Microsoft Research

Microsoft

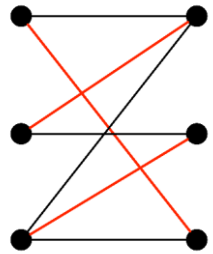
Combinatorial Online Learning

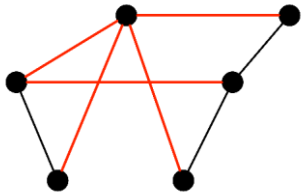
for Combinatorial Optimizations

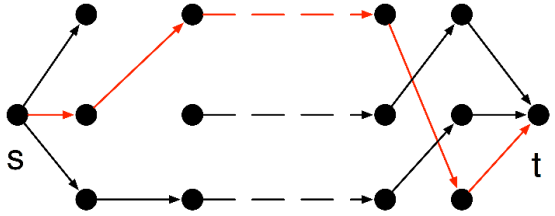
Wei Chen 陈卫
Microsoft Research Asia

Combinatorial optimization

- Well studied
  - classics: max. matchings, shortest paths, min. spanning trees,
  - modern applications: online advertising, viral marketing
- What if the inputs are stochastic, unknown, and has to be learned over time?
  - link delays
  - click-through probabilities
  - influence probabilities in social networks





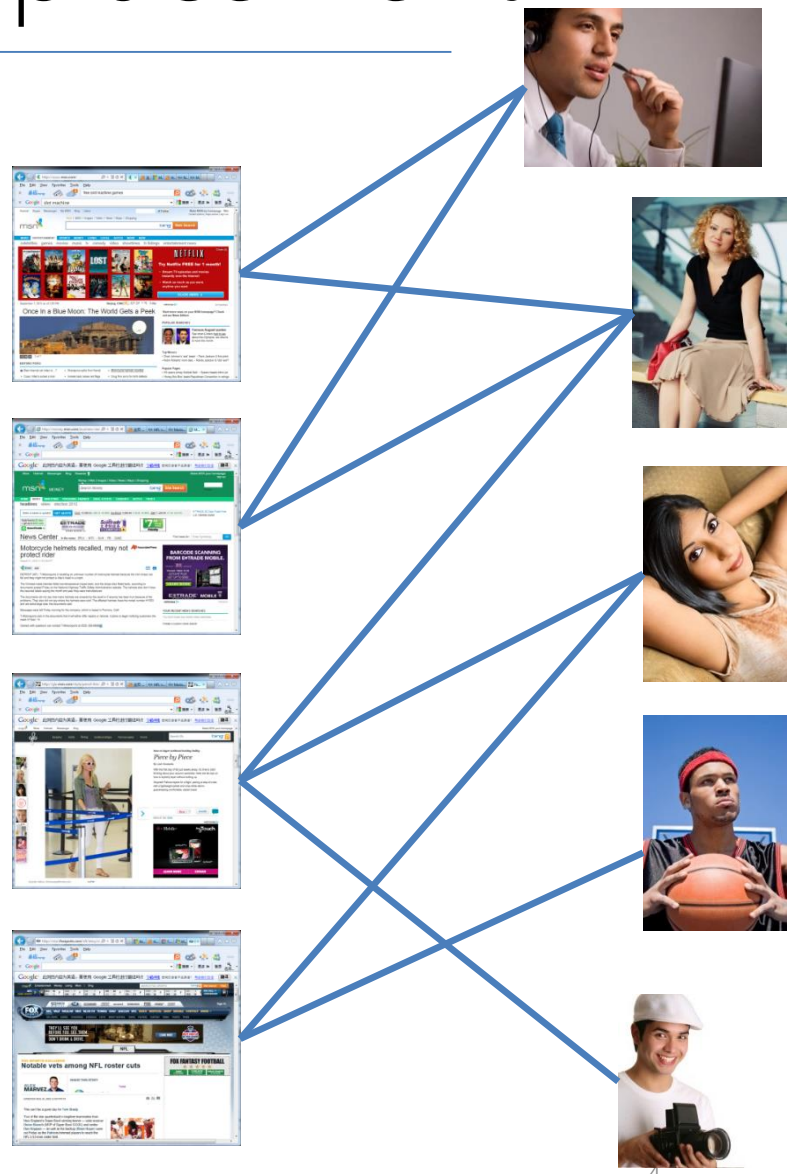


Combinatorial online learning for combinatorial optimizations

- Need new framework for learning and optimization:
- Learn inputs while doing optimization --- combinatorial online learning
- Learning inputs first (and fast) for subsequent optimization --- combinatorial pure exploration

Motivating application: Display ad placement

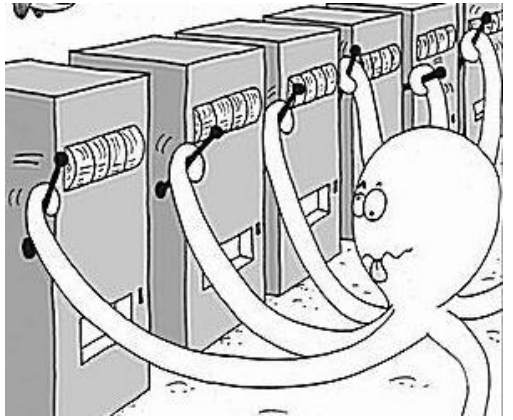
- Bipartite graph of pages and users who are interested in certain pages
  - Each edge has a click-through probability
- Find k pages to put ads to maximize total number of users clicking through the ad
- When click-through probabilities are known, can be solved by approximation
- Question: how to learn click-through prob. while doing optimization?



Multi-armed bandit: the canonical OL problem

- There are $\boldsymbol{m}$ arms (machines)
- Arm $\boldsymbol{i}$ has an unknown reward distribution with unknown mean $\boldsymbol{\mu_i}$
  - best arm $\boldsymbol{\mu^* = \max \mu_i}$
- In each round, the player selects one arm to play and observes the reward

Multi-armed bandit problem



- Regret after playing T rounds:
  - Regret $= T\mu^* - \mathbb{E}[\sum_{t=1}^{T} R_t(i_t^A)\ ]$
- Objective: minimize regret in T rounds
- Balancing exploitation-exploration tradeoff
- Known results:
  - UCB1 (Upper Confidence Bound) [Auer, Cesa-Bianchi, Fischer 2002]
    - Gap-dependent bound $\mathbf{O}(\log T \sum_{i:\Delta_i>0} 1/\Delta_i)$, $\Delta_i = \mu^* - \mu_i$, match lower bound
    - Gap-free bound $\mathbf{O}(\sqrt{mT \log T})$, tight up to a factor of $\sqrt{\log T}$

Naïve application of MAB

- Every set of k webpages is treated as an arm
- Reward of an arm is the total click-through counted by the number of people
- Issues
  - combinatorial explosion
  - ad-user click-through information is wasted

Issues when applying MAB to combinatorial setting

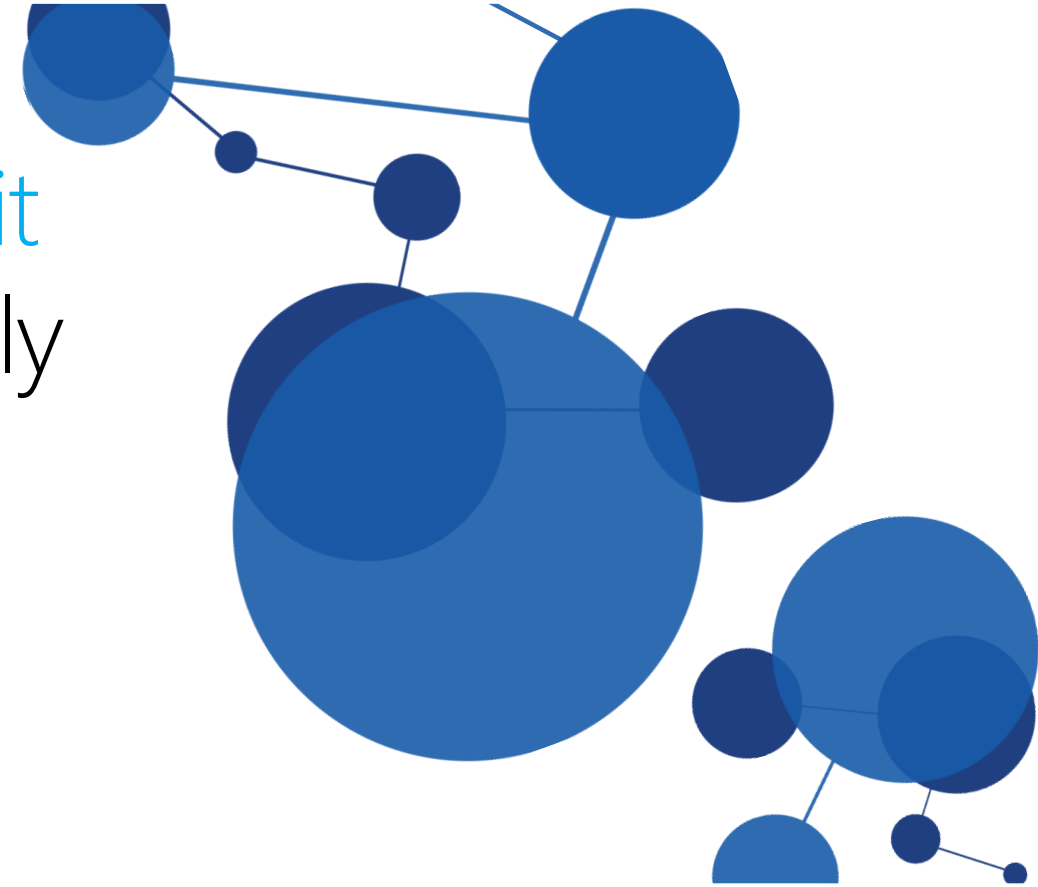
- The action space is exponential
  - Cannot even try each action once
- The offline optimization problem may already be hard
- The reward of a combinatorial action may not be linear on its components
- The reward may depend not only on the means of its component rewards

Combinatorial Online Learning

- General framework
  - Treat offline optimization as an oracle
  - Characterize general learning conditions
  - Focus on online tradeoff between exploration and exploitation
- My existing work
  - ICML'13/JMLR'16: general combinatorial multi-armed bandit (CMAB) framework, apply to non-linear rewards, approximation oracle, probabilistically triggered arms
  - ICML'14: combinatorial partial monitoring
  - NIPS'14: combinatorial pure exploration
  - NIPS'15: online greedy learning
  - ICML'16: contextual combinatorial cascading bandits
  - NIPS'16: CMAB with general reward functions

Microsoft

Combinatorial Multi-Armed Bandit and Its Extension to Probabilistically Triggered Arms

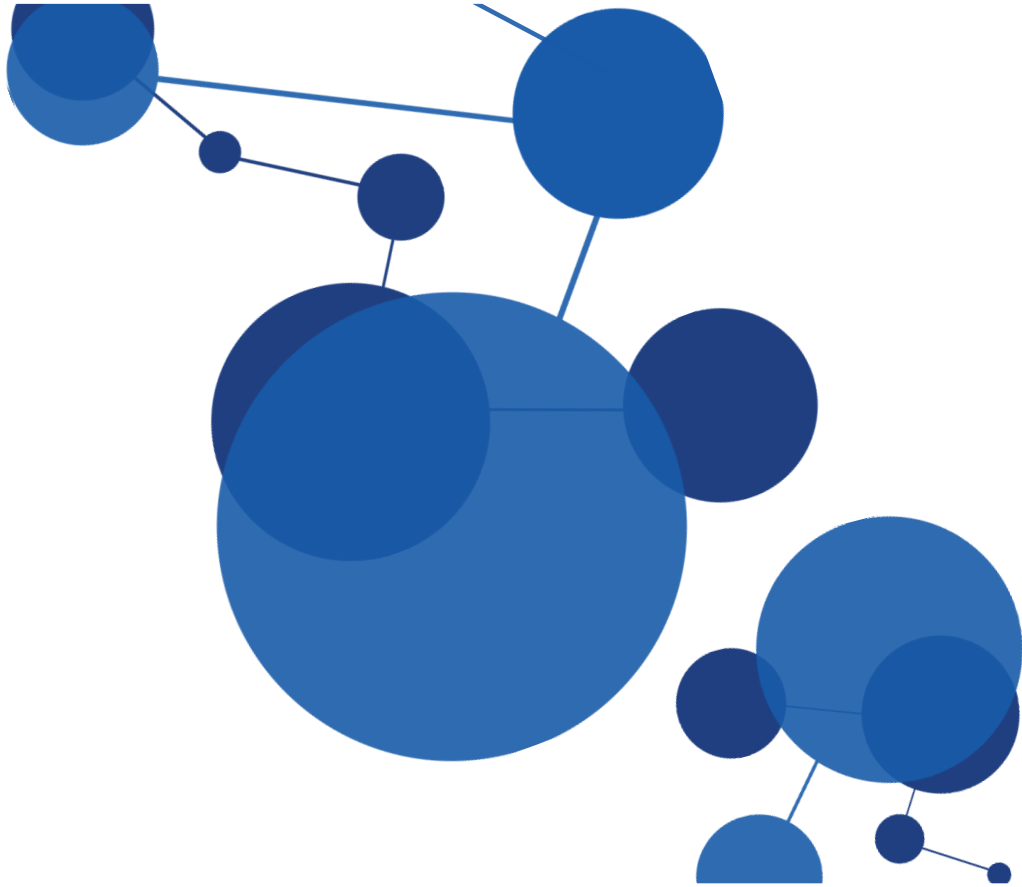


ICML'2013/JMLR'2016, joint work with
Yajun Wang, Microsoft
Yang Yuan, Cornell U.
Qinshi Wang, Tsinghua U.

Contribution of this work

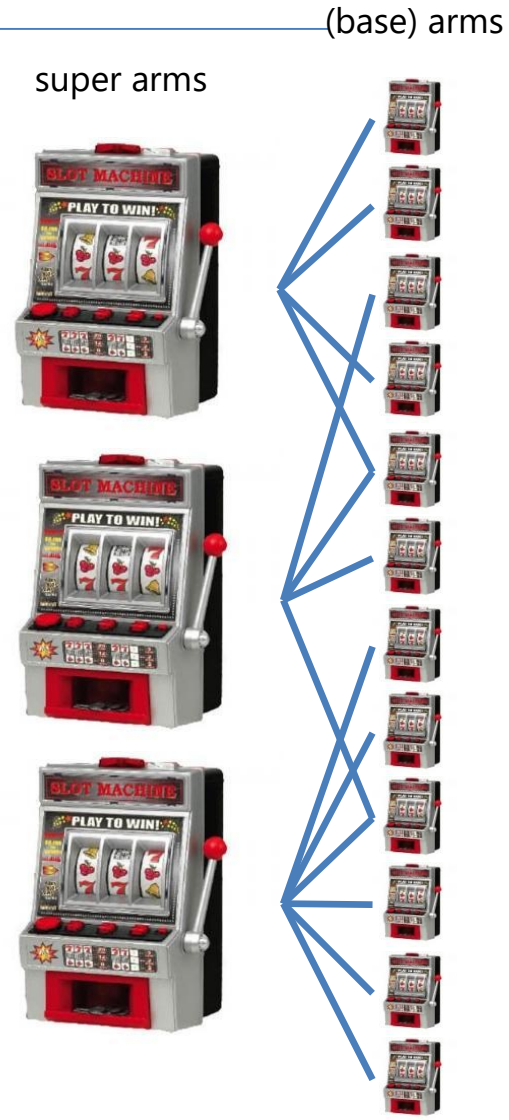
- Stochastic combinatorial multi-armed bandit framework
  - handling non-linear reward functions
  - UCB based algorithm and tight regret analysis
  - extend to probabilistically triggered arms
  - new applications using CMAB framework
- Comparing with related work
  - linear stochastic bandits [Gai et al. 2012]
    - CMAB is more general, and has much tighter regret analysis
  - online submodular optimizations (e.g. [Streeter& Golovin'08, Hazan&Kale'12])
    - for adversarial case, different approach
    - CMAB has no submodularity requirement

CMAB Framework



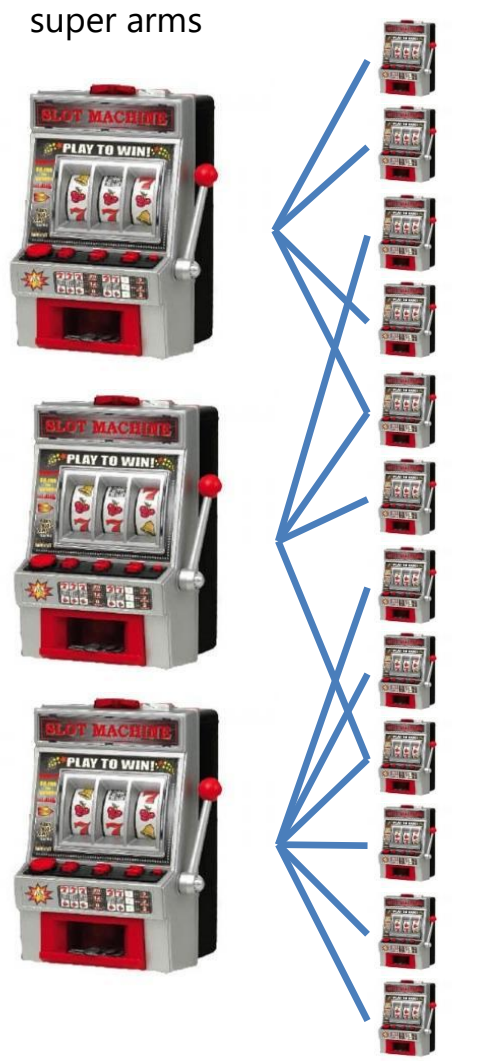
Combinatorial multi-armed bandit (CMAB) framework

- A super arm S is a set of (base) arms, $S \subseteq [m]$
- In round t, a super arm S_t^A is played according algo A
- When a super arm S is played, all based arms in S are played
- Outcomes of all played base arms are observed --- semi-bandit feedback
- Outcome of arm $i \in [m]$ has an unknown distribution with unknown mean μ_i



Rewards in CMAB

- Reward of super arm S_t^A played in round t, $R_t(S_t^A)$, is a function of the outcomes of all played arms
- Expected reward of playing arm S, $\mathbb{E}[R_t(S)]$, only depends on S and the vector of mean outcomes of arms, $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_m)$, denoted $r_{\boldsymbol{\mu}}(S)$
  - e.g. linear rewards, or independent Bernoulli random variables
  - generalization to be discussed later
- Optimal reward: $\mathbf{opt}_{\boldsymbol{\mu}} = \max_S r_{\boldsymbol{\mu}}(S)$



Handling non-linear reward functions --- two mild assumption on $r_{\boldsymbol{\mu}}(S)$

- Monotonicity
  - if $\boldsymbol{\mu} \leq \boldsymbol{\mu}'$ (pairwise), $r_{\boldsymbol{\mu}}(S) \leq r_{\boldsymbol{\mu}'}(S)$, for all super arm S
- Bounded smoothness
  - there exists a strictly increasing function $f(\cdot)$, such that for any two expectation vectors $\boldsymbol{\mu}$ and $\boldsymbol{\mu}'$,
    $|r_{\boldsymbol{\mu}}(S) - r_{\boldsymbol{\mu}'}(S)| \leq f(\Delta)$, where $\Delta = \max_{i \in S} |\mu_i - \mu_i'|$
  - Small change in $\boldsymbol{\mu}$ lead to small changes in $r_{\boldsymbol{\mu}}(S)$
    - A general version of Lipschitz continuity condition
- Rewards may not be linear, a large class of functions satisfy these assumptions

Offline computation oracle --- allow approximations and failure probabilities

- (α, β)-approximation oracle:
  - Input: vector of mean outcomes of all arms $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_n)$,
  - Output: a super arm S, such that with probability at least β the expected reward of S under $\boldsymbol{\mu}$, $r_{\mu}(S)$, is at least α fraction of the optimal reward:

$$\Pr\left[r_{\boldsymbol{\mu}}(S) \geq \alpha \cdot \mathrm{opt}_{\boldsymbol{\mu}}\right] \geq \beta$$

(α, β)-Approximation regret

- Compare against the $\alpha\beta$ fraction of the optimal

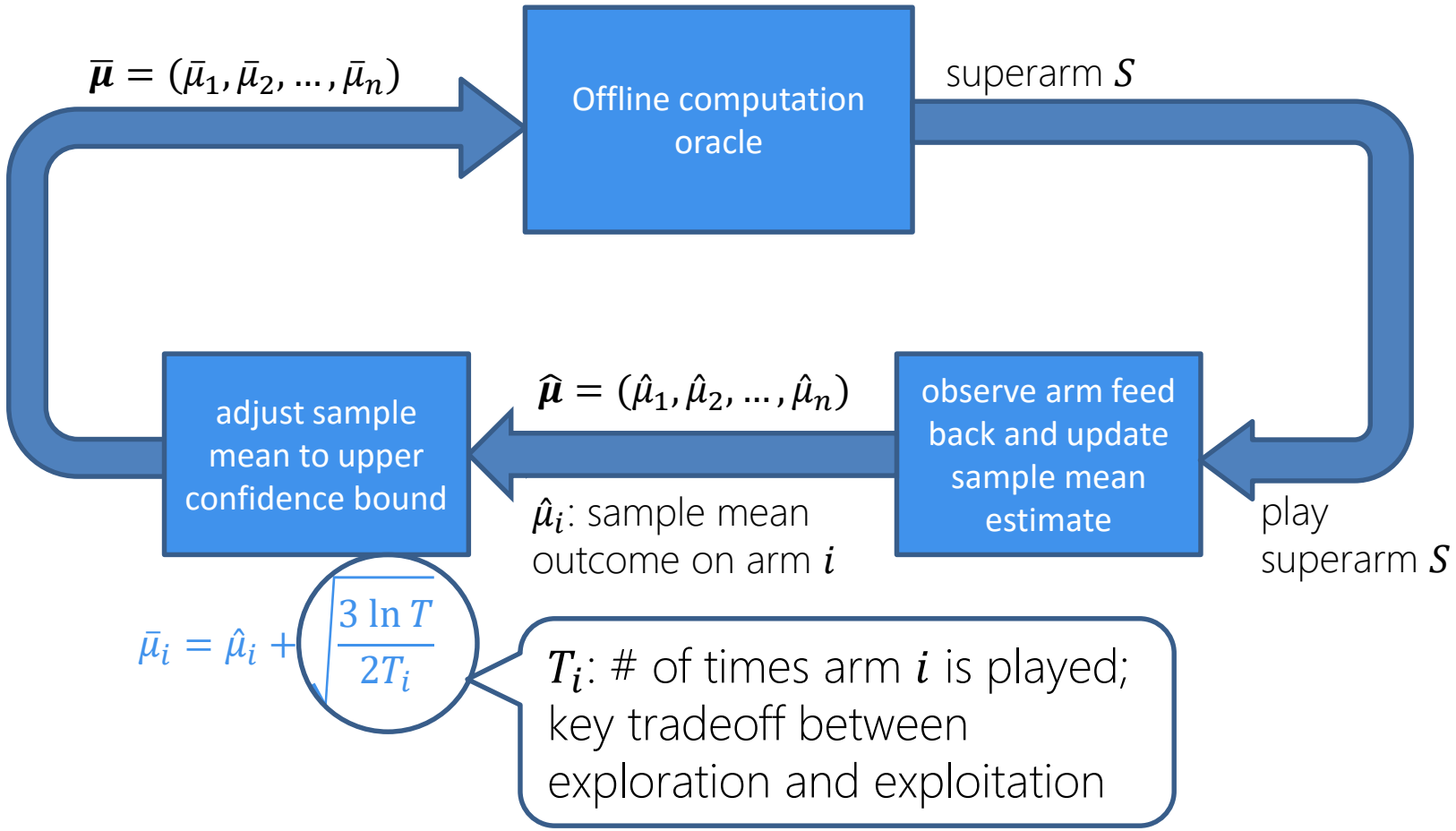
$$\text{Regret} = T \cdot \alpha\beta \cdot \mathbf{opt}_{\boldsymbol{\mu}} - \mathbb{E}[\sum_{i=1}^{T} r_{\boldsymbol{\mu}}(S_t^A)]$$

- Difficulty:
  - do not know arm outcome distribution
  - Oracle treatment: ignore
    - combinatorial structure
    - reward function
    - how oracle computes the solution

Classical MAB as a special case

- Each super arm is a singleton
- Oracle is taking the max, $\alpha = \beta = 1$
- Bounded smoothness function $f(x) = x$

Our solution: CUCB algorithm



Theorem 1: Gap-dependent bound

- The (α, β)-approximation regret of the CUCB algorithm in n rounds using an (α, β)-approximation oracle is at most

$$\sum_{i\in[n],\Delta_{\min}^{i}>0}\left(\frac{6\ln T\cdot\Delta_{\min}^{i}}{(f^{-1}(\Delta_{\min}^{i}))^{2}}+\int_{\Delta_{\min}^{i}}^{\Delta_{\max}^{i}}\frac{6\ln T}{(f^{-1}(x))^{2}}\mathrm{d}x\right)+\left(\frac{\pi^{2}}{3}+1\right)\cdot n\cdot\Delta_{\max}$$

  - $\Delta_{\min}^{i}$ ($\Delta_{\max}^{i}$) are defined as the minimum (maximum) gap between $\alpha\cdot\mathrm{opt}_{\mu}$ and reward of a bad super arm containing i.
    - $\Delta_{\min}=\min_{i}\Delta_{\min}^{i}$, $\Delta_{\max}=\max_{i}\Delta_{\max}^{i}$
    - Here, we define the set of bad super arms as $\mathcal{S}_{\mathrm{B}}=\{S\mid r_{\mu}(S)<\alpha\cdot\mathrm{opt}_{\mu}\}$
- When $f(x)=\gamma\cdot x$, regret bound: $O\left(\sum_{i}\frac{\gamma^{2}\ln T}{\Delta_{\min}^{i}}\right)$
- Match UCB regret for classic MAB

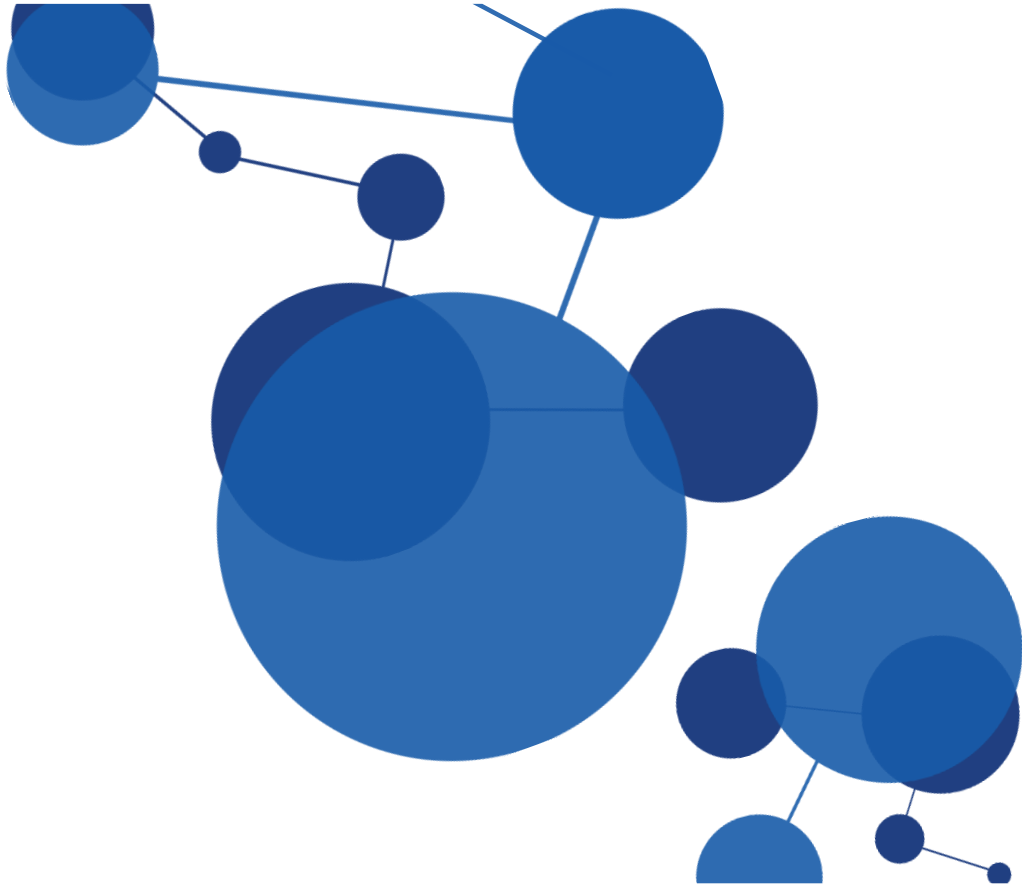
Theorem 2: Gap-free bound

- Consider a CMAB problem with an (α, β)-approximation oracle. If the bounded smoothness function $f(x) = \gamma \cdot x^{\omega}$ for some $\gamma > 0$ and $\omega \in (0,1]$, the regret of CUCB is at most:

$$\frac{2\gamma}{2-\omega} \cdot (6n \ln T)^{\frac{\omega}{2}} \cdot T^{1-\frac{\omega}{2}} + \left(\frac{\pi^2}{3} + 1\right) \cdot n \cdot \Delta_{\max}$$

- When $\omega = 1$, the gap-free bound is $O(\gamma\sqrt{nT \ln T})$

Applications of CMAB



Application to ad placement

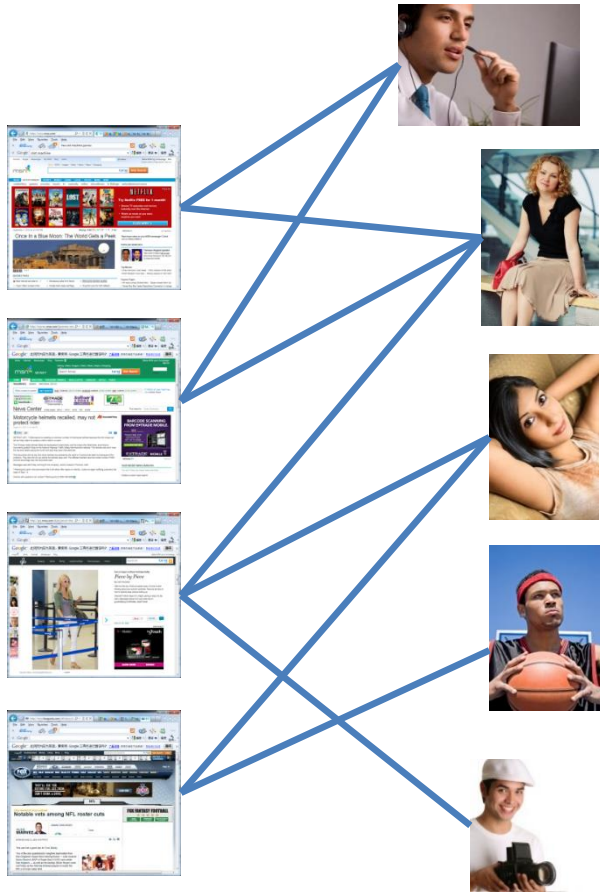
- Bipartite graph $G = (L, R, E)$
- Each edge is a base arm
- Each set of edges linking k webpages is a super arm
- Bounded smoothness function

$$f(\Delta) = |E| \cdot \Delta$$

- $(1 - {}^{1}/_{e}, 1)$-approximation regret

$$\sum_{i \in E, \Delta_{\min}^{i} > 0} \frac{12|E|^2 \ln T}{\Delta_{\min}^{i}} + \left(\frac{\pi^2}{3} + 1\right) \cdot |E| \cdot \Delta_{\max}$$

- improvement based on clustered arms is available



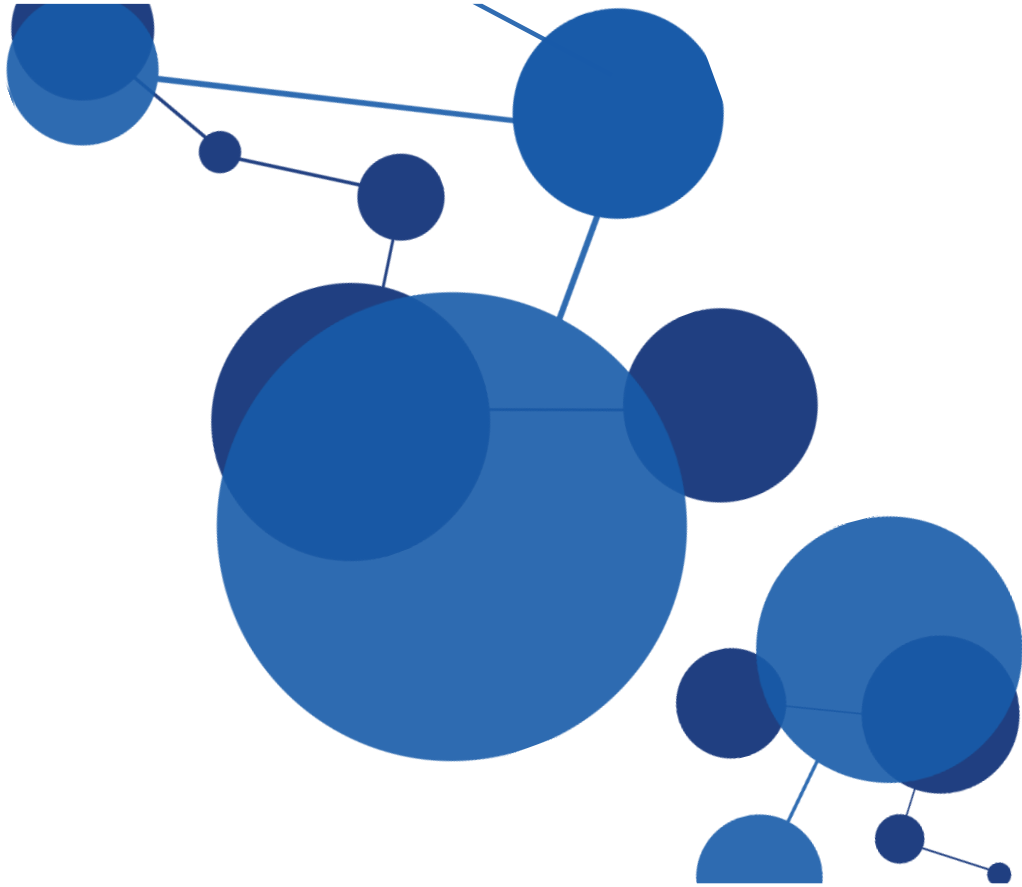
Application to linear bandit problems

- Linear bandits: matching, shortest path, spanning tree (in networking literature)
- Maximize weighted sum of rewards on all arms
- Our result significantly improves the previous regret bound on linear rewards [Gai et al. 2012]
  - Also provide gap-free bound

Application to social influence maximization

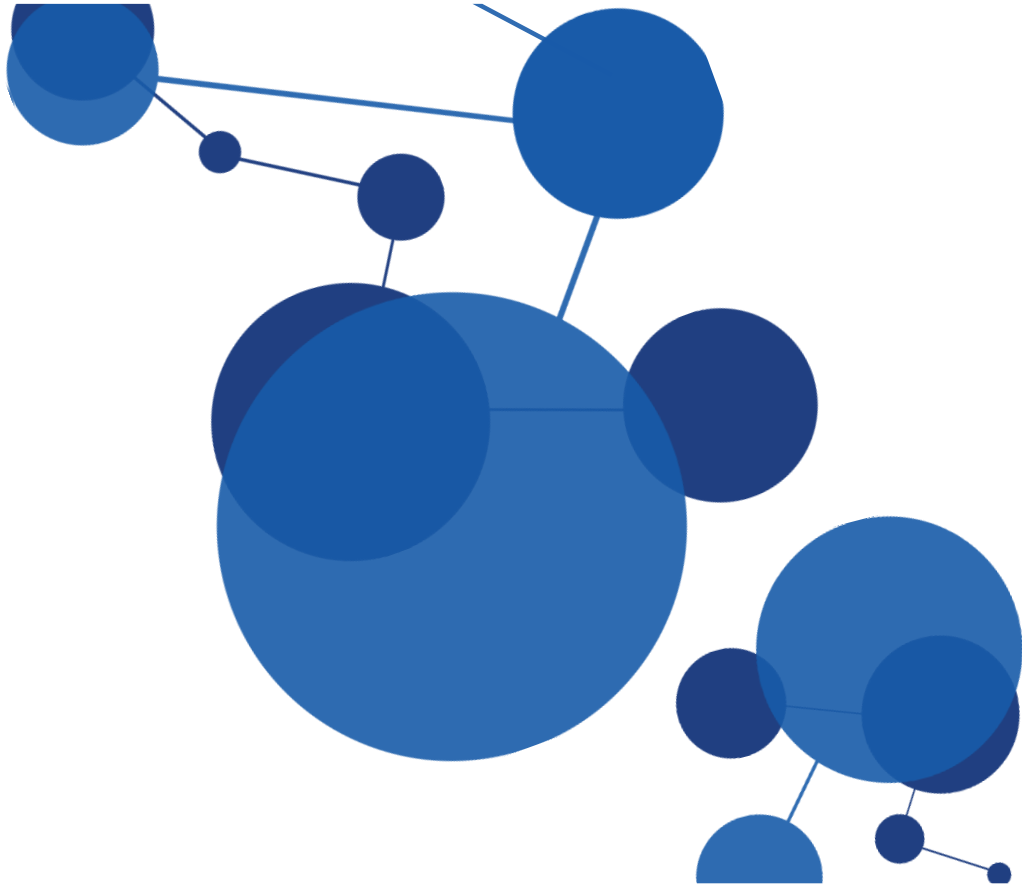
- Each edge is a base arm
- Require a new model extension to allow probabilistically triggered arms
  - Because a played base arm may trigger more base arms to be played --- the cascade effect
- Use the same CUCB algorithm
- Included in the journal version, JMLR'2016

Extensions and Variants



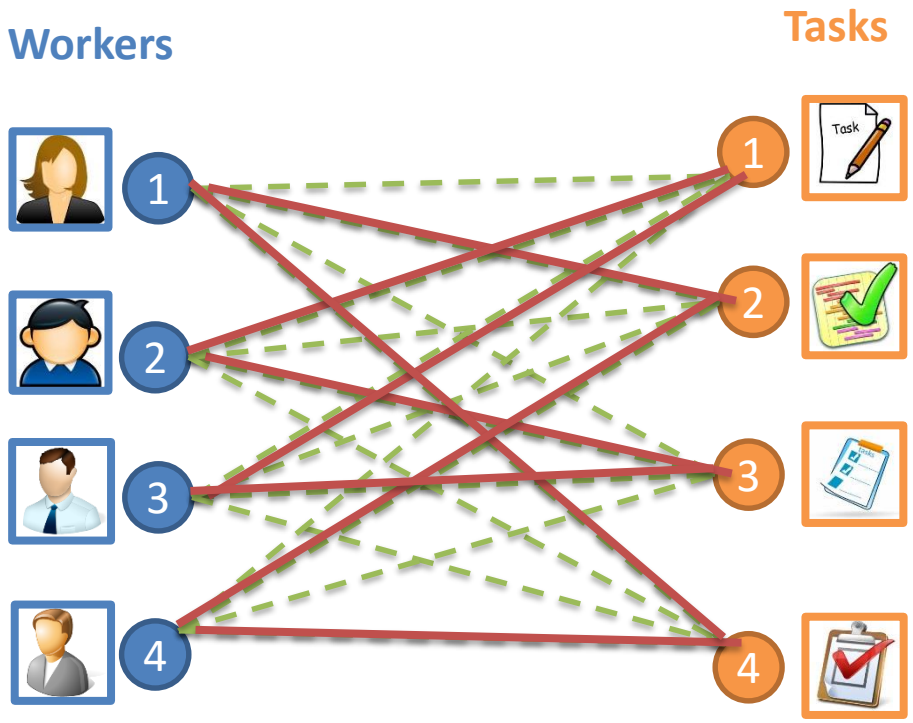
Microsoft

1. What if the feedback is limited?

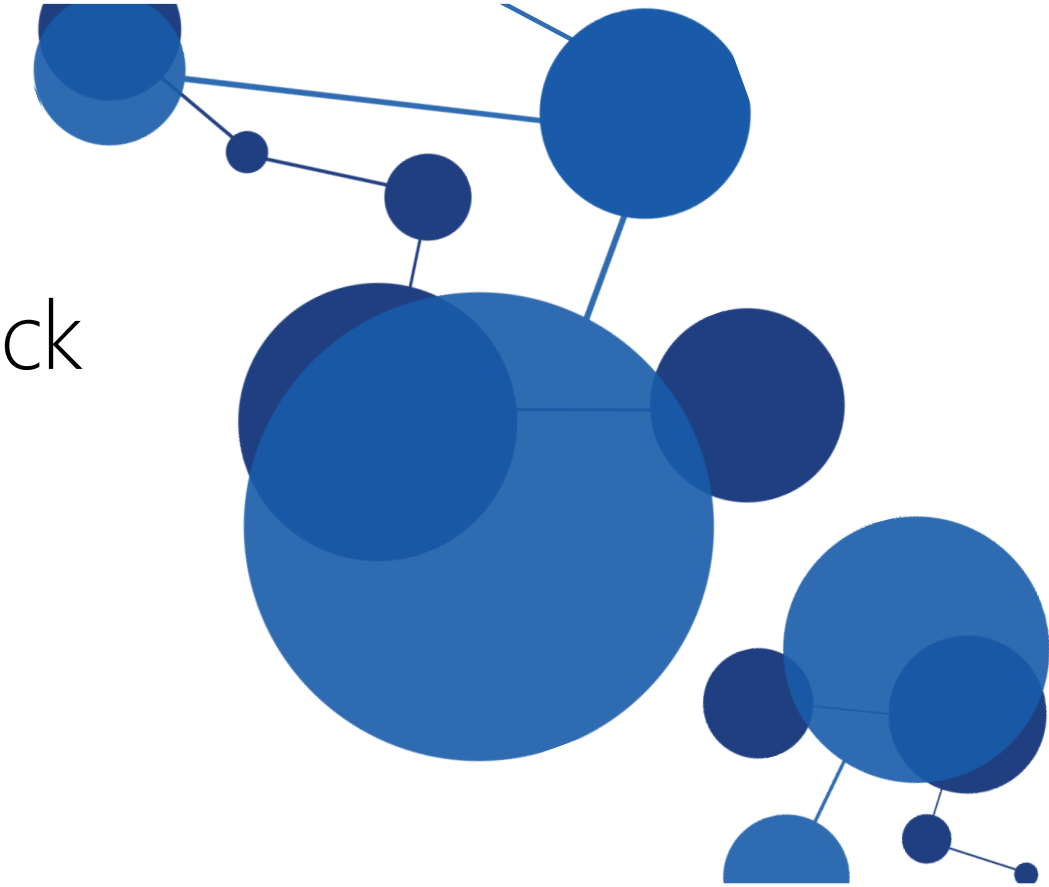


Motivating example: Crowdsourcing

- Matching workers with tasks in a bipartite graph
  - Multiple timeslots: In each timeslot, assign one worker to one task, and the performance is probabilistic
  - Goal: cumulative reward from all timeslots and all worker-task pair performance
- Feedback may be limited:
  - workers may not report their performance
  - Some edges may not be observed in a round.
  - Feedback may or may not equal to reward



Microsoft



See ICML'14: Combinatorial Partial Monitoring Game with Linear Feedback and Its Applications

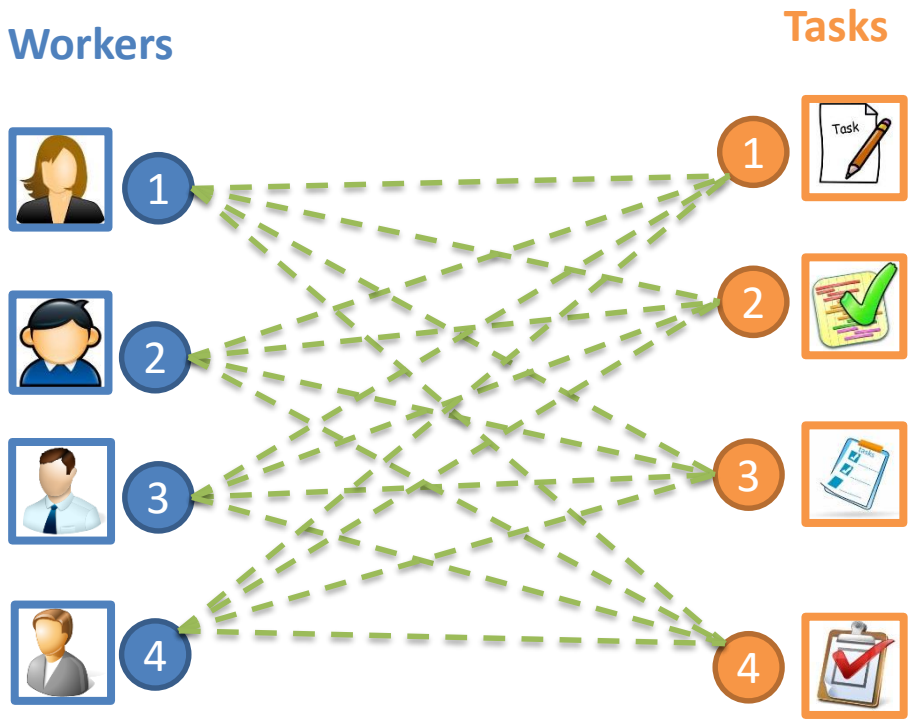
joint work with
Tian Lin, Tsinghua U.
Bruno Abrahao, Robert Kleinberg, Cornell U.
John C.S Lui, CUHK

Microsoft

2. How to test base arms efficiently to find the best super arm?

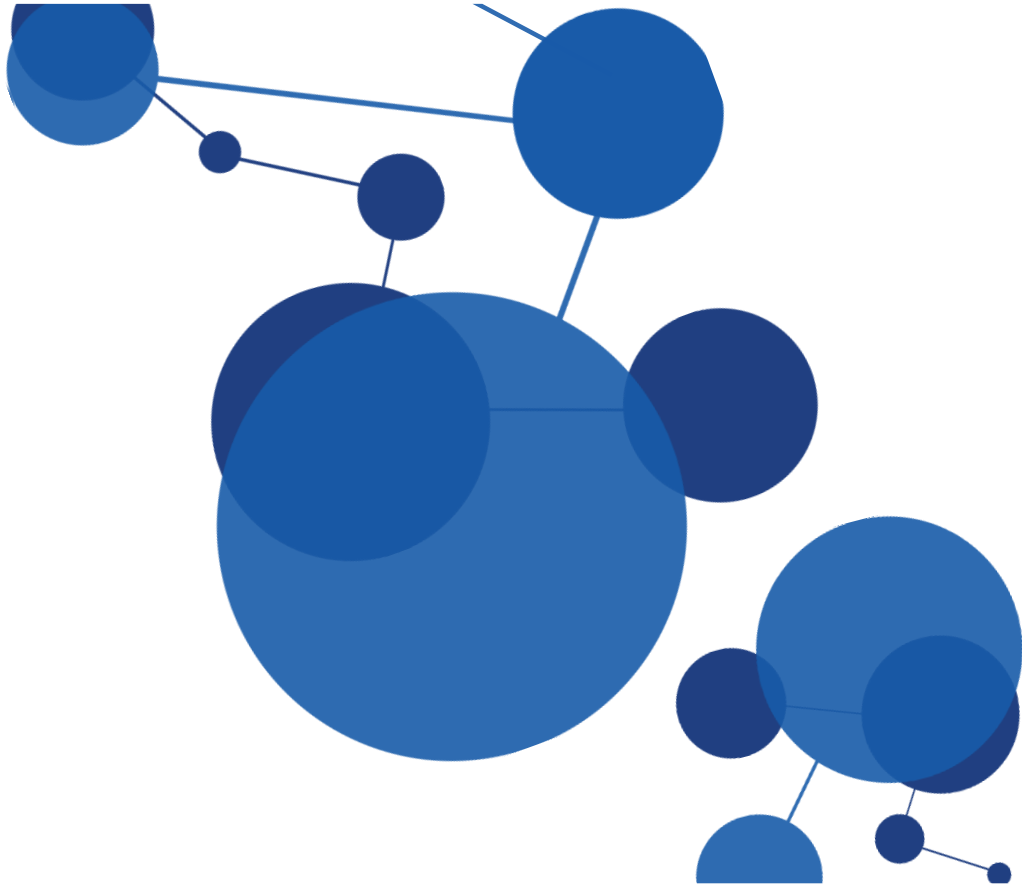
Motivating example: Crowdsourcing

- Matching workers with tasks in a bipartite graph
  - Initial test period: adaptively test worker-task pair performance
  - Goal: at the end of test period, find the best worker-task matching



Microsoft

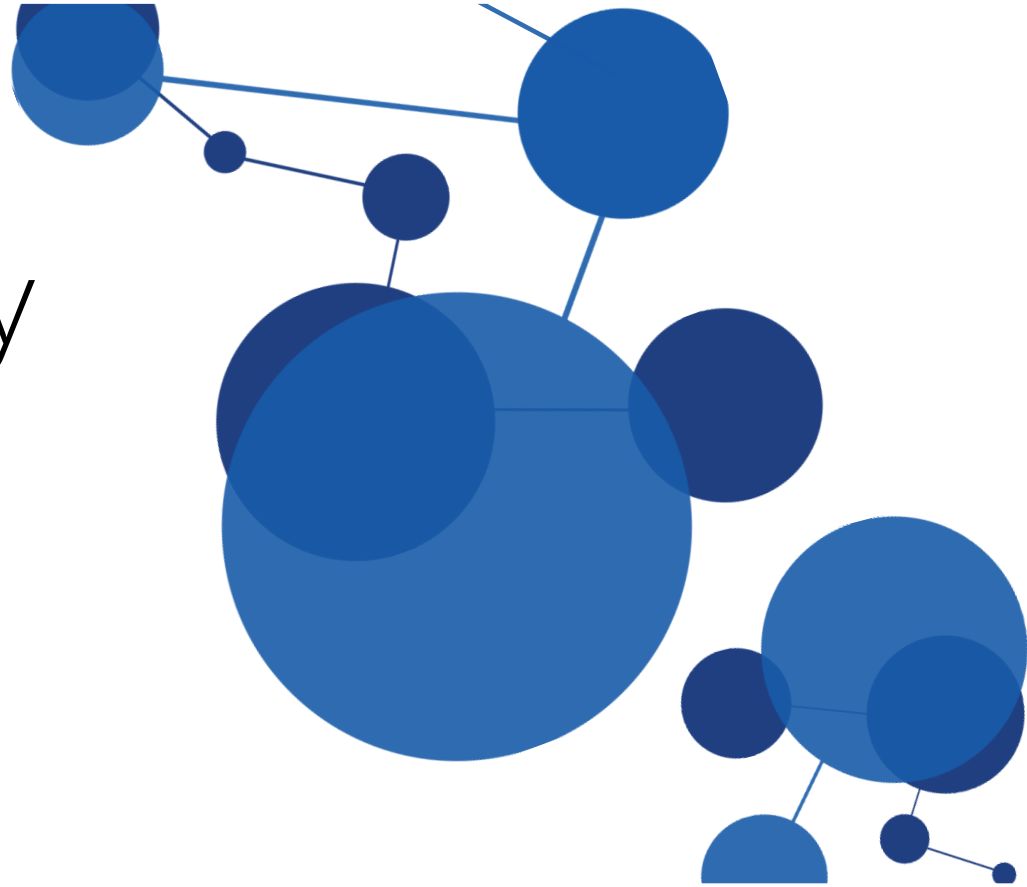
See NIPS'14: Combinatorial Pure Exploration in Multi-Armed Bandits



joint work with
Shouyuan Chen, Irwin King, Michael R. Lyu, CUHK
Tian Lin, Tsinghua U.

Microsoft

3. How to turn offline greedy algorithm to online greedy?

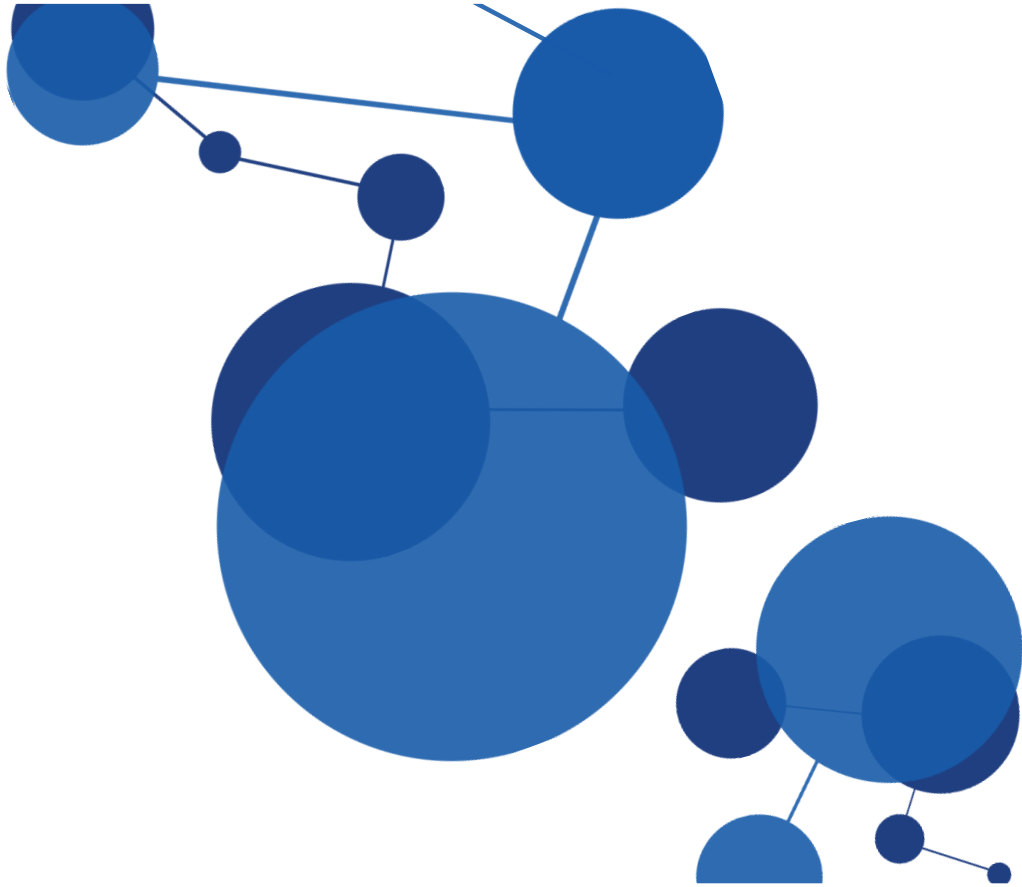


Motivation

- Greedy algorithm is extensively used in optimizations as approximation algorithms or heuristics
- Is there a systematic way of turn offline greedy algorithm into an online greedy learning algorithm?
- Can be viewed as open up the offline oracle to help learning

Microsoft

See NIPS'15: Stochastic Online Greedy Learning with Semi-bandit Feedbacks



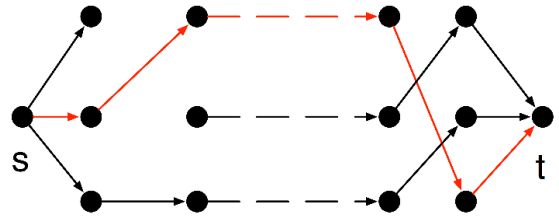
joint work with
Tian Lin, Jian Li, Tsinghua U.

Microsoft

4. What if estimating means of arms is not enough?

Motivating example: graph routing

- Expected Utility Maximization (EUM) Model
  - Each edge i has a random delay X_i
  - Each routing path is a subset of edges, S
  - utility of a routing path S: $u(\sum_{i \in S} X_i)$
    - $u(\cdot)$ is nonlinear, modeling risk-averse or risk-prone behavior
  - Goal: maximize $\mathbb{E}[u(\sum_{i \in S} X_i)]$
- Issue for online learning (when distributions of X_i's are unknown
  - only estimating the mean of X_i is not enough



Microsoft

See NIPS'16: Combinatorial Multi-Armed Bandit with General Reward Functions

joint work with
Wei Hu, Princeton U.
Fu Li, U. of Texas at Austin
Jian Li, Yu Liu, Tsinghua U
Pinyan Lu, Shanghua U. of Finance and Economics

Overall summary

- Central theme
  - deal with stochastic and unknown inputs for combinatorial optimization problems
  - modular approach: separate offline optimization with online learning
    - learning part does not need domain knowledge on optimization
- More wait to be done
  - Many other variants of combinatorial optimizations problems --- as long as it has unknown inputs need to be learned
  - E.g., nonlinear rewards, approximations, adversarial unknown inputs, etc.

Microsoft

Thank you!