Topics in Theoretical CS:
Bandits, Experts, and Games

CMSC 858G Fall 2016
University of Maryland

Alex Slivkins
Microsoft Research NYC

What the course is about?

- algorithms for making **sequential decisions under uncertainty**
  - theoretical CS, machine learning, AI, operations research, economics
  - since 1933, very active in the past decade
  - “bandits” and “experts” – two prominent models
- focus: theory (design & analysis of algorithms)
  - … using tools from Probability
  - … with lots of examples & discussions for motivations & applications
  - … with connections to Economics (game theory and mechanism design)

This lecture

- course organization
- intro to the problem space
- short break (15-20 mins)
- review of Probability (necessary basics)

Prerequisites

- **Algorithm design & mathematical proofs**:
  competence at the level of CMSC 451 (undergrad algorithms course)
- **Probability & statistics**:
  - I'll review the basics later in this lecture
  - deeper familiarity would help, but not required
- **Game theory & mechanism design**:
  - relevant in the last 2-3 lectures
  - prior exposure would help but is not required
- **Programming**: familiarity with programming is not required

Logistics

- Instructor: Alex Slivkins, Senior Researcher, Microsoft Research NYC.
- Schedule: Mondays 2:30pm-5:30pm (short break in the middle), AVW 3258
- Office hours: Mondays 11am-2pm (by appointment), AVW 3171.
- Course webpage: http://www.cs.umd.edu/~slivkins/CMSC858G-fall16/
- Announcements: on homepage and via mailing list cmsc858g-0101-fall16@coursemail.umd.edu
- Q&A: we will use Piazza: https://piazza.com/umd/fall2016/cmsc858g/.
- Contact: slivkins@cs.umd.edu (but please use Piazza if appropriate).
  - please use my UMD email, not my Microsoft email

Coursework and assessment

- 2-3 **homeworks**
  - dates TBA, due in class, no extensions (without a good reason)
  - OK to discuss, but everyone writes his/her own solutions
- **Project**: reading, coding, and/or original research on a course-related topic
  - in groups of 2-3 ppl
- **Scribe** one lecture (in pairs).
  - starting from next lecture, sign-up sheet coming
  - write in LaTeX, template & instructions coming
- **Grading**

  homeworks: 50%

  project: 40%, plus up to 10% bonus for coding and/or original research

  scribing: 10%, plus 5% bonus if scribing another lecture.

Projects

- **topic suggestions** -- I will post soon
- **topic proposal** -- will be due
  - form groups of 2-3 ppl (I'll post a sign-up sheet), each group submits a proposal
- **feedback / discussion**
- **written "final report"** – will be due
- **short presentation** -- in the last two classes

Intro to the problem space

Sequential decisions under uncertainty

(Informal & very stylized) running examples

- **Investment.** Each morning, you choose one stock to invest into, and invest $1.
  In the end of the day, you observe the change in value for each stock.
  Goal: maximize wealth.
- **News site.** When a new user arrives, the site picks a news header to show, observes whether the user clicks. Goal: maximize #clicks.
- **Dynamic pricing.** A store is selling a digital good (e.g., an app or a song).
  When a new customer arrives, the store picks a price.
  Customer buys (or not) and leaves forever. Goal: maximize total profit.

Basic model: multi-armed bandits

- A fixed set of K actions ("arms")
- In each round $t = 1 \ldots T$ algorithm chooses an arm a_t,
  and observes the reward r_t for the chosen arm
- **"Bandit feedback"**: no other rewards are observed!
- **IID rewards:**
  The reward for each arm is drawn independently from a fixed distribution that depends on the arm but not on the round t.
- Time horizon T is known in advance.

Examples

Example	Action	Reward	Other feedback
Investment	a stock to invest into	change in value during the day	change in value for all other stocks
News site	an article to display	1 if clicked, 0 otherwise	NONE
Dynamic pricing	a price p	p if sale, 0 otherwise	sale => sale at any smaller price no sale => no sale at any larger price

- **Full feedback:** reward is revealed for all arms
- **Bandit feedback:** reward is revealed only for the chosen arm
- **Partial feedback:** reward is revealed for some but not (necessarily) for all arms

Exploration-exploitation tradeoff

- Bandit/partial feedback => need to try different arms to acquire new info
  - if algorithm always chooses arm 1, how would it know if arm 2 is better?
- fundamental tradeoff between acquiring info about rewards (***exploration***) and making optimal decisions based on available info (***exploitation***)
  - this tradeoff happens in many scenarios (“reinforcement learning”)
  - multi-armed bandits is a simple model to study this tradeoff

Rich problem space

- full feedback vs bandit feedback vs partial feedback
  - (most of the course will be on bandit & partial feedback)
- … many other distinctions

Distinction #2: where rewards come from?

- **IID rewards:** the reward for each arm is drawn independently from a fixed distribution that depends on the arm but not on the round t.
- **Adversarial rewards:** rewards are chosen by an adversary.
- **Constrained adversary**:
  rewards are chosen by an adversary with known constraints, e.g.:
  - reward of each arm can change by at most ϵ from one round to another
  - reward of each arm can change at most once
- **Stochastic rewards** (beyond IID):
  reward of each arm evolves over time as a random process
  - e.g. random walk: changes by $\pm\epsilon$ in each round

Distinction #3: contexts

In each round, there may be a **context** observable before the decision is made

Example	Action	Reward	context
Investment	a stock to invest into	change in value during the day	current state of the economy
News site	an article to display	1 if clicked, 0 otherwise	user location & demographics
Dynamic pricing	a price p	p if sale, 0 otherwise	Customer's device (e.g.: Windows, Android or Apple?)

Other distinctions

- **Bayesian prior**? (i.e.: problem instance comes from known distribution)
- **Structured rewards:** rewards may have a known structure
  e.g.: arms are points in $[0,1]^d$ and in each round the reward
  is a linear / concave / Lipschitz function of the chosen arm
- **Global constraints:** e.g.: limited #items to sell
- **Complex decisions**
  A news site picks a *slate* of articles
  A store prices many products at once.
- **Complex outcomes** (more than just the reward)
  Dynamic pricing: which items have been sold?
  News site: time spent reading an article?

Some philosophy

- Reality can be complicated … we often study simpler models.
- a good model captures some essential issues
  … present in multiple applications
  … and allows for clean solutions with good performance
  … and provides intuition (if not solutions) for (more) realistic models
- but even a good model typically does not fully capture any one application
  … and that’s OK …
- very rich problem space => why work on problems with shaky motivation?

More examples

Example	Action	Rewards / costs
medical trials	drug to give	health outcomes
internet ads	which ad to display	bid value if clicked, 0 otherwise
content optimization	e.g.: font color or page layout	#clicks
sales optimization	which products to sell at which prices	$$$
recommender systems	suggest a movie, restaurants, etc.	#users that followed suggestions
computer systems	which server(s) to route the job to	job completion time
crowdsourcing systems	which tasks to give to which workers	quality of completed work
	which price to offer?	#completed tasks
wireless networking	which frequency to use?	#successful transmissions
robot control	a “strategy” for a given state & task	#tasks successfully completed
game playing	an action for a given game state	#games won