Lecture 7: Introduction to Techniques

Instructor: Chien-Ju (CJ) Ho

Logistics

- Assignment 2 is out.
  - Due: Feb 19 (Tuesday), 2019

- Questions are follow-ups of the lecture
  - Cooperation and repeated prisoner’s dilemma
  - Eliciting effort with proper scoring rules
  - Peer grading and peer prediction

Logistics

- Presentation
  - You can present in the format you prefer.
  - Try to engage the class as much as possible.
  - One potential format:
    - Summarize the required reading (20-30 min)
    - Discussion (15-20min)
    - Summarize another paper(s)
    - Discussion

  - For more mathematical presentations, spending time **slowly and carefully go through the model and assumptions** before jumping into the results.
  - You might want to **summarize the required reading** carefully. Yes, everyone should have already read it, but everyone might be confused about some parts of the paper.

Logistics Reviews

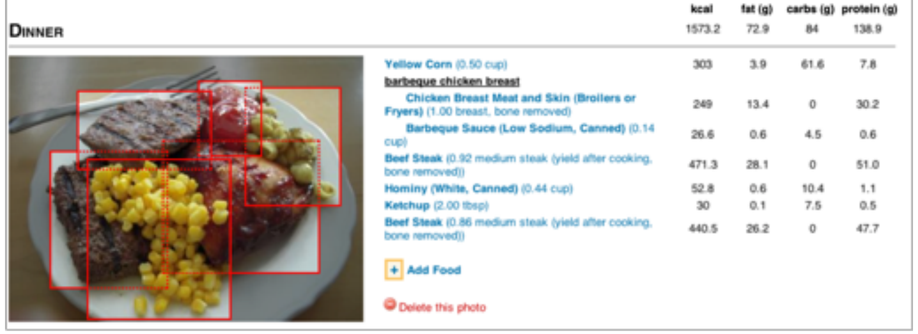
- Standard questions
  - Summarize the paper
  - List 1~3 things you like
  - (maybe) Come up with future directions
    - Might not include this one if there are many questions already
  - (maybe) 2 questions from presenters
- Questions from presenters
  - Please try to come up with around 2 questions when you meet with me a week before your presentations
  - These questions can be used for discussion during your presentation

Logistics

- Project Proposal due on Feb 12
  - Gradescope submission has already been set up

- You have quite a lot flexibility in choosing what to do, as long as the topic is relevant to the course.
- Feel free to discuss with me to get feedback or ask for references.
  - I might not always be helpful, but I’ll try to be.

Example Project 1



- Building Applications: Design an application that utilize the crowd power
  - Analyze the nutrition of the food in a photo [Noronha et al. UIST 2011.]
  - Collecting human decisions in moral scenarios and build a crowd model [Moral machines]
  - Plan a travel with various constraints
  - Measuring and maybe even fixing media biases

- Suggestion:
  - Find a reasonable-scale application that you can build a prototype and use yourself as workers in the preliminary evaluation.
  - Discuss with me early if you want to recruit real workers.
    - There are some logistics within the university to go through. So planning ahead is required.

Example Project 2

- Theory-oriented projects on designing/analyzing incentives.
  - **reputation systems:** enable workers to put in good effort in completing the work?
  - **task recommendation as incentives**: Promise workers that they are going to get recommended better tasks if they do a good job now.
  - **access to information as incentives**? You are only going to see others' reviews on Yelp if you provide good-quality reviews yourself.
  - Maybe other incentives?

- Suggestion:
  - Start by making strong assumptions about how workers behave to get a sense of whether the problem is approachable.
  - Gradually relax the assumptions you find unrealistic.

Example Project 3

- Designing label aggregation algorithms on datasets or simulations
  - For example, you can study how the existence of network might affect label collection and aggregation (or any other human factors/biases).
    - Find an existing dataset
    - Simulate the social network behind the workers (or any other effects you want to study)
    - Propose models on workers behavior behind the data
    - Modify the datasets according to your model
    - Design and test your aggregation algorithms

- Figure Eight has released some datasets that might be interesting
  - https://www.figure-eight.com/datasets/
  - Disclaimer: I didn't carefully check out what they have there…

Example Project 4

- Understand human behavior through data analysis or experiments
  - Is there anything you want to know about the workers/requesters/platforms that are not discussed?

  - Develop methods to obtain the information
    - Are there existing datasets?
    - Can you crawl the data from the web?
    - Can you design behavioral experiments to obtain the data?

Example Project 5

- Improvements over the existing paper.
  - Find a paper of interests
  - Identify what you think can be improved about the paper, e.g., they might make very strong assumptions or their ideas might be able to be applied in a different setting
  - Improve and/or implement the paper

and more….

- Anything that is related to the course (humans in the computational process) should be good.

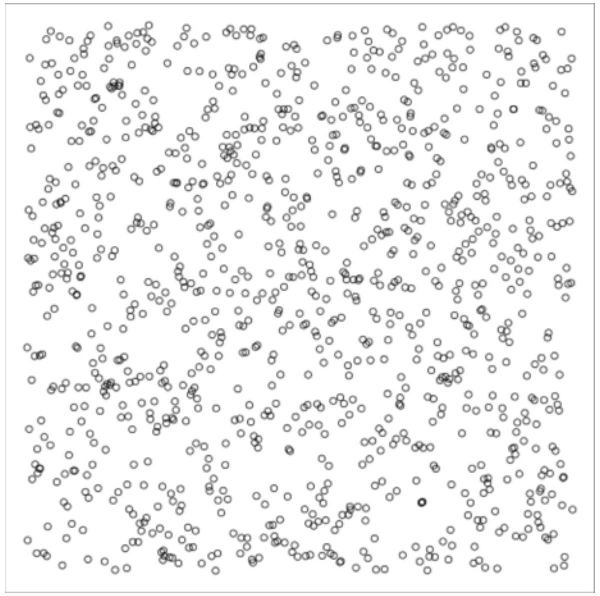
- Try to work on things you feel passionate/enthusiastic about
  - You can convert the project to literature survey before milestone 2

Today’s Lecture

- Optimization frameworks and connection to crowdsourcing
  - Assignment problem and convex optimization
  - Multi-arm bandit problem
  - Markov decision process
- Focus more on formulations instead of algorithms.

In the previous two lectures

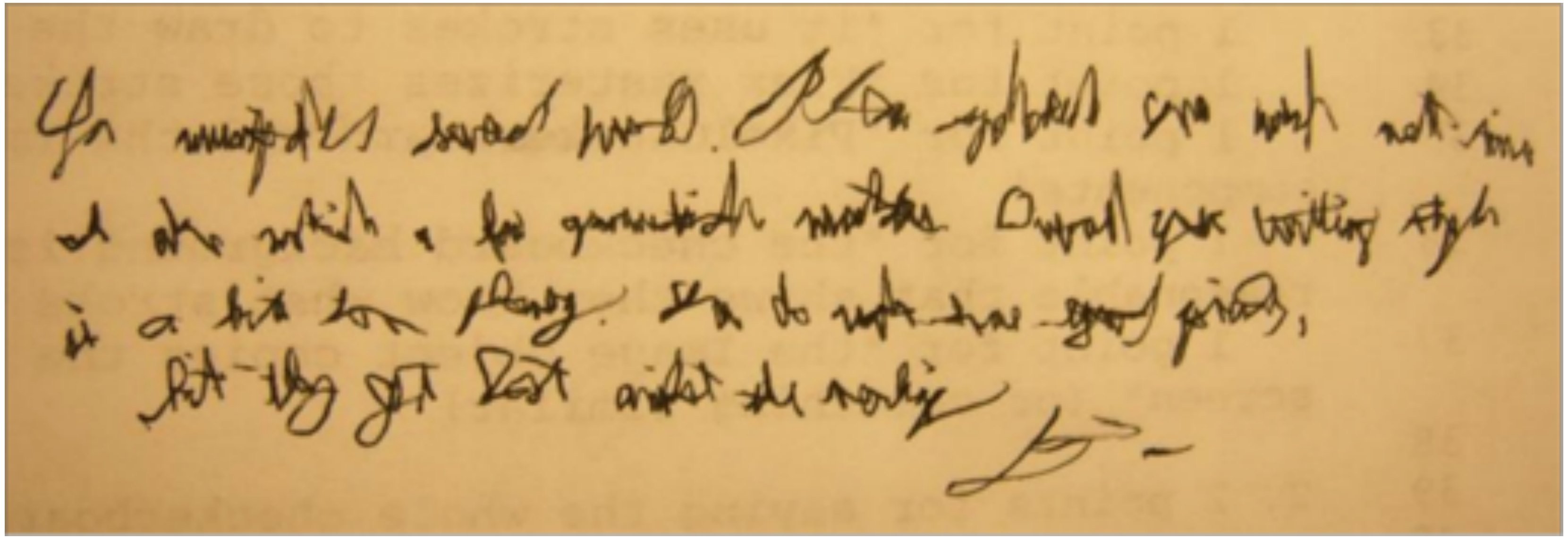
- Modeling incentives and analyzing outcomes (equilibrium analysis)
- Modeling label generation and inferring true labels
- Example:



Can we design a better **workflow**?

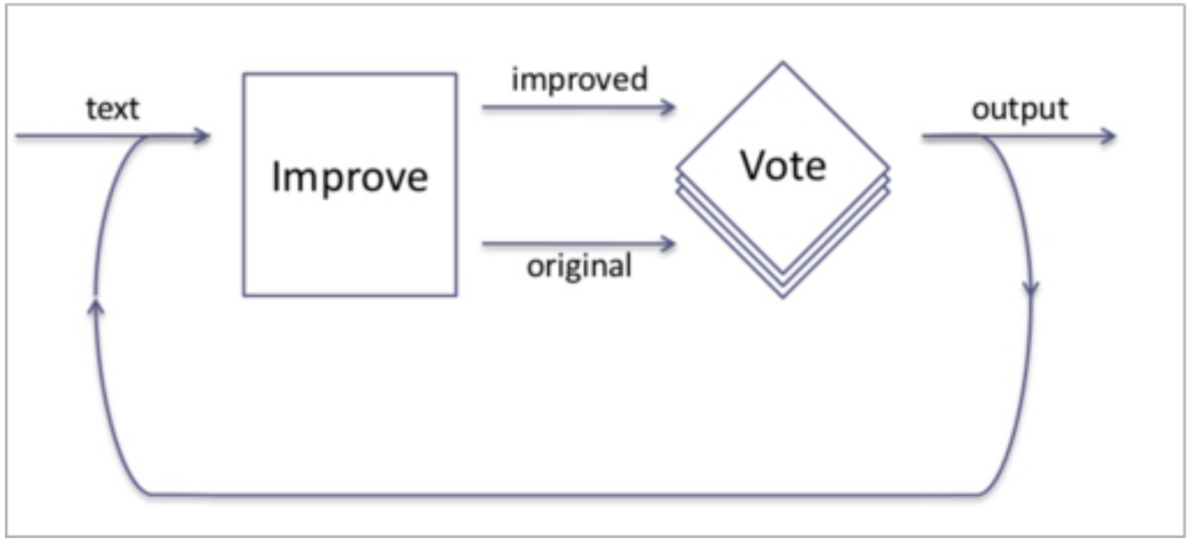
Can we **optimize** the process?

Consider the handwriting transcription task



Little et al. “TurKit: Tools for Iterative Tasks on Mechanical Turk”. HCOMP 2009

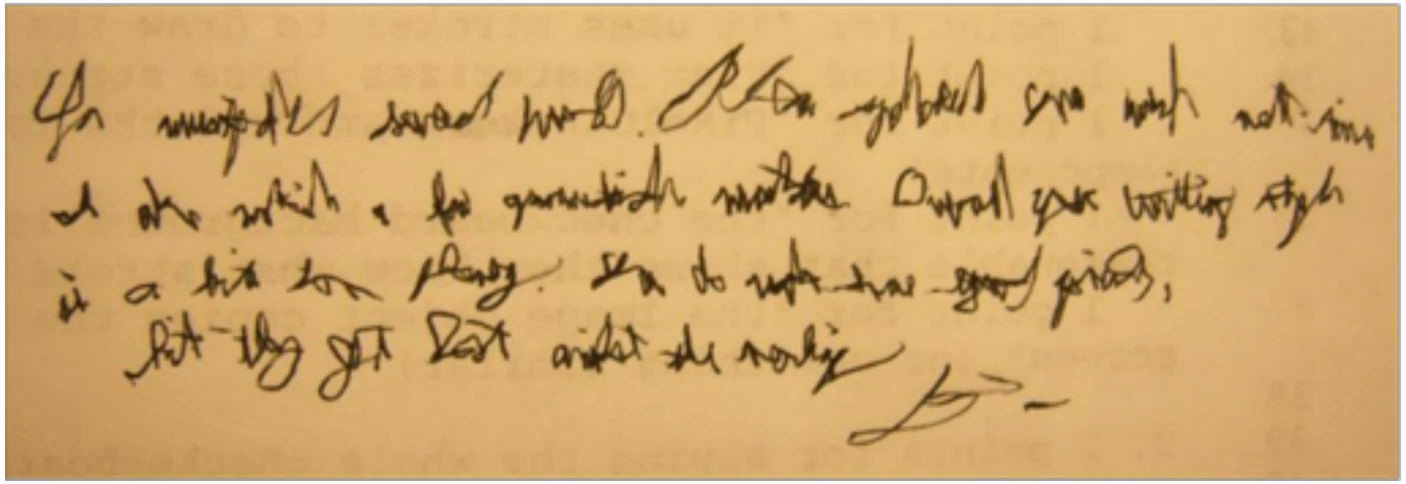
Designing Workflows



- Some workers are asked to perform **improvement** tasks
- Some workers are asked to **vote** on whether the improvement is good

Little et al. "TurKit: Tools for Iterative Tasks on Mechanical Turk". HCOMP 2009

Designing Workflows



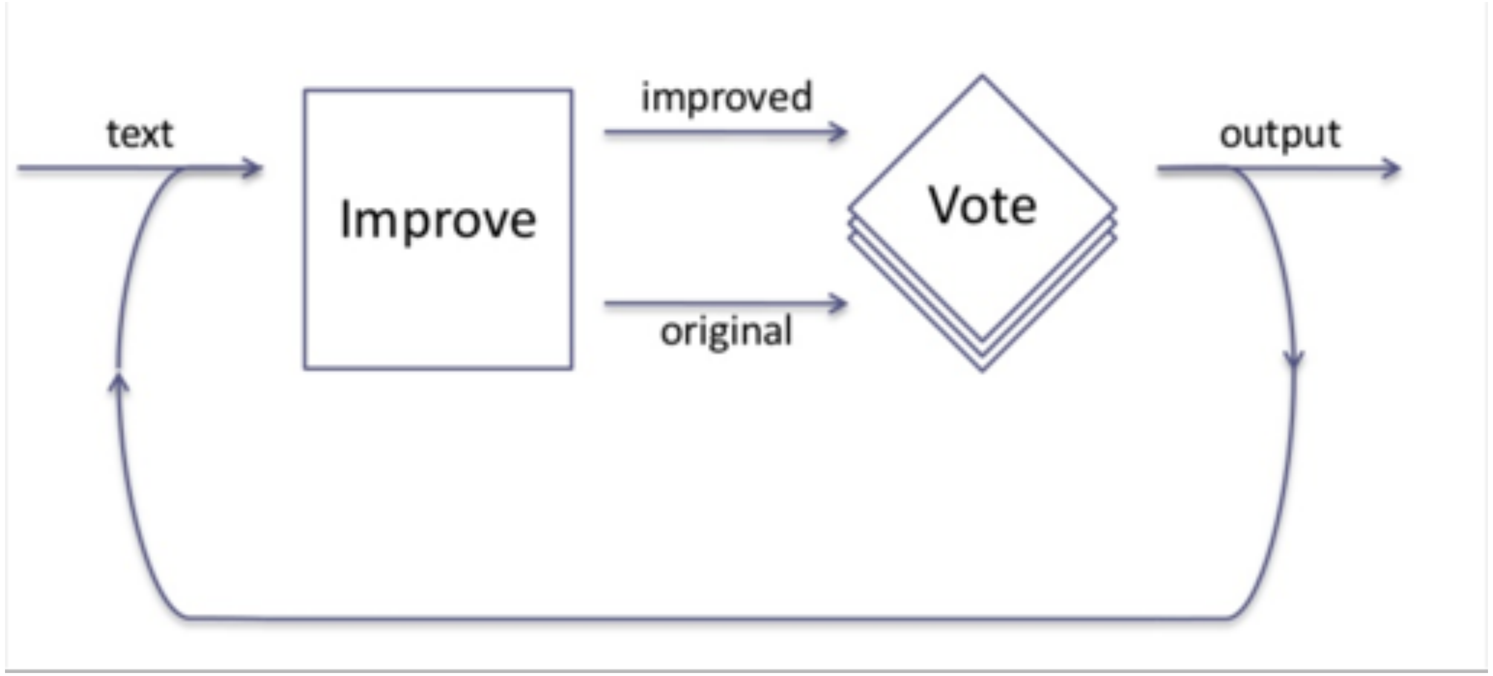
Version 6

You (misspelled) (several) (words). Please spellcheck your work next time. I also notice a few grammatical mistakes. Overall your writing style is a bit too phoney. You do make some good (points), but they got lost amidst the (writing). (signature)

Little et al. “TurKit: Tools for Iterative Tasks on Mechanical Turk”. HCOMP 2009

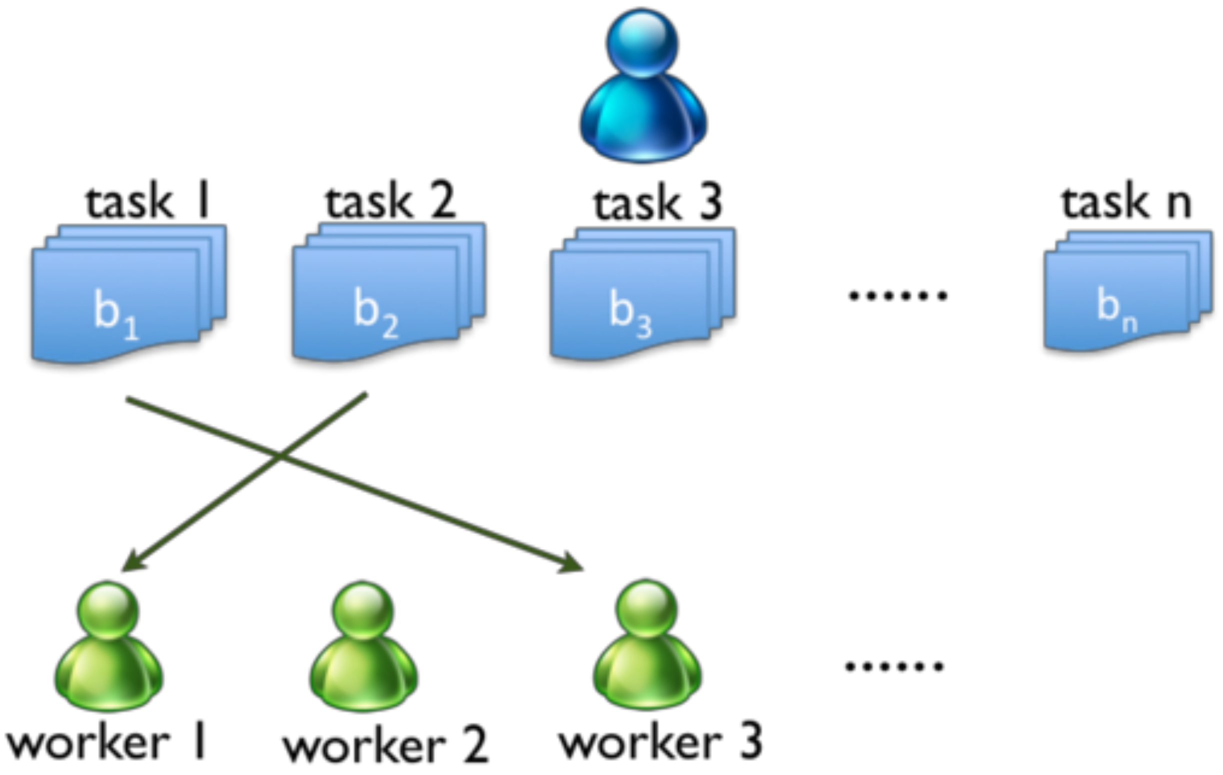
Optimizing Workflows

- How to optimize the process, e.g., how many votes to collect? How to determine if the improvement is better? When to end the process?
- Other types of workflows?



Optimizing Task Assignment / Recommendation

- Every worker might have different abilities for different tasks



Mar 7	Optimization: Task Assignment Presenter: Ben and Ricky	**Required** Online Task Assignment in Crowdsourcing Markets. Ho and Vaughan. AAAI 2012. **Optional** Pick-a-Crowd: Tell Me What You Like, and I'll Tell You What To Do. Difallah et al. WWW 2013. Using Hierarchical Skills for Optimized Task Assignment in Knowledge-Intensive Crowdsourcing. Mavridis et al. WWW 2016. Adaptive Task Assignment for Crowdsourced Classification. Ho, Jabbari, and Vaughan. ICML 2013.
Mar 12	**No Class: Spring Break**	
Mar 14	**No Class: Spring Break**	
Mar 19	Optimization: Workflow Design (Application-Specific) Presenter: Melena and Tonya	**Required** Soylent: A Word Processor with a Crowd Inside. Bernstein et al. UIST 2010. **Optional** Platemate: Crowdsourcing Nutritional Analysis from Food Photographs. Noronha. UIST 2011. Cascade: Crowdsourcing Taxonomy Creation. Chilton et al. CHI 2013. Crowdsourcing Step-by-Step Information Extraction to Enhance Existing How-to Videos. Kim et al. CHI 2014.
Mar 21	Optimization: Workflow Design (More General Purpose) Presenter: CJ	**Required** TurKit: Human Computation Algorithms on Mechanical Turk. Little et al. UIST 2010. **Optional** CrowdForge: Crowdsourcing Complex Work. Kittur et al. UIST 2011. Collaboratively Crowdsourcing Workflows with Turkomatic. Kulkarni et al. CSCW 2012.
Mar 26	Optimization: AI-Assisted Workflow Design Presenter: Peiyun and John	**Required** Decision-theoretic Control of Crowd-sourced Workflows. Dai et al. AAAI 2010. (Extended version at AI Journal 2013.) **Optional** Dynamically Switching between Synergistic Workflows for Crowdsourcing. Lin et al. AAAI 2012. Crowdsourcing Multi-Label Classification for Taxonomy Creation. Bragg et al. HCOMP 2013. Crowdsourcing Complex Workflows under Budget Constraints. Tran-Thanh et al. AAAI 2015.

Today's Lecture

- Optimization frameworks and connection to crowdsourcing
  - Assignment problem and convex optimization
  - Multi-arm bandit problem
  - Markov decision process

- Focus more on **formulations** instead of algorithms.

Assignment Problem and Convex Optimization

Formulations of an Assignment Problem

- An example setting
  - N tasks, indexed by $i = 1, \dots, N$
  - M workers, indexed by $j = 1, \dots, M$
  - $u_{i,j}$: utility to the requester when worker j completes task i
  - $c_{i,j}$: cost the requester needs to pay to worker j to work on task i
  - B: the amount of budget available to the requester
  - Each task can only be done once.

$$
\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{N} \sum_{j=1}^{M} u_{i,j} x_{i,j} \\
\text{subject to} \quad & \sum_{j=1}^{N} x_{i,j} \leq 1, \forall i \\
& \sum_{i=1}^{M} c_{i,j} x_{i,j} \leq B \\
& x_{i,j} \in \{0,1\}, \forall (i,j)
\end{aligned}
$$

- Goal:
  - Maximize total utility and satisfy the budget constraint
  - Define $x_{i,j} \in \{0,1\}$ as the assignment variable

Formulation of an Assignment Problem

Integer Program

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{N} \sum_{j=1}^{M} u_{i,j} x_{i,j} \\ \text{subject to} \quad & \sum_{j=1}^{N} x_{i,j} \le 1, \forall i \\ & \sum_{i=1}^{M} c_{i,j} x_{i,j} \le B \\ & \boxed{x_{i,j} \in \{0, 1\}, \forall (i, j)} \end{aligned}$$

Generally a NP hard problem

Relaxation

Linear Program

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{N} \sum_{j=1}^{M} u_{i,j} x_{i,j} \\ \text{subject to} \quad & \sum_{j=1}^{N} x_{i,j} \le 1, \forall i \\ & \sum_{i=1}^{M} c_{i,j} x_{i,j} \le B \\ & \boxed{0 \le x_{i,j} \le 1, \forall (i, j)} \end{aligned}$$

There exist efficient algorithms

Rounding or other methods
to obtain integer solutions

General Constrained Optimization Problem

Objective

$$
\begin{aligned}
&min_{\vec{w}} && f(\vec{w}) \\
&\text{subject to} && g_i(\vec{w}) \leq 0, \forall i \\
& && h_j(\vec{w}) = 0, \forall j
\end{aligned}
$$

Constraints

- When f, g_i, h_j are all linear, this is a **linear program**
  - There exists efficient algorithms in solving them
- When f and g_i are convex and h_i is linear, this is a **convex program**
  - There also exists efficient algorithms

A Useful Property of Convex Optimization

- For every **primal** convex program, there exists a **dual** program.

Primal

$$\begin{array}{ll} \max & \Sigma_{t=1}^{m} \Sigma_{i=1}^{n} u_{i,j(t)} y_{i,t} \\ \text{s.t.} & \Sigma_{i=1}^{n} y_{i,t} \leq 1, \forall t \\ & \Sigma_{t=1}^{m} y_{i,t} \leq b_i, \forall i \\ & y_{i,t} \geq 0, \forall (i,t) \end{array}$$

$$\begin{array}{ll} \min & \Sigma_{i=1}^{n} b_i x_i + \Sigma_{t=1}^{m} z_t \\ \text{s.t.} & x_i + z_t \geq u_{i,j(t)}, \forall (i,t) \\ & x_i, z_t \geq 0, \forall (i,t) \end{array}$$

Dual

- Strong Duality:
  - **Primal optima = Dual optimal**
- KKT Conditions:
  - **Connections between primal optimal and dual optimal.**

Why the Dual is Useful?

- Provide new insights/interpretations to the original problem
  - Each variable in the dual corresponding to a constraint in the primal
  - Shadow prices in economics
- Provide solutions to seemly unsolvable primal problems
  - Duality is the key to enable kernel tricks in Support Vector Machines (SVM)
  - Deciding assignment online without known the future (More on March 7)

Deriving the Dual from the Primal

- Lagrangian method
  - See Section 5 of Andrew Ng's lecture notes for more details

A Great Amount of Applications

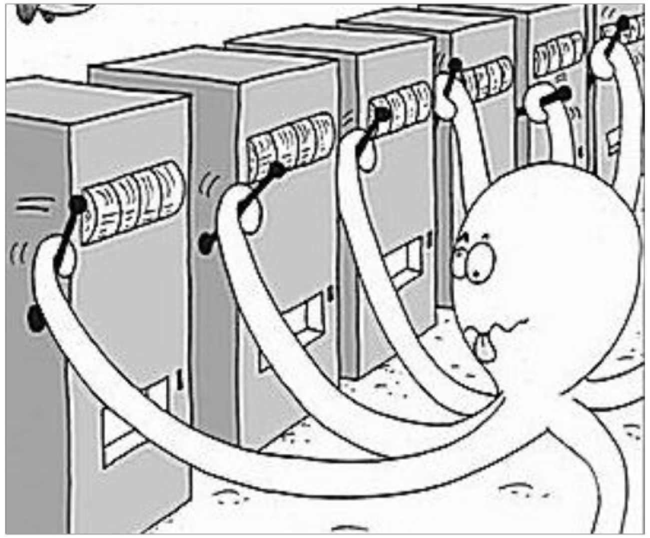
- A great amount of literature in the static setting
  - Matching students to schools
  - Matching kidneys to patients
  - …

- New challenges arrive in online environments
  - Assigning tasks to workers
  - Matching Uber drivers to passengers
  - Matching people on dating website
  - And more…

Multi-Armed Bandit

Multi-Arm Bandit (MAB) Framework

- MAB is a decision making & learning framework
  - Make a sequence of decision on selection, when facing multiple options with unknown statistics.
  - **Q**: which one to select next
  - **Goal**: Maximize total payoff returned by the choice; or regret minimization



Regret:
Utility(OPT) - Utility(ALG)

When the number of "arms" is small, the problem is well-studied!

A classical framework for exploration-exploitation tradeoffs

Upper Confidence Bound (UCB)

- An index-based method for stochastic bandits
  - Maintain an index for each arm *k* at every time *t*
  - Select the arm with the largest index

$$I_k(t) = \bar{X}_k(t) + \sqrt{\frac{L \log t}{n_k(t)}}, \forall k.$$

Empirical mean: exploitation

Confidence interval: exploration

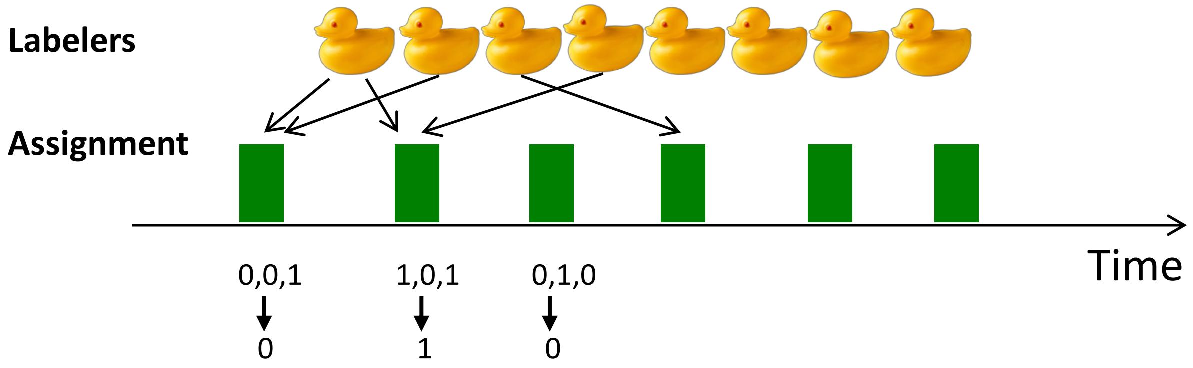
- UCB achieves regret bound *O(log T)* in stochastic settings!

Applying Bandit in Crowdsourcing

Task Assignment

- Choosing which worker to assign the task to
  - Each worker is an arm
- Choosing which task to assign to workers
  - Each task is an arm
- Choosing which task-worker pair to do the assignment
  - Each pair is an arm

- Challenges
  - Budget constraints
  - Unknown ground truth
    - Won’t obtain rewards after pulling arms



Choosing financial incentives

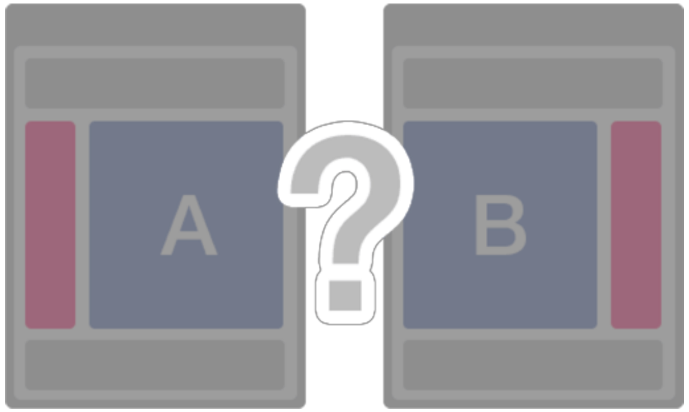
- How much should we offer to workers?
  - Each choice of (x,y) is an arm

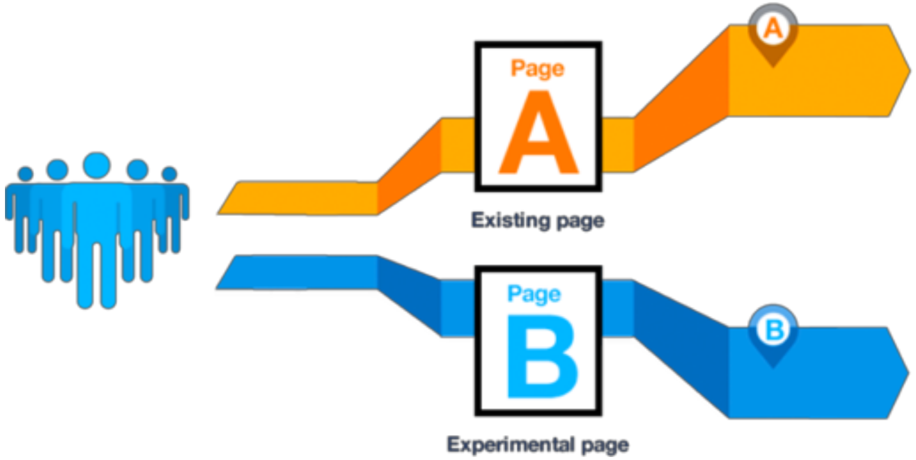
Bad Outcome	Good Outcome
\$x	\$y

- Challenges
  - Infinitely many possible arms

Choosing different layouts/designs

- Traditionally addressed using A/B testing





- Challenges when applying bandits
  - Need to be careful about interactions between different arms

Challenges in Applying Bandits in Crowdsourcing

- The number of arms could be large
  - “nearby” arms might have close rewards
  - Utilize the above structure to reduce explorations [Kleinberg et al. 2008]
- There are budget constraints
  - Using the Lagragian method to turn constrained optimization into unconstrainted
  - Infer the dual and primal simultaneously [Badanidiyuru et al. 2013]
- We may not be able to obtain direct “feedback” after pulling arms
  - “estimate” the rewards given the current knowledge [Abraham et al. 2013]
- Humans are involved and might break common assumptions


Made With
VivaVideo

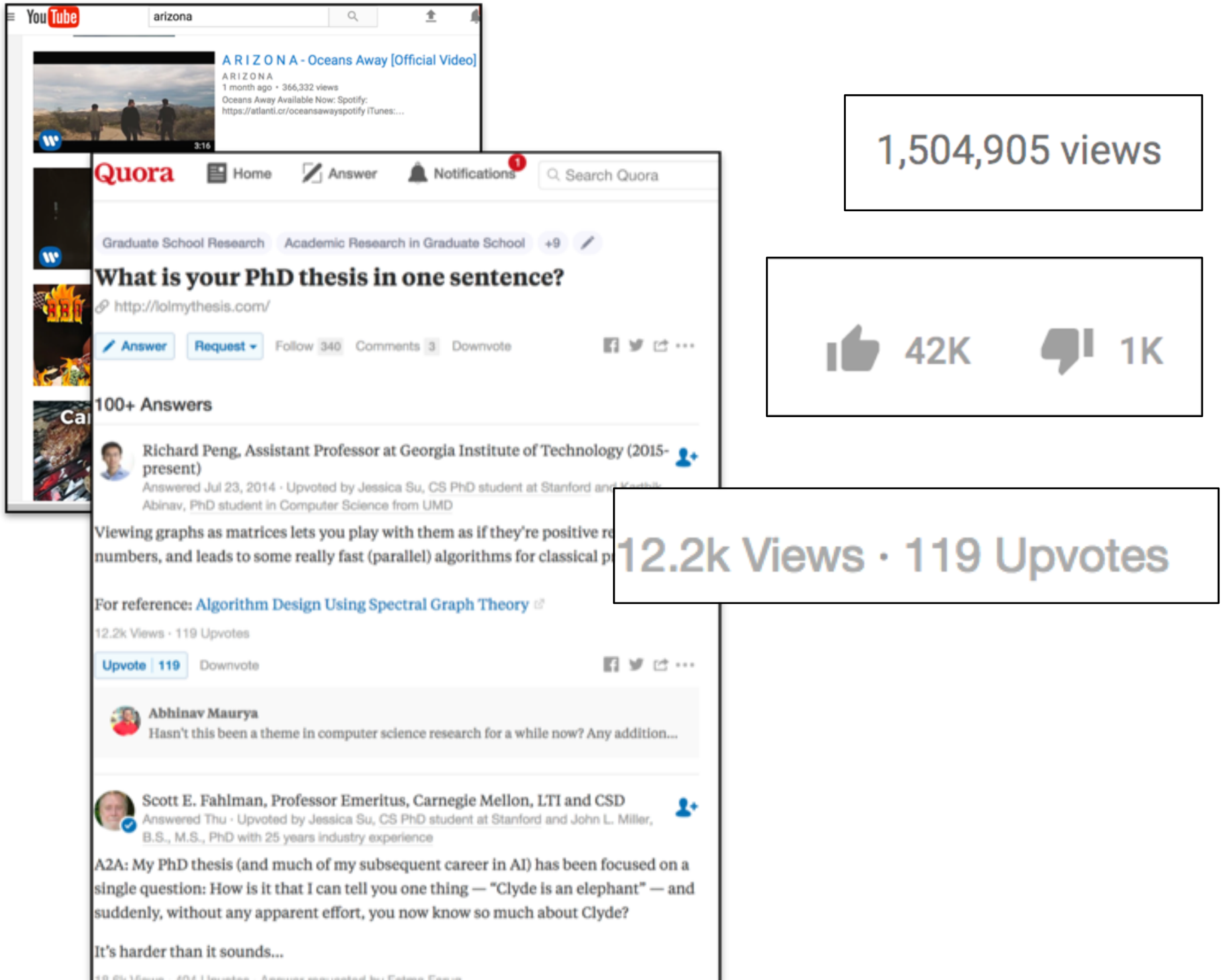
- This happens every day

Bandit Learning with Biased Feedback

joint work with Wei Tang

To appear in AAMAS 2019

User Generated Content Platforms



A Bandit Formulation

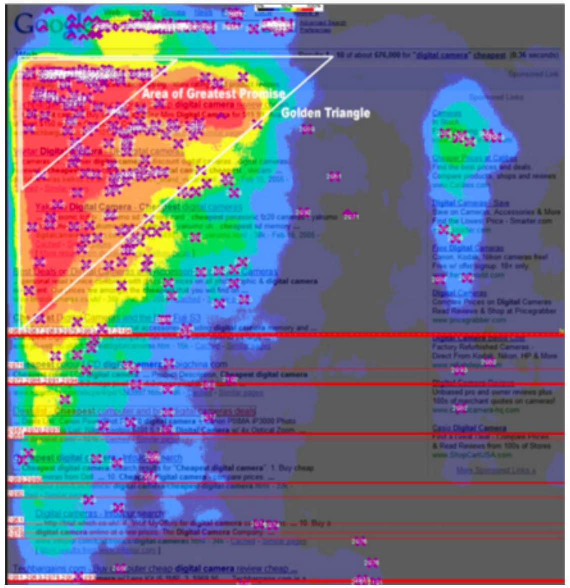
- When each new user arrives
  - Show the user some (set of) content
  - Obtain feedback (upvotes, likes, shares, etc) from the user

- Goal:
  - maximize the total number of positive feedback (user happiness)

Assume the user feedback is "unbiased". ?

Users' feedback might be biased

- Bias in selecting items
- Bias in voting
- and others…



Feedback Model 1

- Model
  - Users feedback depends on
    - their own experience
    - average feedback of others

- Positive results
  - Collectively, users are performing online gradient descent on a latent function.
  - **Sublinear regret is achievable** under mild conditions using techniques from online optimization.

Feedback Model 2

- Model
  - Users feedback depends on
    - their own experience
    - average feedback of others
    - length of the feedback history
- Impossibility results
  - The average feedback converges to a random variable with non-zero variance.
  - **No algorithm can achieve sublinear regrets.**

Markov Decision Process

Markov Decision Process (MDP)

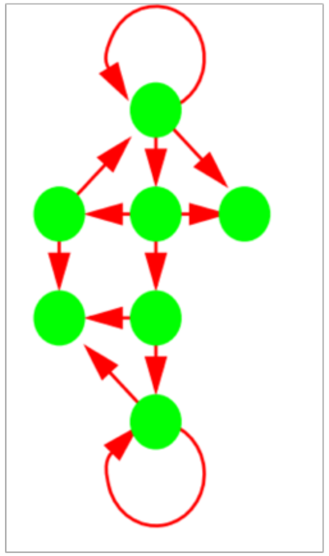
- Both bandits and Markov decision process are commonly discussed frameworks in *reinforcement learning*.

- In standard bandits, the reward depends only on the actions we take. Focus on the **exploration-exploitation** tradeoff.

- However, in some scenarios, the reward could depending on both the action and the current ***state***. Markov decision process incorporates the ideas of states in the model.

Markov Decision Process

- Formally, a Markov decision process model contains
  - A set of possible state S
  - A set of possible actions A
  - A real-valued reward function $R(s, a)$ for $s \in S, a \in A$
  - A transition function $T_a(s, s')$ which specifies the probability of ending in state s' from state s after taking action a



- Assumption of **Markov property**:
  - The effect of an action taking in a state only depends on that state and not on the prior history.

How to Define Objective (Utility)

- In Markov decision process, we take a (possibly infinite) series of actions. How should we define the total utility we obtain?

- One common choice: Time-discounted utility
  - There is a positive probability you will leave the system at every time step.
  - We discount future rewards to account for this.

  - Say u_t is the reward we expect to receive at time t. For $\delta \in (0,1)$

$$U = \sum_{t=1}^{\infty} \delta^t u_t$$

Solving the Optimal Policy

- Policy $\pi: S \to A$ specifies that actions to take at each state.
  - We consider the deterministic action here to simplify the expression
- Value function $V_\pi: S \to \mathbb{R}$ is the value of using policy π starting at s
- Bellmen equation

$$V_\pi(s) = R(s, \pi(s)) + \delta \sum_{s' \in S} T_{\pi(s)}(s, s') V_\pi(s')$$

$$V_{\pi^*}(s) = \mathrm{argmax}_a \left\{ R(s, \pi(s)) + \delta \sum_{s' \in S} T_{\pi(s)}(s, s') V_\pi(s') \right\}$$

Solvable using dynamic-programming style algorithms (value iteration) or linear program.

Partially Observable MDP (POMDP)

- In MDP, we assume we know the state we are in. However, sometimes states are not known, and we only have noisy observations about the state.

- Formally, a POMDP contains
  - A set of possible state S
  - A set of possible actions A
  - A real-valued reward function $R(s, a)$ for $s \in S, a \in A$
  - A transition function $T_a(s, s')$ which specifies the probability of ending in state s' from state s after taking action a
  - A set of observations O
  - A conditional observation probability $P(o|s)$ for all o $\in O, s \in S$

Application to Crowdsourcing

- Task assignment
  - State: workers’ answers we obtained so far
  - Action: which worker to recruit or which task to assign
  - Reward: the value of workers’ answers
  - Transition matrix: defined by the action

- And many others…

References

- The extended version of the required reading on Mar 26 provides explanations on MDPs, POMDPs, and their applications to crowdsourcing.