Dueling Bandits

CS 159

Aman Agarwal, Kushal Agarwal, Fabian Boemer, and Jialin Song

May 24, 2016
Overview

1. Motivation

2. Problem Formulation

3. The Algorithm

4. Extensions
Motivation
How does Clickthrough Data Reflect Retrieval Quality? [Radlinski 2008]

Given a **QUERY** \( q \) and a collection \( D \) of documents that match the **QUERY**, the problem is to rank the documents in \( D \) according to some criterion so that the “best” results appear early in the result list displayed to the user.
Example: Evaluation Search Rankings

**Retrieval Function A**

CS 159 Purdue University
web.ics.purdue.edu/~cs159/ Purdue University
Aug 16, 2012 - CS 159 introduces the tools of software development that have become essential for creative problem solving in Engineering. Educators and ...

CS 159: Introduction to Parallel Processing | People | San Jo...
www.sjsu.edu › ... › Chun, Robert K › Courses › San Jose State University
Jan 20, 2015 - Description. A combination hardware architecture and software development class focused on multi-threaded, parallel processing algorithms ...

CS 159: Introduction to Parallel Processing - Info.sjsu.edu
info.sjsu.edu › ... › Courses › San Jose State University
CS 159. Introduction to Parallel Processing. Description Major parallel architectures: shared memory, distributed memory, SIMD, MIMD. Parallel algorithms: ...

Guy falls asleep in CS159 lab Purdue - YouTube
https://www.youtube.com/watch?v=vVciOgZwLag
Mar 24, 2011 - Uploaded by james brand
Guy falls asleep in our 7:30 am lab so we take his phone turn the volume up to full and call him.

CS 159: Advanced Topics in Machine Learning - Yisong Yue
www.yisongyue.com/courses/cs159/ Yisong Yue
CS 159: Advanced Topics in Machine Learning (Spring 2016). Course Description. This course will cover a mixture of the following topics: Online Learning ...

**Retrieval Function B**

Guy falls asleep in CS159 lab Purdue - YouTube
https://www.youtube.com/watch?v=vVciOgZwLag
Mar 24, 2011 - Uploaded by james brand
Guy falls asleep in our 7:30 am lab so we take his phone turn the volume up to full and call him.

CS 159 Purdue University
web.ics.purdue.edu/~cs159/ Purdue University
Aug 16, 2012 - CS 159 introduces the tools of software development that have become essential for creative problem solving in Engineering. Educators and ...

CS159: Introduction to Parallel Processing | People | San Jo.
www.sjsu.edu › ... › Chun, Robert K › Courses › San Jose State University
Jan 20, 2015 - Description. A combination hardware architecture and software development class focused on multi-threaded, parallel processing algorithms ...

CS 159: Introduction to Parallel Processing - Info.sjsu.edu
info.sjsu.edu › ... › Courses › San Jose State University
CS 159. Introduction to Parallel Processing. Description Major parallel architectures: shared memory, distributed memory, SIMD, MIMD. Parallel algorithms: ...

CS 159: Advanced Topics in Machine Learning - Yisong Yue
www.yisongyue.com/courses/cs159/ Yisong Yue
CS 159: Advanced Topics in Machine Learning (Spring 2016). Course Description. This course will cover a mixture of the following topics: Online Learning ...

CS159: Introduction to Computational Complexity
cs.brown.edu/courses/cs159/home.html Brown University
Home | Course Info | Assignments | Syllabus And Lectures | Staff and Hours | LaTeX. An early model of parallel computation... Home Courses.
Example: Evaluation Search Rankings

Retrieval Function A

CS 159 Purdue University
web.ics.purdue.edu/~cs159/ ▸ Purdue University ▸
Aug 16, 2012 - CS 159 introduces the tools of software development that have become essential for creative problem solving in Engineering. Educators and ...

CS 159: Introduction to Parallel Processing | People | San Jo. www.sjsu.edu › ... › Chun, Robert K › Courses ▸ San Jose State University ▸
CS 159. Introduction to Parallel Processing. Description Major parallel architectures: shared memory, distributed memory, SIMD, MIMD. Parallel algorithms: ...

Guy falls asleep in CS159 lab Purdue - YouTube
https://www.youtube.com/watch?v=vVcIoGzWlAg
Mar 24, 2011 - Uploaded by james brand
Guy falls asleep in our 7:30 am lab so we take his phone turn the volume up to full and call him.

CS 159: Advanced Topics in Machine Learning | People | San Jo. www.yisongyue.com/courses/cs159/ ▸
CS 159: Advanced Topics in Machine Learning (Spring 2016). Course Description. This course will cover a mixture of the following topics: Online Learning ...

Retrieval Function B

Guy falls asleep in CS159 lab Purdue - YouTube
https://www.youtube.com/watch?v=vVcIoGzWlAg
Mar 24, 2011 - Uploaded by james brand
Guy falls asleep in our 7:30 am lab so we take his phone turn the volume up to full and call him.

CS 159 Purdue University
web.ics.purdue.edu/~cs159/ ▸ Purdue University ▸
Aug 16, 2012 - CS 159 introduces the tools of software development that have become essential for creative problem solving in Engineering. Educators and ...

CS 159: Introduction to Parallel Processing | People | San Jo. www.sjsu.edu › ... › Chun, Robert K › Courses ▸ San Jose State University ▸
Jan 20, 2015 - Description. A combination hardware architecture and software development class focused on multi-threaded, parallel processing algorithms ...

CS 159: Introduction to Parallel Processing - Info.sjsu.edu info.sjsu.edu › ... › Courses ▸ San Jose State University ▸
CS 159. Introduction to Parallel Processing. Description Major parallel architectures: shared memory, distributed memory, SIMD, MIMD. Parallel algorithms: ...

CS 159: Introduction to Parallel Processing - Info.sjsu.edu info.sjsu.edu › ... › Courses ▸ San Jose State University ▸
Jan 20, 2015 - Description. A combination hardware architecture and software development class focused on multi-threaded, parallel processing algorithms ...

CS 159: Advanced Topics in Machine Learning - Yisong Yue www.yisongyue.com/courses/cs159/ ▸
CS 159: Advanced Topics in Machine Learning (Spring 2016). Course Description. This course will cover a mixture of the following topics: Online Learning ...

CS 159: Introduction to Computational Complexity cs.brown.edu/courses/cs159/home.html ▸ Brown University ▸
Home | Course Info | Assignments | Syllabus And Lectures | Staff and Hours | LaTeX. An early model of parallel computation... Home Courses.
Example: Evaluation Search Rankings

Retrieval Function A
CS 159 Purdue University
web.ics.purdue.edu/~cs159/ Purdue University
Aug 16, 2012 - CS 159 introduces the tools of software development that have become essential for creative problem solving in Engineering. Educators and ...

CS 159: Introduction to Parallel Processing | People | San Jo.
www.sjsu.edu › ... › Chun, Robert K › Courses › San Jose State University
CS 159. Introduction to Parallel Processing. Description Major parallel architectures: shared memory, distributed memory, SIMD, MIMD. Parallel algorithms ...

Guy falls asleep in CS159 lab Purdue - YouTube
https://www.youtube.com/watch?v=vVci6C ...
Mar 24, 2011 - Uploaded by james brand
Guy falls asleep in our 7:30 am lab so we take his phone turn the volume up to full and call him.

Retrieval Function B
Guy falls asleep in CS159 lab Purdue - YouTube
https://www.youtube.com/watch?v=vVci6C ...
Mar 24, 2011 - Uploaded by james brand
Guy falls asleep in our 7:30 am lab so we take his phone turn the volume up to full and call him.

CS 159 Purdue University
web.ics.purdue.edu/~cs159/ Purdue University
Aug 16, 2012 - CS 159 introduces the tools of software development that have become essential for creative problem solving in Engineering. Educators and ...

CS 159: Introduction to Parallel Processing - Info.sjsu.edu
info.sjsu.edu › ... › Courses › San Jose State University
CS 159. Introduction to Parallel Processing. Description Major parallel architectures: shared memory, distributed memory, SIMD, MIMD. Parallel algorithms ...

Guy falls asleep in CS159 lab Purdue - YouTube
https://www.youtube.com/watch?v=vVci6C ...
Mar 24, 2011 - Uploaded by james brand
Guy falls asleep in our 7:30 am lab so we take his phone turn the volume up to full and call him.

CS 159: Advanced Topics in Machine Learning - Yisong Yue
www.yisongyue.com/courses/cs159/ Yisong Yue
CS 159: Advanced Topics in Machine Learning (Spring 2016). Course Description. This course will cover a mixture of the following topics: Online Learning ...

CS 159: Introduction to Computational Complexity
cs.brown.edu/courses/cs159/home.html Brown University
Home | Course Info | Assignments | Syllabus And Lectures | Staff and Hours | LaTeX. An early model of parallel computation... Home Courses.
Evaluating retrieval functions

- **Explicit tests**
  - Cranfield methodology
    - Quality measure (recall, precision)
  - Expensive
  - Slow turnaround

- **Implicit judgments**
  - Effectively no cost (no experts needed)
  - Real time
  - Reflects values of users
  - Based on user behavior?

What measurements reflect retrieval quality?
Evaluation Methods

○ Absolute Metrics
  ○ **Assumption:** retrieval quality impacts observable user behavior in an ‘absolute sense’
  ○ Abandonment rate
  ○ Reformulation rate
  ○ Queries per session
  ○ Clicks per query
  ○ Max reciprocal rank
  ○ Mean reciprocal rank
  ○ Time to first click
  ○ Time to last click

○ Paired Comparison Tests
  ○ **Assumption:** Users can identify preferred alternative in direct comparison
    ○ Given, $A, B$, give preference $A > B$, or $B > A$
    ○ Inspires dueling bandits bandits
Showing results 1 through 11 (of 11 total) for all:(dueling AND bandits)

1. arXiv:1605.01677 [pdf, other]
   Copeland Dueling Bandit Problem: Regret Lower Bound, Optimal Algorithm, and Computationally Efficient Algorithm
   Junpei Komiyama, Junya Honda, Hiroshi Nakagawa
   Subjects: Machine Learning (stat.ML); Learning (cs.LG)

2. arXiv:1604.07101 [pdf, other]
   Double Thompson Sampling for Dueling Bandits
   Huasen Wu, Xin Liu, R. Srikant
   Comments: 27 pages, 5 figures
   Subjects: Learning (cs.LG); Machine Learning (stat.ML)

   Indistinguishable Bandits Dueling with Decoys on a Poset
   Julien Audiffren (CMLA), Ralaivola Liva (LIF)
   Subjects: Learning (cs.LG); Artificial Intelligence (cs.AI)

   A Relative Exponential Weighing Algorithm for Adversarial Utility-based Dueling Bandits
   Pratik Gajane, Tanguy Urvoy, Fabrice Clérot (FT R and D)
   Subjects: Learning (cs.LG)
Experimental Design

○ **Assumption:** Click indicates user preference

○ Method of presentation: interleaved rankings
  ○ Two rankings should be:
    ○ Blind to user
    ○ Not substantially alter search experience
    ○ Lead to clicks that reflect user’s preference
  ○ More clicks from ranking A than B indicates preference for A over B
Constructing Rankings

Comparison Triplets

- Orig > Flat > Rand
  - **Orig**: Hand-tuned ranking function
  - **Flat**: No field weights
  - **Rand**: Randomize top 11 results in Flat
  - Substantial distinction

- Orig > Swap2 > Swap4
  - **Swap2**: Orig with 2 pairs swapped
  - **Swap4**: Orig with 4 pairs swapped
  - More subtle distinction
Presenting Rankings

- Balanced Interleaving
- Team-Draft Interleaving
  - Analogous to sports captains choosing teammates
  - At each time, a coin flip decides which captain can choose his next teammate
Team-Draft Interleaving Example

**Ranking A**
1. CS 159 Purdue University
2. CS 159: Introduction to Parallel Processing | People | San Jose
3. CS159: Introduction to Parallel Processing - Info.sjsu.edu
4. Guy falls asleep in CS159 lab Purdue - YouTube
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity

**Ranking B**
1. Guy falls asleep in CS159 lab Purdue - YouTube
2. CS 159 Purdue University
3. CS 159: Introduction to Parallel Processing | People | San Jose
4. CS159: Introduction to Parallel Processing - Info.sjsu.edu
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity
Team-Draft Interleaved Ranking

1. CS 159 Purdue University
2. CS 159: Introduction to Parallel Processing | People | San Jose
3. CS159: Introduction to Parallel Processing - Info.sjsu.edu
4. Guy falls asleep in CS159 lab Purdue - YouTube
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity
Team-Draft Interleaving

**Ranking A**
1. CS 159 Purdue University
2. CS 159: Introduction to Parallel Processing | People | San Jose
3. CS159: Introduction to Parallel Processing - Info.sjsu.edu
4. Guy falls asleep in CS159 lab Purdue - YouTube
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity

**Ranking B**
1. Guy falls asleep in CS159 lab Purdue - YouTube
2. CS 159 Purdue University
3. CS 159: Introduction to Parallel Processing | People | San Jose
4. CS159: Introduction to Parallel Processing - Info.sjsu.edu
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity

**Team-Draft Interleaved Ranking**
1. CS 159 Purdue University
TeamDraft Interleaving

**Ranking A**
1. CS 159 Purdue University
2. CS 159: Introduction to Parallel Processing | People | San Jose
3. CS159: Introduction to Parallel Processing - Info.sjsu.edu
4. Guy falls asleep in CS159 lab Purdue - YouTube
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity

**Ranking B**
1. Guy falls asleep in CS159 lab Purdue - YouTube
2. CS 159 Purdue University
3. CS 159: Introduction to Parallel Processing | People | San Jose
4. CS159: Introduction to Parallel Processing - Info.sjsu.edu
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity

**Team-Draft Interleaved Ranking**
1. CS 159 Purdue University
2. Guy falls asleep in CS159 lab Purdue - YouTube
Team-Draft Interleaved Ranking

1. CS 159 Purdue University
2. Guy falls asleep in CS159 lab Purdue - YouTube
Team-Draft Interleaving

**Ranking A**
1. CS 159 Purdue University
2. CS 159: Introduction to Parallel Processing | People | San Jose
3. CS159: Introduction to Parallel Processing - Info.sjsu.edu
4. Guy falls asleep in CS159 lab Purdue - YouTube
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity

**Ranking B**
1. Guy falls asleep in CS159 lab Purdue - YouTube
2. CS 159 Purdue University
3. CS 159: Introduction to Parallel Processing | People | San Jose
4. CS159: Introduction to Parallel Processing - Info.sjsu.edu
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity

**Team-Draft Interleaved Ranking**
1. CS 159 Purdue University
2. Guy falls asleep in CS159 lab Purdue - YouTube
3. CS 159: Introduction to Parallel Processing | People | San Jose
TeamDraft Interleaving

**Ranking A**
1. CS 159 Purdue University
2. CS 159: Introduction to Parallel Processing | People | San Jose
3. CS159: Introduction to Parallel Processing - Info.sjsu.edu
4. Guy falls asleep in CS159 lab Purdue - YouTube
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity

**Ranking B**
1. Guy falls asleep in CS159 lab Purdue - YouTube
2. CS 159 Purdue University
3. CS 159: Introduction to Parallel Processing | People | San Jose
4. CS159: Introduction to Parallel Processing - Info.sjsu.edu
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity

**Team-Draft Interleaved Ranking**
1. CS 159 Purdue University
2. Guy falls asleep in CS159 lab Purdue - YouTube
3. CS 159: Introduction to Parallel Processing | People | San Jose
4. CS159: Introduction to Parallel Processing - Info.sjsu.edu
TeamDraft Interleaving

**Ranking A**
1. CS 159 Purdue University
2. CS 159: Introduction to Parallel Processing | People | San Jose
3. CS159: Introduction to Parallel Processing - Info.sjsu.edu
4. Guy falls asleep in CS159 lab Purdue - YouTube
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity

**Ranking B**
1. Guy falls asleep in CS159 lab Purdue - YouTube
2. CS 159 Purdue University
3. CS 159: Introduction to Parallel Processing | People | San Jose
4. CS159: Introduction to Parallel Processing - Info.sjsu.edu
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity

**Team-Draft Interleaved Ranking**
1. CS 159 Purdue University
2. Guy falls asleep in CS159 lab Purdue - YouTube
3. CS 159: Introduction to Parallel Processing | People | San Jose
4. CS159: Introduction to Parallel Processing - Info.sjsu.edu
TeamDraft Interleaving

**Ranking A**
1. CS 159 Purdue University
2. CS 159: Introduction to Parallel Processing | People | San Jose
3. CS159: Introduction to Parallel Processing - Info.sjsu.edu
4. Guy falls asleep in CS159 lab Purdue - YouTube
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity

**Ranking B**
1. Guy falls asleep in CS159 lab Purdue - YouTube
2. CS 159 Purdue University
3. CS 159: Introduction to Parallel Processing | People | San Jose
4. CS159: Introduction to Parallel Processing - Info.sjsu.edu
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity

**Team-Draft Interleaved Ranking**
1. CS 159 Purdue University
2. Guy falls asleep in CS159 lab Purdue - YouTube
3. CS 159: Introduction to Parallel Processing | People | San Jose
4. CS159: Introduction to Parallel Processing - Info.sjsu.edu
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
### Team-Draft Interleaved Ranking

1. CS 159 Purdue University
2. Guy falls asleep in CS159 lab Purdue - YouTube
3. CS 159: Introduction to Parallel Processing | People | San Jose
4. CS159: Introduction to Parallel Processing - Info.sjsu.edu
5. CS 159: Advanced Topics in Machine Learning - Yisong Yue
6. CS 159: Introduction to Computational Complexity
### Team-Draft Interleaved Ranking

1. **CS 159 Purdue University**
2. **Guy falls asleep in CS159 lab Purdue - YouTube**
3. **CS 159: Introduction to Parallel Processing | People | San Jose**
4. **CS159: Introduction to Parallel Processing - Info.sjsu.edu**
5. **CS 159: Advanced Topics in Machine Learning - Yisong Yue**
6. **CS 159: Introduction to Computational Complexity**
Algorithm 2 Team-Draft Interleaving

Input: Rankings $A = (a_1, a_2, \ldots)$ and $B = (b_1, b_2, \ldots)$
Init: $I \leftarrow ()$; $TeamA \leftarrow \emptyset$; $TeamB \leftarrow \emptyset$;
while $(\exists i : A[i] \notin I) \land (\exists j : B[j] \notin I)$ do
    if $(|TeamA| < |TeamB|) \lor
        ((|TeamA| = |TeamB|) \land (RandBit() = 1))$ then
        $k \leftarrow \min_i \{i : A[i] \notin I\} \ldots$ top result in $A$ not yet in $I$
        $I \leftarrow I + A[k]$; \hspace{1cm} \ldots$ append it to $I$
        $TeamA \leftarrow TeamA \cup \{A[k]\}$ \hspace{1cm} \ldots credits credited to $A$
    else
        $k \leftarrow \min_i \{i : B[i] \notin I\} \ldots$ top result in $B$ not yet in $I$
        $I \leftarrow I + B[k]$ \hspace{1cm} \ldots append it to $I$
        $TeamB \leftarrow TeamB \cup \{B[k]\}$ \hspace{1cm} \ldots credits credited to $B$
    end if
end while

Output: Interleaved ranking $I, TeamA, TeamB$

Radlinski et.al 2008
# Absolute Metrics: Hypothesis

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Hypothesized Change as Quality Falls</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abandonment Rate</td>
<td>% of queries with no click</td>
<td>Increase</td>
</tr>
<tr>
<td>Reformulation Rate</td>
<td>% of queries that are followed by reformulation</td>
<td>Increase</td>
</tr>
<tr>
<td>Queries per Session</td>
<td>Session = no interruption of more than 30 minutes</td>
<td>Increase</td>
</tr>
<tr>
<td>Clicks per Query</td>
<td>Number of clicks</td>
<td>Decrease</td>
</tr>
<tr>
<td>Clicks @ 1</td>
<td>Clicks on top results</td>
<td>Decrease</td>
</tr>
<tr>
<td>pSkip [Wang et al ’09]</td>
<td>Probability of skipping</td>
<td>Increase</td>
</tr>
<tr>
<td>Max Reciprocal Rank*</td>
<td>1/rank for highest click</td>
<td>Decrease</td>
</tr>
<tr>
<td>Mean Reciprocal Rank*</td>
<td>Mean of 1/rank for all clicks</td>
<td>Decrease</td>
</tr>
<tr>
<td>Time to First Click*</td>
<td>Seconds before first click</td>
<td>Increase</td>
</tr>
<tr>
<td>Time to Last Click*</td>
<td>Seconds before final click</td>
<td>Decrease</td>
</tr>
</tbody>
</table>

(*) only queries with at least one click count

[From Yisong Yue]
None of the metrics reliably reflect expected order.
Results: Pairwise Preferences

<table>
<thead>
<tr>
<th>Comparison Pair</th>
<th>Query Based</th>
<th>User Based</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A &gt; B</td>
<td></td>
</tr>
<tr>
<td></td>
<td>A wins</td>
<td>B wins</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Balanced Interleaving</td>
<td>Orig &gt; Flat</td>
<td>30.6%</td>
</tr>
<tr>
<td></td>
<td>Flat &gt; Rand</td>
<td>28.0%</td>
</tr>
<tr>
<td></td>
<td>Orig &gt; Rand</td>
<td>40.9%</td>
</tr>
<tr>
<td></td>
<td>Orig &gt; Swap2</td>
<td>18.1%</td>
</tr>
<tr>
<td></td>
<td>Swap2 &gt; Swap4</td>
<td>33.6%</td>
</tr>
<tr>
<td></td>
<td>Orig &gt; Swap4</td>
<td>32.1%</td>
</tr>
<tr>
<td>Team-Draft Interleaving</td>
<td>Orig &gt; Flat</td>
<td>47.7%</td>
</tr>
<tr>
<td></td>
<td>Flat &gt; Rand</td>
<td>46.7%</td>
</tr>
<tr>
<td></td>
<td>Orig &gt; Rand</td>
<td>55.6%</td>
</tr>
<tr>
<td></td>
<td>Orig &gt; Swap2</td>
<td>44.4%</td>
</tr>
<tr>
<td></td>
<td>Swap2 &gt; Swap4</td>
<td>44.2%</td>
</tr>
<tr>
<td></td>
<td>Orig &gt; Swap4</td>
<td>47.7%</td>
</tr>
</tbody>
</table>

Radlinski et.al 2008

- **RECALL**: Orig > Flat > Rand. Orig > Swap2 > Swap4.
- Correct implications. Significant.
- Let $\Delta_{AB} := \text{wins}(A) - \text{wins}(B)$. Note, for $A > B > C$, $\Delta_{AC} > \max\{\Delta_{AB}, \Delta_{BC}\}$, indicating Strong Stochastic Transitivity
Deployment on Yahoo! Search Engine

Comparing Two Ranking Functions

- Interleaving is more sensitive and more reliable

Absolute Metrics
E.g., #Clicks@1, Total #Clicks, etc.

Each ranking function receives 50% traffic

[Chapelle, Joachims, Radlinski & Yue, TOIS 2012]
[From Yisong Yue]
Problem Formulation
Recall the Standard Multi-armed Bandit Problem

Definitions:

- $T$ rounds
- A set of bandits $\{b_1, \ldots, b_K\}$
- Each bandit has a stationary reward distribution

Standard Multi-armed Bandits Procedure

- **Choose** one bandit $b_i$ from $\{b_1, \ldots, b_K\}$ each round
- **Receive Reward** drawn from $b_i$’s distribution
- **Receive Feedback** by being told your reward
Example: Retrieval Functions

Suppose Google has developed 10 new retrieval functions

**Goal:** Interactivity learn the best retrieval function

**What if we Apply Standard Multi-armed Bandits?**

- Each function is a bandit
- Assumes clicks ⇒ explicit absolute feedback
- As described at the beginning of the talk, this won’t work
The Dueling Bandit Problem

Definitions:

- $T$ rounds
- A set of bandits $\{b_1, \ldots, b_K\}$
- The probability of $b_i$ beating $b_j$ depends only on $i$ and $j$

Dueling Bandits Procedure

- **Choose** two bandits $b_i, b_j$ from $\{b_1, \ldots, b_K\}$ each round
- **Receive Reward** based on the (unknown) probabilities that $b_i$ and $b_j$ individually beat the best bandit
- **Receive Feedback** by being told the winner of the duel between $b_i$ and $b_j$

(Maximum reward is if the best bandit always duels itself)
Example: Retrieval Functions

Suppose Google has developed 10 new retrieval functions

**Goal:** Interactivity learn the best retrieval function

### How to Apply Dueling Bandits

- $\{b_1, ..., b_K\}$ = the set of retrieval functions
- For each user query, you interleave the results from two ranking algorithms: $b_1^{(t)}$ and $b_2^{(t)}$ to present to the user to elicit a pairwise comparison
- You want to present the best possible ranking. Hence the necessity of the regret formulation to minimize:

$$R_T = \sum_{t=1}^{T} \text{avg}\{\epsilon(b^*, b_1^{(t)}), \epsilon(b^*, b_2^{(t)})\}$$
Visualizing the Example

Interleave A vs B

<table>
<thead>
<tr>
<th></th>
<th>Left wins</th>
<th>Right wins</th>
</tr>
</thead>
<tbody>
<tr>
<td>A vs B</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>A vs C</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B vs C</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

[From Yisong Yue]
Visualizing the Example

Interleave A vs C

<table>
<thead>
<tr>
<th></th>
<th>Left wins</th>
<th>Right wins</th>
</tr>
</thead>
<tbody>
<tr>
<td>A vs B</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>A vs C</td>
<td>0</td>
<td><strong>1</strong></td>
</tr>
<tr>
<td>B vs C</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

[From Yisong Yue]
Visualizing the Example

Interleave B vs C

<table>
<thead>
<tr>
<th></th>
<th>Left wins</th>
<th>Right wins</th>
</tr>
</thead>
<tbody>
<tr>
<td>A vs B</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>A vs C</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>B vs C</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

[From Yisong Yue]
Visualizing the Example

<table>
<thead>
<tr>
<th></th>
<th>Left wins</th>
<th>Right wins</th>
</tr>
</thead>
<tbody>
<tr>
<td>A vs B</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>A vs C</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>B vs C</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

[From Yisong Yue]
Dueling Bandits Problem

Goal: Maximize total user utility

Exploit: run C
(interleave C with itself)

Explore: interleave A vs B

Best: A
(interleave A with itself)

How to interact optimally?

<table>
<thead>
<tr>
<th></th>
<th>Left wins</th>
<th>Right wins</th>
</tr>
</thead>
<tbody>
<tr>
<td>A vs B</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>A vs C</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>B vs C</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

[From Yisong Yue]
Formal Framework and Notation

- **Given**
  - \(\{b_1, ..., b_K\}\) = the set of \(K\) bandits (aka. arms, actions)
  - \(T\) = Time horizon (aka. number of rounds)

- **Assume**
  - The probability that \(b_i\) defeats \(b_j\) in a duel depends only on \(i, j\) and is unknown
    - \(P(b_i > b_j)\) is denoted by \(\epsilon(b_i, b_j) + \frac{1}{2}\) or \(\epsilon_{i,j} + \frac{1}{2}\)
    - Can be interpreted as the fraction of users that prefer \(b_i\) to \(b_j\)
    - Each duel is independent
  - The strongest bandit is denoted \(b^*\)

- **For each round** \(t\)
  - Algorithm selects two bandits, \(b_1^{(t)}\) and \(b_2^{(t)}\) to duel
  - Add \(\text{AVG of } \{\epsilon(b^*, b_1^{(t)}), \epsilon(b^*, b_2^{(t)})\}\) to our regret
  - Algorithms is told the winner of the duel.

- **Goal:** minimize total regret at time \(T\): \(R_T\)
What This Means: The $\epsilon_{i,j}$ Matrix

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>0.03</td>
<td>0.04</td>
<td>0.06</td>
<td>0.10</td>
<td>0.11</td>
</tr>
<tr>
<td>B</td>
<td>-0.03</td>
<td>0</td>
<td>0.03</td>
<td>0.05</td>
<td>0.08</td>
<td>0.11</td>
</tr>
<tr>
<td>C</td>
<td>-0.04</td>
<td>-0.03</td>
<td>0</td>
<td>0.04</td>
<td>0.07</td>
<td>0.09</td>
</tr>
<tr>
<td>D</td>
<td>-0.06</td>
<td>-0.05</td>
<td>-0.04</td>
<td>0</td>
<td>0.05</td>
<td>0.07</td>
</tr>
<tr>
<td>E</td>
<td>-0.10</td>
<td>-0.08</td>
<td>-0.07</td>
<td>-0.05</td>
<td>0</td>
<td>0.03</td>
</tr>
<tr>
<td>F</td>
<td>-0.11</td>
<td>-0.11</td>
<td>-0.09</td>
<td>-0.07</td>
<td>-0.03</td>
<td>0</td>
</tr>
</tbody>
</table>

• Values are $\Pr(\text{row} > \text{col}) - 0.5$
• Bandit Order: $A > B > C > D > E > F$

[From Yisong Yue]
Assumptions on $\epsilon_{i,j}$

Recall $\epsilon_{i,j} = P(b_i > b_j) - \frac{1}{2}$

- **Symmetry** $\epsilon_{i,j} = -\epsilon_{j,i}$ (implicit in this is that $\epsilon_{i,i} = 0$)
- **Total Ordering** $\exists$ an ordering where $b_i > b_j \Rightarrow \epsilon_{i,j} > 0$
- **Strong Stochastic Transitivity** $b_i > b_j \Rightarrow \forall k \epsilon_{i,k} \geq \epsilon_{j,k}$
- **Stochastic Triangle Inequality** $b_i > b_j \Rightarrow \forall k \epsilon_{i,j} \leq \epsilon_{i,k} + \epsilon_{k,j}$ (or the weaker condition: $\frac{\epsilon_{i,j}}{\epsilon_{i,k} + \epsilon_{k,j}}$ is bounded)
What This Means: Strong Stochastic Transitivity

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>0.03</td>
<td>0.04</td>
<td>0.06</td>
<td>0.10</td>
</tr>
<tr>
<td>B</td>
<td>-0.03</td>
<td>0</td>
<td>0.03</td>
<td>0.05</td>
<td>0.08</td>
</tr>
<tr>
<td>C</td>
<td>-0.04</td>
<td>-0.03</td>
<td>0</td>
<td>0.04</td>
<td>0.07</td>
</tr>
<tr>
<td>D</td>
<td>-0.06</td>
<td>-0.05</td>
<td>-0.04</td>
<td>0</td>
<td>0.05</td>
</tr>
<tr>
<td>E</td>
<td>-0.10</td>
<td>-0.08</td>
<td>-0.07</td>
<td>-0.05</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>-0.11</td>
<td>-0.11</td>
<td>-0.09</td>
<td>-0.07</td>
<td>-0.03</td>
</tr>
</tbody>
</table>

- Values are $\Pr(\text{row} > \text{col}) - 0.5$
- Bandit Order: $A > B > C > D > E > F$

[From Yisong Yue]
What This Means: Stochastic Triangle Inequality

The probability of a bandit winning will exhibit diminishing returns as it becomes increasingly superior

\[ \mathcal{E}_{ik} \leq \mathcal{E}_{ij} + \mathcal{E}_{jk} \]

Red \leq Blue + Green

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>0.03</td>
<td>0.04</td>
<td>0.06</td>
<td>0.10</td>
<td>0.11</td>
</tr>
<tr>
<td>B</td>
<td>-0.03</td>
<td>0</td>
<td>0.03</td>
<td>0.05</td>
<td>0.08</td>
<td>0.11</td>
</tr>
<tr>
<td>C</td>
<td>-0.04</td>
<td>-0.03</td>
<td>0</td>
<td>0.04</td>
<td>0.07</td>
<td>0.09</td>
</tr>
<tr>
<td>D</td>
<td>-0.06</td>
<td>-0.05</td>
<td>-0.04</td>
<td>0</td>
<td>0.05</td>
<td>0.07</td>
</tr>
<tr>
<td>E</td>
<td>-0.10</td>
<td>-0.08</td>
<td>-0.07</td>
<td>-0.05</td>
<td>0</td>
<td>0.03</td>
</tr>
<tr>
<td>F</td>
<td>-0.11</td>
<td>-0.11</td>
<td>-0.09</td>
<td>-0.07</td>
<td>-0.03</td>
<td>0</td>
</tr>
</tbody>
</table>

- Values are \( \Pr(\text{row} > \text{col}) - 0.5 \)
- Bandit Order: \( A > B > C > D > E > F \)

[From Yisong Yue]
The Algorithm
Main algorithm has 2 phases:

1. **EXPLORE**: Find best bandit, $\hat{b}$
   - If the algorithm works with probability $\geq 1 - \frac{1}{T}$ there is only a constant penalty in terms of expected regret

2. **EXPLOIT**: As long as there are rounds left, play $(\hat{b}, \hat{b})$
   - If the explore phase correctly found the best bandit ($\hat{b}$), then this phase has no regret
What if we Explored using a Tournament Bracket?

**Idea:** Setup a tournament bracket. Each pair "duels" until the winner has statistical significance.

But if we have two bad but equal bandits, they will play each other for a long time:

Problem: two equally bad bandits

⇒ large regret
Interleaved Filter (IF): A Better EXPLORE Algorithm

Each cycle is a “round”

Choose candidate bandit at random

Make noisy comparisons (Bernoulli trial) against all other bandits simultaneously

• Maintain mean and confidence interval for each pair

...until another bandit is better

• With confidence $1 - \delta$

Change our estimate of the best bandit

• Remove all empirically worse bandits

Repeat above cycle until 1 candidate left

[From Yisong Yue]
Interleaved Filter (IF): A Better EXPLORE Algorithm

**Input:** $\mathcal{B} = \{b_1, ..., b_K\}$
Choose $\hat{b} \in \mathcal{B}$ randomly (best bandit found so far)
$W \leftarrow \mathcal{B} \setminus \{\hat{b}\}$ (remaining potential best bandit)

while $W \neq \emptyset$ do
  \[
  \forall b \in W: \text{run the duel} \ (\hat{b}, b) \ \text{and add result to a running tally}
  \]
  Remove all $b \in W$ where $b < \hat{b}$ within a confidence interval
  if $\exists b' \in W$ s.t. $b' > \hat{b}$ within a confidence interval then
    Remove all $b \in W$ which lost more often than they won
    $\hat{b} \leftarrow b'$, $W \leftarrow W \setminus \{b'\}$
  end
end
return $\hat{b}$

(The confidence interval = (empirical mean) $\pm \sqrt{4 \log(TK^2)/t}$)
Theorem 1. Using IF as the EXPLORE Algorithm with $\mathcal{B} = \{b_1, \ldots, b_K\}$, time horizon $T \ (T \geq K)$, and IF incurs expected regret bounded by

$$E[R_T] = O(E[R_{IF}^T]) = O\left(\frac{K}{\epsilon_{1,2}} \log T\right)$$

We prove Theorem 1 by showing the following three lemmas.
Lemma 1. The probability that IF makes a mistake resulting in the elimination of the best bandit $b_1$ is at most $\frac{1}{T}$.

Lemma 2. Assuming IF is mistake-free, then, with high probability, $R_{IF}^T = O\left(\frac{K \log K}{\epsilon_{1,2}} \log T\right)$.

Lemma 3. Assuming IF is mistake-free, then $E[R_{IF}^T] = O\left(\frac{K}{\epsilon_{1,2}} \log T\right)$. 
Once we prove Lemma 1 and 3, Theorem 1 follows because IF correctly returns the best bandit with probability at least $1 - \frac{1}{T}$. Correspondingly, a suboptimal bandit is returned with probability at most $\frac{1}{T}$, in which case we can assume maximal regret of $O(T)$. Then

$$E[R_T] \leq (1 - \frac{1}{T})E[R_{IF}^T] + \frac{1}{T}O(T)$$

$$= O(E[R_{IF}^T] + 1)$$

$$= O(E[R_{IF}^T])$$

$$= O\left(\frac{K}{\epsilon_{1,2}} \log T\right)$$
To aid our analysis, we introduce

Confidence Intervals:

IF maintains a number \( \hat{P}_{i,j} = \frac{\# \text{ \# wins}}{\# \text{ comparisons}} \) in a match between \( b_i \) and \( b_j \). In the following presentation, we drop the subscripts \((i, j)\) and use \( \hat{P}_t \), where \( t \) is the \# of comparisons.

IF also maintains a confidence interval \( \hat{C}_t = (\hat{P}_t - c_t, \hat{P}_t + c_t) \) where \( c_t = \sqrt{\frac{4 \log(1/\delta)}{t}} \) where \( \delta = 1/(TK^2) \).
Lemma 4. For $\delta = \frac{1}{TK^2}$, the number of comparisons in a match between $b_i$ and $b_j$ is with high probability at most $O\left(\frac{1}{e_{i,j}^2} \log(TK)\right)$. Moreover, the probability that the inferior bandit is declared the winner at some time $t \leq T$ is at most $\delta$.

Proof Sketch:

Stopping condition of a match between $b_i$ and $b_j$: 
$\exists t, \hat{P}_t - c_t > \frac{1}{2}$. (Corresponds to the confidence interval condition in the Algorithm)

Let $n$ be the number of comparisons between $b_i$ and $b_j$, then 
$P(n > t) \leq P(\hat{P}_t - c_t \leq \frac{1}{2}) = P(\hat{P}_t - \frac{1}{2} - \epsilon_{i,j} \leq c_t - \epsilon_{i,j}) = P(E[\hat{P}_t] - \hat{P}_t \geq \epsilon_{i,j} - c_t)$. Apply Hoeffding’s inequality to get the desired result. Similar procedure applies to showing the second part.
Lemma 4 bounds the number of comparisons in each match, next we bound the resulting regret of each match.

**Lemma 5.** Assuming $b_1$ has not been removed and $T \geq K$, then with high probability the accumulated regret from any match is at most $O\left(\frac{1}{\epsilon_{1,2}} \log T\right)$.

**Proof Sketch:**
Suppose the candidate $\hat{b} = b_j$ is playing a match against $b_i$. By Lemma 4, a match played by $b_j$ contains at most $O\left(\frac{1}{\epsilon_{1,j}} \log(TK)\right) = O\left(\frac{1}{\epsilon_{1,2}} \log(TK)\right)$ comparisons.
Since $\min\{\epsilon_{1,j}, \epsilon_{1,i}\} \leq \epsilon_{1,j}$, so the accumulated (weak) regret is bounded by

$$
\epsilon_{1,j} O\left(\frac{1}{\epsilon_{1,j}^2} \log(TK)\right) = O\left(\frac{1}{\epsilon_{1,j}} \log(TK)\right)
$$

$$
= O\left(\frac{1}{\epsilon_{1,2}} \log(TK)\right)
$$

$$
= O\left(\frac{1}{\epsilon_{1,2}} \log T\right) \text{ (since } K \leq T)\n$$

■
Analysis of Algorithm

We need one more lemma that bounds the probability that IF makes a mistake.

**Lemma 6.** For all triples of bandits $b, b', \hat{b}$ such that $b > b'$, the probability that IF eliminates $b$ in a pruning step in which $b'$ wins a match against the incumbent bandit $\hat{b}$ (i.e. $\hat{P}_{\hat{b}, b'} < \frac{1}{2}$) while $b$ is found to be empirically inferior to $\hat{b}$ (i.e. $\hat{P}_{\hat{b}, b} > \frac{1}{2}$) is at most $\delta = \frac{1}{TK^2}$. 
Recall **Lemma 1**: The probability that IF makes a mistake resulting in the elimination of the best bandit $b_1$ is at most $\frac{1}{T}$.

**Proof of Lemma 1:**
By Lemma 4, the probability that $b_1$ loses to any $b_i$ in a direct match is at most $\delta$. By a union bound, the probability that $b_1$ is eliminated in a direct match is at most $(K - 1)\delta$.

By Lemma 6, the probability that $b_1$ is eliminated in a pruning step is at most $\delta$. By a union bound, the probability that $b_1$ is eliminated in a pruning step is at most $(K - 1)^2\delta$.

So the probability that IF eliminate $b_1$ is at most $\delta[(K - 1) + (K - 1)^2] < \delta K^2 = \frac{1}{T}$ ■
Next we sketch the proof for lemma 2, which states the mistake-free executions of IF satisfy $R_{IF}^T = O\left(\frac{K \log K}{\epsilon_{1,2}} \log T\right)$. The key point in the proof is to obtain an upper bound on the number of matches IF plays. To do this, we introduce a random walk model.
Definition (Random Walk Model). Define a random walk graph with $K$ nodes labeled $b_1, \cdots, b_K$ (these will correspond to the similarly named bandits). Each node $b_j (j > 1)$ transitions to $b_i$ for $j > i \geq 1$ with probability $\frac{1}{j-1}$. The final node $b_1$ is an absorbing node.

![Random Walk Graph Illustration](image)

Fig. 1. An illustrative example of a sequence of candidate bandits. The incumbent candidate in each round is shaded in grey.
A path in the Random Walk Model corresponds to a sequence of candidate bandits taken by IF where $\epsilon_{1,j} = \epsilon_{2,j} = \cdots = \epsilon_{j-1,j}$ for all $j > 1$

It turns out that the number of rounds in the execution of IF is stochastically bounded by the path length of a random walk in the Random Walk Model, i.e. if $S$ and $\tilde{S}$ are random variables corresponding to the two quantities, then

$\forall x : P(S \geq x) \leq P(\tilde{S} \geq x)$. Using this property, we can show with high probability, a mistake-free execution of IF runs for $O(\log K)$ rounds. And lemma 2 follows.
We now prove Lemma 3, which claims that mistake-free executions of IF satisfy $E[R_{T}^{IF}] = O(\frac{K}{\epsilon_{1,2}} \log T)$.

Recall that by Lemma 5, for a mistake-free execution of IF and $T \geq K$, with high probability the accumulated regret from any match is at most $O(\frac{1}{\epsilon_{1,2}} \log T)$.

Lemma 3 directly follows from Lemma 5 and the following: **Lemma 9.** Assuming IF is mistake-free, there are $O(K)$ matches in expectation.
**Proof Sketch:**

Let $B_j$ be the number of matches played by $b_j$ when it is not the incumbent. Furthermore, let $A_j$ be the number of matches played by $b_j$ against $b_i$ for $i > j$, and $G_j$ be the number of matches when $i < j$ ($b_i$ is incumbent).

Then the expected number of matches is

$$
\sum_{j=1}^{K-1} E[B_j] = \sum_{j=1}^{K-1} E[A_j] + E[G_j].
$$

Leveraging the Random Walk Model, it can be shown that $E[A_j] \leq 1 + H_{K-1} - H_j$, where $H_j$ is the harmonic sum.
Next, we show that $E[G_j] \leq 2$.

Quick justification: probability that $b_j$ is not pruned in a match against a superior incumbent bandit $b_i$ is less than half. So, in expectation, it takes two such matches for $b_j$ to get pruned. ■

That completes the proof of Lemma 3, and thus Theorem 1. The bound in Theorem 1 is in fact information theoretically optimal upto constant factors, which is the content of Theorem 2.
Theorem 2. For any fixed $\epsilon > 0$ and any algorithm $\phi$ for the dueling bandits problem, there exists a problem instance such that

$$R^\phi_T = \Omega\left(\frac{K}{\epsilon \log T}\right)$$

where $\epsilon = \min_{b \neq b^*} P(b^* > b)$.  


The proof is similar to the lower bound proof for multi-armed bandit problem and is omitted. However, here is a heuristic explanation for why we might suspect the theorem to be true: Given a bandit \( b \), suppose we need to determine with high probability whether \( b \) is the best bandit. We know that given two bandits, we can identify the better bandit with probability at least \( 1 - 1/T \) after \( O(\log T/\epsilon^2) \) comparisons. Since there are \( K \) bandits, we can expect to take \( K \) times \( O(\log T/\epsilon^2) \) comparisons to determine whether bandit \( b \) is the best.
Extensions
where $Q_{ij}$ is the probability arm $i$ beats arm $j$. 

Busa-Fekete 2014
Assumption: There exists a Condorcet winner: a bandit that beats all other bandits in expectation.

RUCB doesn't require an input horizon $T$.
- Interleaved Filter, Beat-the-mean, SAVAGE require finite-time horizon.

No need to guess exploration horizon
- More useful in practice

Finite-time regret bound of order $O(\log t)$
RUCB Algorithm

For each time $t = 1, \ldots$:

1. Put all arms in a pool of potential champions.

2. Compare each arm $a_i$ against all other arms optimistically:
   - Compute $u_{ij}(t) = \mu_{ij}(t) + c_{ij}(t)$, where $c_{ij}(t)$ is confidence bound, $\mu_{ij}(t)$ is estimate of $p_{ij}$ so far.
   - If $u_{ij} < \frac{1}{2}$ for any $j$, remove $a_i$ from pool of champions.
   - Randomly choose champion arm $a_c$ from remaining potential champions
     - **Intuition:** Comparisons in 2. are optimistic, so $a_c$ becomes champion easily.

3. Perform regular UCB using $a_c$ as benchmark. Choose $d = \arg \max_j u_{jc}$
   - **Intuition:** Avoids comparing $a_c$ with itself unless $a_c$ is Condorcet winner.

4. Choose $(a_c, a_d)$ to compare.
Algorithm 1 Relative Upper Confidence Confidence Bound

Input: $\alpha > \frac{1}{2}$, $T \in \{1, 2, \ldots\} \cup \{\infty\}$

1: $W = [w_{ij}] \leftarrow 0_{K \times K}$ // 2D array of wins: $w_{ij}$ is the number of times $a_i$ beat $a_j$

2: for $t = 1, \ldots, T$ do

3: $U := [u_{ij}] = \frac{W}{W + W^T} + \sqrt{\frac{\alpha \ln t}{W + W^T}}$ // All operations are element-wise; $\frac{x}{0} := 1$ for any $x$.

4: $u_{ii} \leftarrow \frac{1}{2}$ for each $i = 1, \ldots, K$.

5: Pick any $c$ satisfying $u_{cj} \geq \frac{1}{2}$ for all $j$. If no such $c$, pick $c$ randomly from $\{1, \ldots, K\}$.

6: $d \leftarrow \arg \max_j u_{jc}$

7: Compare arms $a_c$ and $a_d$ and increment $w_{cd}$ or $w_{dc}$ depending on which arm wins.

8: end for

Return: An arm $a_c$ that beats the most arms, i.e., $c$ with the largest count $\# \left\{ j \mid \frac{w_{cj}}{w_{cj} + w_{jc}} > \frac{1}{2} \right\}$.
RUCB Results

Zoghi 2014
What if Condorcet winner doesn’t exist?
- Stochastic transitivity not satisfied (sports competitions)
- Copeland winner always exists
  - Copeland score: Number of pairwise victories minus number of pairwise defeats.

$O(K \log T)$ regret bound without restrictions
- Previous results: $O(K^2 \log T)$ or $O(K \log T)$ with restrictions
Other extensions

- **Borda winner**: arm $a_b$ that satisfies $\sum_j p_{bj} \geq \sum_j p_{ij}$ for all $i = 1, \ldots, K$
  - When averaged across all arms, $a_b$ has the highest probability of winning a given comparison

- **von Neumann winner**: has at least a 50% chance of winning
  - Allows for randomized policies

- **Convex, continuous setting**
  - Actions are comparisons between $w, w' \in \mathcal{W}$, for compact, convex set $\mathcal{W}$.
  - Sublinear regret $O(T^{3/4})$
Application: Personalized Clinical Treatment

(with Yanan Sui, Vincent Zhuang and Joel Burdick)

Each patient is unique
$10^9$ possible configurations!

[From Yisong Yue]
Conclusion

- Absolute metrics are insufficient in comparing rankings
- Dueling bandits useful for pairwise comparisons
- Interleaved filter: explore, then exploit
  - Achieves sublinear regret $O\left(\frac{K}{\epsilon_{1,2}} \log T\right)$
  - Matches theoretical lower bound
- Extensions
  - RUCB
  - Copeland bandits
  - Personalized Medical Treatment
References

- ‘A Survey of Preference-Based Online Learning with Bandit Algorithms’, Róbert Busa-Fekete and Eyke Hüllermeier, 2014
- ‘Contextual Dueling Bandits’, Dudik et. al., 2015.