Microsoft Research

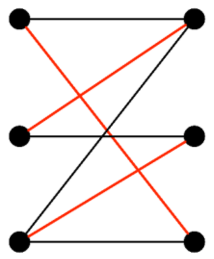
Microsoft

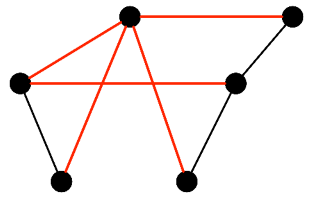
Combinatorial Learning for Combinatorial Optimization --- A Trilogy

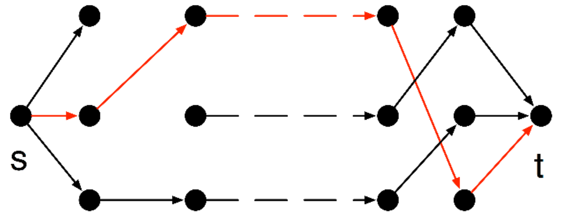
Wei Chen 陈卫
Microsoft Research Asia

Combinatorial optimization

- Well studied
  - classics: shortest paths, min. spanning trees, max. matchings
  - 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





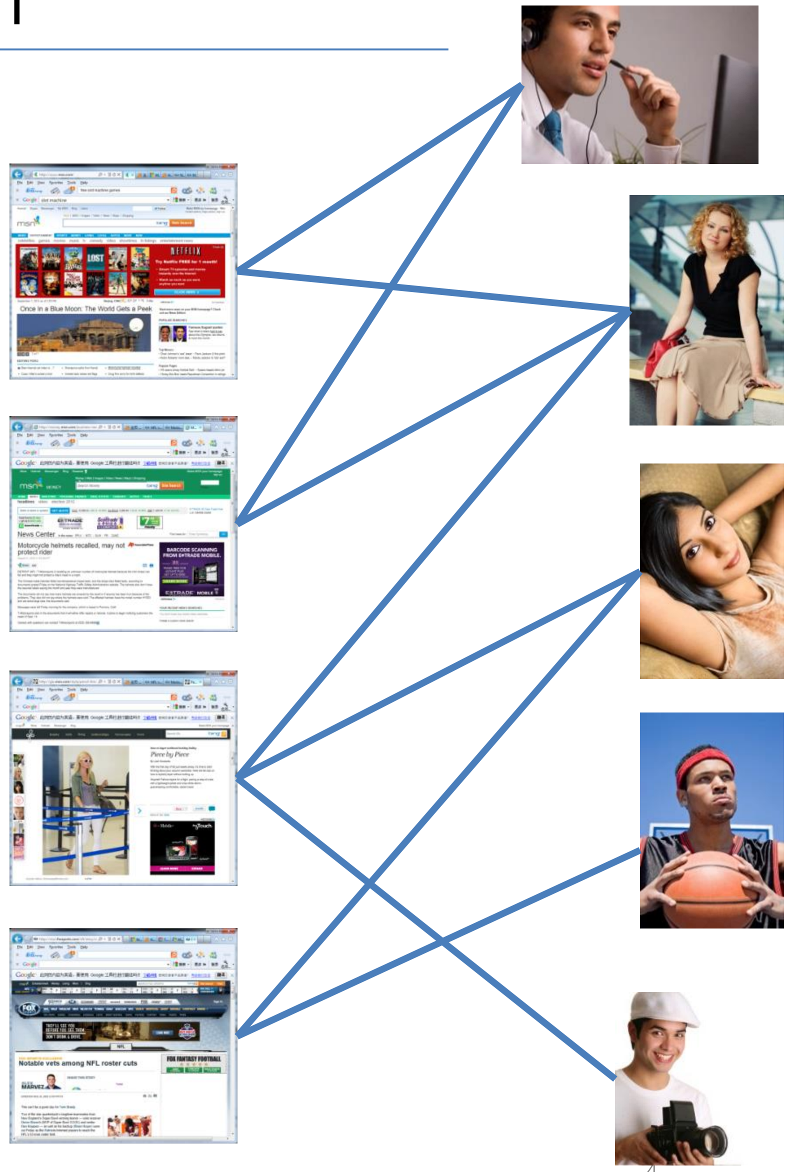


Combinatorial learning for combinatorial optimizations

- 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

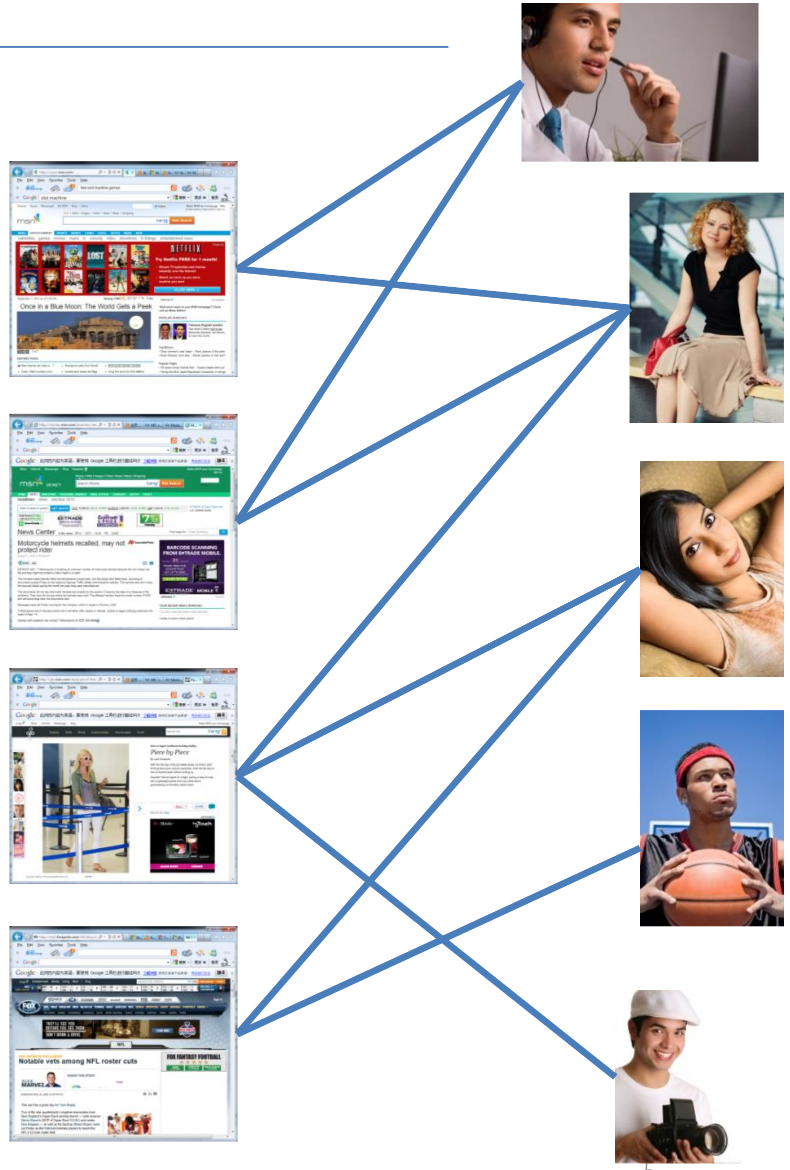
Motivating application: Display ad placement

- 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?



Main difficulties

- Combinatorial in nature
- Non-linear optimization objective, based on underlying random events
- Offline optimization may already be hard, need approximation
- Online learning: learn while doing repeated optimization



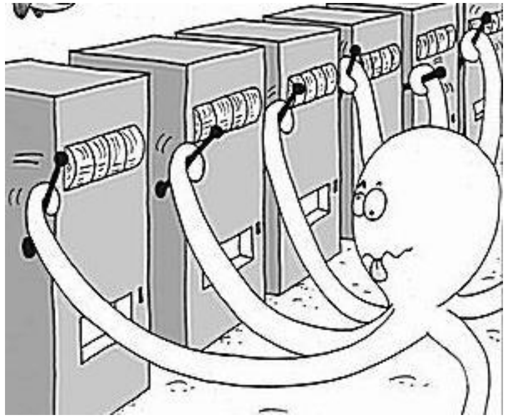
Multi-armed bandit: the canonical OL problem

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

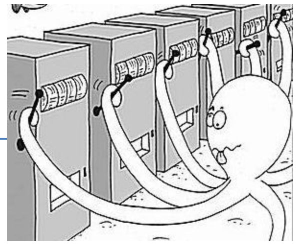
Multi-armed bandit problem

– 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 $\mathbf{O}(\sqrt{nT \log T})$, 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

A COL Trilogy

- On stochastic setting: Only a few scattered work exist before
- ICML'13: Combinatorial multi-armed bandit framework
  - On cumulative rewards / regrets
  - Handling nonlinear reward functions and approximation oracles
- ICML'14: Combinatorial partial monitoring
  - Handling limited feedback with combinatorial action space
- NIPS'14: Combinatorial pure exploration
  - On best combinatorial arm identification
  - Handling combinatorial action space

The unifying theme

- Separate online learning from offline optimization
  - Assume offline optimization oracle
- General combinatorial online learning framework
  - Apply to many problem instances, linear, non-linear, exact solution or approximation

Chapter I:
Combinatorial Multi-Armed Bandit: General Framework, Results and Applications

ICML'2013, joint work with
Yajun Wang, Microsoft
Yang Yuan, Cornell U.

Contribution of this work

- Stochastic combinatorial multi-armed bandit framework
  - handling non-linear reward functions
  - UCB based algorithm and tight regret analysis
  - 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, $S \subseteq [n]$
- 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 $i \in [n]$ 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, $\boldsymbol{\mu} = (\mu_1, \mu_2, \dots, \mu_n)$, denoted $r_{\boldsymbol{\mu}}(S)$
  – e.g. linear rewards, or independent Bernoulli random variables
- 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 $\boldsymbol{\mu} = (\mu_1, \mu_2, \dots, \mu_n)$,
  - Output: a super arm $\boldsymbol{S}$, such that with probability at least $\boldsymbol{\beta}$ the expected reward of $\boldsymbol{S}$ under $\boldsymbol{\mu}$, $\boldsymbol{r_{\mu}(S)}$, is at least $\boldsymbol{\alpha}$ 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 $\boldsymbol{\alpha\beta}$ fraction of the optimal

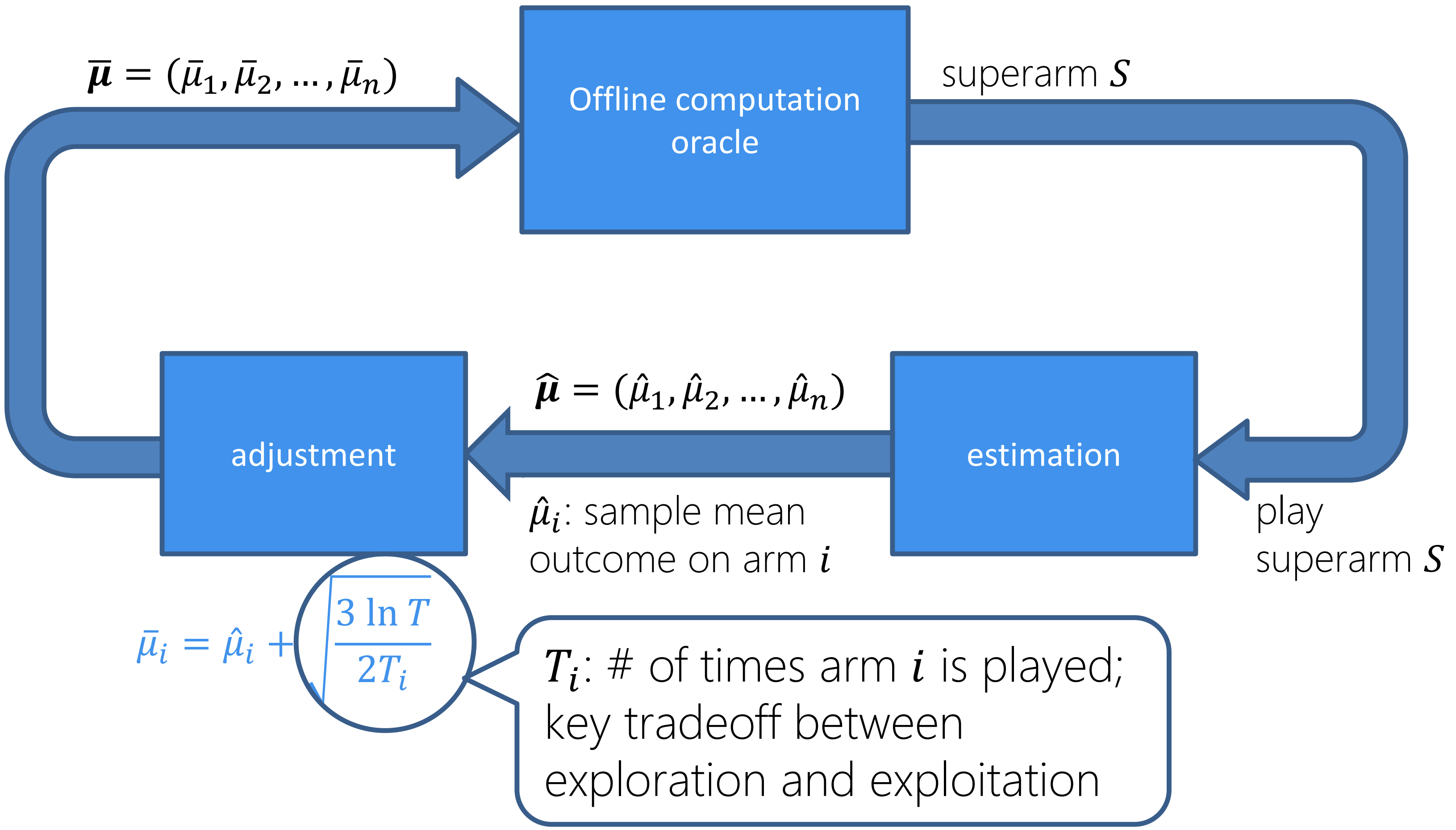
$$\text{Regret} = T \cdot \alpha\beta \cdot \mathbf{opt}_{\boldsymbol{\mu}} - \mathbb{E}[\sum_{i=1}^{T} r_{\boldsymbol{\mu}}(S_t^A)]$$

- Difficulty: do not know
  – combinatorial structure
  – reward function
  – arm outcome distribution
  – 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 $\boldsymbol{n}$ 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 $\boldsymbol{\alpha} \cdot \mathbf{opt}_{\boldsymbol{\mu}}$ and reward of a bad super arm containing $\boldsymbol{i}$.
    - $\Delta_{\min} = \min_i \Delta_{\min}^i$, $\Delta_{\max} = \max_i \Delta_{\max}^i$
    - Here, we define the set of bad super arms as

$$\mathcal{S}_{\mathrm{B}} = \{S \mid r_{\boldsymbol{\mu}}(S) < \alpha \cdot \mathrm{opt}_{\boldsymbol{\mu}}\}$$

- Match UCB regret for classic MAB

Proof ideas (for a looser bound)

- Each base arm has a sampling threshold $\ell_t = \frac{6 \ln t}{\left(f^{-1}(\Delta_{\min})\right)^2}$
  - $T_{i,t-1} > \ell_t$: base arm i is sufficiently sampled at time t
  - $T_{i,t-1} \le \ell_t$: base arm i is under-sampled at time t
- At round t, with high probability $(1 - 2nt^{-2})$, the round is nice --- empirical means of all base arms are within their confidence radii:
  - $\forall i \in [n], |\hat{\mu}_{i,T_{i,t-1}} - \mu_i| \le \Lambda_{i,t}$, $\Lambda_{i,t} = \sqrt{\frac{3 \ln t}{2T_{i,t-1}}}$ (by Hoeffding inequality)
- In a nice round t with selected super arm S_t, if all base arms of S_t are sufficiently sampled, then using their UCBs the oracle will not select a bad super arm S_t
  - Continuity and monotonicity conditions

Why bad super arm cannot be selected in a nice round when its base arms are sufficiently sampled

- define $\Lambda = \sqrt{\frac{3 \ln t}{2\ell_t}}$, $\Lambda_t = \max\{\Lambda_{i,t} | i \in S_t\}$, thus $\Lambda > \Lambda_t$ (by sufficient sampling condition)
- $\forall i \in [n], \bar{\mu}_{i,t} \geq \mu_i$, and $\forall i \in S_t, |\bar{\mu}_{i,t} - \mu_i| \leq 2\Lambda_t$ (since $\bar{\mu}_{i,t} = \hat{\mu}_{i,T_{i,t-1}} + \Lambda_{i,t}$)
- Then we have:

$$
\begin{aligned}
r_{\boldsymbol{\mu}}(S_t) + f(2\Lambda) &> r_{\boldsymbol{\mu}}(S_t) + f(2\Lambda_t) && \{\text{strict monotonicity of } f\} \\
&\geq r_{\bar{\boldsymbol{\mu}}_t}(S_t) && \{\text{bounded smoothness of } r_{\boldsymbol{\mu}}(S)\} \\
&\geq \alpha \cdot \mathrm{opt}_{\bar{\boldsymbol{\mu}}_t} && \{\alpha\text{-approximation w.r.t. } \bar{\boldsymbol{\mu}}_t\} \\
&\geq \alpha \cdot r_{\bar{\boldsymbol{\mu}}_t}(S^*_{\boldsymbol{\mu}}) && \{\text{definition of } \mathrm{opt}_{\bar{\boldsymbol{\mu}}_t}\} \\
&\geq \alpha \cdot r_{\boldsymbol{\mu}}(S^*_{\boldsymbol{\mu}}) = \alpha \cdot \mathrm{opt}_{\boldsymbol{\mu}} && \{\text{monotonicity of } r_{\boldsymbol{\mu}}(S)\}
\end{aligned}
$$

- Since $f(2\Lambda) = \Delta_{\min}$, by the def'n of $\Delta_{\min}$, S_t is not a bad super arm with probability $1 - 2nt^{-2}$.

Counting the regret

- Sufficiently sampled part:
  - $\sum_{t=1}^{T} 2nt^{-2} \cdot \Delta_{\max} \leq \frac{\pi^2}{3} \cdot n \cdot \Delta_{\max}$
- Under-sampled part: pay regret $\boldsymbol{\Delta_{\max}}$ for each under-sampled round
  - If a round is under-sampled (meaning some of the base arms of the played super arm is under-sampled), the under-sampled base arms must be sampled once
  - Thus total number of under-sampled round is at most $m\,(\ell_T + 1) = \left(\frac{6 \ln T}{(f^{-1}(\Delta_{\min}))^2} + 1\right) \cdot n$
- . Thus, getting a loose bound:

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

- To tighten the bound, fine-tune sufficient sampling condition and under-sampled part regret computation.

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

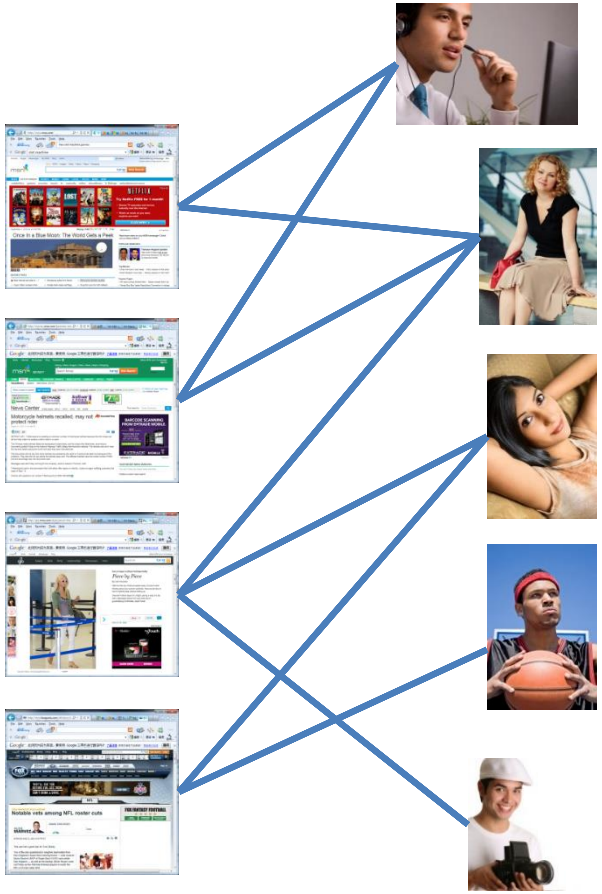
- Bipartite graph $G = (L, R, E)$
- Each edge is a base arm
- Each set of edges linking k webpages is a super arm
- Bounded smoothness function

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

- $(1 - {}^{1}/_{e}, 1)$-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
- See full report arXiv:1111.4279 for complete details

Summary and future work

- Summary
  - Avoid combinatorial explosion while utilizing low-level observed information
  - Modular approach: separation between online learning and offline optimization
  - Handles non-linear reward functions
  - New applications of the CMAB framework, even including probabilistically triggered arms
- Future work
  - Improving algorithm and/or regret analysis for probabilistically triggered arms
  - Combinatorial bandits in contextual bandit settings
  - Investigate CMABs where expected reward depends not only on expected outcomes of base arms

Chapter II: Combinatorial Partial Monitoring Game with Linear Feedback and Its Applications

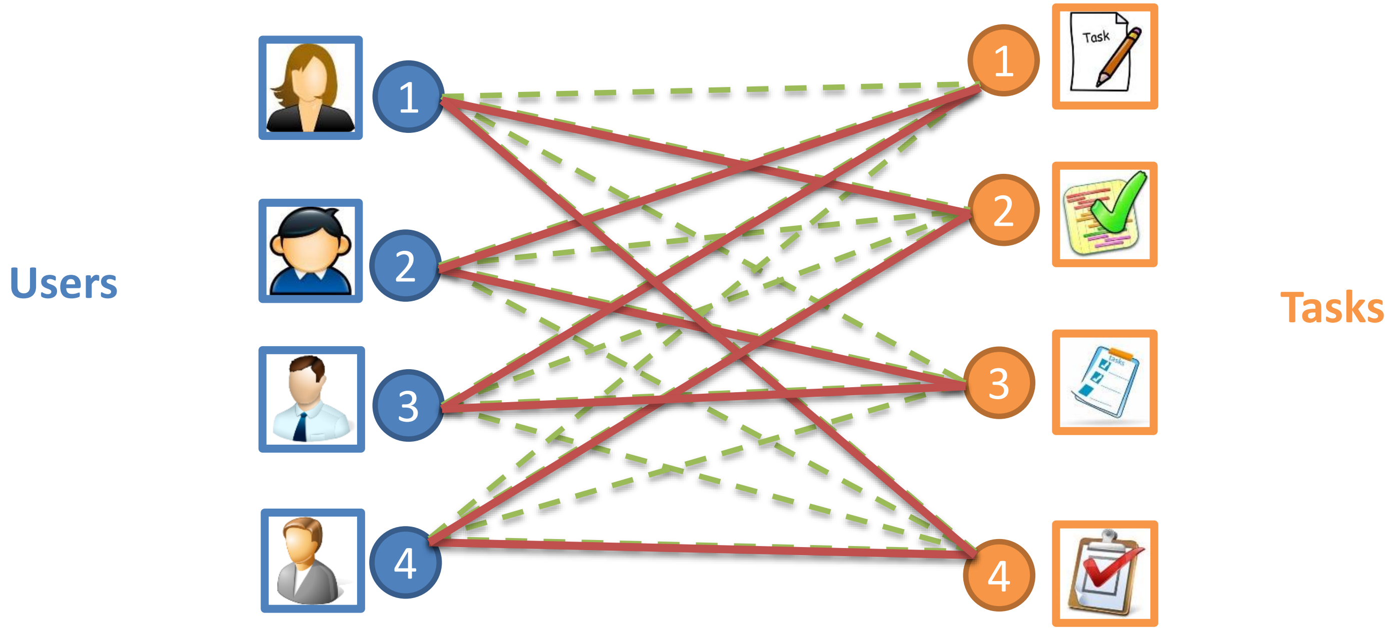
ICML'2014, joint work with
Tian Lin, Tsinghua U.
Bruno Abrahao, Robert Kleinberg, Cornell U.
John C.S Lui, CUHK

New question to address: What if the feedback is limited?

Motivating example: Crowdsourcing

– In each timeslot, one user works on one task, and the performance is probabilistic

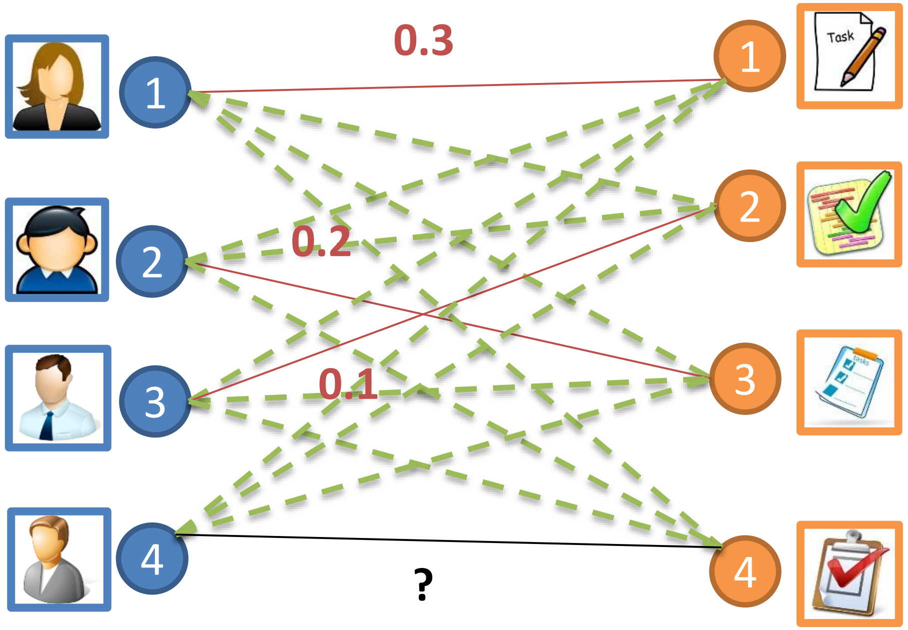
- Matching workers with tasks in a bipartite graph $G = (V, E)$.
- The total reward is based on the performance of the matching.
- Want to find the matching yielding the best performance



The total number of possible matchings is exponentially large!

Motivating example: Crowdsourcing

- 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.



Question: Can we maximize rewards by learning the best matching?

Features of the problem

- Features of the problem：
  - Combinatorial learning
    - Possible choices are exponentially large
  - Stochastic model: e.g. human behaviors are stochastic
  - Limited feedback:
    - Users may not want to provide feedback (need extra work)
- Other examples in combinatorial recommendation
  - Learning best matching in online advertising, buyer-seller markets, etc.
  - Learning shortest path in traffic monitoring and planning, etc.

Related work

	Sufficient Feedback (easier)	Limited Feedback (harder)
Simple action space $\lvert\mathcal{X}\rvert = \mathrm{poly}(n)$	Full information [Littlestone & Warnuth, 1989] MAB [Robbin, 1985; Auer et al. 2002]	Finite partial monitoring [Piccolboni & Schindelhauer, 2001; Cesa et al., 06; Antos et al., 12] Issue: algorithm and regret linearly depends on $\lvert\mathcal{X}\rvert$
Combinatorial action space $\lvert\mathcal{X}\rvert = \exp(n)$	CMAB [Cesa-Bianchi et al., 2010; Gai et al., 2012; Chen et al., 2012] Issue: require sufficient feedback	?

← (CPM: The first step towards this problem)

Our contributions

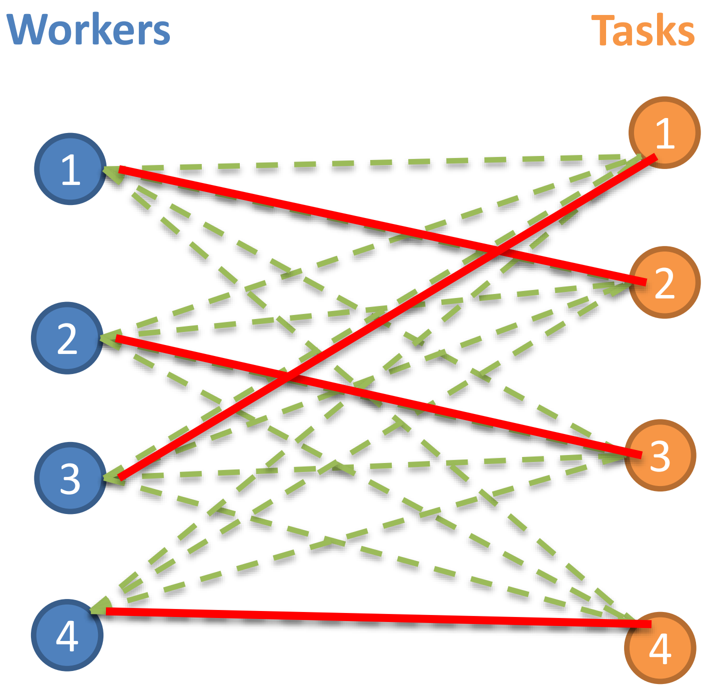
- Generalize FPM to Combinatorial Partial Monitoring Games (CPM):
  - Action set $|\mathcal{X}|$: $\mathbf{poly}(n) \rightarrow \mathbf{exp}(n)$
  - Environment outcomes: Finite set $\{1, 2, \cdots, M\} \rightarrow$ Continuous space $[0,1]^n$ ($\boldsymbol{n}$ base arms)
  - Reward: linear → non-linear (with Lipschitz continuity)
  - Algorithm only needs a weak feedback assumption
  - use information from a set of actions jointly
- Achieve regret bounds: distribution-independent $\mathbf{O}\left(T^{\frac{2}{3}}(\log T + \log|\mathcal{X}|)\right)$ and distribution-dependent $\mathbf{O}(\log T + \log|\mathcal{X}|)$
  - Regret depends on $\log|\mathcal{X}|$ instead of $|\mathcal{X}|$

Our solution

- Ideas: consider actions jointly
  - Use a small set of actions to “observe” all actions
    - Borrowing linear regression idea
  - One action only provides limited feedback, but their combination may provide sufficient information.

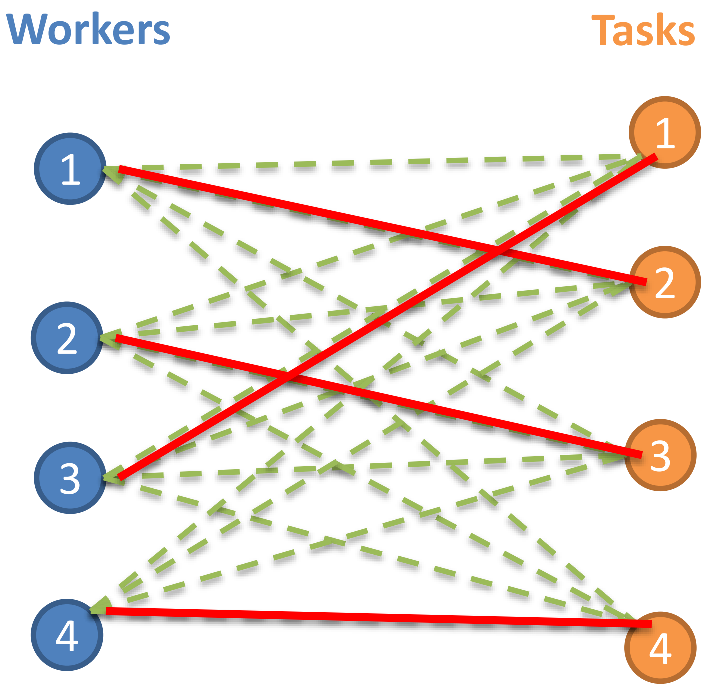
Example application to crowdsourcing

- Model: Matching workers with tasks, bipartipe $G = (V, E)$
  - Each edge e_{ij} is a base arm (the outcome v_{ij} is the utility of worker i on the task j)
  - each matching is a super arm, or an action $\boldsymbol{x}$
  - Find a matching x to maximize total utilities $\operatorname{argmax}_x \mathbf{E}[\sum_{e_{ij} \in x} v_{ij}]$



Example application to crowdsourcing

- Feedback: Only for certain observable actions, observe the a partial sum of three edge outcomes
  - Represented by a transformation matrix $\boldsymbol{M_x}$
  - Outcome of edges in vector $\boldsymbol{v}$
  - $\boldsymbol{M_x} \cdot \boldsymbol{v}$ is the feedback of action $\boldsymbol{x}$
  - When stacking $\boldsymbol{M_x}$ together, it is full column rank
- Algorithm solution:
  - Use these observable actions to explore
  - Use linear regression to estimate and find best action and explore
  - Properly set switching condition between exploration and exploitation



Conclusion and future work

- Propose CPM model:
  - Exponential number of actions/Infinite outcomes/non-linear reward
  - Succinct representation by using transformation matrices
- Global observer set:
  - Use combination of action for limited feedbacks, and it is small
- Algorithm and results:
  - Use global confidence bound to raise the probability of finding the optimal action
  - Guarentee $\tilde{O}(T^{2/3})$ and $O(\log T)$ (assume unique optimum), only linearly depends on $\log |X|$
- Future work:
  - More flexible feedback model
  - More applications

Chapter III: Combinatorial Pure Exploration in Multi-Armed Bandits

NIPS'2014, joint work with
Shouyuan Chen, Irwin King, Michael R. Lyu, CUHK
Tian Lin, Tsinghua U.

Pure exploration

Multi-armed bandit vs. Pure exploration bandit

You go to Vegas trying to explore different slot machines while gaining as much as possible --- cumulative reward

You and your boss go to Vegas together trying to explore the slot machines and find the best machine for your boss to win --- best machine identification

Pure exploration bandit

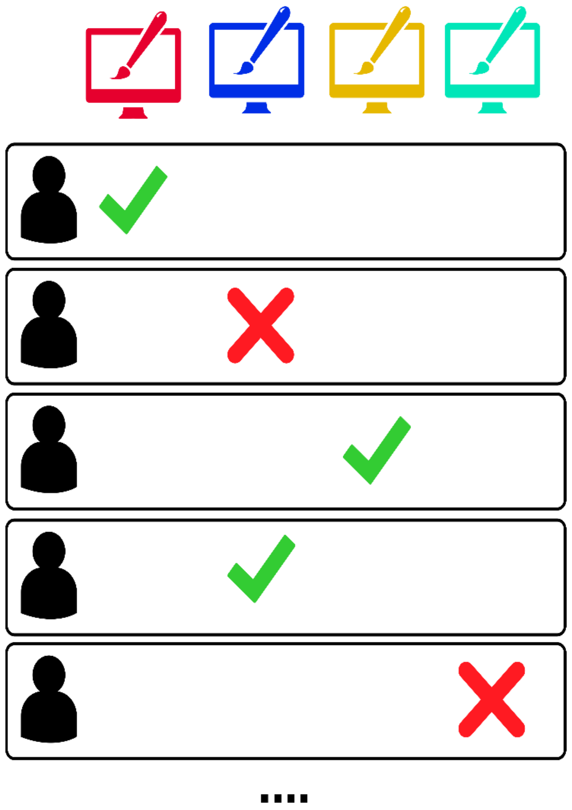
- n arms
- Fixed budget model --- with a fixed time period T
  - Learn in first T rounds, and output one arm at the end
  - Maximize the probability of outputting the best arm
- Fixed confidence model --- with a fixed error confidence δ
  - Explore arms and output one arm in the end
  - Guarantee that the output arm is the best arm with probability of error at most δ
  - Minimize the number of rounds needed for exploration
- How to adaptively explore arms to be more effective
  - Arms less (more) likely to be the best one should be explored less (more)

Pure exploration vs. Online learning

Online learning	Pure exploration
Learning while optimization	A dedicate learning period, with a learning output for subsequent optimization
Adaptive for both learning and optimization	Adaptive for more effective learning
Exploration vs. exploitation tradeoff	Focus on adaptive exploration in the learning period
Multi-armed bandit	Pure exploration bandit

Application of pure exploration

- A/B testing
- Others: clinical trials, wireless networking (e.g. finding the best route, best spanning tree)



Combinatorial pure exploration

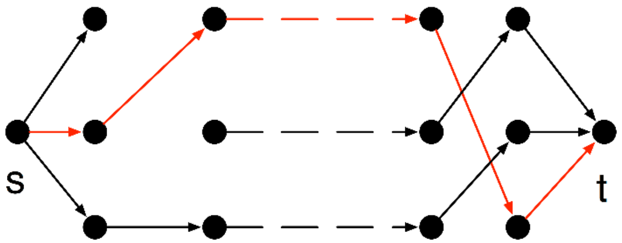
- Play one arm at each round
- Find the optimal set of arms $\boldsymbol{M_*}$ satisfying certain constraint

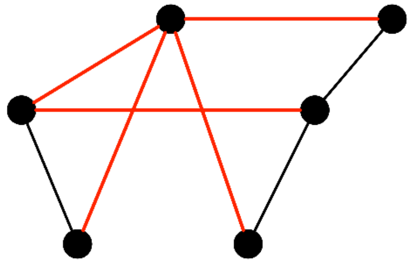
$$M_* = \arg\max_{M \in \mathcal{M}} \sum_{e \in M} w(e)$$

  - $\mathcal{M} \subseteq 2^{[\boldsymbol{n}]}$ decision class with certain combinatorial constraint
    - E.g. k-sets, spanning trees, matchings, paths
  - maximize the sum of expected rewards of arms in the set
- Prior work
  - Find top-k arms [KS10, GGL12, KTPS12, BWV13, KK13, ZCL14]
  - Find top arms in disjoint groups of arms (multi-bandit) [GGLB11, GGL12, BWV13]
  - Separated treatments, no unified framework

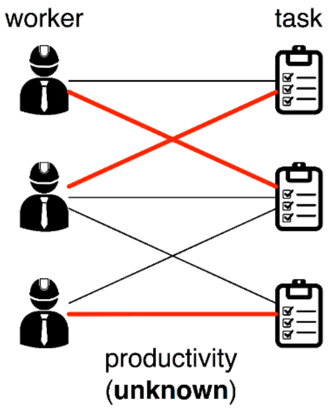
Applications of combinatorial pure exploration

- Wireless networking
  - Explore the links, and find the expected shortest paths or minimum spanning trees





- Crowd sourcing
  - Explore the worker-task pair performance, and find the best matching

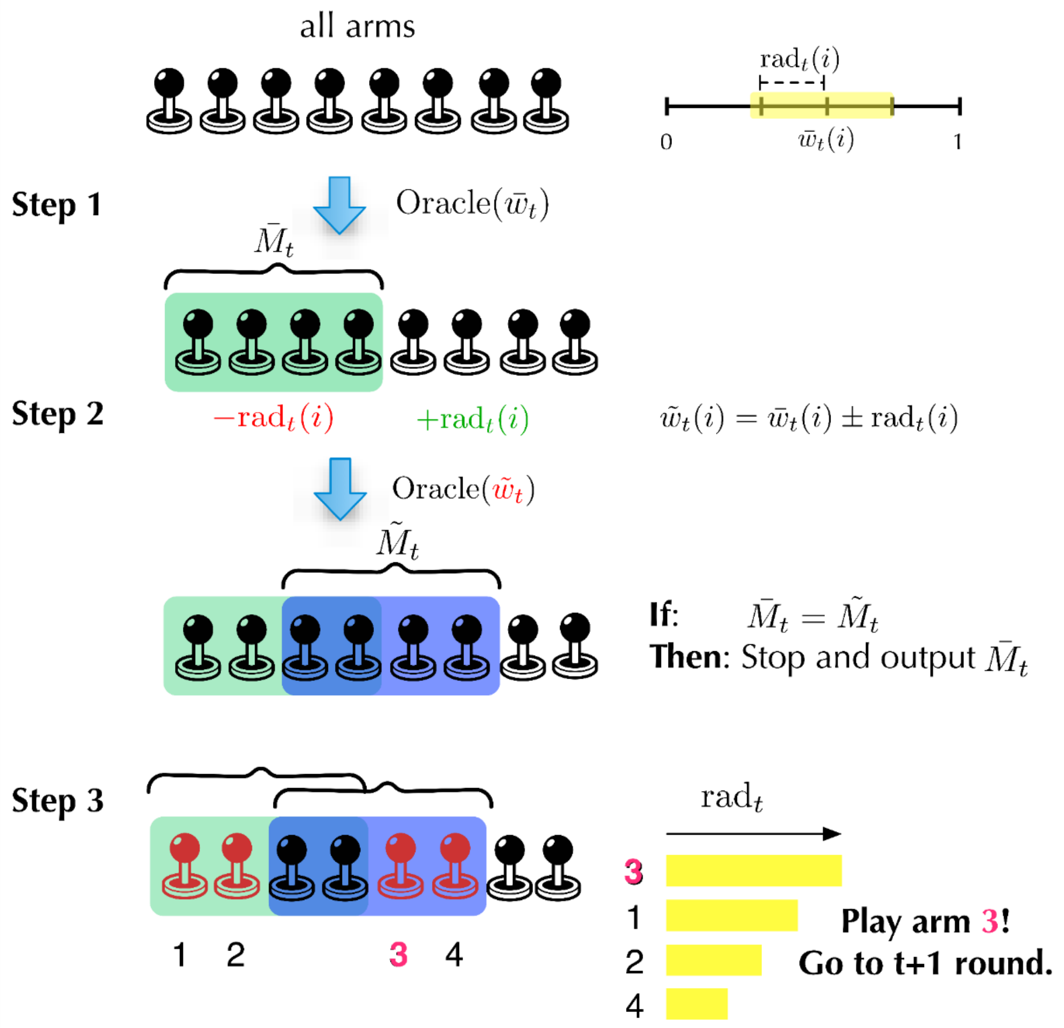


Goal:
1) estimate the productivities from tests.
2) find the optimal **1-1 assignment**.

CLUCB: fixed-confidence algo

input parameter: $\delta \in (0,1)$
(max. allowed probability of error)

maximization oracle:
Oracle(): $R^n \rightarrow \mathcal{M}$
$\text{Oracle}(v) = \arg\max_{M \in \mathcal{M}} \sum_{i \in M} v(M)$ for
weights $v \in R^n$



CLUCB result

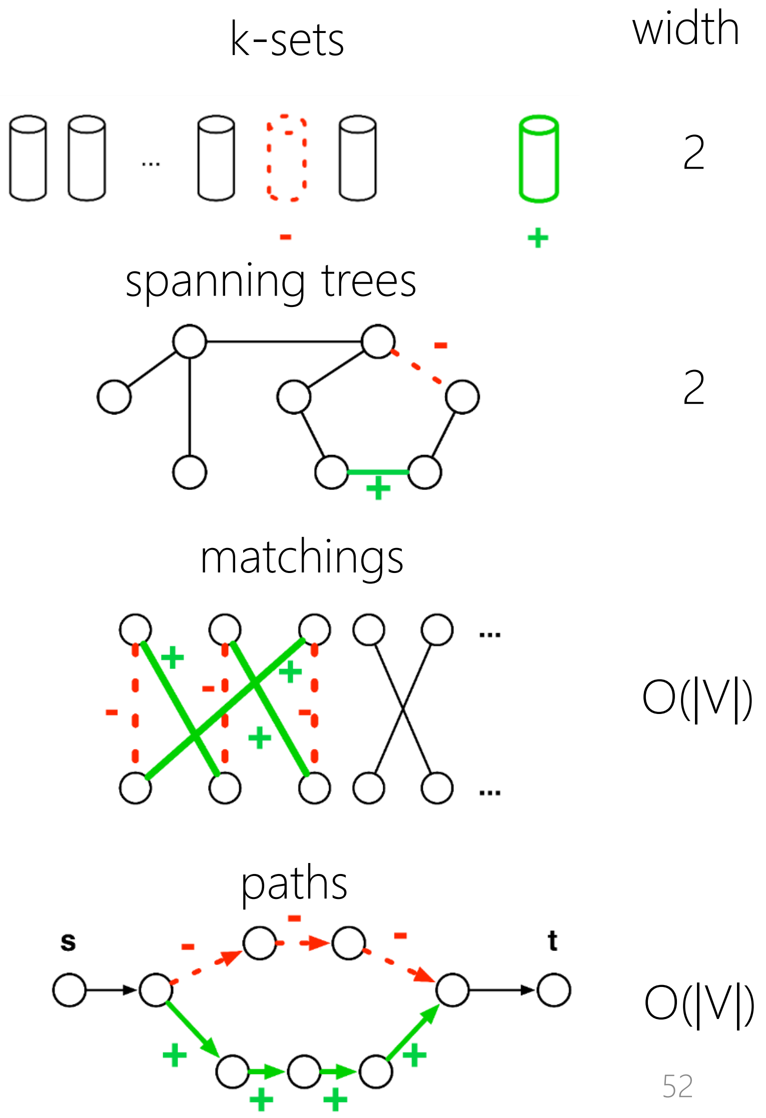
- With probability at least $\mathbf{1-\delta}$
  - Correctly find the optimal set
  - Uses at most $O\left(\mathrm{width}^2(\mathcal{M})\mathrm{H}\log\left(\frac{n\mathrm{H}}{\delta}\right)\right)$ rounds
    - H: hardness, $\mathrm{width}(\mathcal{M})$: width of the decision class
- Hardness:
  - Δ_e: Gap of arm $\boldsymbol{e}$

$$\Delta_e = \begin{cases} w(M_*) - \max_{M \in \mathcal{M}: e \in M} w(M) & \text{if } e \notin M_*, \\ w(M_*) - \max_{M \in \mathcal{M}: e \notin M} w(M) & \text{if } e \in M_*, \end{cases}$$

  - $\mathbf{H} = \sum_{e \in [n]} \Delta_e^{-2}$
  - Recover previous definitions of H for the top-1, top-K and multi-bandit problems.

Exchange class and width --- arm interdependency measure

- **exchange class:** a unifying method for analyzing different decision classes
  - a ``proxy" for the structure of decision class
  - An exchange class B is a collection of ``patches"
  - (b_+, b_-) (where $b_+, b_- \subseteq [n]$) are used to interpolate between valid sets $M' = M \cup b_+ \setminus b_-$ $(M, M' \in \mathcal{M})$
- **width** of exchange class B: size of largest patch
  - $\mathrm{width}(B) = \max_{(b_+,b_-)\in B} (|b_+| + |b_-|)$
- **width** of decision class $\mathcal{M}$: width of the ``thinnest" exchange class
  - $\mathrm{width}(\mathcal{M}) = \min_{B \in \mathrm{Exchange}(\mathcal{M})} \mathrm{width}(B)$



Other results

- Lower bound: $\widetilde{\Omega}(\mathrm{H})$
- Fixed budget algo: CSAR
  - successive accepting / rejecting arms
  - Correct with probability at least $1 - 2^{\tilde{O}\left(-\frac{T}{\mathbf{width}^2(\mathcal{M})\mathrm{H}}\right)}$
- Extend to PAC learning (allow ε off from optimal)

Future work

- Narrow down the gap (dependency on the width)
- Support approximation oracles
- Support nonlinear reward functions

Overall summary on combinatorial learning

- 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, expected rewards depending not only on means of arm outcomes, adversarial unknown inputs, etc.

Microsoft

Thank you!