IEOR 8100-001: Learning and optimization for sequential decision making

Instructor: Shipra Agrawal

Industrial Engineering and Operations Research
Columbia University

...

Outline

Marovian bandits, Gittins Index

Markovian state-dependent inputs

In MAB rewards from an arm were i.i.d. from **stationary distribution**. Two properties:

- IID across time
- Independence across arms

Every time you pulled an arm the reward was generated independently from a stationary distribution, irrespective of how many times it was pulled before and what happened in the previous pulls of the arm. Past pulls may affect your decision, not the actual reward distribution.

Markovian state-dependent inputs

What if pulling an arm changes the state of the arm, changing the response from the same arm in the future?

- Scheduling problem: jobs are arms, pulling means scheduling it. Every time you schedule it, a part of it gets finished – you will receive some reward when it is finished completely, and reward 0 before that.
- Single user, Many movies (arms): You recommend a movie to a user, he liked it, what happens when you recommend it again?
- Single user, many ads (arms): A sustained ad campaign can increase the consumer's chance to purchase that product.

Markovian Multi-armed bandit problem: Two main features

- Reward distribution of an arm can depend on past but only through an observable *state of that arm* (Markovian dependence on past)
  - state of job: how much of the job is remaining
  - state of movie: how many times it has been recommended/watched before
  - state of ad: brand image of product (not observable)
- Pulling an arm changed the future reward distribution of *that* arm only. (independence across arms)
  - scheduling a job changes how much of it is remaining
  - if a movie is recommended and the user watches it, may not want to watch it again next time – doesn't changes her preferences for other movies
  - showing an ad changes only the brand image of that product.

"Markovian Multi-armed bandit problem", also referred to simply as "Multi-armed bandit problem".

Markovian multi-armed bandit

N arms, $a = 1, \ldots, N$.

Every arm a is described by $(\mathcal{S}_a, r_a, p_a, s_{a,1})$

- State space $\mathcal{S}_a$. starting state $s_{a,1}$
- Reward function $r_a(s)$ for $s \in \mathcal{S}_a$. (bounded reward, no noise)
- Transition probabilities $p_a(s, s')$ for $s, s' \in \mathcal{S}_a$: probability of going from state s to s'.

Here, $\sum_{s' \in \mathcal{S}} p_a(s, s') = 1, \forall s \in \mathcal{S}, a \in \mathcal{A}$. Every row of p_a is a distribution over states.

Markovian multi-armed bandit

Starting states $s_{1,1}, \ldots, s_{N,1}$ for arms $1, \ldots, N$.

At time t,

- Observe state $s_{a,t} \in \mathcal{S}_a$, for every arm a.
- Take an action $a_t \in \mathcal{A}$.
- Receive reward $r_t = r_{a_t}(s_{a_t,t})$.
- State of arm a_t transitions from $s = s_{a_t,t}$ to state $s' = s_{a_t,t+1} \in \mathcal{S}_a$ with probability $p_{a_t}(s, s')$.

Goal

Let algorithm plays arm a_t at time t, and $s_{a_t,t}$ is the state of arm a_t at time t. Let $r_t = r_{a_t}(s_{a_t,t})$ Reward accumulated in time T:

$$\sum_{t=1}^{T} r_t$$

Discounted reward (discount factor $\gamma < 1$)

$$\lim_{T\to\infty} \sum_{t=1}^{T} \gamma^{t-1} r_t$$

Average reward:

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} r_t$$

Example: scheduling

Arms: each job is an arm. On scheduling a job a, a random fraction u of the job gets completed where u is generated i.i.d. from some distribution D_a.

- State space $\mathcal{S}_a$ has infinite states, one each corresponding to s =fraction of job remaining.
- Reward function $r_a(s) = 0$, for $s > 0$, $r_a(0) = \mu_a$, and a dummy state $s = -1$, with reward $r_a(-1) = 0$.
- Transition probabilities $p_a(s, s') = \Pr_{u \sim D_a}(u = s - s')$, $s, s' \geq 0$, $p_a(0, -1) = 1, p_a(-1, -1) = 1$.
- **Discounted** reward (discount factor $\gamma < 1$)

$$\lim_{T \to \infty} \sum_{t=1}^{T} \gamma^{t-1} r_t = \sum_a \gamma^{\tau_a - 1} r_a(s_{a,\tau_a}) = \sum_a \gamma^{\tau_a - 1} \mu_a$$

where τ_a is the time at which job a got completed. You want to complete larger jobs earlier. **Finite time reward** is total reward for jobs completed in time T. For infinite number of jobs **infinite horizon average reward** is rate of reward for completed jobs.

Problem with known transition matrix and reward function

Solve the problem assuming $p_a(\cdot, \cdot)$ and $r_a(\cdot)$ are known for every arm a.

- At time t, after observing the current states of every arm which arm should you pick?

Optimal policy

Our goal is to maximize expected discounted reward. Let algorithm plays arm a_t at time t, and $s_{a_t,t}$ is the state of arm a_t at time t. Let $r_t = r_{a_t}(s_{a_t,t})$. Expected Discounted reward (discount factor $\gamma < 1$):

$$\mathbb{E}[\sum_{t=1}^{\infty} \gamma^{t-1} r_t]$$

What is the optimal policy?

- A policy is a plan or strategy: what action will be taken at time t, given the history at time t (history includes states of all arms at time t).
- Without state dependence (and known r_a for every arm) optimal policy would be simply be to pick arm $\arg\max_a r_a$ at every time t.
- In Markovian problem, we cannot simply pick $\arg\max_a r_a(s_{a,t})$. We need to evaluate the effect of action in time t on what future states the arm will transition to, and therefore future rewards.

Evaluating a policy

Given a policy $\pi : (t, \mathcal{H}_t) \rightarrow \mathcal{A}$, expected discounted reward

$$\mathbb{E}[\sum_{t=1}^{\infty} \gamma^{t-1} r_{a_t}(s_{a_t,t})]$$

where $a_t = \pi(t, \mathcal{H}_t)$.
Given infinite computational power, we could evaluate all possible policies, to compute optimal policy, but not only that would be computationally expensive, the optimal policy could have an infinite description.

A simplification

- Consider Markovian stationary policies, i.e. policies that map states to action: $\pi : \mathcal{S}_1 \times \cdots \times \mathcal{S}_N \to \mathcal{A}$.
  - For example, policies which will just look at how much of each job is remaining and take decision based on that, irrespective of time and history.
- Remarkable result from dynamic programming: there always exist a Markovian stationary policy which is optimal. We will see a proof of this later in the class for more general Markovian sequential decision making processes.

Even if consider only policies that are Markovian and stationary, i.e. maps states to actions, the description will be at least exponential in number of arms (possible values of $(s_{t,a_1}, \ldots, s_{t,a_N})$ is $\Pi_a |\mathcal{S}_a|$, i.e., exponential in N even if each $\mathcal{S}_a$ is a finite set).

Gittins Index

"A dynamic allocation index for the design of experiments". J. C. Gittins and D. M. Jones, 1974.
"Multi-armed bandit Allocation Indices". J.C. Gittins. 1989

Informal Theorem statement:

Theorem

For every arm a, and every state $s \in \mathcal{S}_a$, an index $G_a(s)$ (called Gittins Index) can be computed such that given states $s_{1,t}, \ldots, s_{N,t}$ at time t, an optimal policy is to pull the arm

$$a_t := \arg\max_a G_a(s_{a,t})$$

Why useful? Policy description reduced from $|S|^N$ to $N|S|$.

Gittins Index

$$G_a(s) := \sup_{\tau \geq 1} \frac{\mathbb{E}\left[\sum_{t=1:s_{a,1}=s}^{\tau} r_a(s_{a,t})\gamma^{t-1}\right]}{\mathbb{E}[\sum_{t=1}^{\tau} \gamma^{t-1}]}$$

- numerator: expected total discounted reward if the arm a is continuously played starting from state s until some stopping time τ.
- denominator: expected total discounted *time* until stopping time τ.
- Stopping time τ: Stopping time is a time which can be recognized when it occurs.
- In fact, it can be shown that τ attains the supremum when

$$\tau = \min\{t : G_a(s_{a,t}) \leq G_a(s_{a,1}), t > 1\}$$

that is, τ is the first time at which the process reaches a state in which the Gittins index is no greater than it was initially.

Gittins Index

Two things to note:

- Decision at time t depends only on states of the arms at time t. Given the states, history and time index does not matter: Markovian stationary policy: maps states to actions.
- Index of arm a depends only on $p_a(\cdot, \cdot)$ and $r_a(\cdot)$, i.e. transition and reward function for that arm.

Gittins Index: example

Job scheduling : for each job states are fraction of job remaining. From a given starting state s, let $\tau_{s,a}$ be the time at which the job finishes. That is, the state at $\tau_{s,a}$ pull is $s_{a,\tau_a} = 0$, reward will be μ_a at $(\tau_{s,a})^{th}$ pull of arm a. Therefore, the inner term in evaluation of Gittins index $G_s(a)$ will be

- 0 for all $\tau < \tau_{s,a}$
- for $\tau = \tau_{s,a}$, it will be

$$\frac{\mathbb{E}[\gamma^{\tau_{s,a}-1}\mu_a]}{\mathbb{E}[(1+\gamma+\cdots+\gamma^{\tau_{s,a}-1})]} = \frac{\mathbb{E}[\gamma^{\tau_{s,a}-1}](1-\gamma)\mu_a}{\mathbb{E}[(1-\gamma^{\tau_{s,a}})]}$$

- and will started decreasing afterwards (more terms will be added to denominator but not to numerator).

Therefore, $\tau_{s,a}$ will attain the supremum, will be given by the expression in the second bullet above: depends on just $\mathbb{E}[\gamma^{\tau_{s,a}}]$ Given the distribution of how much of job gets finished every time it is scheduled, this can be computed (or estimated to very good accuracy).

Gittins index solution for job scheduling: Given state $s_{a,t}$ for arm a at time t, compute

$$\Gamma_{a,t} = \mathbb{E}[\gamma^{\tau_{s_{a,t},a}}], \forall a$$

where $\tau_{s_{a,t},a}$ is number of pulls it will take to finish the *remaining* job if you keep pulling arm a.

Pull arm

$$a_t = \arg\max_a \frac{\Gamma_{a,t}}{1-\Gamma_{a,t}} \frac{(1-\gamma)\mu_a}{\gamma}$$