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

Information retrieval: Ranking Problem

- 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 ...

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: ...

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**/ ▾
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.

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**/ ▾
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 ...

CS159: Introduction to Parallel Processing | Pe... o...
www.sjsu.edu › ... › Chun, Robert K › Courses ▾ San Jose...
Jan 20, 2015 - Description. A combination hardware architecture ... development class focused on multi-threaded, parallel processing algorith...s ...

Click!

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...
www.yisongyue.com/courses/**cs159**/ ▾
CS 159: Advanced Topics in Machine Learning (Spring 2016). Course ... Th... course will cover a mixture of the following topics: Online Learning ...

Click!

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.

Retrieval Function B

Guy falls asleep in CS159 lab Purdue - YouTube

https://www.youtube.com/watch?v=vV...
Mar 24, 2011 - Uploaded by james brand
Guy falls asleep in our 7:30 am lab so we tak... the volume up to full and call him.

Click!

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 ...

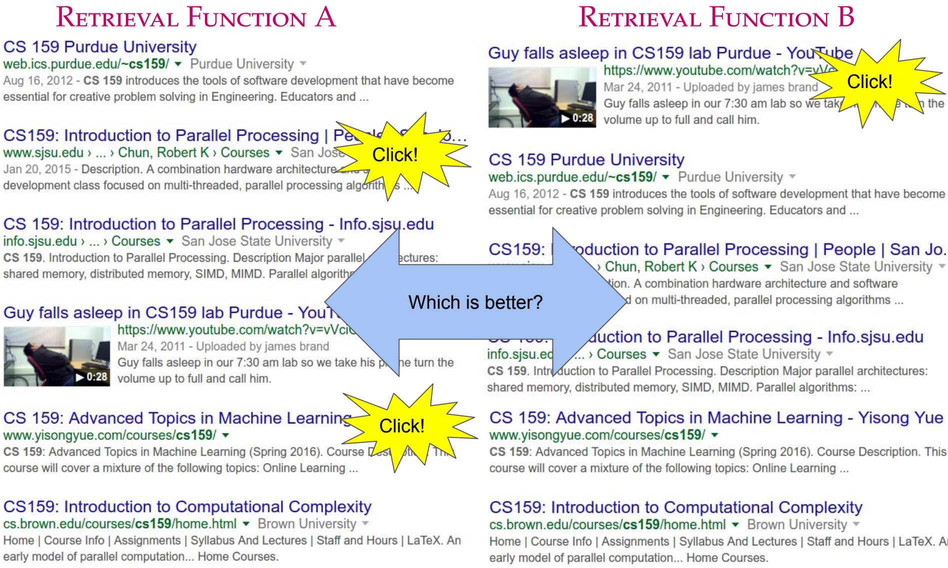
CS159: Introduction to Parallel Processing | People | San Jo...
www.sjsu.edu › ... › Chun, Robert K › Courses ▾ San Jose State University ▾
Jan 20, 2015 - Description. A combination hardware architecture and software development class focused on multi-threaded, parallel processing algorithms ...

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**/ ▾
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. A... early model of parallel computation... Home Courses.

Example: Evaluation Search Rankings



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 $A \succ B$, or $B \succ A$
    - Inspires dueling bandits bandits

Smaller scale: arXiv

Cornell University Library

We gratefully acknowledge support from
the Simons Foundation
and California Institute of Technology

arXiv.org > search

Search or Article-id

(Help | Advanced search)

All papers ▼ Go!

arXiv.org Search Results

Back to Search form

The URL for this search is http://arxiv.org:443/find/all/1/all:+AND+dueling+bandits/0/1/0/all/0/1

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)

3. **arXiv:1602.02706 [pdf, other]**
   Indistinguishable Bandits Dueling with Decoys on a Poset
   Julien Audiffren (CMLA), Ralaivola Liva (LIF)
   Subjects: **Learning (cs.LG)**; Artificial Intelligence (cs.AI)

4. **arXiv:1601.03855 [pdf, other]**
   A Relative Exponential Weighing Algorithm for Adversarial Utility-based Dueling Bandits
   Pratik Gajane, Tanguy Urvoy, Fabrice Clérot (FT R and D)
   Journal-ref: The 32nd International Conference on Machine Learning, Jul 2015, Lille, France. 37, pp.218-227, Proceedings of The 32nd International Conference on Machine Learning
   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 $\succ$ Flat $\succ$ Rand
  - ORIG: Hand-tuned ranking function
  - FLAT: No field weights
  - RAND: Randomize top 11 results in Flat
  - Substantial distinction
- Orig $\succ$ Swap2 $\succ$ 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

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

Team-Draft Interleaved Ranking

1.

A

http://www.thefreedictionary.com/flip

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

Team-Draft Interleaved Ranking

1. CS 159 Purdue University

A

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

Team-Draft Interleaved Ranking

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

A

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

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

B

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

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

B

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

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

B

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

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

A

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

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

A

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

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

A

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

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

TeamDraft Interleaving

Algorithm 2 Team-Draft Interleaving

Input: Rankings $A = (a_1, a_2, \dots)$ and $B = (b_1, b_2, \dots)$
Init: $I \leftarrow (); TeamA \leftarrow \emptyset; TeamB \leftarrow \emptyset;$
while $(\exists i : A[i] \notin I) \wedge (\exists j : B[j] \notin I)$ **do**
  if $(|TeamA| < |TeamB|) \vee$
    $((|TeamA| = |TeamB|) \wedge (RandBit() = 1))$ **then**
    $k \leftarrow \min_i \{i : A[i] \notin I\}$ ..... *top result in A not yet in I*
    $I \leftarrow I + A[k];$ .... *append it to I*
    $TeamA \leftarrow TeamA \cup \{A[k]\}$ ...... *clicks credited to A*
  else
    $k \leftarrow \min_i \{i : B[i] \notin I\}$ ..... *top result in B not yet in I*
    $I \leftarrow I + B[k]$ .... *append it to I*
    $TeamB \leftarrow TeamB \cup \{B[k]\}$ ...... *clicks credited to B*
  end if
end while
Output: Interleaved ranking I, $TeamA$, $TeamB$

Radlinski et.al 2008

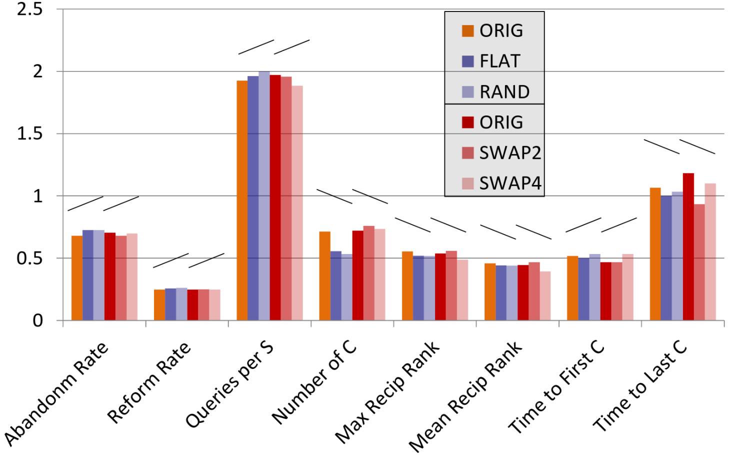
Absolute Metrics: Hypothesis

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

(*) only queries with at least one click count

[From Yisong Yue]

Absolute Metrics: Results



[Radlinski et al. 2008]
[From Yisong Yue]

None of the metrics reliably reflect expected order.

Results: Pairwise Preferences

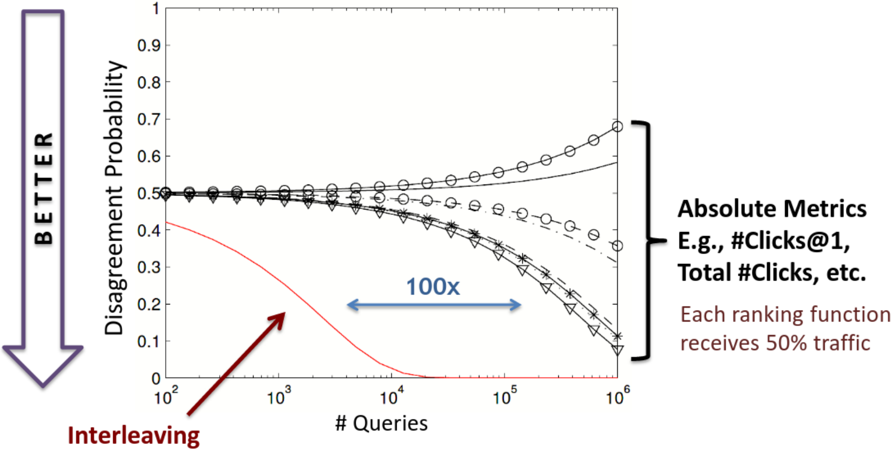
	Comparison Pair A ≻ B	Query Based			User Based		
		A wins	B wins	# queries	A wins	B wins	# users
Balanced Interleaving	Orig ≻ Flat	**30.6%**	**21.9%**	857	**33.3%**	**23.8%**	538
	Flat ≻ Rand	**28.0%**	**22.9%**	907	**31.8%**	**23.3%**	529
	Orig ≻ Rand	**40.9%**	**30.1%**	930	**41.0%**	**27.1%**	553
	Orig ≻ Swap2	**18.1%**	**14.6%**	1035	**23.1%**	**17.1%**	589
	Swap2 ≻ Swap4	**33.6%**	**27.5%**	1061	35.1%	30.0%	606
	Orig ≻ Swap4	**32.1%**	**24.5%**	1173	**37.7%**	**26.7%**	591
Team-Draft Interleaving	Orig ≻ Flat	**47.7%**	**37.3%**	1272	**49.6%**	**36.0%**	667
	Flat ≻ Rand	**46.7%**	**39.7%**	1376	**46.3%**	**36.8%**	646
	Orig ≻ Rand	**55.6%**	**29.8%**	1095	**58.7%**	**28.6%**	622
	Orig ≻ Swap2	44.4%	40.3%	1170	**44.7%**	**37.4%**	693
	Swap2 ≻ Swap4	44.2%	40.3%	1202	45.1%	39.8%	703
	Orig ≻ Swap4	**47.7%**	**37.8%**	1332	**47.2%**	**35.0%**	697

Radlinski et.al 2008

- Recall: Orig > Flat > Rand. Orig > Swap2 > Swap4.
- Correct implications. Significant.
- Let $\Delta_{AB} := wins(A) - wins(B)$. 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**

[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 $\Rightarrow$ 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

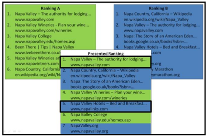
Goal: Interactivity learn the best retrieval function

How to Apply Dueling Bandits

- $\{b_1, \ldots, b_K\}$ = 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



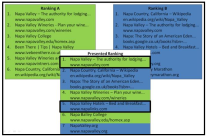
Interleave A vs B

. . .

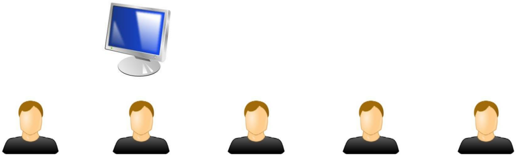
	Left wins	Right wins
A vs B	0	**1**
A vs C	0	0
B vs C	0	0

[From Yisong Yue]

Visualizing the Example



Interleave A vs C

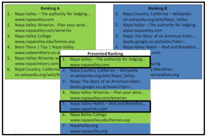


. . .

	Left wins	Right wins
A vs B	0	1
A vs C	0	**1**
B vs C	0	0

[From Yisong Yue]

Visualizing the Example



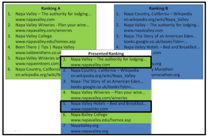
Interleave B vs C

. . .

	Left wins	Right wins
A vs B	0	1
A vs C	0	1
B vs C	0	1

[From Yisong Yue]

Visualizing the Example



Interleave A vs C

	Left wins	Right wins
A vs B	0	1
A vs C	1	1
B vs C	0	1

[From Yisong Yue]

Visualizing the Example

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?

	Left wins	Right wins
A vs B	0	1
A vs C	1	1
B vs C	0	1

[From Yisong Yue]

Formal Framework and Notation

- Given
  - $\{b_1, \ldots, b_K\}$ = 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 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

	A	B	C	D	E	F
A	**0**	**0.03**	**0.04**	**0.06**	**0.10**	**0.11**
B	-0.03	0	0.03	0.05	0.08	0.11
C	-0.04	-0.03	0	0.04	0.07	0.09
D	-0.06	-0.05	-0.04	0	0.05	0.07
E	-0.10	-0.08	-0.07	-0.05	0	0.03
F	-0.11	-0.11	-0.09	-0.07	-0.03	0

Regrets

- Values are Pr(row > 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

Monotonic →

↑ Monotonic

	A	B	C	D	E	F
A	0	0.03	0.04	0.06	0.10	0.11
B	-0.03	0	0.03	0.05	0.08	0.11
C	-0.04	-0.03	0	0.04	0.07	0.09
D	-0.06	-0.05	-0.04	0	0.05	0.07
E	-0.10	-0.08	-0.07	-0.05	0	0.03
F	-0.11	-0.11	-0.09	-0.07	-0.03	0

- Values are Pr(row > 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

$$\varepsilon_{ik} \leq \varepsilon_{ij} + \varepsilon_{jk}$$

Red ≤ Blue + Green

	A	B	C	D	E	F
A	0	0.03	0.04	0.06	0.10	0.11
B	-0.03	0	0.03	0.05	0.08	0.11
C	-0.04	-0.03	0	0.04	0.07	0.09
D	-0.06	-0.05	-0.04	0	0.05	0.07
E	-0.10	-0.08	-0.07	-0.05	0	0.03
F	-0.11	-0.11	-0.09	-0.07	-0.03	0

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

[From Yisong Yue]

The Algorithm

Explore then Exploit

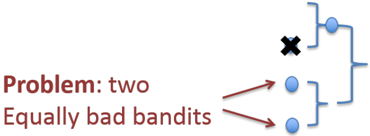
Main algorithm has 2 phases:

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

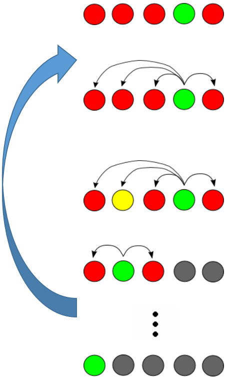
But if we have 2 bad but equal bandits, they will play each other for a long time:



$\Rightarrow$ 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: $\mathcal{B} = \{b_1, \ldots, b_K\}$
Choose $\hat{b} \in \mathcal{B}$ randomly (best bandit found so far)
$W \leftarrow \mathcal{B} \backslash \{\hat{b}\}$ (remaining potential best bandit)
while $W \neq \emptyset$ **do**
  $\forall b \in W$: run the duel $(\hat{b}, b)$ 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
    $\hat{b} \leftarrow b', W \leftarrow W \backslash \{b'\}$
    Reset the running tally
  end
end
return $\hat{b}$

(The confidence interval = (empirical mean) $\pm\sqrt{4\log(TK^2)/t}$)

Analysis of Algorithm

Theorem 1. Using IF as the EXPLORE Algorithm with $\mathcal{B} = \{b_1, \cdots, b_K\}$, time horizon T $(T \geq K)$, and IF incurs expected regret bounded by

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

We prove Theorem 1 by showing the following three lemmas.

Analysis of Algorithm

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, $R_T^{IF} = O(\frac{K \log K}{\epsilon_{1,2}} \log T)$.

Lemma 3. Assuming IF is mistake-free, then $E[R_T^{IF}] = O(\frac{K}{\epsilon_{1,2}} \log T)$.

Analysis of Algorithm

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

$$\begin{aligned} E[R_T] &\leq (1 - \frac{1}{T})E[R_T^{IF}] + \frac{1}{T}O(T) \\ &= O(E[R_T^{IF}] + 1) \\ &= O(E[R_T^{IF}]) \\ &= O(\frac{K}{\epsilon_{1,2}} \log T) \end{aligned}$$

Analysis of Algorithm

To aid our analysis, we introduce

Confidence Intervals:

IF maintains a number $\hat{P}_{i,j} = \frac{\# \, b_i \text{ wins}}{\# \text{ comparisons}}$ 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 $\delta = 1/(TK^2)$.

Analysis of Algorithm

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 $O(\frac{1}{\epsilon_{i,j}^2}\log(TK))$. Moreover, the probability that the inferior bandit is declared the winner at some time $t \leq T$ is at most δ.

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. ■

Analysis of Algorithm

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 $O(\frac{1}{\epsilon_{1,2}} \log T)$.

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

Analysis of Algorithm

Since $\min\{\epsilon_{1,j}, \epsilon_{1,i}\} \leq \epsilon_{1,j}$, so the accumulated (weak) regret is bounded by

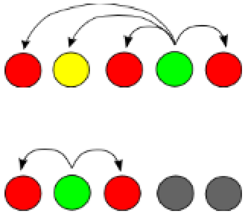
$$\begin{aligned}
\epsilon_{1,j} O(\frac{1}{\epsilon_{1,j}^2} \log(TK)) &= O(\frac{1}{\epsilon_{1,j}} \log(TK)) \\
&= O(\frac{1}{\epsilon_{1,2}} \log(TK)) \\
&= O(\frac{1}{\epsilon_{1,2}} \log T)(\text{ since } K \leq T)
\end{aligned}$$

■

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 $b \succ b'$, 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 $\delta = \frac{1}{TK^2}$.



Analysis of Algorithm

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 δ. 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 δ. 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 $\delta[(K-1)+(K-1)^2] < \delta K^2 = \frac{1}{T}$ ■

Analysis of Algorithm

Next we sketch the proof for lemma 2, which states the mistake-free executions of IF satisfy $R_T^{IF} = O(\frac{K \log K}{\epsilon_{1,2}} \log T)$. 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.

Analysis of Algorithm

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.

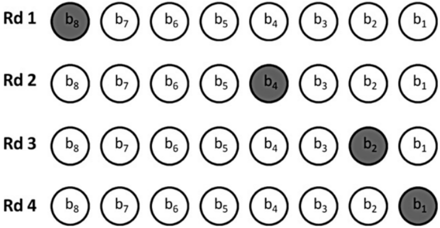


Fig. 1. An illustrative example of a sequence of candidate bandits. The incumbent candidate in each round is shaded in grey.

Analysis of Algorithm

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.

Analysis of Algorithm

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.

Analysis of Algorithm

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.

Analysis of Algorithm

Next, we show that $E[G_j] <= 2$.
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.

Analysis of Algorithm

Theorem 2. For any fixed $\epsilon > 0$ and any algorithm ϕ for the dueling bandits problem, there exists a problem instance such that

$$R_T^{\phi} = \Omega(\frac{K}{\epsilon} logT)$$

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

Analysis of Algorithm

The proof is similar to the lower bound proof for multi-armed bandit problem and is omited.
However, here is a heuristic explanation for why we might suspect the theorem to 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 $O(logT/\epsilon^2)$ comparisons. Since there are K bandits, we can expect to take K times $O(logT/\epsilon^2)$ comparions to determine whether bandit b is the best.

Extensions

Other models

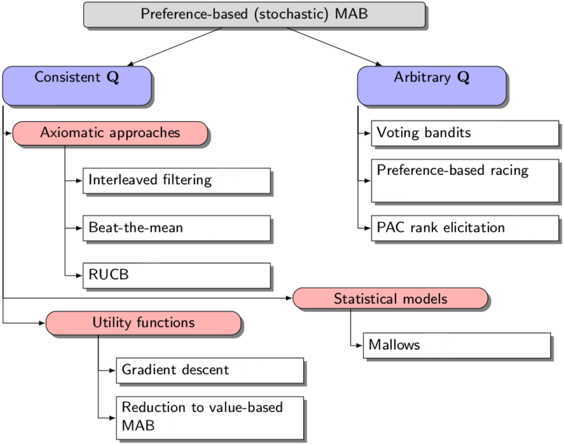


Fig. 1. A taxonomy of (stochastic) PB-MAB algorithms

Busa-Fekete 2014

where Q_{ij} is the probability arm i beats arm j.

RUCB (Zohgi 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.

RUCB Algorithm

Algorithm 1 Relative Upper Confidence Bound

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

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

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

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

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

5: $\quad$ 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: $\quad d \leftarrow \arg\max_j u_{jc}$

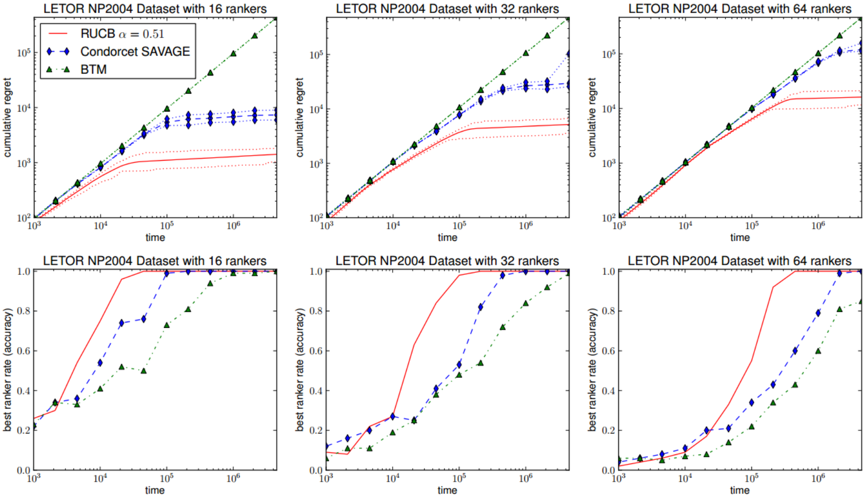
7: $\quad$ Compare arms a_c and a_d and increment w_{cd} or w_{dc} depending on which arm wins.

8: **end for**

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

Zoghi 2014

RUCB Results



Zoghi 2014

Copeland Bandits

- 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)

10 mm

Medtronic human array

49 mm

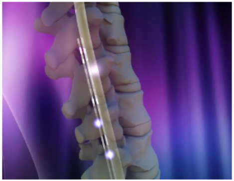


Image source: williamcapicottomd.com

SCI Patient

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 $O(\frac{K}{\epsilon_{1,2}} \log T)$
  - 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
- ‘Relative Upper Confidence Bound for the K-Armed Dueling Bandit Problem’, Zoghi et. al, 2013.
- ‘Interactively Optimizing Information Retrieval Systems as a Dueling Bandits Problem’, Yue & Joachims, 2009.
- ‘Contextual Dueling Bandits’, Dudik et. al., 2015.