}{T}\right] = \mathbb{E}[\mu^*] - \mathbb{E}\left[\frac{\sum_{t=1}^T \hat{r}_t}{T}\right] = 1 - \frac{\sum_k \mu_k}{n}$$
$$= 1 - \frac{n(n+1)}{2n^2} = 1 - \frac{n+1}{2n} = 1 - \frac{1}{2} - \frac{1}{2n} = \frac{1}{2} - \frac{1}{2n} = \frac{1}{2} - \frac{1}{6} = \frac{1}{3}$$

Discussion

- What problems you encounter in your daily life can be viewed / modeled as an MAB problem?

- In those problems, how do you choose the arm to pull in each time step?

Probably approximately correct (PAC) and MAB

- Probably approximately correct (PAC): with high probability, it is close to being correct
$$\Pr(error \leq \epsilon) \geq 1 - \delta$$
- PAC version of zero-regret strategy
$$\Pr(\lim_{T \to \infty} \frac{\rho}{T} \leq \epsilon) \geq 1 - \delta$$

Upper Confidence Bound in Binary MAB

- In binary MAB, reward is either 0 or 1
- Let $N(k)$ be the number of times that k is chosen
- Let $H(k)$ be the number of times that k is chosen and reward is 1
- Let $\widehat{\mu_k} = H(k)/N(k)$, average reward when k is chosen

- Given $N(k), H(k), \widehat{\mu_k}$, we can estimate the range of μ_k

Upper Confidence Bound in Binary MAB

- $\widehat{\mu_k} = H(k)/N(k)$
- According to Chernoff-Hoeffding Bound
  - $\Pr(\widehat{\mu_k} \geq \mu_k + a) \leq e^{-2a^2 N(k)}$
  - $\Pr(\widehat{\mu_k} \leq \mu_k - a) \leq e^{-2a^2 N(k)}$
  - So $\Pr(\widehat{\mu_k} - a \leq \mu_k \leq \widehat{\mu_k} + a) \geq 1 - 2e^{-2a^2 N(k)}$
- What do we know from this?
  - If we set $a = \sqrt{\frac{2 \ln t}{N(k)}}$ and $\mu_{LB}^k = \widehat{\mu_k} - a, \mu_{UB}^k = \widehat{\mu_k} + a$
  - Then $\Pr\left(\mu_{LB}^k \leq \mu_k \leq \mu_{UB}^k\right) \geq 1 - 2t^{-4}$

Upper Confidence Bound in Binary MAB

- Chernoff-Hoeffding Bound: Let $X_1, X_2, \ldots, X_n$ be independent random variables in the range $[0, 1]$ with $\mathbb{E}[X_i] = \mu$. Then for $a > 0$

$$\Pr\left(\frac{1}{n}\sum_{i=1}^{n} X_i \geq \mu + a\right) \leq e^{-2a^2 n}$$

$$\Pr\left(\frac{1}{n}\sum_{i=1}^{n} X_i \leq \mu - a\right) \leq e^{-2a^2 n}$$

- That is, with high probability, the observed average value of X_i is very close to the expected value of X_i

Upper Confidence Bound in Binary MAB

- UCB1 Algorithm:
  - Always choose the arm with the highest upper confidence bound defined as $\mu_{UB}^k = \widehat{\mu_k} + \sqrt{\frac{2 \ln t}{N(k)}}$
  - Intuition: If μ_{UB}^k is large, either arm k is a good arm or $N(k)$ is small (not enough data is gathered)
  - General principle: optimism in the face of uncertainty

Upper Confidence Bound in Binary MAB

UCB1 Algorithm

Initialize $H(\cdot) = 0, N(\cdot) = 0, \mu_{UB}^{\cdot} = 0, \widehat{\mu_{\cdot}} = 0$

For $t = 1..T$

Choose arm $k_t = \text{argmax}_k\, \mu_{UB}^k$

Get reward $r_t \sim Bernoulli(\mu_k)$

Update $H(k_t) \leftarrow H(k_t) + r_t$

Update $N(k_t) \leftarrow N(k_t) + 1$

Update $\widehat{\mu_{k_t}} = \frac{H(k_t)}{N(k_t)}$

For $k = 1..N$

Update $\mu_{UB}^k \leftarrow \widehat{\mu_k} + \sqrt{\frac{2\ln t}{N(k)}}$

Upper Confidence Bound in Binary MAB

UCB1 Algorithm

Initialize $H(\cdot) = 0, N(\cdot) = 0, \mu_{UB}^{\cdot} = 0, \widehat{\mu_{\cdot}} = 0$
For $t = 1..T$
  Choose arm $k_t =$
  Get reward $r_t \sim$

  Update $H(k_t) \leftarrow H(k_t) + r_t$
  Update $N(k_t) \leftarrow N(k_t) + 1$
  Update $\widehat{\mu_{k_t}} = \frac{H(k_t)}{N(k_t)}$

  For $k = 1..N$
    Update $\mu_{UB}^{k} \leftarrow$

Poll 2

$\Pr(\mu_{LB}^k \leq \mu_k \leq \mu_{UB}^k) \geq 1 - 2t^{-4}$

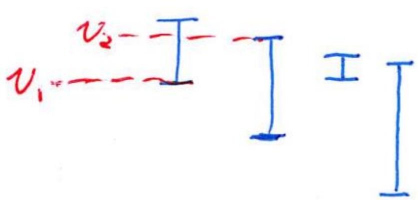
- (T/F) Follow UCB1 algorithm. Assume that in round t, arm 1 has the highest upper confidence bound μ_{UB}^k among all arms
- If $\mu_{UB}^1 - \mu_{LB}^1 \leq \epsilon$, then the difference between the optimal expected reward among all arms and the average value of arm 1 is guaranteed to be no larger than ϵ with high probability, i.e.,

$$\Pr\left(\max_k \mu_k - \mu_1 \leq \epsilon\right) \geq 1 - 2t^{-4}?$$

Poll 2

- (T/F) If $\mu_{UB}^1 - \mu_{LB}^1 \leq \epsilon$, then the difference between the optimal expected reward among all arms and the average value of arm 1 is guaranteed to be no larger than ϵ with high probability, i.e.,

$$\Pr\left(\max_k \mu_k - \mu_1 \leq \epsilon\right) \geq 1 - 2t^{-4}?$$



$v_2 - v_1 \leq \epsilon$

$\mu_k \geq \mu_{UB}^k - \epsilon$

$\mu_{k'} \leq \mu_{UB}^{k'} \leq \mu_{UB}^k$

So $\mu_{k'} - \mu_{k'} \leq \epsilon$

A Practical Problem: Video Recommendation

- You need to recommend videos to a user
- Each arm corresponds to a video
- Pull an arm: recommend the video to the user
- Reward: whether user clicks/likes the video

- Q: Can we use UCB1 to decide which video to recommend? What are the issues with this approach if we want to deploy it on Youtube?

Discussion

- In food rescue domain that we have discussed, is there any problem that can be formulated as a MAB problem?

Contextual Bandits

- K arms
- In round $t \in \{1 \dots T\}$, we observe context $\mathbf{x}_{k,t}$ for all arms $k \in \{1 \dots K\}$, then choose an arm I_t, and receive reward r_t which depends on I_t and $\mathbf{x}_{I_t,t}$
- In video recommendation, $\mathbf{x}_{k,t}$ can be features of the user-video pair
- In food rescue, $\mathbf{x}_{k,t}$ can be features of the rescue-volunteer pair

LinUCB Overview (Disjoint linear models)

- Assume reward is a arm-dependent linear function of context vector + noise
  - $r_t = \mathbf{x}_{I_t,t}^T \boldsymbol{\theta}_{I_t}^* + \epsilon_t$
  - $\mathbb{E}[r_t] = \mathbf{x}_{I_t,t}^T \boldsymbol{\theta}_{I_t}^*$
  - θ_k^* are unknown coefficient vector associated with each arm
- After a number of rounds, for each arm k, apply linear regression on existing data $(\mathbf{x}_{I_t,t}, r_t)$ where $I_t = k$ to get estimated $\widehat{\boldsymbol{\theta}}_k^*$
- For a new round t, calculate the UCB for each arm k based on $\mathbf{x}_{k,t}^T \widehat{\boldsymbol{\theta}}_k^*$
- Choose the arm with the highest UCB

LinUCB Overview (Hybrid linear models)

- $\mathbb{E}[r_t] = \mathbf{z}_{I_t,t}^T \boldsymbol{\beta}^* + \mathbf{x}_{I_t,t}^T \boldsymbol{\theta}_{I_t}^*$
- $\boldsymbol{\beta}^*$ is a shared coefficient vector across all arms
- Can still apply linear regression to get estimated $\widehat{\boldsymbol{\beta}}^*$ and $\widehat{\boldsymbol{\theta}}_k^*$
- For a new round t, calculate the UCB for each arm k based on $\mathbf{z}_{k,t}^T \widehat{\boldsymbol{\beta}}^* + \mathbf{x}_{k,t}^T \widehat{\boldsymbol{\theta}}_k^*$
- Choose the arm with the highest UCB

- Q: What if we already have some data, but we don't believe the reward is a linear function of context?

Outline

- Multi-Armed Bandit (MAB) Problems
- Bandit Data-Driven Optimization in Food Rescue (Optional)
- Markov Decision Process
- Restless Multi-Armed Bandits

Volunteer engagement as iterative prediction-prescription

Features
Volunteer & rescue attributes

ML prediction →

Predicted labels
Claim probability of each volunteer

Optimization ↓

Interventions
Send notifications to whom

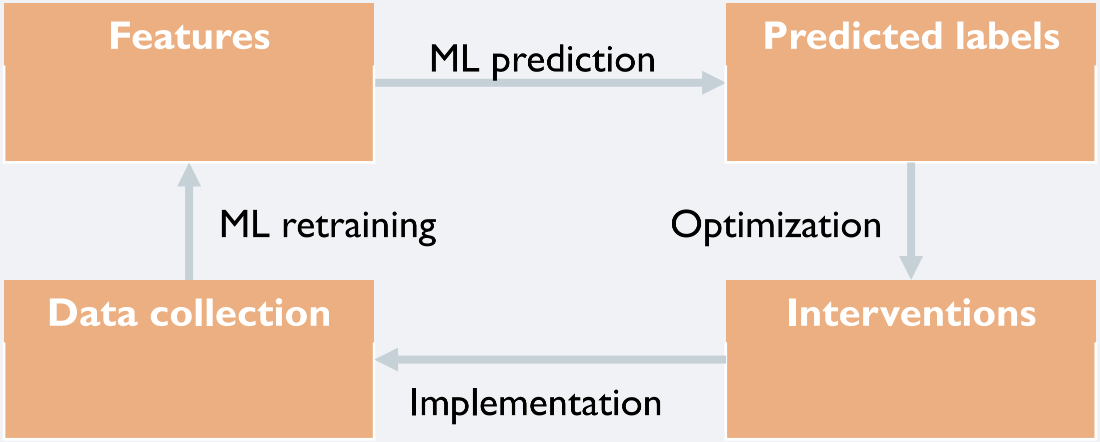
Implementation ←

Data collection
Observe which volunteer claims it

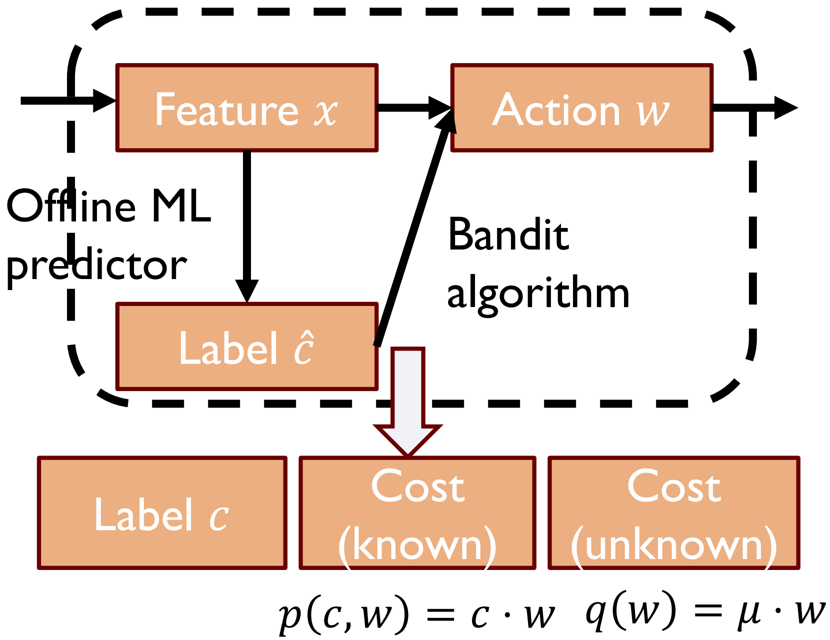
ML retraining ↑

Application-independent iterative prediction-prescription

- Data-driven optimization
  [Bertsimas and Kallus, 2020; Elmachtoub and Grigas, 2017]
- Decision-focused learning
  [Donti et al., 2017]
- Contextual/linear bandit
  [Dani et al., 2008; Lai and Robbins, 1985]



Bandit data-driven optimization



Optimal policy: $\pi(\mathbf{x}) = \arg\min_{\mathbf{w}} \mathbb{E}_{\mathbf{c},\eta|\mathbf{x}}[u(\mathbf{c}, \mathbf{w})]$

Regret: $R_T = \mathbb{E}_{x,c,\eta}\left[\sum_{t=1}^{T} \left(u(\mathbf{c}^t, \mathbf{w}^t) - u(\mathbf{c}^t, \pi(\mathbf{x}^t))\right)\right]$

PROOF: PRedict-then-Optimize with Optimism in Face of uncertainty

Algorithm 2: PROOF: PREDICT-THEN-OPTIMIZE WITH OPTIMISM IN FACE OF UNCERTAINTY

Initialize:

- Find a barycentric spanner $b_1, \dots, b_d$ for W
- Set $A_i^1 = \sum_{j=1}^{d} b_j b_j^\dagger$ and $\hat{\mu}_i^1 = 0$ for all $i = 1, 2, \dots, n$

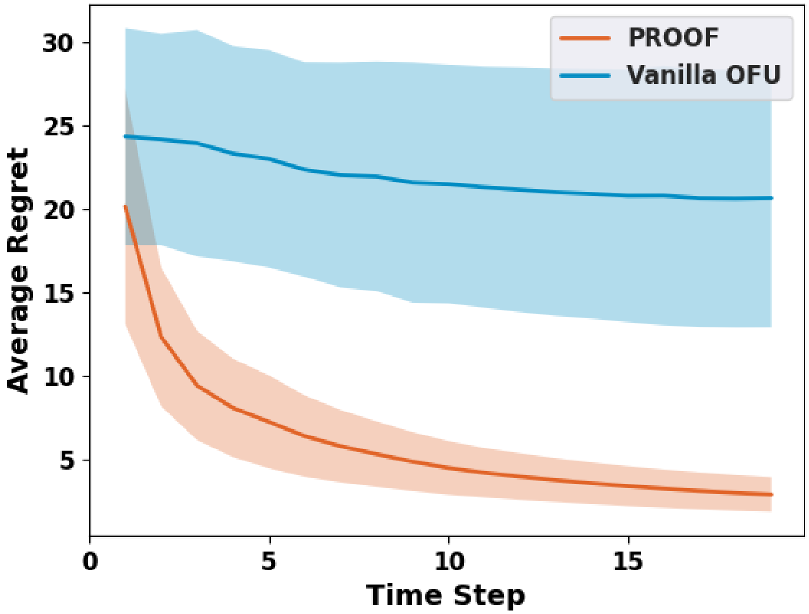
Receive initial dataset $\mathcal{D} = \{(x_i^0, c_i^0; w_i^0)_{i=1,\dots,n}\}$ from distribution D on (X, C).

for $t = 1, 2, \dots, T$ **do**

- Train the ML model & use it to make a prediction
- Set the confidence radius for UCB
- Select action by integrating UCB with offline ML model
- Receive the true labels and cost
- Update the bandit cost estimate

Theorem.
Assuming ordinary least squares regression, the PROOF algorithm has regret $\tilde{O}\left(n\sqrt{dmT}\right)$ with probability $1 - \delta$.

Numerical simulations



PROOF converges

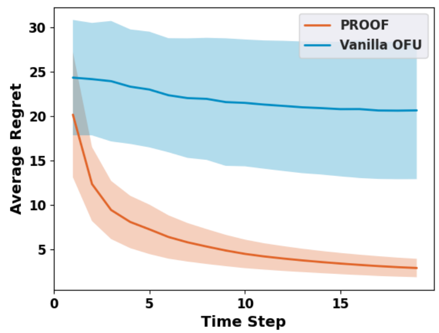
- much faster, and
- with smaller variance

than vanilla bandit.

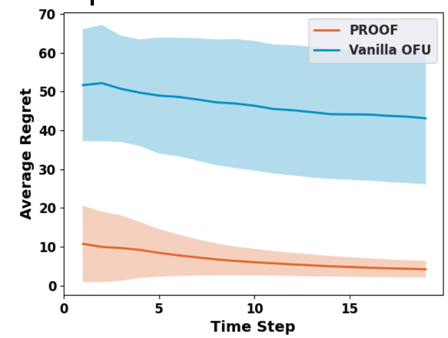
Numerical simulations

PROOF outperforms vanilla bandit in both convergence speed and variance.

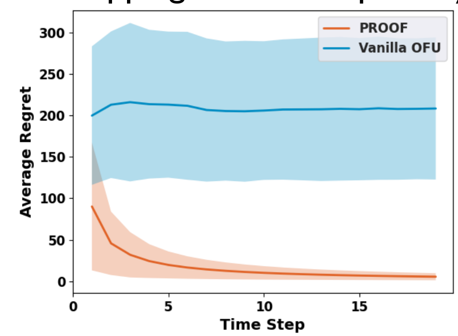
Small scale base case



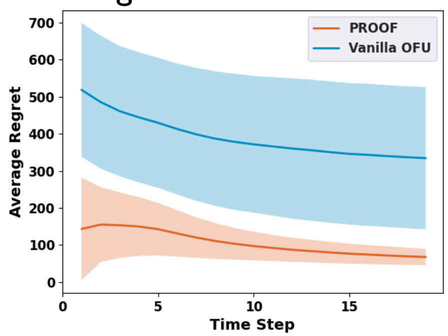
Data/step increased from 20 to 40



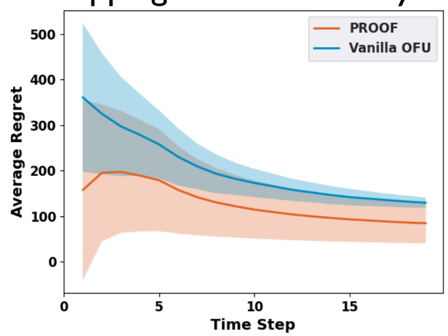
Linear mapping norm multiplied by 10



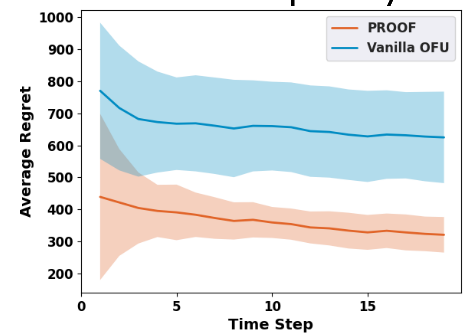
Large scale base case



Linear mapping norm divided by 10



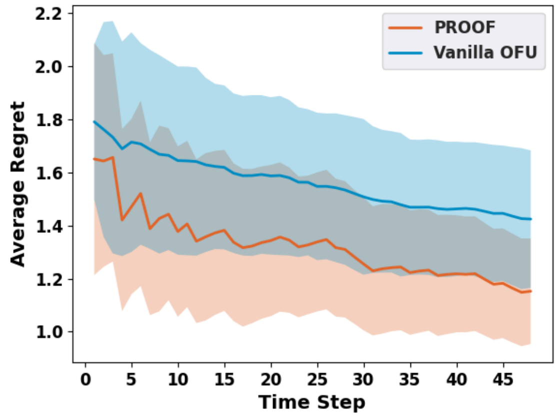
Data noise multiplied by 5



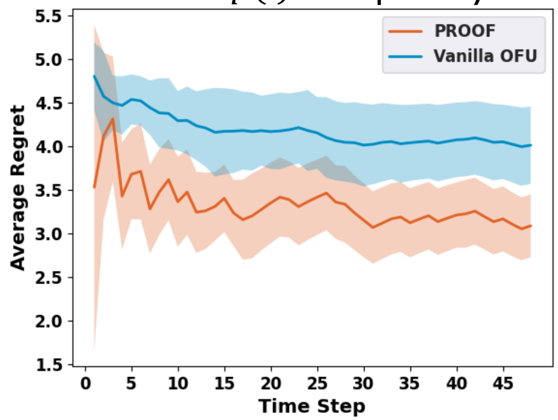
PROOF for food rescue volunteer recommendation

Feature x	Volunteer-rescue pair features
Label $c \in \{0,1\}^d$	whether volunteer claimed the rescue
Action $w \in \{0,1\}^d$	whether to send push notifications to each volunteer
Known cost $p(c, w)$	whether we send push notifications to the "right" volunteer
Unknown cost $q(w)$	how volunteers might react to notifications

Base case



Known cost $p(\cdot)$ multiplied by 4

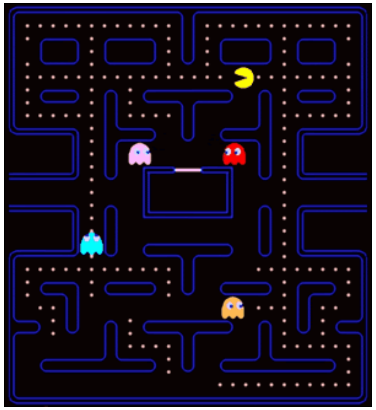


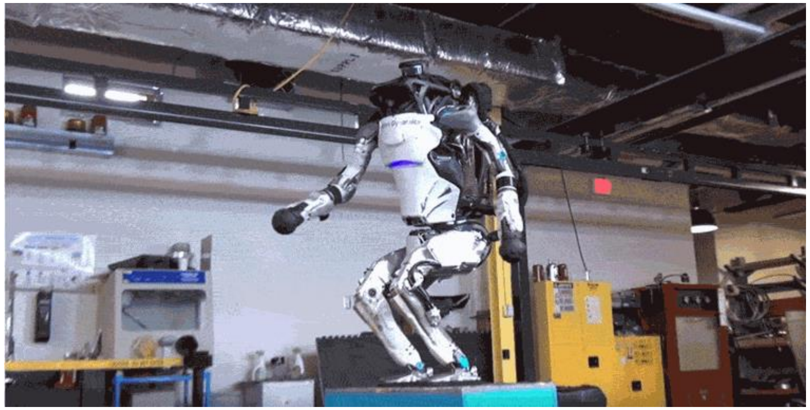
Outline

- Multi-Armed Bandit (MAB) Problems
- Bandit Data-Driven Optimization in Food Rescue
- Markov Decision Process
- Restless Multi-Armed Bandits

General Sequential Decision Making Problems

- The agent move from state to state
- At each time step, the agent choose an action
- Action bring the agent to a successor state
- Actions and / or states lead to reward
- A rational agent acts so as to maximize the expected utility in total (which is some function of the reward)





Markov Decision Process (MDP)

- Special case of sequential decision problems
- MDP = (S, A, T, R, γ)
  - S: set of states, $s_t \in S$ (where can agent be?)
  - A: set of actions, $a_t \in A$ (what can agent do?)
  - T: transition function $T(s_t, a_t, s_{t+1}) = \mathbb{P}(s_{t+1} | s_t, a_t)$ (what happens next?)

Next state only depends on the current state, not previous states! (Markovian)

  - R: reward function $r_t = R(s_t)$ or $R(s_t, a_t)$ or $R(s_t, a_t, s_{t+1})$ (what do I gain?)
  - $\gamma \in [0, 1]$ (discount factor)

Markov Decision Process (MDP)

- Assume an agent starts at s_0, takes action a_0, gets reward r_0, arrives at s_1, takes action a_1, …

- Agent's utility = accumulated reward with discount = $\sum_t \gamma^t r_t$

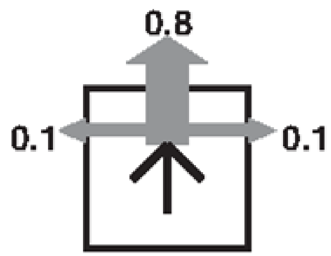
Markov Decision Process (MDP)

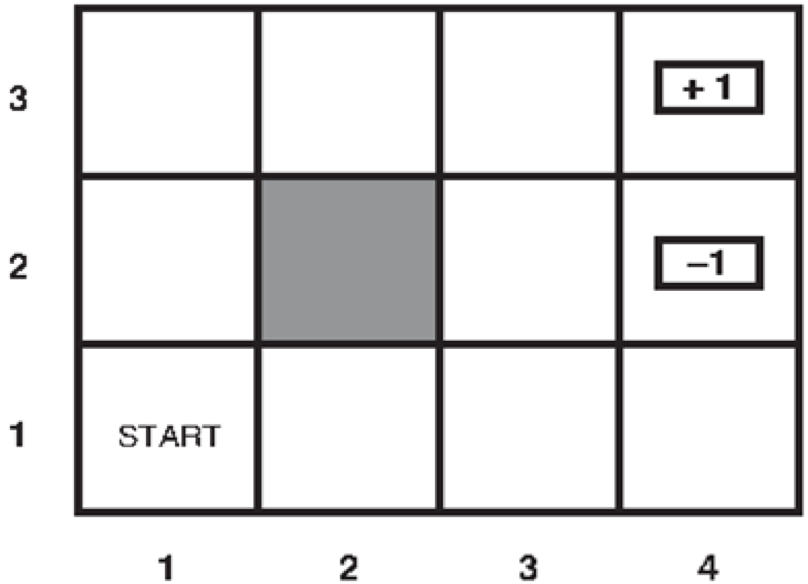
- MDP = (S, A, T, R, γ)
- (Deterministic) Policy $\pi: S \rightarrow A$
  - Maps state to action, defines a *plan*
- Given a policy π, we can sample history $h = \{s_0, a_0, s_1, a_1, \dots\}$
- Goal: find π to maximize utility
  - Expected
  - $\pi^* = \text{argmax}_\pi \mathbb{E}_{h\sim\pi}\left[\sum_t \gamma^t R\left(s_t, \pi(s_t)\right)\right]$
- Myopic strategy does not work: a_t affects future states

Example

MDP = (S, A, T, R, γ)

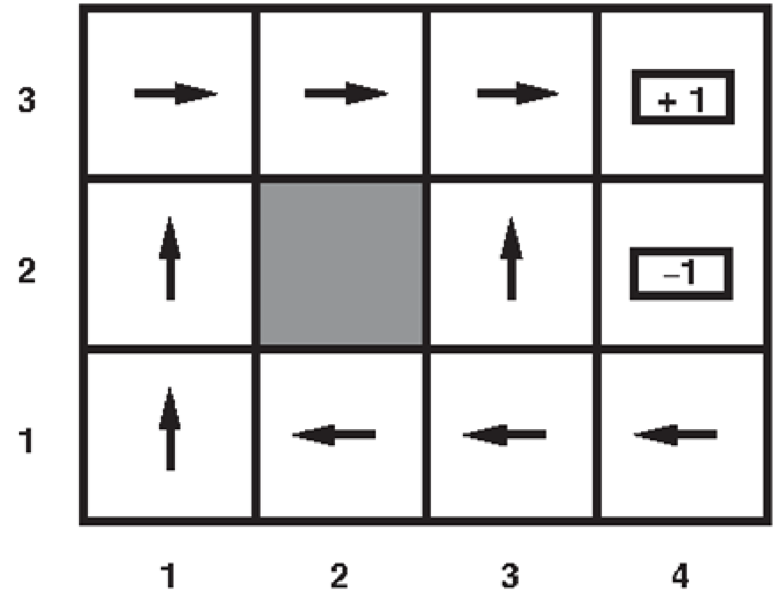
Q: Optimal policy with $R(s,a) = -100000$?





Optimal policy

with $R(s,a) = -0.04 \; \forall s, a$

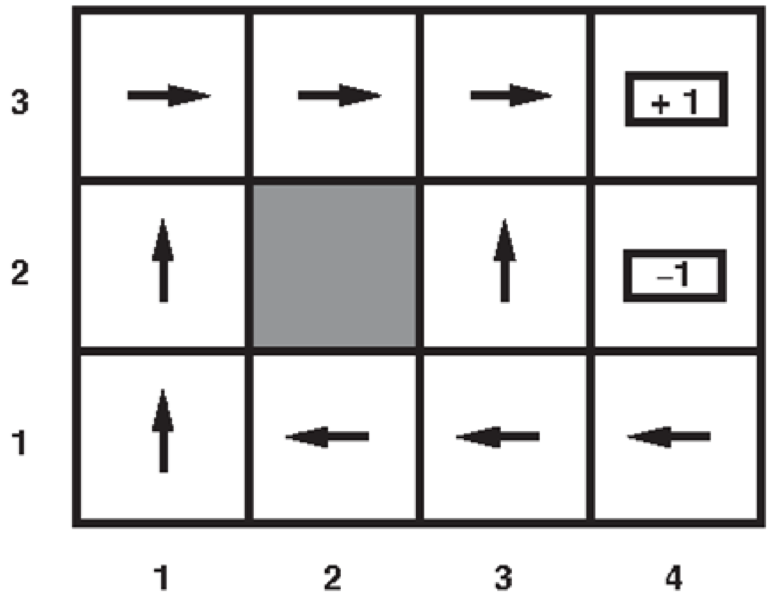


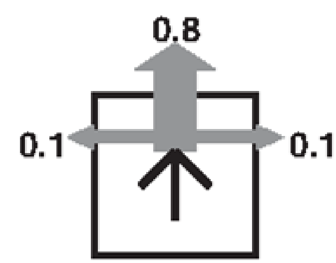
Actions = {Up, Down, Left, Right}

Example

- If we want to find optimal policy through brute force search: Enumerate all possible policies and compare expected utility

- How to compute/estimate the expected utility?





Value Function

- The value function of a given policy π describes the expected accumulated reward with discount starting from a state

$$V^{\pi}(s) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R\big(s_t, \pi(s_t)\big)]$$

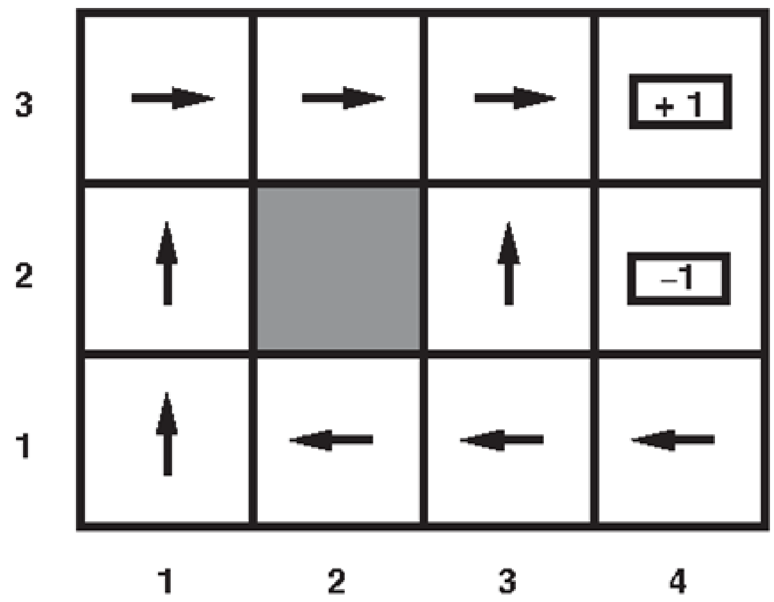
Where $s_0 = s$

Sometimes called state value function or V-value function

Value Function

- The goal can be restated as finding the policy π that lead to the optimal $V^{\pi}(s)$, i.e., $\operatorname{argmax}_{\pi} V^{\pi}(s)$

$$V(s) = V^*(s) = \max_{\pi} V^{\pi}(s) = V^{\pi^*}(s)$$

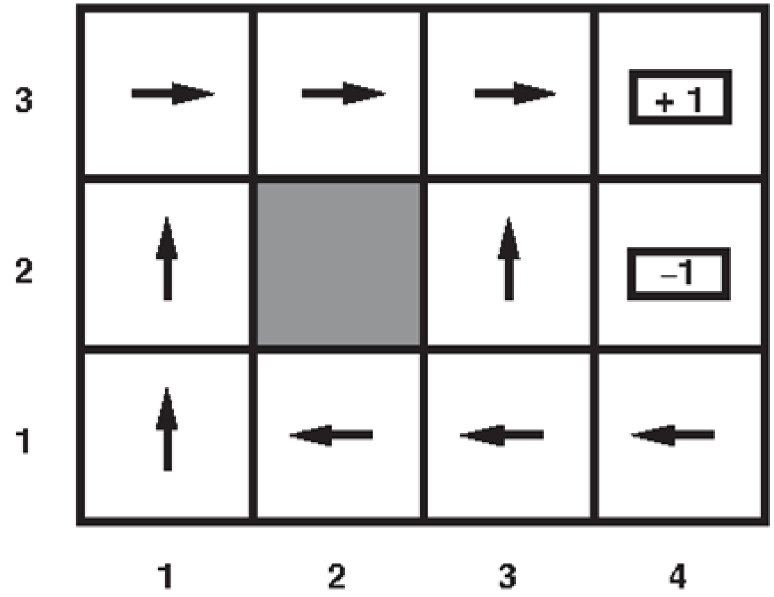


Value Function

Q: Given $V^*(s)$, how to find the optimal policy?

Pick the action which maximizes *current + future reward* (assuming continued optimal behavior)

$$\pi^*(s) = \underset{a \in A}{\operatorname{argmax}} \left[R(s,a) + \gamma \sum_{s'} \mathbb{P}(s'|s,a) V^*(s') \right], \forall s$$



Bellman Equation

- $V^*(s)$ satisfy the following Bellman Equation

$$V^*(s) = \max_{a \in A} [R(s,a) + \gamma \sum_{s'} \mathbb{P}(s'|s,a) V^*(s')]$$

Necessary and sufficient condition for optimality!

Q-Value Function

- Similar to state value function $V^{\pi}(s)$, but defined on state-action pair
- $Q^{\pi}(s,a)$: expected total reward from state s onward if taking action a in state s, and follow policy π afterward

That is $Q^{\pi}(s,a) = R(s) + \gamma \sum_{s'} P(s'|s,a) V^{\pi}(s')$

Obviously $V^{\pi}(s) = Q^{\pi}(s, \pi(s))$

So $Q^{\pi}(s,a) = R(s) + \gamma \sum_{s'} P(s'|s,a) Q^{\pi}(s', \pi(s'))$

Optimal Q-Value

Recall $V^\pi(s) = R(s) + \gamma \sum_{s'} P(s'|s,\pi(s))V^\pi(s')$
And $Q^\pi(s,a) = R(s) + \gamma \sum_{s'} P(s'|s,a)V^\pi(s')$
And $V^\pi(s) = Q^\pi(s,\pi(s))$

- When using optimal policy π^*, we will take the action that leads to maximum total utility at each state

$$\pi^*(s) = \operatorname*{argmax}_a Q^*(s,a)$$

- Therefore

$$V^*(s) = Q^*\big(s,\pi^*(s)\big) = \max_a Q^*(s,a)$$

$$Q^*(s,a) = R(s) + \gamma \sum_{s'} P(s'|s,a)V^*(s')$$

$$= R(s) + \gamma \sum_{s'} P(s'|s,a) \max_{a'} Q^*(s',a')$$

Bellman Equation

Again, necessary and sufficient condition for optimality!

How to Solve MDPs (Non-Exhaustive List)

- Exact
  - Value Iteration
  - Policy Iteration
  - Linear Programming

- Approximate
  - Sampling-based
  - Function approximation

How to Solve MDPs in Practice

- Call MDP solvers
  - E.g., MDP toolbox in python

```
>>> import mdptoolbox, mdptoolbox.example
>>> P, R = mdptoolbox.example.rand(10, 3)
>>> pi = mdptoolbox.mdp.PolicyIteration(P, R, 0.9)
>>> pi.run()
```


https://pymdptoolbox.readthedocs.io/en/latest/api/mdp.html

Outline

- Multi-Armed Bandit (MAB) Problems
- Bandit Data-Driven Optimization in Food Rescue
- Markov Decision Process
- Restless Multi-Armed Bandits

Restless Multi-Armed Bandit

MDP = (S, A, T, R, γ)

- N arms
- Each arm is a 2-action MDP
  - In time t, an arm is in some state
  - Two possible actions: active (pull the arm), or passive (not pull the arm)
  - Action brings the planner reward and brings the arm into a successor state in time $t + 1$
  - "Restless": state transition happens even if the arm is not pulled
- Planner can observe the state of each arm, and pull αN arms in each time step

Example

- You take 5 courses this semester
- Each arm is a course
- Everyday, you choose two courses and spend 4 hours on each of them
- The state of each arm is your level of understanding of the course material
- If you choose a course A in one day, your level of understanding for course $A \uparrow$
- If you do not choose A in one day, your level of understanding for course $A \downarrow$ (we forget things ☹)

Discussion

- What real-world problems can be viewed / modeled as a restless MAB problem?
- Can you design a heuristic way of choosing the arms to pull?

Restless Multi-Armed Bandit

- Assuming you know the MDP associated with each arm, how to choose the arms to pull in each time step given the observed states of the arms?
- Whittle solution approach:
  - Key idea: "passive subsidy" – a hypothetical reward offered to the planner, in addition to the original reward function, for choosing the passive action
  - Whittle Index: Infimum subsidy that makes the planner indifferent between the "active" and the "passive" actions

$$W(s) = \inf_{\lambda}\{\lambda: Q_{\lambda}(s, passive) = Q_{\lambda}(s, active)\}$$

  - Choose the arms with highest $W(s)$

Backup Slides

Value Iteration

Value Iteration

Initialize $V_0^*(s) \leftarrow 0$

Typical termination condition: difference$< \epsilon$

Iterate $V_{i+1}^*(s) \leftarrow \max_{a \in A} [R(s,a) + \gamma \sum_{s'} \mathbb{P}(s'|s,a) V_i^*(s')]$

Based on state-value Bellman Equations

$$V^*(s) = \max_{a \in A} [R(s,a) + \gamma \sum_{s'} \mathbb{P}(s'|s,a) V^*(s')]$$

- Consider the finite horizon view: Pretend we have a really long horizon
- aka 'Value update', 'Bellman backups/updates'
- Guaranteed to converge to $V^*(s)$

What is the optimal policy π^*?

Policy Iteration

- Not necessary to get an accurate estimate of V^* to induce π^*

- Policy iteration
  - Compute optimal policy π^* directly
  - Iterate between 2 steps
    - Policy Evaluation (check how good current policy is)
    - Policy Improvement (get a 'better' policy)

Policy Iteration

- Policy Evaluation
  - Method 1: Iterative approach
    - $V_{i+1}^{\pi}(s) \leftarrow R(s,\pi(s)) + \gamma \sum_{s'} \mathbb{P}(s'|s,a) V_i^{\pi}(s'), V_0^{\pi}(s) \leftarrow 0$
  - Method 2: Solve system of linear equations
    - $V^{\pi}(s) = R(s,\pi(s)) + \gamma \sum_{s'} \mathbb{P}(s'|s,a) V^{\pi}(s')$
- Policy Improvement
  - $\pi^{new}(s) \leftarrow \underset{a \in A}{\operatorname{argmax}} \left[R(s,a) + \gamma \sum_{s'} \mathbb{P}(s'|s,a) V^{\pi}(s')\right]$
  - Note that when π is optimal, π^{new} is the same as π
- Policy iteration converges to the optimal policy in a finite number of steps
- Often converge faster than value iteration

LinUCB

- Assume reward is a arm-dependent linear function of context vector + noise
  - $r_t = \mathbf{x}_{I_t,t}^T \theta_{I_t}^* + \epsilon_t$
  - $\mathbb{E}[r_t] = \mathbf{x}_{I_t,t}^T \theta_{I_t}^*$
  - θ^* are unknown coefficient vector associated with each arm