Dueling Bandit Review

Bowen Xu

Sep 2022

Table of Contents

Table of Contents

Motivation

Drawback of Conventional MAB problem:

- When payoff is a **relative comparison** result rather than an **absolute value**, it is difficult to apply conventional MAB Algorithm.

What if Rewards aren't Directly Measurable?

Motivation

Some scenarios where the traditional MAB algorithm cannot be applied:

- User-perceived quality of a set of retrieval results
- taste of food
- product attractiveness

Motivation

Characteristics of dueling bandit algorithm:

- Only **binary feedback** about the **relative reward** of two chosen strategies is available
- Pairwise comparison is made in this problem.
- Dueling Bandit Problem applies where a system must adapt interactively to specific user bases

Motivation

Suitable application scenarios:

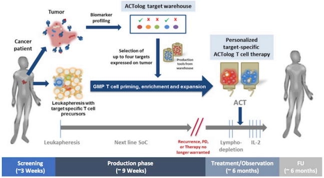
- Search-engine user prefers ranking r_1 over r_2 for a given query
- Online Advertising
- Recommender Systems: (e.g. Restaurant recommend app)
- Ongoing work: Personalized Clinical Treatment.

Application of Dueling Bandit Algorithm

(a) search engine

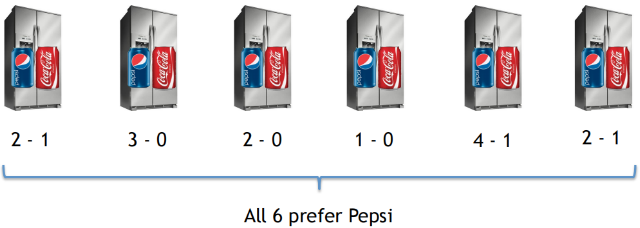
(b) online advertisement

(c) recommend system



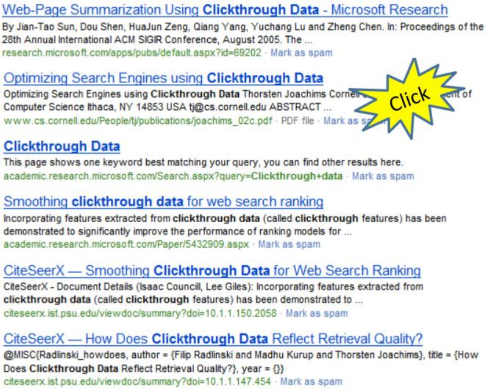
(d) personalized clinical treatment

- Experiment 1: Pepsi or Coca
- Model: A 2-arm dueling bandit problem



Compare the Coca and Pepsi cola and we will find that the Pepsi cola is more preferred than Coca cola relatively.

- Experiment 2: Rank the customers' preference of websites
- Model: A Dueling bandit problem



Search Engine: Interleave

Interleave the two reference ranking into the presented ranking.
Method: At each time, a coin flip decides which captain can choose his next teammate.

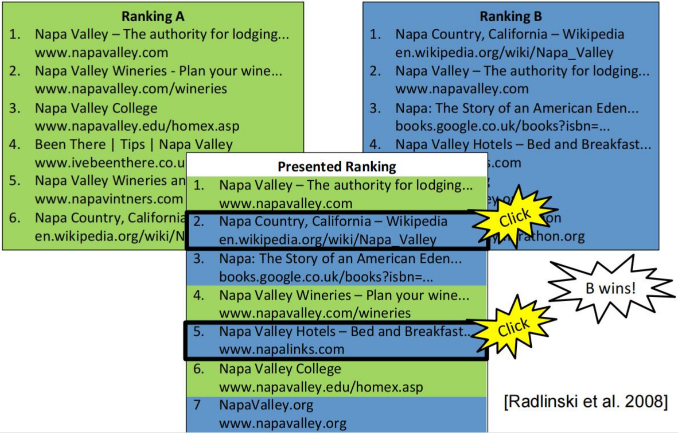
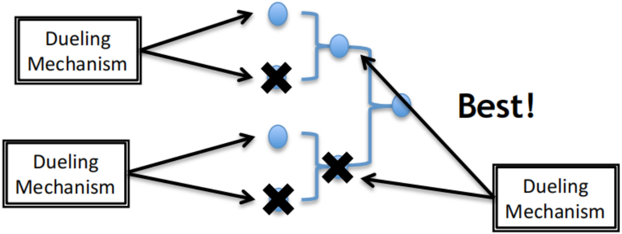


Table of Contents

Dueling Mechanism

Each pair duels until statistical significance



Dueling Mechanism

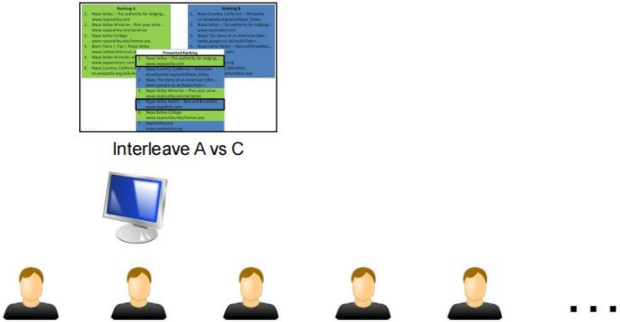
Interleave A vs B

...

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

[From Yisong Yue]

Dueling Mechanism



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

[From Yisong Yue]

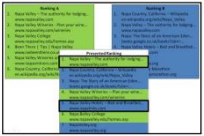
Dueling Mechanism

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]

Dueling Mechanism



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]

Dueling Mechanism

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]

Problem Setting

Dueling Bandits:

- Process: At each time step: $t = 1, 2, \ldots, T$
  - The algorithm chooses a pair of actions a_i, a_j from K available actions.
  - The world provides (independent stochastic) preference feedback of which action is more preferred.
  - The preference feedback satisfy the probability of $P(a_i > a_j) = P_{ij}$

Problem Setting

Dueling Bandits:

- Process: At each time step: $t = 1, 2, \ldots, T$
  - The algorithm chooses a pair of actions a_i, a_j from K available actions.
  - The world provides (independent stochastic) preference feedback of which action is more preferred.
  - The preference feedback satisfy the probability of $P(a_i > a_j) = P_{ij}$
- Regret: $R(T) = \sum_{t=1}^{T} r(t)$
  - $r = \Delta_{1i} + \Delta_{1j}$, $\Delta_{ij} = P_{ij} - 0.5$

Problem Setting

Dueling Bandits:

- Process: At each time step: $t = 1, 2, \ldots, T$
  - The algorithm chooses a pair of actions a_i, a_j from K available actions.
  - The world provides (independent stochastic) preference feedback of which action is more preferred.
  - The preference feedback satisfy the probability of $P(a_i > a_j) = P_{ij}$
- Regret: $R(T) = \sum_{t=1}^{T} r(t)$
  - $r = \Delta_{1i} + \Delta_{1j}$, $\Delta_{ij} = P_{ij} - 0.5$
- Goal: Minimize the cumulative regret: $R(T)$ from pulling the suboptimal arms.

Problem Setting: Winner Setting

- Condorcet Winner: the arm beat all other arms with probability of 0.5 ($P_{ij} > 0.5$)
  - e.g. IF, BTM, Sparring, RUCB
- Copeland Winner: the arm beat the most other arms with probability of 0.5.
  - e.g. CCB, RCB, RMED, DTS

Problem Setting

Two Styles of Algorithm Design:

- Asymmetric Algorithms: choosing a reference arm and a exploration arm.
  - e.g. IF, BtM, SAVAGE, Doubler, RUCB, MergeRUCB, RCS, and DTS.
- Symmetric Algorithms: treats the choice of the two arms symmetrically.
  - e.g. Sparring, Self-Sparring.

Problem Setting

Basic Assumption:

- Ordered Arms: We can relabel arms as $P_{ij} > 0.5$ for all i, j.

Problem Setting

Basic Assumption:

- Ordered Arms: We can relabel arms as $P_{ij} > 0.5$ for all i, j.
- Stochastic Triangle Inequality (STI): $\Delta_{ik} \leq \Delta_{ij} + \Delta_{jk} (i \leq j \leq k)$.

Problem Setting

Basic Assumption:

- Ordered Arms: We can relabel arms as $P_{ij} > 0.5$ for all i, j.
- Stochastic Triangle Inequality (STI): $\Delta_{ik} \leq \Delta_{ij} + \Delta_{jk} (i \leq j \leq k)$.
- Transitivity Conditions:
    - Strong Stochastic Transitivity (SST): $\Delta_{ik} \geq \max\{\Delta_{ij}, \Delta_{jk}\}$
    - Relaxed Stochastic Transitivity (RST): $\gamma\Delta_{ik} \geq \max\{\Delta_{ij}, \Delta_{jk}\}$

Strong Stochastic Transitivity

$$\Delta_{ik} \geq \max\{\Delta_{ij}, \Delta_{jk}\}.$$

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

Monotonic ↑

Strong Triangle Inequality

$$\Delta_{ik} \leq \Delta_{ij} + \Delta_{jk}.$$

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

Development (Main Algorithm)

- Intereaved Filter [Yue et al., 2009]
- Beat the Mean [Yue, Joachims., 2011]
- SAVAGE [Urvoy et al., 2013]
- Sparring [Alion et al., 2014]
- RMED [Komiyama et al., 2015]
- RUCB [Zoghi et al., 2014; 2015]
- CCB, SCB [Masrour Zoghi et al., 2015]
- DTS [Wu, Liu, 2016]
- SelfSparring [Yanan Sui et al., 2017]
- ......

Problem Setting

Two main Model Assumption:

- Condorcet Winner
- Copeland Winner [Zoghi et al., 2015]

Other Model Assumption:

- Borda Winner [Jamieson et al., 2015]
- Von Neuman Winner [Dudik et al., 2015]
- General Tournament Solu@ons [Ramamohan et al. 2016]

Table of Contents

Interleaved Filter (IF)

The basic algorithm of K-arm Dueling Bandits problem.

- Motivation: solve the problem where absolute rewards have no natural scale or are difficult to measure.
  (K-arm Bandit problem)

Interleaved Filter (IF)

The basic algorithm of K-arm Dueling Bandits problem.

- Motivation: solve the problem where absolute rewards have no natural scale or are difficult to measure.
  (K-arm Bandit problem)
- Algorithm: IF1, IF2

IF Algorithm: IF1

- Idea: combine estimate $\hat{P}$ with confidence interval $\hat{C}$.
  $\hat{P}_{b',b} = P(b' > b)$, $\hat{C}_t = (\hat{P}_t - c_t, \hat{P}_t + c_t)$, $c_t = \sqrt{log(1/\delta)/t}$

IF Algorithm: IF1

- Idea: combine estimate $\hat{P}$ with confidence interval $\hat{C}$.
  $\hat{P}_{b',b} = P(b' > b), \hat{C}_t = (\hat{P}_t - c_t, \hat{P}_t + c_t), c_t = \sqrt{log(1/\delta)/t}$
- Assumption: SST, finite horizon

IF Algorithm: IF1

- Idea: combine estimate $\hat{P}$ with confidence interval $\hat{C}$.
  $\hat{P}_{b',b} = P(b' > b), \hat{C}_t = (\hat{P}_t - c_t, \hat{P}_t + c_t), c_t = \sqrt{log(1/\delta)/t}$
- Assumption: SST, finite horizon
- Regret: $E(R_T^{IF1}) = O(\frac{KlogK}{\Delta_{1,2}} logT)$ ($\Delta_{1,2}$ means the distinguishability between the two best bandits)

Algorithm: IF1

Algorithm 2 Interleaved Filter 1 (IF1)

Input: $T, \mathcal{B} = \{b_1, \ldots, b_K\}$
$\delta \leftarrow 1/(TK^2)$
Choose $\hat{b} \in \mathcal{B}$ randomly
$W \leftarrow \{b_1, \ldots, b_K\} \setminus \{\hat{b}\}$
$\forall b \in W$, maintain estimate $\hat{P}_{\hat{b},b}$ of $P(\hat{b} > b)$
$\forall b \in W$, maintain $1 - \delta$ confidence interval $\hat{C}_{\hat{b},b}$ of $\hat{P}_{\hat{b},b}$
while $W \neq \emptyset$ **do**
  for $b \in W$ **do**
    compare $\hat{b}$ and b
    update $\hat{P}_{\hat{b},b}$, $\hat{C}_{\hat{b},b}$
  end for
  while $\exists b \in W$ s.t. $\left(\hat{P}_{\hat{b},b} > 1/2 \wedge 1/2 \notin \hat{C}_{\hat{b},b}\right)$ **do**
    $W \leftarrow W \setminus \{b\}$
  end while
  if $\exists b' \in W$ s.t. $\left(\hat{P}_{\hat{b},b'} < 1/2 \wedge 1/2 \notin \hat{C}_{\hat{b},b'}\right)$ **then**
    $\hat{b} \leftarrow b', \; W \leftarrow W \setminus \{b'\}$ *//new round*
    $\forall b \in W$, reset $\hat{P}_{\hat{b},b}$ and $\hat{C}_{\hat{b},b}$
  end if
end while
$\hat{T} \leftarrow$ Total Comparisons Made
return $(\hat{b}, \hat{T})$

IF Algorithm: IF2

- Idea: IF1 + **pruning**.

IF Algorithm: IF2

- Idea: IF1 + **pruning**.
- Assumption: SST, finite horizon

IF Algorithm: IF2

- Idea: IF1 + **pruning**.
- Assumption: SST, finite horizon
- Regret: $E(R_T^{IF2}) = O(\frac{K}{\Delta_{1,2}} log T)$

Algorithm: IF2

Algorithm 3 Interleaved Filter 2 (IF2)

Input: T, $\mathcal{B} = \{b_1, \dots, b_K\}$
$\delta \leftarrow 1/(TK^2)$
Choose $\hat{b} \in \mathcal{B}$ randomly
$W \leftarrow \{b_1, \dots, b_K\} \setminus \{\hat{b}\}$
$\forall b \in W$, maintain estimate $\hat{P}_{\hat{b},b}$ of $P(\hat{b} > b)$
$\forall b \in W$, maintain $1 - \delta$ confidence interval $\hat{C}_{\hat{b},b}$ of $\hat{P}_{\hat{b},b}$
while $W \neq \emptyset$ **do**
    for $b \in W$ **do**
        compare $\hat{b}$ and b
        update $\hat{P}_{\hat{b},b}$, $\hat{C}_{\hat{b},b}$
    end for
    while $\exists b \in W$ s.t. $\left(\hat{P}_{\hat{b},b} > 1/2 \wedge 1/2 \notin \hat{C}_{\hat{b},b}\right)$ **do**
        $W \leftarrow W \setminus \{b\}$
    end while
    if $\exists b' \in W$ s.t. $\left(\hat{P}_{\hat{b},b'} < 1/2 \wedge 1/2 \notin \hat{C}_{\hat{b},b'}\right)$ **then**
        while $\exists b \in W$ s.t. $\hat{P}_{\hat{b},b} > 1/2$ **do**
            $W \leftarrow W \setminus \{b\}$ *//pruning*
        end while
        $\hat{b} \leftarrow b'$, $W \leftarrow W \setminus \{b'\}$ *//new round*
        $\forall b \in W$, reset $\hat{P}_{\hat{b},b}$ and $\hat{C}_{\hat{b},b}$
    end if
end while
$\hat{T} \leftarrow$ Total Comparisons Made
return $(\hat{b}, \hat{T})$

Interleaved Filter

Reference: The K-armed Dueling bandits problem, 2009

Beat the Mean (BTM)

- motivation: Extend the Dueling Bandits Problem to a **relaxed setting (RST)** where preference magnitudes can violate transitivity

Beat the Mean (BTM)

- motivation: Extend the Dueling Bandits Problem to a **relaxed setting (RST)** where preference magnitudes can violate transitivity
- Idea: Compare the fewest recorded comparison bandit with the mean bandit (random sample to obtain the mean)

Beat the Mean (BTM)

- motivation: Extend the Dueling Bandits Problem to a **relaxed setting (RST)** where preference magnitudes can violate transitivity
- Idea: Compare the fewest recorded comparison bandit with the mean bandit (random sample to obtain the mean)
- Assumption: **RST**, ordered arms

Beat the Mean (BTM)

- motivation: Extend the Dueling Bandits Problem to a **relaxed setting (RST)** where preference magnitudes can violate transitivity
- Idea: Compare the fewest recorded comparison bandit with the mean bandit (random sample to obtain the mean)
- Assumption: **RST**, ordered arms
- Setting: PAC(confident near-optimal bandit), online

Beat the Mean (BTM)

Algorithm 1 Beat-the-Mean

Input: $\mathcal{B} = \{b_1, \ldots, b_K\}$, N, T, $c_{\delta,\gamma}(\cdot)$
$W_1 \leftarrow \{b_1, \ldots, b_K\}$ *//working set of active bandits*
$\ell \leftarrow 1$ *//num rounds*
$\forall b \in W_\ell$, $n_b \leftarrow 0$ *//num comparisons*
$\forall b \in W_\ell$, $w_b \leftarrow 0$ *//num wins*
$\forall b \in W_\ell$, $\hat{P}_b \equiv w_b/n_b$, or $1/2$ if $n_b = 0$
$n^* \equiv \min_{b \in W_\ell} n_b$
$c^* \equiv c_{\delta,\gamma}(n^*)$, or 1 if $n^* = 0$ *//confidence radius*
$t \leftarrow 0$ *//total number of iterations*
while $|W_\ell| > 1$ **and** $t < T$ **and** $n^* < N$ **do**
  $b \leftarrow \operatorname{argmin}_{b \in W_\ell} n_b$ *//break ties randomly*
  select $b' \in W_\ell$ at random, compare b vs b'
  if b wins, $w_b \leftarrow w_b + 1$
  $n_b \leftarrow n_b + 1$
  $t \leftarrow t + 1$
  if $\min_{b' \in W_\ell} \hat{P}_{b'} + c^* \leq \max_{b \in W_\ell} \hat{P}_b - c^*$ **then**
    $b' \leftarrow \operatorname{argmin}_{b \in W_\ell} \hat{P}_b$
    $\forall b \in W_\ell$, delete comparisons with b' from w_b, n_b
    $W_{\ell+1} \leftarrow W_\ell \setminus \{b'\}$ *//update working set*
    $\ell \leftarrow \ell + 1$ *//new round*
  end if
end while
return $\operatorname{argmax}_{b \in W_\ell} \hat{P}_b$

Beat the Mean (BTM)

Regret:

- $(\epsilon - \delta)$-PAC: $O(KN) = O(\frac{K\gamma^6}{\epsilon^2} log \frac{KN}{\delta})$
- Online: $O(\sum_{l=1}^{K-1} min\{\frac{\gamma^7}{\epsilon_l}, \frac{\gamma^5 \epsilon_l}{\epsilon_*^2}\} log T) = O(\frac{\gamma^7 K}{\epsilon_*} log T)$

Another Perspective

Borda Score: $\sum_j \frac{1}{K} p_{ij}$

Theorem:

- The Borda score of the Condorcet winner is always greater than or equal to 0.5.
- The Condorcet winner remains optimal after removing other bandits.

Idea: Keep eliminating Borda losers, eventually the Condorcet winner.

Beat the Mean

Reference: Beat the Mean Bandit, 2011

UBDB Problem

Utility-based dueling bandits problem

- Average utility: $U_t^{av} = (u_t + v_t)/2$

UBDB Problem

Utility-based dueling bandits problem

- Average utility: $U_t^{av} = (u_t + v_t)/2$
- Reward(utility) not observed

UBDB Problem

Utility-based dueling bandits problem

- Average utility: $U_t^{av} = (u_t + v_t)/2$
- Reward(utility) not observed
- Link function: $\Pr[b_t = 1|(u_t, v_t)] = \Phi(u_t, v_t)$

UBDB Algorithm

- Motivation: Reducing the Dueling Bandits problem to the conventional (stochastic) **Multi-Armed Bandits** problem

UBDB Algorithm

- Motivation: Reducing the Dueling Bandits problem to the conventional (stochastic) **Multi-Armed Bandits** problem
- Idea: View the dueling bandit as the dueling of two arms with different strategy (left and right)

UBDB Algorithm

- Motivation: Reducing the Dueling Bandits problem to the conventional (stochastic) **Multi-Armed Bandits** problem
- Idea: View the dueling bandit as the dueling of two arms with different strategy (left and right)
- Algorithm: Doubler, MultiSBM, Sparring

UBDB Algorithm: Doubler

Structured: Large or possibly infinite set of arms X

- left arm: random choice
- right arm: SBM

Regret (SBM is UCB): $O(Hlog^2 T)$, where $H = \sum_{i=2}^{K} \Delta_i^{-1}$

UBDB Algorithm: Doubler

Algorithm 2 (Doubler): Reduction for finite and infinite X with internal structure.

```
S ← new SBM over X
ℒ ← an arbitrary singleton in X
i ← 1, t ← 1
while true do
    reset(S)
    for j = 1...2^i do
        choose x_t uniformly from ℒ
        y_t ← advance(S)
        play (x_t, y_t), observe choice b_t
        feedback(S, b_t)
        t ← t + 1
    end for
    ℒ ← the multi-set of arms played as y_t in the last for-loop
    i ← i + 1
end while
```

UBDB Algorithm: MultiSBM

Unstructured: The elements of X typically have no structure

- left arm: arm of the last round
- right arm: SBM

Regret (SBM is UCB): $O(H\alpha lnT + H\alpha(KlnK + KlnlnT - \sum_{x \neq x^*} ln\Delta_x))$

UBDB Algorithm: MultiSBM

Algorithm 3 (MultiSBM): Reduction for unstructured finite X by using K SBMs in parallel.

```
For all x ∈ X: S_x ← new SBM over X, reset(S_x)
y_0 ← arbitrary element of X
t ← 1
while true do
    x_t ← y_{t−1}
    y_t ← advance(S_{x_t})
    play (x_t, y_t), observe choice b_t
    feedback(S_{x_t}, b_t)
    t ← t + 1
end while
```

UBDB Algorithm: Sparring

- left arm: SBM1
- right arm: SBM2

Upper bound of Regret: a constant times the regret of the two SBMs

UBDB Algorithm: Sparring

Algorithm 4 (Sparring): Reduction to two SBMs.

$S_L, S_R \leftarrow$ two new SBMs over X
reset(S_L), reset(S_R), $t \leftarrow 1$
while true **do**
    $x_t \leftarrow$ advance(S_L); $y_t \leftarrow$ advance(S_R)
    play (x_t, y_t), observe choice $b_t \in \{0, 1\}$
    feedback$(S_L, \mathbf{1}_{b_t=0})$; feedback$(S_R, \mathbf{1}_{b_t=1})$
    $t \leftarrow t + 1$
end while

Relative UCB (RUCB)

- Motivation: Extends the **Upper Confidence Bound (UCB)** algorithm in relative setting to apply dueling bandit

Relative UCB (RUCB)

- Motivation: Extends the **Upper Confidence Bound (UCB)** algorithm in relative setting to apply dueling bandit
- Idea: Implement two comparisons:
  - optimistically: u_{uj} (line 6)
  - pessimistically: u_{jc} (line 13)

  objective: avoid auto-comparisons (self-comparisons obtains no information) until is great certainty of Condorcet winner.

Relative UCB (RUCB)

- Motivation: Extends the **Upper Confidence Bound (UCB)** algorithm in relative setting to apply dueling bandit
- Idea: Implement two comparisons:
  - optimistically: u_{uj} (line 6)
  - pessimistically: u_{jc} (line 13)

  objective: avoid auto-comparisons (self-comparisons obtains no information) until is great certainty of Condorcet winner.

Relative UCB (RUCB)

- Motivation: Extends the **Upper Confidence Bound (UCB)** algorithm in relative setting to apply dueling bandit
- Idea: Implement two comparisons:
  - optimistically: u_{uj} (line 6)
  - pessimistically: u_{jc} (line 13)

  objective: avoid auto-comparisons (self-comparisons obtains no information) until is great certainty of Condorcet winner.
- Regret: $O(K^2 + K log T)$

Relative UCB (RUCB)

- Strengths: RUCB achieves high-probability regret bounds while making minimal assumptions other than assuming a Condorcet winner.
- Weaknesses: Compare Condorcet winner with itself both optimistically and pessimistically, where it is overly prudent.

RUCB Algorithm

Algorithm 1 Relative Upper Confidence Bound

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

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

$\mathcal{B} = \varnothing$

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

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

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

  $\mathcal{C} \leftarrow \left\{a_c \,|\, \forall j : u_{cj} \geq \frac{1}{2}\right\}$.

  If $\mathcal{C} = \varnothing$, then pick c randomly from $\{1, \ldots, K\}$.

  $\mathcal{B} \leftarrow \mathcal{B} \cap \mathcal{C}$.

  If $|\mathcal{C}| = 1$, then $\mathcal{B} \leftarrow \mathcal{C}$ and a_c to be the unique element in $\mathcal{C}$.

  if $|\mathcal{C}| > 1$ **then**

    Sample a_c from $\mathcal{C}$ using the distribution:

$$p(a_c) = \begin{cases} 0.5 & \text{if } a_c \in \mathcal{B}, \\ \frac{1}{2^{|\mathcal{B}|}|\mathcal{C} \setminus \mathcal{B}|} & \text{otherwise.} \end{cases}$$

  end if

  $d \leftarrow \arg\max_j u_{jc}$, with ties broken randomly. Moreover, if there is a tie, d is not allowed to be equal to c.

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

end for

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

RUCB Algorithm: MergeRUCB

- Idea: partition the K arms into small batches and compares arms within each batch.

RUCB Algorithm: MergeRUCB

- Idea: partition the K arms into small batches and compares arms within each batch.
- Regret: $O(K log T)$
  MergeRUCB does not require global pairwise comparisons between all pairs of arms than RUCB.

Relative UCB (RUCB)

Reference: Reference: Relative Upper Confidence Bound for the K-Armed Dueling Bandit Problem, 2014

Table of Contents

Copaland Bandits

Mainly generalized from traditional MAB algorithms along two lines:

- UCB-type: RUCB, CCB
- MED(minimal empirical divergence)-type: RMED

SAVAGE Algorithm

- Motivation: solve the N dimensional parametric decision problem with high confidence.

SAVAGE Algorithm

- Motivation: solve the N dimensional parametric decision problem with high confidence.
- Idea:
  - work by reducing progressively a box-shaped confidence set H until a single decision remains in f(H).
  - (more explicitly) eliminate the arms loss with high probability to another arm until leave with Condorcet(Copeland) winner.

SAVAGE Algorithm

- Motivation: solve the N dimensional parametric decision problem with high confidence.
- Idea:
  - work by reducing progressively a box-shaped confidence set H until a single decision remains in f(H).
  - (more explicitly) eliminate the arms loss with high probability to another arm until leave with Condorcet(Copeland) winner.
- Regret: $O(K^2 log T)$ outperform IF and BTM for moderate K

SAVAGE Algorithm

Algorithm 1 SAVAGE algorithm

Input: $\mathbf{X} = (X_1, \ldots, X_N)$, f, $\mathcal{F}$, T, δ
Initialization:
$\mathcal{W} := \{1, \ldots, N\}$, $\mathcal{H} := \mathcal{F}$, $s := 1$
$\forall i \in \mathcal{W} : \hat{\mu}_i := 1/2$, and $t_i := 0$
while $\neg$**Accept**$(f, \mathcal{H}, \mathcal{W}) \wedge s \leqslant T$ **do**
  Pick a variable index $i \in \arg\min_{\mathcal{W}} \{t_1, \ldots, t_N\}$
  $t_i := t_i + 1$
  Sample the i^{th} distribution $x_i \leftarrow X_i$
  $\hat{\mu}_i := (1 - \frac{1}{t_i})\hat{\mu}_i + \frac{1}{t_i} x_i$
  $\mathcal{H} := \mathcal{H} \cap \{\mathbf{x} \mid |x_i - \hat{\mu}_i| < c(t_i)\}$
  $\mathcal{W} := \mathcal{W} \backslash \{j \parallel$ **IndepTest**$(f, \mathcal{H}, j)\}$
  $s := s + 1$
end while
return $\hat{d} \in f(\mathcal{H})$

SAVAGE Algorithm

Reference: Reference: Generic Exploration and K-armed Voting Bandits, 2013

Copeland Confidence Bounds (CCB)

- Motivation: design for small number of arms Copeland winner problem

Copeland Confidence Bounds (CCB)

- Motivation: design for small number of arms Copeland winner problem
- Principle: optimism followed by pessimism
  - optimistic copeland winner: have confidence to beat other arms
  - pessimistic opponent: have confidence to beat potential winner

Copeland Confidence Bounds (CCB)

- Motivation: design for small number of arms Copeland winner problem
- Principle: optimism followed by pessimism
  - optimistic copeland winner: have confidence to beat other arms
  - pessimistic opponent: have confidence to beat potential winner
- Regret: $O(K^2 + KlogT)$

Copeland Confidence Bounds (CCB)

Algorithm 1 Copeland Confidence Bound

Input: A Copeland dueling bandit problem and an exploration parameter $\alpha > \frac{1}{2}$.

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: $\mathcal{B}_1 = \{a_1, \dots, a_K\}$ // potential best arms
3: $\mathcal{B}_1^i = \varnothing$ for each $i = 1, \dots, K$ // potential to beat a_i
4: $\overline{L}_C = K$ // estimated max losses of a Copeland winner
5: **for** $t = 1, 2, \dots$ **do**
6: $\mathbf{U} := [u_{ij}] = \frac{\mathbf{W}}{\mathbf{W}+\mathbf{W}^T} + \sqrt{\frac{\alpha \ln t}{\mathbf{W}+\mathbf{W}^T}}$ and $\mathbf{L} := [l_{ij}] = \frac{\mathbf{W}}{\mathbf{W}+\mathbf{W}^T} - \sqrt{\frac{\alpha \ln t}{\mathbf{W}+\mathbf{W}^T}}$, with $u_{ii} = l_{ii} = \frac{1}{2}, \forall i$
7: $\overline{\mathrm{Cpld}}(a_i) = \#\left\{k \mid u_{ik} \geq \frac{1}{2}, k \neq i\right\}$ and $\underline{\mathrm{Cpld}}(a_i) = \#\left\{k \mid l_{ik} \geq \frac{1}{2}, k \neq i\right\}$
8: $\mathcal{C}_t = \{a_i \mid \overline{\mathrm{Cpld}}(a_i) = \max_j \overline{\mathrm{Cpld}}(a_j)\}$
9: Set $\mathcal{B}_t \leftarrow \mathcal{B}_{t-1}$ and $\mathcal{B}_t^i \leftarrow \mathcal{B}_{t-1}^i$ and update as follows:

A. Reset disproven hypotheses: If for any i and $a_j \in \mathcal{B}_t^i$ we have $l_{ij} > 0.5$, reset $\mathcal{B}_t$, $\overline{L}_C$ and $\mathcal{B}_t^k$ for all k (i.e. set them to their original values as in Lines 2–4 above).

B. Remove non-Copeland winners: For each $a_i \in \mathcal{B}_t$, if $\overline{\mathrm{Cpld}}(a_i) < \underline{\mathrm{Cpld}}(a_j)$ holds for any j, set $\mathcal{B}_t \leftarrow \mathcal{B}_t \setminus \{a_i\}$, and if $|\mathcal{B}_t^i| \neq \overline{L}_C + 1$, then set $\mathcal{B}_t^i \leftarrow \{a_k | u_{ik} < 0.5\}$. However, if $\mathcal{B}_t = \varnothing$, reset $\mathcal{B}_t$, $\overline{L}_C$ and $\mathcal{B}_t^k$ for all k.

C. Add Copeland winners: For any $a_i \in \mathcal{C}_t$ with $\overline{\mathrm{Cpld}}(a_i) = \underline{\mathrm{Cpld}}(a_i)$, set $\mathcal{B}_t \leftarrow \mathcal{B}_t \cup \{a_i\}$, $\mathcal{B}_t^i \leftarrow \varnothing$ and $\overline{L}_C \leftarrow K - 1 - \underline{\mathrm{Cpld}}(a_i)$. For each $j \neq i$, if we have $|\mathcal{B}_t^j| < \overline{L}_C + 1$, set $\mathcal{B}_t^j \leftarrow \varnothing$, and if $|\mathcal{B}_t^j| > \overline{L}_C + 1$, randomly choose $\overline{L}_C + 1$ elements of $\mathcal{B}_t^j$ and remove the rest.

10: With probability $1/4$, sample (c, d) uniformly from the set $\{(i, j) \mid a_j \in \mathcal{B}_t^i$ and $0.5 \in [l_{ij}, u_{ij}]\}$ (if it is non-empty) and skip to Line 14.
11: If $\mathcal{B}_t \cap \mathcal{C}_t \neq \varnothing$, then with probability $2/3$, set $\mathcal{C}_t \leftarrow \mathcal{B}_t \cap \mathcal{C}_t$.
12: Sample a_c from $\mathcal{C}_t$ uniformly at random.
13: With probability $1/2$, choose the set $\mathcal{B}^i$ to be either $\mathcal{B}_t^i$ or $\{a_1, \dots, a_K\}$ and then set $d \leftarrow \arg\max_{\{j \in \mathcal{B}^i \mid l_{jc} \leq 0.5\}} u_{jc}$. If there is a tie, d is not allowed to be equal to c.
14: Compare arms a_c and a_d and increment w_{cd} or w_{dc} depending on which arm wins.
15: **end for**

Scalable Confidence bounds (SCB)

- Motivation: design for large-scale arms Coepland winner problem

Scalable Confidence bounds (SCB)

- Motivation: design for large-scale arms Coepland winner problem
- Algorithm difference from CCB:
  - Approximate Copeland Solver
  - KL-based arm-elimination indentification

Scalable Confidence bounds (SCB)

- Motivation: design for large-scale arms Coepland winner problem
- Algorithm difference from CCB:
  - Approximate Copeland Solver
  - KL-based arm-elimination indentification
- Regret: $O(KlogKlogT)$

Scalable Confidence Bounds (SCB)

Algorithm 3 Scalable Copeland Bandits

Input: A Copeland dueling bandit problem with preference matrix $\mathbf{P} = [p_{ij}]$

for all $r = 1, 2, \ldots$ **do**
  Set $T = 2^{2^r}$ and run Algorithm 2 with failure probability $\log(T)/T$ in order to find an exact Copeland winner ($\epsilon = 0$); force-terminate if it requires more than T queries.
  Let T_0 be the number of queries used by invoking Algorithm 2, and let a_i be the arm produced by it; query the pair (a_i, a_i) $T - T_0$ times.
end for

CCB, SCB

Reference: Copeland Dueling Bandits, 2015

Relative Minimal Empirical Divergence (RMED)

- Idea: Minimal Empirical Divergence

Relative Minimal Empirical Divergence (RMED)

- Idea: Minimal Empirical Divergence
- Theory: Empirical Divergence $I(t) = \sum_{\{j|p_{ij<.5}\}} d(\hat{p}_{ij}, .5)N_{ij}(t))$:
  - defines the minimum number of comparisons to prove i is inferior to j
  - likelihood function of Condorcet winner: $exp(-I(t))$

Relative Minimal Empirical Divergence (RMED)

- Idea: Minimal Empirical Divergence
- Theory: Empirical Divergence $I(t) = \sum_{\{j|p_{ij}<.5\}} d(\hat{p}_{ij}, .5)N_{ij}(t))$:
  - defines the minimum number of comparisons to prove i is inferior to j
  - likelihood function of Condorcet winner: $exp(-I(t))$
- Algorithm: RMED1, RMED2, RMED2FH
  (Unlike RMED1, RMED2 keeps drawing pairs of arms i, j at least $\alpha loglogt$ times (Line 10 in Algorithm 1))

Relative Minimal Empirical Divergence (RMED)

- Idea: Minimal Empirical Divergence
- Theory: Empirical Divergence $I(t) = \sum_{\{j|p_{ij<.5}\}} d(\hat{p}_{ij}, .5)N_{ij}(t))$:
  - defines the minimum number of comparisons to prove i is inferior to j
  - likelihood function of Condorcet winner: $exp(-I(t))$
- Algorithm: RMED1, RMED2, RMED2FH
  (Unlike RMED1, RMED2 keeps drawing pairs of arms i, j at least $\alpha loglogt$ times (Line 10 in Algorithm 1))
- Regret: $R_T \geq \sum_{k=2}^{K} \min_{\{j|p_{ij}<.5\}} \frac{(\Delta_{1i}+\Delta_{1j})logT}{2d(p_{ij},.5)}$

Relative Minimal Empirical Divergence (RMED)

Algorithm 1 Relative Minimum Empirical Divergence (RMED) Algorithm

Input: K arms, $f(K) \geq 0$. $\alpha > 0$ (RMED2FH, RMED2). T (RMED2FH).

$L \leftarrow \begin{cases} 1 & \text{(RMED1, RMED2)} \\ \lceil \alpha \log \log T \rceil & \text{(RMED2FH)} \end{cases}$.

Initial phase: draw each pair of arms L times. At the end of this phase, $t = L(K-1)K/2$.

if RMED2FH **then**

  For each arm $i \in [K]$, fix $\hat{b}^\star(i)$ by (6).

end if

$L_C, L_R \leftarrow [K], L_N \leftarrow \emptyset$.

while $t \leq T$ **do**

  if RMED2 **then**

    Draw all pairs (i, j) until it reaches $N_{i,j}(t) \geq \alpha \log \log t$. $t \leftarrow t + 1$ for each draw.

  end if

  for $l(t) \in L_C$ in an arbitrarily fixed order **do**

    Select $m(t)$ by using $\begin{cases} \text{Algorithm 2} & \text{(RMED1)} \\ \text{Algorithm 3} & \text{(RMED2, RMED2FH)} \end{cases}$.

    Draw arm pair $(l(t), m(t))$.

    $L_R \leftarrow L_R \setminus \{l(t)\}$.

    $L_N \leftarrow L_N \cup \{j\}$ (without a duplicate) for any $j \notin L_R$ such that $\mathcal{J}_j(t)$ holds.

    $t \leftarrow t + 1$.

  end for

  $L_C, L_R \leftarrow L_N, L_N \leftarrow \emptyset$.

end while

Relative Minimal Empirical Divergence (RMED)

Reference: Regret Lower Bound and Optimal Algorithm, 2015

Double Thompson Sampling

- Motivation:
  - Extends the Thompson Sampling in relative setting to apply dueling bandit
  - TS achieves lower regret in practice and more robust than other algorithms

Double Thompson Sampling

- Motivation:
  - Extends the Thompson Sampling in relative setting to apply dueling bandit
  - TS achieves lower regret in practice and more robust than other algorithms
- Idea: Sample from the parameters representing the number of winning: B_{ij}

Double Thompson Sampling

- Motivation:
  - Extends the Thompson Sampling in relative setting to apply dueling bandit
  - TS achieves lower regret in practice and more robust than other algorithms
- Idea: Sample from the parameters representing the number of winning: B_{ij}
- Elimination:
  - RUCB-based: choose the first arm based on upper confidence bound
  - RLCB-based: choose the second arm only from the uncertain pair (with high lower confidence bound)
  - Objective: Avoid falling into the suboptimal comparisons.

Double Thompson Sampling

- Motivation:
  - Extends the Thompson Sampling in relative setting to apply dueling bandit
  - TS achieves lower regret in practice and more robust than other algorithms
- Idea: Sample from the parameters representing the number of winning: B_{ij}
- Elimination:
  - RUCB-based: choose the first arm based on upper confidence bound
  - RLCB-based: choose the second arm only from the uncertain pair (with high lower confidence bound)
  - Objective: Avoid falling into the suboptimal comparisons.
- Regret: $O(KlogT + K^2 loglogT)$

Double Thompson Sampling

DT-S+

- change the selection of the first arms
- minimize the regret of comparison $\tilde{R}_i^{(1)}(t)$

Double Thompson Sampling

Algorithm 1 D-TS for Copeland Dueling Bandits

Init: $\boldsymbol{B} \leftarrow \boldsymbol{0}_{K \times K}$; // *$B_{ij}$ is the number of time-slots that the user prefers arm i to j.*

for $t = 1$ **to** T **do**

  // *Phase 1: Choose the first candidate $a^{(1)}$*

  $\boldsymbol{U} := [u_{ij}]$, $\boldsymbol{L} := [l_{ij}]$, where $u_{ij} = \frac{B_{ij}}{B_{ij}+B_{ji}} + \sqrt{\frac{\alpha \log t}{B_{ij}+B_{ji}}}$, $l_{ij} = \frac{B_{ij}}{B_{ij}+B_{ji}} - \sqrt{\frac{\alpha \log t}{B_{ij}+B_{ji}}}$, if $i \neq j$, and $u_{ii} = l_{ii} = 1/2, \forall i$; // *$\frac{x}{0} := 1$ for any x.*

  $\hat{\zeta}_i \leftarrow \frac{1}{K-1} \sum_{j \neq i} \mathbb{1}(u_{ij} > 1/2)$; // *Upper bound of the normalized Copeland score.*

  $\mathcal{C} \leftarrow \{i : \hat{\zeta}_i = \max_j \hat{\zeta}_j\}$;

  for $i, j = 1, \ldots, K$ with $i < j$ **do**

    Sample $\theta_{ij}^{(1)} \sim \mathrm{Beta}(B_{ij}+1, B_{ji}+1)$;

    $\theta_{ji}^{(1)} \leftarrow 1 - \theta_{ij}^{(1)}$;

  end for

  $a^{(1)} \leftarrow \arg\max_{i \in \mathcal{C}} \sum_{j \neq i} \mathbb{1}(\theta_{ij}^{(1)} > 1/2)$; // *Choosing from $\mathcal{C}$ to eliminate likely non-winner arms; Ties are broken randomly.*

  // *Phase 2: Choose the second candidate $a^{(2)}$*

  Sample $\theta_{ia^{(1)}}^{(2)} \sim \mathrm{Beta}(B_{ia^{(1)}}+1, B_{a^{(1)}i}+1)$ for all $i \neq a^{(1)}$, and let $\theta_{a^{(1)}a^{(1)}}^{(2)} = 1/2$;

  $a^{(2)} \leftarrow \arg\max_{i: l_{ia^{(1)}} \leq 1/2} \theta_{ia^{(1)}}^{(2)}$; // *Choosing only from uncertain pairs.*

  // *Compare and Update*

  Compare pair $(a^{(1)}, a^{(2)})$ and observe the result w;

  Update $\boldsymbol{B}$: $B_{a^{(1)}a^{(2)}} \leftarrow B_{a^{(1)}a^{(2)}} + 1$ if $w = 1$, or $B_{a^{(2)}a^{(1)}} \leftarrow B_{a^{(2)}a^{(1)}} + 1$ if $w = 2$;

end for

Table of Contents

Dependent Arms

- Motivation: solve the problem of dueling bandit arm with dependent arms

Dependent Arms

- Motivation: solve the problem of dueling bandit arm with dependent arms
- Gaussian Process:
  - use covariance between arms to model the dependency
  - posterior inference updates the mean reward estimates for all arms and update the dependency

Self-Sparring

- Idea:
  - reduces the multi-dueling bandits problem to a conventional MAB bandit
  - simultaneously dueling multiple arms as well as modeling dependencies between arms using a kernel

Self-Sparring

- Idea:
  - reduces the multi-dueling bandits problem to a conventional MAB bandit
  - simultaneously dueling multiple arms as well as modeling dependencies between arms using a kernel
- Assumption: approximate linearity (using linear function to estimate the preference based on utility u)

Self-Sparring

- Idea:
  - reduces the multi-dueling bandits problem to a conventional MAB bandit
  - simultaneously dueling multiple arms as well as modeling dependencies between arms using a kernel
- Assumption: approximate linearity (using linear function to estimate the preference based on utility u)
- Algorithm: Ind-Self-Sparring, Kernel-Self-Sparring

Algorithm 2 SELFSPARRING

input arms $1, \ldots, K$ in space S, m the number of arms drawn at each iteration, η the learning rate

1: Set prior D_0 over S
2: **for** $t = 1, 2, \ldots$ **do**
3:     **for** $j = 1, \ldots, m$ **do**
4:         select arm $i_j(t)$ using D_{t-1}
5:     **end for**
6:     Play m arms $\{i_j(t)\}_j$ and observe $m \times m$ pairwise feedback matrix $R = \{r_{ij} \in \{0, 1, \emptyset\}\}_{m \times m}$
7:     update D_{t-1} using R to obtain D_t
8: **end for**

Ind-Self-Sparring

Ind-Self-Sparring:

- independent cases
- SBM: Beta-Bernoulli Thompson Sampling
- Regret: $O(K log T/\Delta)$

Ind-Self-Sparring

Algorithm 3 IndSelfSparring

input m the number of arms drawn at each iteration, η the learning rate

For each arm $i = 1, 2, \cdots, K$, set $S_i = 0$, $F_i = 0$.
for $t = 1, 2, \ldots$ **do**
  for $j = 1, \ldots, m$ **do**
    For each arm $i = 1, 2, \cdots, K$, sample θ_i from $Beta(S_i + 1, F_i + 1)$
    Select $i_j(t) := \text{argmax}_i \, \theta_i(t)$
  end for
  Play m arms $\{i_j(t)\}_j$, observe pairwise feedback matrix $R = \{r_{jk} \in \{0, 1, \emptyset\}\}_{m \times m}$
  for $j, k = 1, \ldots, m$ **do**
    if $r_{jk} \neq \emptyset$ **then**
      $S_j \leftarrow S_j + \eta \cdot r_{jk}$, $F_j \leftarrow F_j + \eta(1 - r_{jk})$
    end if
  end for
end for

Kernel-Self-Sparring

- dependent cases

Kernel-Self-Sparring

- dependent cases
- Idea:
  - use Gaussian processes $GP((b), k(b, b'))$. to model the preference function $f(b)$
  - apply Bayesian update using $(i_j(t), r_{jk})$ to obtain (μ_t, σ_t)

Kernel-Self-Sparring

- dependent cases
- Idea:
  - use Gaussian processes $GP((b), k(b, b'))$. to model the preference function $f(b)$
  - apply Bayesian update using $(i_j(t), r_{jk})$ to obtain (μ_t, σ_t)
- Regret: $O(KlogT)$ (A remaining opening problem)

Kernel-Self-Sparring

Algorithm 4 KERNELSELFSPARRING

input Input space S, GP prior (μ_0, σ_0), m the number of arms drawn at each iteration

for $t = 1, 2, \ldots$ **do**
  for $j = 1, \ldots, m$ **do**
    Sample f_j from $(\mu_{t-1}, \sigma_{t-1})$
    Select $i_j(t) := \text{argmax}_x\, f_j(x)$
  end for
  Play m arms $\{i_j(t)\}_j$, observe pairwise feedback matrix $R = \{r_{jk} \in \{0, 1, \emptyset\}\}_{m \times m}$
  for $j, k = 1, \ldots, m$ **do**
    if $r_{jk} \neq \emptyset$ **then**
      apply Bayesian update using $(i_j(t), r_{jk})$ to obtain (μ_t, σ_t)
    end if
  end for
end for

Self-Sparring

Reference: Multi-dueling Bandits with Dependent Arms, 2017

Dependent Arms Application

CoSpar Algorithm:

- the user can suggest improvements in the form of coactive feedback
- Reference: Reference Preference-Based Learning for Exoskeleton Gait Optimization, 2020

Table of Contents

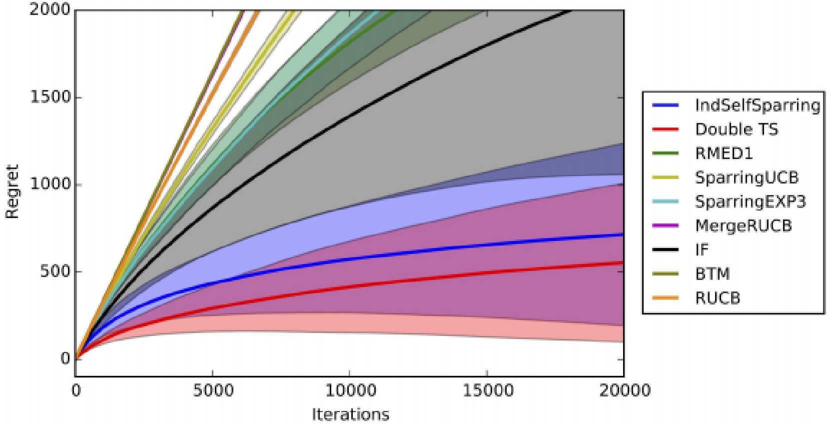
Application Scenarios of Main Algorithms

- IF(Interleaved Filter, 2009): strong stochastic transitivity (**SST**)
- BTM (2011): relaxed stochastic transitivity (**RST**)
- UBDB (2014): reducing the Dueling Bandits problem to the **conventional Multi-Armed** Bandits problem
- RUCB (2014): Apply UCB in relative setting (by **upper bound**)
- CCB (2015): Using Upper bound to solve **Copeland Bandit** problem
- RMED (2015): **Minimum Empirical Divergence**
- DTS (2016): Extends the **Thompson Sampling** in relative setting to apply dueling bandit
- Self-Sparring (2017): Solve Dueling Bandit problem with **dependent arms**

Performance of Main Algorithm

Algorithm	Regret
IF	$O(K log T/\Delta_{min})$
BTM	$O(\frac{\gamma^7 K}{\Delta_{min}} log T)$
Sparring	$O(K log T)$
RUCB	$O(K^2 + K log T)$
SAVAGE	$O(K^2 log T)$
CCB	$O(K^2 + K log T)$
RMED	$R_T \geq \sum_{k=2}^{K} \min_{\{j \mid p_{ij} < .5\}} \frac{(\Delta_{1i} + \Delta_{1j}) log T}{2d(p_{ij}, .5)}$
DTS	$O(K log T + K^2 log log T)$
Self-Sparring	$O(K log T/\Delta_{min})$?

Performance of Main Algorithm



Algorithm Basis

- Upper Bound: RUCB, CCB
- Thompson Sampling: DTS
- Minimum Empirical Divergence: RMED
- Reduce to Conventional MAB: Sparring, SelfSparring

Explore-Exploit Tradeoff

Conclusion

- Elicits preference feedback
  - Motivated by human-centric personalization
  - Explore-exploit Tradeoff
- Algorithm Basis
  - Upper Bound
  - Thompson Sampling
  - Minimum Empirical Divergence
  - Reduce to Conventional MAB

Prospect

Ongoing research

- Personalized clinical treatment (High-Dimensional Multi-arms dueling bandit)
- Dependent Arms (regret bound)
- Apply more basic MAB algorithms in Dueling Bandit (e.g. Gradient Descent, Upper Bound in Dependent arms)
- Complex dueling mechanisms (Algorithms different with conventional MAB problem)

Reference

- The K-armed Dueling Bandits Problem, Yisong Yue, Josef Broder, Robert Kleinberg and Thorsten Joachims, COLT 2009
- Beat the Mean Bandit, by Yisong Yue and Thorsten Joachims, ICML 2011
- Generic Exploration and K-armed Voting Bandits, Tanguy Urvoy, Fabrice Clerot, Raphael Feraud, Sami Naamane, 2013
- Relative Upper Confidence Bound for the K-armed Dueling Bandit Problem, Masrour Zoghi, Shimon Whiteson, Remi Munos and Maarten de Rijke, ICML, 2014
- Reducing Dueling Bandits to Cardinal Bandits, Nir Ailon, Zohar Karnin and Thorsten Joachims, ICML 2014
- Copeland Dueling Bandits, Masrour Zoghi, Zohar Karnin, Shimon Whiteson, and Maarten de Rijke, 2015
- Regret Lower Bound and Optimal Algorithm in Dueling Bandit Problem, Junpei Komiyama, Junya Honda, Hisashi Kashima, Hiroshi Nakagawa, 2015

Reference

- Double Thompson Sampling for Dueling Bandits, Huasen Wu, Xin Liu, NIPS 2016
- Multi-dueling Bandits with Dependent Arms, Yanan Sui, Vincent Zhuang, Joel W. Burdick, Yisong Yue, 2017
- Preference-Based Learning for Exoskeleton Gait Optimization, Maegan Tucker, Ellen Novoseller, Claudia Kann, Yanan Sui, Yisong Yue, Joel W. Burdick, and Aaron D. Ames, 2020