Assistive Dueling Bandits: No-Regret Algorithms for Assisting No-Regret Users

Mark Bedaywi, Cassidy Laidlaw, Austin Tripp, Nika Haghtalab
UC Berkeley

Abstract

We study a stochastic bandit problem in which learning is split between two agents. In each round, a human observes rewards but can choose only between a pair of arms selected by an assistant, while the assistant observes the human's behavior but not realized rewards. We call this model the **assistive dueling bandit**. The human is modeled as a learning agent whose regret on any subset of arms is bounded by a function $g(T)$ unknown to the assistant. This is a natural model for recommender systems or AI assistants that must present a slate of options to a user who may be still learning about their own preferences.

We provide a general reduction from assistive dueling bandits to the problem of **max-finding with imprecise feedback.** With it, we design assistant algorithms that cooperate with any $g(T)$-regret human to achieve a joint regret of $\tilde{\mathcal{O}}(K \cdot g(T/K))$, even without knowledge of $g(T)$. Crucially, when $g(T) \in \mathcal{O}(\sqrt{T})$, our method recovers the nearly optimal minimax rate of $\tilde{\mathcal{O}}(\sqrt{KT})$, implying that splitting up the responsibilities of learning in this way can be done without suffering any excess regret asymptotically. We complement our theoretical analysis with experimental evidence that this algorithm outperforms an assistant implemented via standard dueling bandit algorithms.

1 Introduction

The alignment of AI systems with human values relies fundamentally on the quality of the feedback those systems receive. The vast majority of preference learning literature—whether in Multi-Armed Bandits (MAB) [Lattimore and Szepesvári, 2020], Dueling Bandits [Yue and Joachims, 2009, Yue et al., 2012], or Reinforcement Learning from Human Feedback (RLHF) [Christiano et al., 2017]—rests on a static assumption: that the human is an oracle with a fully formed, latent preference function. Under this view, any deviation from the optimal choice is treated merely as stochastic noise.

In reality, human preferences are rarely static or pre-defined. Users are often learners themselves, discovering their tastes through interaction with the environment. Consider a video recommendation platform such as Netflix or YouTube that presents a small slate of content from a larger library. When a user selects an item from this slate, their motivation is often ambiguous. They may choose a video because they are certain it aligns with their interests, or they may be motivated by a desire to explore—to try a new genre, indulge a contrarian impulse, or to simply refine their understanding of their own tastes.

This ambiguity poses a central challenge for recommendation algorithms. Standard preference-learning methods (e.g., those used in dueling bandits), which interpret every choice as a revealed preference, can be easily misled by exploratory or contrarian behavior. Treating such actions as genuine preferences risks locking the system into recommending a slate of uninteresting content to the user. Ignoring user choices altogether is also problematic, leading to irrelevant recommendations, failing to help users discover enjoyable content or meaningfully refine their tastes. This raises a critical algorithmic question:

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

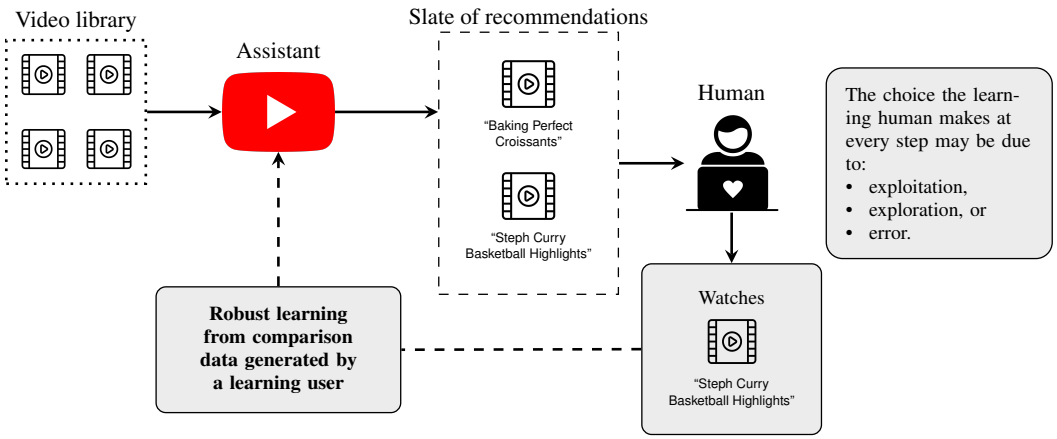


Figure 1: **The Assistive Dueling Bandit Framework.** We model the recommendation problem as a cooperative game between an AI assistant and a learning human. (Left) The assistant selects a menu of arms to present. (Right) The Human, modeled as a learning agent rather than a fixed oracle, selects an arm based on their current (potentially incomplete) preference model. The assistant does not know the human's learning algorithm, does not know when the human is choosing to explore, to exploit, or is making an error, and does not observe any of the rewards the human is receiving. The assistant only observes the choices the human makes. Unlike standard settings where feedback is static, the bandit must account for the user's learning process that leads to non-stationary behavior. The assistant's goal is, with very little information, to recommend in such a way that a perfect version of the human that fully understands their preferences does not regret their actions much in hindsight.

How can we design recommender systems that assist a human in finding their optimal preference when the human does not yet know what that preference is?

Assistive Dueling Bandits: To rigorously answer this, we introduce the Assistive Dueling Bandit framework. This setting is built on top of a stochastic K-armed bandit problem, where each arm is associated with an unknown reward distribution, and each arm pull generates a stochastic reward from the corresponding distribution. An assistive dueling bandit models the interaction between an assistant and a human as a cooperative game over T rounds in this environment. At every round,

- **The Assistant:** Selects a small menu of arms (in our case, 2 arms) to present to the user. It observes the human's choice, but never observes the reward directly.
- **The Human:** Chooses one of the arms from this menu and experiences the reward. The human is modeled not as a noisy oracle, but as an agent whose choices lead to cumulative regret bounded by an unknown function $g(T)$.

The assistive dueling bandit framework sits in the middle ground between two single-agent perspectives in bandits. Unlike the standard MAB setting (where the learner explores alone), the human is only exposed to a small subset of arms selected by the assistant. Unlike dueling bandits, the human's choices are not drawn from a stationary distribution parameterized by the mean rewards, but can instead be consistent with any no-regret learning algorithm. The key challenge in designing assistant algorithms is therefore to robustly learn from unreliable human feedback, while simultaneously producing menus of arms that guide the human's learning toward options they are more likely to enjoy.

Algorithm Design for Assistive Dueling Bandits The core challenge is that feedback from a learning human is unreliable, at least until they have experienced enough comparisons to distinguish higher-quality from lower-quality arms. While as a whole, the feedback becomes more reliable as the human gains additional exposure, the rate at which it does so depends both on the human's unknown learning rate (namely $g(\cdot)$) and the unknown reward gap between the arms they are recommended.

This insight allows us to reduce the problem of learning to assist a learning agent as to the problem of *finding maxima with imprecise feedback*, as studied by Ajtai et al. [2015], Acharya et al. [2018], Gopi et al. [2020], Trevisan [2024]. The latter problem is concerned with finding the entry max

of a list $v_1, \ldots, v_K$ from feedback that takes the following form: A query 'is v_i larger than v_j' is only guaranteed to be correctly answered when $|v_i - v_j| \geq \tau$, for some threshold $\tau \in \mathbb{R}$. For this reduction to work, we of course have to ensure that human regret remains low, even as we gradually work with smaller values of τ, that relate to the learning rate of the agent. To achieve this, we treat the user's choices as a tournament, where an arm i beats arm j if a human historically preferred i to j. By solving for the winner of this tournament in phases of decreasing τ, we safely retain arms that are likely to be of high mean reward.

This reduction will allow us to develop provably optimal assistants that have the human learn at a rate only a logarithmic factor worse than if the human were playing the full multi-armed bandit on all K arms alone.

2 Related Work

The assistive dueling bandit setting is similar to a special case of assistance games. Introduced by Hadfield-Menell et al. [2016], an assistance game models cooperation between a human who knows some hidden preference and an assistant who does not. Despite being a decade old, only recently have people started studying solutions to assistance games [Laidlaw et al., 2025]. Unlike most work on assistance games, we do not assume that the human initially knows their preferences.

Another extension of the bandit problem to assistance is the assistive multi-armed bandit proposed by Chan et al. [2019]. In an assistive multi-armed bandit, a human picks arms to pull and an assistant can intervene to override the human's pull; only the human sees the reward. The goal of this work is to build assistants that can strategically change which arms a user pull's to help the user learn faster when the user is suboptimal. While this setting is interesting and related, they do not propose any algorithms that provably minimize regret. Instead, they assume that the human's learning strategy and a prior over rewards are known and solve the problem as a POMDP. Our goal, in contrast, is to build assistants that when paired with any human minimizes that human's regret, even when the human's policy and the distribution over possible preferences are not known in advance.

Some work has explored how to recommend content to a user that is still learning their preferences. Dzyabura and Hauser [2019] give experimental evidence that users of online marketplaces learn about their preferences as they receive product recommendations, and propose an algorithm for sorting recommendations to help users learn their preferences faster. Carroll et al. [2024] study more broadly what happens when the choices of an assistant change the preferences of the user they are assisting.

A large literature has emerged on the study of dueling bandits, a variant of the multi-armed bandit where feedback is ordinal rather than cardinal. First proposed by Yue and Joachims [2009], Yue et al. [2012], a number of very interesting algorithms have been proposed to get optimal dueling bandit regret. These include algorithms that estimate confidence intervals from feedback like the Relative UCB algorithm [Zoghi et al., 2014], algorithms that solve for different notions of regret like Copeland dueling bandits [Zoghi et al., 2015], or algorithms which reduce dueling bandits to multi-armed bandits [Ailon et al., 2014]. The sparring dueling bandit algorithm introduced by Ailon et al. [2014] and developed in further work like Gajane et al. [2015], Balsubramani et al. [2016] solves the dueling bandit with a self-play procedure. Saha et al. [2021] introduce a class of dueling bandit algorithms that are robust to adversarially generated feedback. We refer the reader to the survey Bengs et al. [2021] for a more complete comparison of existing algorithms.

3 Preliminaries

3.1 Notation

Let $[N]$ be the set of integers $\{1, \ldots, N\}$. For any set A and natural number k, let A^k denote the collection of all k-tuples of A. For any set A, let $\Delta(A)$ denote the set of probability distributions over A. The uniform distribution over A is denoted by $\mathrm{Unif}(A)$. Let $\mathbb{R}_+$ denote the non-negative reals. For any $x \in \mathbb{R}$, let $x_+ = \max(0, x)$.

3.2 The Assistive Dueling Bandit Setting

Our model of interaction, the **Assistive Dueling Bandit**, involves an environment (adversary), an assistant, and a human user. The interaction proceeds for T rounds over a set of K arms.

Environment. At the start of the interaction, the environment fixes a reward distribution $\mathcal{D} \in \Delta([0,1]^K)$ for the arms. At each round, a reward vector is drawn from $\mathcal{D}$ independently across rounds.

For each arm $i \in [K]$, let $\mathcal{D}_i$ denote its marginal reward distribution and $\mu_i = \mathbb{E}_{\mathcal{D}_i}[r_i] \in [0,1]$ denote its mean reward. We assume the reward distribution is fixed; as we discuss in Appendix A.2, learning is impossible if $\mathcal{D}$ evolves adaptively and adversarially over time.

Interaction Protocol. At each round $t \in [T]$:

1. The assistant selects a slate of a pair of arms $S_t \in [K]^2$ to present to the human.
2. The human selects one arm $i_t \in S_t$ to pull and observes a reward $r_t \sim \mathcal{D}_{i_t}$.
3. The assistant observes the human's choice i_t, but *not* the reward r_t.

We analyze performance by studying the *human's pseudo-regret*. This is the optimality of the arms actually pulled by the human.

Definition 1 (Human Regret)**.** *A sequence of arm pulls $i_1, \ldots, i_T$ achieves a **human regret** of:*

$$R_{\text{user}}(T) = T \cdot \max_{i^* \in [K]} \mu_{i^*} - \sum_{t=1}^{T} \mu_{i_t}.$$

Note that the human regret is not defined with respect to the best arms they are shown, but rather to the overall best arm. Thus, the only way for the assistant algorithm to ensure that the human's average regret tends towards zero is to learn to recommend them near-optimal arms, *while never observing any reward signal at all.*

Pairwise Statistics. To analyze the human's feedback, we track the history of pairwise comparisons. Let $N_t(i,j)$ denote the number of times the pair $\{i,j\}$ has been presented up to time t. Let $N_t(i \succ j)$ denote the number of times the human chose arm i when presented with $\{i,j\}$.

Definition 2 (Empirical Beat)**.** *At time t, we say arm i **beats** arm j if i has been chosen at least as often as j in direct comparisons: $N_t(i \succ j) \geq N_t(i,j)/2$.*

Finally, for any pair of arms i, j, let $\Delta_{i,j} = |\mu_i - \mu_j|$ be the mean gap, and let $\Delta_i = \max_k \mu_k - \mu_i$ be the sub-optimality gap of arm i.

3.3 The Human Model

We model the human as a learning agent whose cumulative pseudo-regret on any subset of arms is bounded, a property we term $g(\cdot)$-rationality.

Definition 3 (g-Rationality)**.** *For a pair of arms $\{i,j\}$, let*

$$I_{i,j}^T = \{t \leq T : S_t = \{i,j\}\}, \qquad N_T(i,j) = |I_{i,j}^T|.$$

*A human policy is $g(\cdot,\cdot)$-**rational** if for every horizon T, every pair of arms $\{i,j\}$, and every $\delta \in (0,1)$, with probability at least $1-\delta$,*

$$\max_{i^* \in \{i,j\}} \sum_{t \in I_{i,j}^T} (\mu_{i^*} - \mu_{i_t}) \leq g(N_T(i,j), \delta)\,.$$

The probability is over the human's choices, reward observations, and the assistant policy.

g-rationality is a fairly conservative assumption on the human's behavior, as it does not imply transitivity: a $g(\cdot,\cdot)$-rational human who has been exposed to pairs of arms $\{i,j\}$ and $\{j,k\}$ many times may be able to correctly answer queries $\{i,j\}$ and $\{j,k\}$, but no assumption is made on the accuracy of the learning human's responses when asked to compare $\{i,k\}$. Our algorithms rely solely on pairwise rationality and function despite inconsistent human behavior.

We make the following standard assumptions on the regret bound $g : \mathbb{R}_+ \to \mathbb{R}$.

Definition 4. *We say $g : \mathbb{R}_+ \times \mathbb{R} \to \mathbb{R}$ satisfies the learning-human assumption when, for all fixed $\delta > 0$,*

1. *$g(t,\delta)$ is concave and weakly increasing in t,*
2. *$g(0,\delta) = 0$ and $g(t,\delta) > 0$ for all $t \geq 1$.*

Any reasonable regret bound must be positive and weakly increasing, as each $g(t, \delta)$ is an upper bound on the sum of all the regret incurred up until time t. Concavity is consistent with virtually all no-regret learning bounds. Regret being concave also aligns with experimental findings in human learning psychology [Heidari et al., 2016].

A useful consequence of this definition we will use repeatedly is that $g(t, \delta)/t$ is weakly decreasing for any g satisfying the learning-human assumption. This is because for any $x > y \geq 0$, by concavity

$$g(y, \delta) = g\left(\frac{y}{x} \cdot x + \left(1 - \frac{y}{x}\right) \cdot 0, \delta\right) \geq \frac{y}{x} \cdot g(x, \delta) + \left(1 - \frac{y}{x}\right) \cdot g(0, \delta) = \frac{y}{x} \cdot g(x, \delta).$$

Rearranging gives $g(y, \delta)/y \geq g(x, \delta)/x$.

4 Optimal g-Agnostic Assistant Algorithms

In this section, we describe our main algorithm for solving assistive dueling bandit problems. We describe a general framework of algorithms that let us plug and play any algorithm for finding maxima under imprecise comparisons. We show that these algorithms are optimal for any human they assist without needing to know a priori how rational the human is.

Our approach is to give a general reduction framework to the problem of max finding with imprecise feedback. We proceed in four conceptual steps. In Section 4.1, we formalize how after sufficiently many exposures to a pair $\{i, j\}$, the arm with sufficiently higher reward will empirically beat the other. In Section 4.2, we denote these empirical beats as edges of a tournament graph. Our main insight there is that short *beat paths*—directed paths where each edge denotes an empirical beating between the two ends—certify that the human has achieved low regret. In Section 4.3 we use this fact to design an iterative procedure that repeatedly builds sets in which exploration is safe. The algorithm will proceed in phases where in phase s we build a safe set W_s such that the maximum regret of any arm in W_s is at most $g(2^s)/(2^s)$. In Section 4.4 we describe the L-max finding algorithms used. All together, these findings result in an assistive dueling bandit algorithm with a regret rate of $\tilde{\mathcal{O}}(Kg(T/K))$.

4.1 Regret Bounds for Imprecise Feedback

If the human selects an arm j at least n times out of t, they must incur a regret of at least $n(\max_i \mu_i - \mu_j)$. Since this is at most $g(t)$, the regret of playing any arm must be upper bounded.

Observation 1 (Pairwise Comparisons Are In Probably Meaningful Eventually)**.** *For any iteration $t \in [T]$ and any pair of arms $\{i, j\} \in [K]^2$, and for any $\delta > 0$, suppose that*

$$2\frac{g(N_t(i, j), \delta)}{N_t(i, j)} \leq \mu_i - \mu_j.$$

Then with probability $1 - \delta$, i empirically beats j, i.e., $N_t(i \succ j) \geq N_t(j \succ i)$.

Proof. With probability at least $1 - \delta$, by g-rationality,

$$(\mu_i - \mu_j)N_t(j \succ i) \leq g(N_t(i, j), \delta).$$

Since $N_t(i \succ j) > N_t(j \succ i)$ and $N_t(i \succ j) + N_t(j \succ i) = N_t(i, j)$, we know that $N_t(i \succ j) \geq N_t(i, j)/2$. Therefore, with probability $1 - \delta$,

$$\mu_i - \mu_j \leq 2\frac{g(N_t(i, j), \delta)}{N_t(i, j)}.$$ □

Operationally, comparing a pair $\{i, j\}$ for L rounds yields one of two outcomes, depending on how close together the rewards of the arms are:

- **Separated arms:** If $|\mu_i - \mu_j| \geq 2g(L, \delta)/L$, the empirical winner is correct with high probability.
- **Near-ties:** If $|\mu_i - \mu_j| < 2g(L, \delta)/L$, the outcome may be arbitrary.

The more the assistant proposes a particular pair of arms, the smaller this threshold $g(L, \delta)/L$ becomes. The speed at which this happens depends on how fast the human learns. Achieving the desired precision is not free: comparing $\{i, j\}$ for L rounds necessarily incurs up to $L \cdot \max\{\Delta_i, \Delta_j\} + g(L, \delta)$ human regret, and so getting a precise comparison is very expensive when both arms are bad, or the human learns slowly. And because the assistant does not know $|\mu_i - \mu_j|$, it can never know whether or not it has shown the human the pair enough for a comparison to have been precise.

4.2 Graph-Theoretic Abstraction: The Domination Graph

How can we use human feedback to learn what the best arm is, when any feedback may be imprecise and arbitrary, and especially when we do not have knowledge of the rate at which the human is learning? The key insight is to look at the *paths* formed in the corresponding graph describing wins and losses.

Definition 5 (Domination Graph)**.** *Let G_t be a tournament graph on $[K]$ where an edge $i \to j$ indicates that i empirically beats j at time t, i.e., $N_t(i \succ j) \geq N_t(j \succ i)$ (ties broken arbitrarily). G_t is said to be the domination graph of the assistive dueling bandit at iteration t.*

Definition 6 (Beat Paths)**.** *An arm i is said to beat j via a* strength-s path of length L *if there exists a directed path $i = i_1 \to i_2 \to \cdots \to i_L = j$ in G_t such that $N_t(i_\ell, i_{\ell+1}) \geq s$ for all $\ell \in [L-1]$.*

*An arm i is an L-*max of strength s *if it admits such a path to every other arm.*

Beat paths allow us to chain the guarantee in Observation 1 together. An L-max i must have a beat path of length L to all nodes, and so in particular must have one to i^*. Union bounding the errors on each edge, this must mean that for any $\delta > 0$, any arm that is an L-max in the domination graph must have in regret at most $\Delta_i \leq 2L \cdot \frac{g(s,\delta)}{s}$, assuming we are not in the δ probability event that a comparison has failed.

The problem now becomes finding L-maxes of large strength in the domination graph. There does not have to exist a 1-max in the domination graph; in a cycle graph, every node has some incoming edge. Surprisingly, in any tournament graph, there always exists a 2-max. In particular, the node that beats most other nodes is always a 2-max. Intuitively, although cycles may prevent a Condorcet winner, every tournament admits a node that is within two steps of all others. This is stated in the following lemma.

Lemma 1. *[Landau, 1953] In a tournament graph G, the node in V with maximum out-degree is a 2-max.*

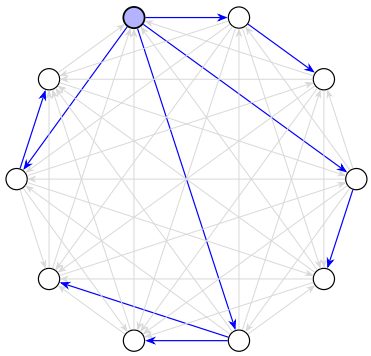


Figure 2: An illustration of Lemma 1 where a 2-max of a tournament is highlighted. Even though most tournaments do not have a vertex that beats all other vertices — a 1-max — every tournament provably must have a vertex that has a beat path of length at most two to every other vertex — a 2-max. Regardless of how inconsistent a human's preference is, there is always at least one option i such that for any option j, either i beats j, or i beats some option ℓ that beats j. Recommending an L-max is the correct way of dealing with a learning user's inconsistent preferences.

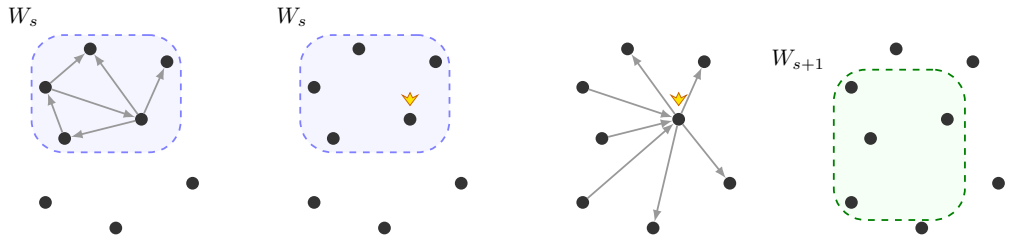
4.3 Iterative Refinement via Imprecise Max-Finding

We now present our main algorithm. The goal is to identify an arm that is a short beat path away from all others—and thus low regret—while minimizing the cost of exploration. To accomplish this, we offer a general reduction from L-max algorithms to bandit algorithms, a reduction that is able to induce asymptotically optimal assistant algorithms for minimizing regret.

Ideally, we would want to run a max-finding procedure directly on the assistive dueling bandit. But procedures for finding maxes could incur large regret if they fixate on low reward arms. To circumvent this, we employ a bootstrapping strategy. If we were to already know that an arm i has large reward, there is a simple way to curate a set of arms that do not have reward much less than that of i. We use Observation 1 to (imprecisely) compare every arm $j \in [K] \setminus \{i\}$ with arm i a total N times, and maintain a "winning set" W of arms that are chosen over i more often than not.

We proceed in phases $s = 1, 2, \ldots$. In each phase, we maintain the winning set W_s of candidate arms. We use an L-max finding oracle $\mathcal{A}$ to identify a winner $i_s \in W_s$ using imprecise queries with precision 2^s. We then filter the full set of arms $[K]$, retaining only i_s and the arms that empirically beat it. By Observation 1, arms in W_{s+1} are guaranteed to be close in utility to i_s, ensuring that the regret of arms in the safe set shrinks as the precision increases.

The intuition for why this algorithm works is that we always ensure that there is a short path from arms we play to any other arm. As the algorithm goes on, we ask that the paths become more and more strengthened. In doing so, we do not ever need to have been told how fast the human is learning, our regret only depends on how fast they can correctly orient edges along paths.



(a) Run $\mathcal{A}$ on the previous winning set W_s, making each comparison 2^s times

(b) Use $\mathcal{A}$ to pick an L-max from W_s

(c) Compare the winner with everything 2^s times each

(d) Build a new winning set W_{s+1}

Figure 3: This figure describes the main reduction of our algorithm. We are given a max finding algorithm $\mathcal{A}$ and want to use it to find good arms. To avoid having to pay too much, we bootstrap off of previous good arms in the following way. We split execution into phases, where as the phases go along, we ask for double the level of preciseness in the comparison compared to last phase. At every phase s, a winning set W_s is maintained. This is the set of arms that are safe to play without suffering much regret. In (a) we run $\mathcal{A}$ on W_s. In (b), we use the output of $\mathcal{A}$ to name a winner $i_s^* \in W_s$. In (c), the winner is then used to filter out all bad arms, and in (d) we build a new winning set W_{s+1} out of i_s^* and all the arms that beat i_s^*. This allows us to slowly bootstrap off of $\mathcal{A}$ without incurring excessive regret.

Algorithm 1 A g-Agnostic Assistant via Imprecise Max-Finding

Require: L-Max finding algorithm $\mathcal{A}$ with query complexity Q.

Initialize winning set $W_0 \leftarrow [K]$, and phase $s \leftarrow 0$.
while number of iterations is at most T **do**

Step 1: Run the Oracle With Precision 2^s

  Initialize $\mathcal{A}$ on the subgraph induced by W_s.
  while $\mathcal{A}$ has not terminated **do**
    Get query pair (i, j) from $\mathcal{A}$.
    Present $\{i, j\}$ to human 2^s times.
    Return edge $i \to j$ to $\mathcal{A}$ if $N_t(i \succ j) \geq N_t(j \succ i)$, else $j \to i$.
  end while
  Let i_s be the candidate returned by $\mathcal{A}$.

Step 2: Filter the Set of Arms

  for each $j \in [K] \setminus \{i_s\}$ **do**
    Present $\{i_s, j\}$ to human 2^{s+1} times.
  end for
  Update W_{s+1} to be

$$\{i_s\} \cup \{\, j \in [K] \mid N_t(j \succ i_s) \geq N_t(i_s \succ j)\,\}.$$

  Increment $s \leftarrow s + 1$.
end while

Formally, a max finding oracle can be defined like so.

Definition 7 (L-Max Finding Oracles)**.** *Let G be an arbitrary tournament graph on vertices V. An L-max finding oracle with query complexity Q, $\mathcal{A}$, is an algorithm that adaptively*

1. *Outputs a query $\{i, j\} \subseteq V$,*
2. *And takes as input an orientation: either $i \to j$ or $j \to i$.*

After Q rounds, the algorithm outputs some L-max $i \in V$.

We are now ready to describe the main theorem of this work.

Theorem 4.1 (Reduction to Imprecise Max Finding)**.** *Let $\mathcal{A}$ be an L-max oracle with query complexity Q and let $\delta > 0$ be arbitrary. Running Algorithm 1 with $\mathcal{A}$ with a g-rational human, for any choice of g, achieves a human regret of*

$$\mathcal{O}\left(Q + L(K+Q)\log({}^{T}/_{(Q+K)}) \cdot g\left(\frac{T}{(Q+K)\log({}^{T}/_{(Q+K)})}, \frac{\delta}{T\binom{K}{2}}\right)\right).$$

with probability at least $1-\delta$. When $g(t,\delta) \in \mathcal{O}\left(t^{\alpha} \cdot f(\delta)\right)$ for some constant $\alpha > 0$ and function $f : [0,1] \to [0,\infty)$, this can be improved to

$$\mathcal{O}\left(Q + Lf\left(\frac{\delta}{T\binom{K}{2}}\right) \cdot (K+Q)^{1-\alpha} \cdot T^{\alpha}\right).$$

We have reduced the problem of assisting a learning human to the well-studied literature on finding L-maxes in tournaments. It remains to describe what algorithms compute max finding oracles.

4.4 L-Max Finding Oracles

A line of work from the past decade has proposed algorithms for this kind of max finding [Ajtai et al., 2015, Acharya et al., 2018, Gopi et al., 2020, Trevisan, 2024]. A simple algorithm that is effective when K is small is to simply run a knock-out tournament. We describe this in Algorithm 2. This deterministically returns a $\lceil \log_2(K) \rceil$-max in $2K$ queries. Ajtai et al. [2015] give an algorithm that finds $\mathcal{O}\left(\log\log(K)\right)$-maxes in $\mathcal{O}\left(K\right)$ queries by cleverly recursing knock-out tournaments. We describe this in Appendix B.3. In Appendix B, we describe these algorithms in detail.

Corollary 1 (Nearly Minimax Optimal Assistants)**.** *The assistant running the max-finding algorithm in Algorithm 5 with Algorithm 1 achieves a regret of*

$$\mathcal{O}\left(K + \log\log(K)\log\left({}^{T}/_{K}\right)K \cdot g\left(\frac{{}^{T}/_{2K}}{\log({}^{T}/_{2K})}, \frac{\delta}{T\binom{K}{2}}\right)\right).$$

with probability at least $1-\delta$. In particular, when $g(t,\delta) = \sqrt{t\log(1/\delta)}$, we achieve a regret of

$$\mathcal{O}\left(K + \log\log(K)\sqrt{TK\log({}^{TK}/_{\delta})}\right)$$

with probability at least $1-\delta$.

4.5 Extensions

In the appendix, we describe a few extensions of our algorithm. First, in Appendix F, we describe how the algorithm can be easily extended to work in the case where the assistant recommends a menu of $R > 2$ items at a time. This involves generalizing the algorithms in Ajtai et al. [2015] to work for larger, batched queries. Second, in Appendix D we show that a variant of our algorithm that runs for only one phase minimizes a stronger form of regret that penalizes the assistant for showing bad arms to the human, even if the human did not pick the bad arm. Finally, in Appendix E we study the performance of our algorithm for standard dueling bandits with underlying utilities for each arm and show that our algorithm works as is with no modification.

5 Experiments

In this section we showcase the performance of our algorithms empirically. Because of the novelty of the setting, there are no baselines to compare to that have been designed for assistance. In lieu of this, we compare our algorithms to a range of standard dueling bandit algorithms in the literature. Of course, standard dueling bandit algorithms are not designed to work well here and have no guarantees on assistive dueling bandits. A description of each of the baselines tested is given in the Appendix G.

We run our Algorithm 1 with two choices of max finding algorithms: 1) the $\log(K)$-max $2K$ query tournament algorithm that is effective for small values of K; and 2) the $\mathcal{O}\left(\log\log K\right)$ max $\mathcal{O}\left(K\right)$ query algorithm of Ajtai et al. [2015]. We test our algorithms on various types of human behavior. The UCB and EXP3-IX humans run a separate instance of UCB and EXP3-IX respectively for every pair of arms. The maximally bad human with regret function g at time step t will always make a mistake if regret so far on the given pair up to iteration t is less than $g(t)$. We test against maximally

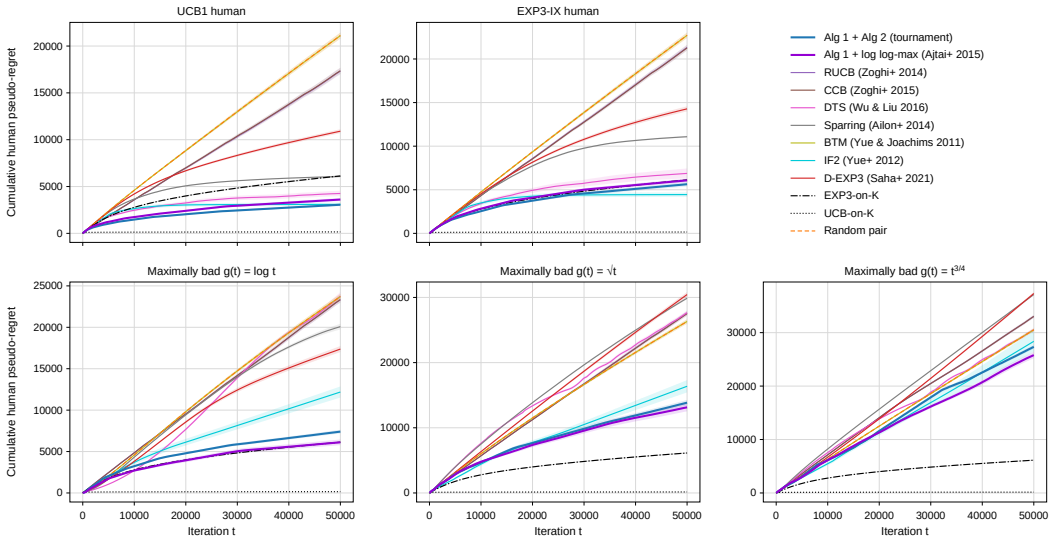


Figure 4: The cumulative regret for each algorithm and baseline, with $K = 100$ arms averaged over 25 seeds over the five types of humans to assist for $50{,}000$ iterations. Standard Error is shaded in. For each baseline, a sweep of hyperparameters was performed. The dark purple and blue lines denote our algorithms. The dotted orange line denotes the performance of recommending random arms to the user. The dotted black lines denote the performance of a user playing the 100 arm bandit alone without an assistant, either with EXP3 or UCB; these are not directly comparable to our algorithms but can be used to contextualize the assistive setting relative to the non-assistive setting.

bad humans with $g(t) \in \{\log(t), \sqrt{t}, t^{3/4}\}$. A hyperparameter sweep was performed on all baselines, with full details for reproducibility given in the appendix. We test the agents on Bernoulli bandits with initially uniformly randomly sampled means for each arm in $[0, 1]$.

In Figure 4, we see that our algorithm is consistently among the best performing in all settings. It is worth noting that our algorithms have no hyperparameters at all and require no tuning, as opposed to all the other algorithms. Despite this, they can adapt to all kinds of learning behavior. The algorithm IF2 from Yue et al. [2012] is surprisingly effective at assisting EXP3-IX and UCB humans, but begins to perform poorly on maximally bad no-regret humans. We see that a $g(t) = t^{3/4}$ regret human is particularly nasty, misleading most dueling bandit algorithms into performing much worse than even random play. Performance of the algorithms on smaller values of K and for different distributions over rewards are shown in Appendix G.

6 Discussion

We have designed efficient, optimal, and surprisingly robust algorithms that split the responsibilities of a multi-armed bandit between two agents—an assistant and human—without suffering much additional regret. The assistant does not need to know any information about how fast the human learns—that is, it does not need to know g. Additionally, our work highlights the potential of max finding algorithms as a useful algorithmic primitive for learning from preference feedback more generally.

A number of open questions remain. One limitation of this work is that it requires the distribution over rewards to be independent and unchanging. As we show in Proposition 2, learning in this setting impossible when rewards are set by an oblivious adversary. Finding a model of co-learning that can be solved under changing reward distributions is still wide open. Another limitation of this work is that it assumes the human will pick a recommendation from the menu of options presented to them. Finally, one open question is to usefully model and solve an assistive dueling bandit where the human can choose not to pick any item in the current batch of recommendations and instead ask for a new batch of items, akin to a user of a social media platform continuing to scroll without clicking on any suggestion.

Acknowledgements

This work was supported in part by the National Science Foundation under grant CCF-2145898, by the Office of Naval Research under grant N00014-24-1-2159, a Google Research Scholar Award, an Alfred P. Sloan fellowship, a Schmidt Science AI2050 fellowship. Mark Bedaywi is supported by a Cooperative AI PhD Fellowship and a Canada Graduate Research Scholarship.

References

Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

Yisong Yue and Thorsten Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1201–1208, 2009.

Yisong Yue, Josef Broder, Robert Kleinberg, and Thorsten Joachims. The k-armed dueling bandits problem. *Journal of Computer and System Sciences*, 78(5):1538–1556, 2012.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Miklós Ajtai, Vitaly Feldman, Avinatan Hassidim, and Jelani Nelson. Sorting and selection with imprecise comparisons. *ACM Transactions on Algorithms (TALG)*, 12(2):1–19, 2015.

Jayadev Acharya, Moein Falahatgar, Ashkan Jafarpour, Alon Orlitsky, and Ananda Theertha Suresh. Maximum selection and sorting with adversarial comparators. *Journal of Machine Learning Research*, 19(59):1–31, 2018.

Sivakanth Gopi, Gautam Kamath, Janardhan Kulkarni, Aleksandar Nikolov, Zhiwei Steven Wu, and Huanyu Zhang. Locally private hypothesis selection. In *Conference on Learning Theory*, pages 1785–1816. PMLR, 2020.

Christopher Trevisan. Sorting and selection in rounds with adversarial comparisons. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1099–1119. SIAM, 2024.

Dylan Hadfield-Menell, Stuart J Russell, Pieter Abbeel, and Anca Dragan. Cooperative inverse reinforcement learning. *Advances in neural information processing systems*, 29, 2016.

Cassidy Laidlaw, Eli Bronstein, Timothy Guo, Dylan Feng, Lukas Berglund, Justin Svegliato, Stuart Russell, and Anca Dragan. Assistancezero: Scalably solving assistance games. In *International Conference on Machine Learning*, pages 32278–32305. PMLR, 2025.

Lawrence Chan, Dylan Hadfield-Menell, Siddhartha Srinivasa, and Anca Dragan. The assistive multi-armed bandit. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 354–363. IEEE, 2019.

Daria Dzyabura and John R Hauser. Recommending products when consumers learn their preference weights. *Marketing Science*, 38(3):417–441, 2019.

Micah Carroll, Davis Foote, Anand Siththaranjan, Stuart Russell, and Anca Dragan. Ai alignment with changing and influenceable reward functions. *arXiv preprint arXiv:2405.17713*, 2024.

Masrour Zoghi, Shimon Whiteson, Remi Munos, and Maarten Rijke. Relative upper confidence bound for the k-armed dueling bandit problem. In *International conference on machine learning*, pages 10–18. PMLR, 2014.

Masrour Zoghi, Zohar S Karnin, Shimon Whiteson, and Maarten De Rijke. Copeland dueling bandits. *Advances in neural information processing systems*, 28, 2015.

Nir Ailon, Zohar Karnin, and Thorsten Joachims. Reducing dueling bandits to cardinal bandits. In *International Conference on Machine Learning*, pages 856–864. PMLR, 2014.

Pratik Gajane, Tanguy Urvoy, and Fabrice Clérot. A relative exponential weighing algorithm for adversarial utility-based dueling bandits. In *International Conference on Machine Learning*, pages 218–227. PMLR, 2015.

Akshay Balsubramani, Zohar Karnin, Robert E Schapire, and Masrour Zoghi. Instance-dependent regret bounds for dueling bandits. In *Conference on Learning Theory*, pages 336–360. PMLR, 2016.

Aadirupa Saha, Tomer Koren, and Yishay Mansour. Adversarial dueling bandits. In *International Conference on Machine Learning*, pages 9235–9244. PMLR, 2021.

Viktor Bengs, Róbert Busa-Fekete, Adil El Mesaoudi-Paul, and Eyke Hüllermeier. Preference-based online learning with dueling bandits: A survey. *Journal of Machine Learning Research*, 22(7): 1–108, 2021.

Hoda Heidari, Michael J Kearns, and Aaron Roth. Tight policy regret bounds for improving and decaying bandits. In *IJCAI*, pages 1562–1570, 2016.

Hyman G Landau. On dominance relations and the structure of animal societies. iii. the condition for a score structure. *Bull. Math. Biophys*, 15(2):143–148, 1953.

Jean-Yves Audibert and Sébastien Bubeck. Minimax policies for adversarial and stochastic bandits. In *COLT*, pages 217–226, 2009.

Lucas Maystre and Matthias Grossglauser. Just sort it! a simple and effective approach to active preference learning. In *International Conference on Machine Learning*, pages 2344–2353. PMLR, 2017.

Huasen Wu and Xin Liu. Double thompson sampling for dueling bandits. *Advances in neural information processing systems*, 29, 2016.

Yisong Yue and Thorsten Joachims. Beat the mean bandit. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 241–248, 2011.

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002a.

Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2002b.

A Lower Bounds

To strengthen the lower bounds in this section, we will show lower bounds not on human regret, but on a weaker form of regret, one that does not penalize the assistant for bad human actions. Namely, when the assistant recommends set $S \subseteq [K]$ of arms, even if the human does not choose to pull the best arm in S, we will only have the assistant suffer regret $\max_{i^* \in [K]} \mu_{i^*} - \max_{i \in S} \mu_i$, as opposed to the usually larger $\max_{i^* \in [K]} \mu_{i^*} - \mu_i$ when $i \in S$ is the arm picked by the human.

Definition 8 (Recommendation Regret)**.** *A sequence of recommendations by the assistant $S_1, \ldots, S_T$ achieves a **recommendation regret** of:*

$$R_{\mathrm{rec}}(T) = T \cdot \max_{i^* \in [K]} \mu_{i^*} - \sum_{t=1}^{T} \max_{i \in S_t} \mu_i.$$

Notice that, because the human on iteration t pulls an arm in the recommended set S_t, recommendation regret can only be lower than human regret. Therefore, any lower bound on recommendation regret must also apply to human regret.

A.1 Matching Lower Bounds on Regret

In this section, we show that, up to logarithmic factors, our algorithm is optimal. Intuitively this must be true for the following reason. The human-AI team together can act as a multi-armed bandit algorithm. Since the human can always achieve $g(T) = \sqrt{T}$ error by running a minimax optimal learning algorithm, if an assistive dueling bandit can do better than $o(Kg(T/K))$, then the human-AI team must achieve a regret of $o(K\sqrt{T/K}) = o(\sqrt{TK})$, which is better than the standard minimax lower bound $\Omega(\sqrt{TK})$ on multi-armed bandits [Audibert and Bubeck, 2009, Lattimore and Szepesvári, 2020].

We make this intuition formal in Proposition 1, proving that this lower bound holds for all possible choices of speeds at which the human learns. This proves that there cannot exist better algorithms for assistive dueling bandits, even if we weaken the evaluation criteria of our algorithms to a weaker form of regret, and even if we give the assistant all the information about how fast the human they are assisting is learning.

Proposition 1. *For any choice of human learning function $g : \mathbb{R} \to \mathbb{R}$ there exists a g-rational human that deterministically makes $g(t)$ error, that when paired with any possible choice of assistive dueling bandit algorithm must suffer a recommendation regret of at least*

$$\Omega(\min(T, Kg(2T/K))),$$

with probability $1/2$.

This lower bound holds even when the assistant is told the function denoting how fast the human learns g, and even when we measure human regret instead.

Proof. Fix an arbitrary choice of human regret rate $g : \mathbb{R} \to \mathbb{R}$, and a number of iterations T.

Consider the following human behavior. For any pairs of arms $\{i, j\} \subseteq [K]$, every time the human is shown $\{i, j\}$ by the assistant, the human will alternate between picking arm i and picking arm j, with some arbitrary fixed starting arm. After being shown the pair of arms n times, once the regret of alternating would exceed $g(n)$, the human will pick the optimal arm. Such a human is g-rational by definition at all steps.

We will show that assisting this a human that uses a burn-in period to play optimally is hard. We continue by Yao's principle. Instead of constructing a hard instance and arguing no stochastic assistant can do well, we instead construct a distribution over bandit problems and argue that recommendation regret is high with probability at least $1/2$. By Yao's minimax principle, this suffices.

Let

$$m = \left\lfloor \frac{4T}{K} \right\rfloor, \qquad \Delta = \min\left(1, \frac{g(m)}{m}\right).$$

For each $i \in [K]$, define the bandit instance B_i to be the one where each arm i has reward Δ deterministically, and every other arm has reward zero. Sample one of these uniformly at random initially.

We will show that with high enough probability, any choice of assistant must suffer $\Omega(Kg(2T/K, \delta))$ recommendation regret on this distribution of bandit problems. Notice that the human has two types of behavior on bandit B_i.

1. When shown arms $\{j, k\} \subseteq [K]$ with $j, k \neq i$, the human will always alternate.
2. When shown arms $\{i, j\} \subseteq [K]$ for some j, the human will alternate for the first m iterations. Indeed, for any $1 \leq n \leq m$, the regret within the pair of any n choices of arms is at most

$$\begin{aligned} n\Delta &\leq \frac{n}{m} g(m) \\ &= g(n). && \text{(By concavity of } g\text{)} \end{aligned}$$

   Therefore, g-rationality is preserved, so the human will continue to alternate for the first m iterations.

Notice that regardless of which bandit instance B_i is chosen, the human will always alternate everywhere, so long as the optimal arm i is shown to it at most m times. Because the assistant can

only see human behavior, while i is shown to the human at most m times in bandit B_i, the assistant can never distinguish between the bandit instances and always receives the same feedback.

The distribution of assistant arm pulls only depends on the human's response. While the human alternates deterministically, the assistant must therefore play the same arms.

Because the assistant can only recommend two arms at a time, at most $2T$ arms are shown to the human. Because there are K arms, notice that by Markov's inequality at least half of the arms are shown at most $4T/K = m$ times in expectation.

Because the bandit instances B_i are chosen uniformly at random, it then follows that with probability at least $1/2$, a bandit instance B_i is sampled where the assistant shows the human arm i at most m times. This means that the maximum reward that can be incurred by the human-AI team is at most

$$m \cdot \Delta$$

whereas in hindsight, had the human-AI team played optimally by always pulling arm i in bandit instance B_i, they would have achieved reward

$$T \cdot \Delta$$

which means a regret of at least

$$(T-m)\Delta = \min\left(T-m,\ T \cdot \frac{g(m)}{m} - g(m)\right) \in \Omega(\min(T, Kg(2T/K))).$$

By Yao's minimax principle, there therefore exists a hard bandit instance where when paired with this human, no assistant can achieve a recommendation regret of better than

$$\Omega(\min(T, Kg(2T/K))). \qquad \square$$

A.2 A Linear Regret Lower Bound in the Oblivious Adversarial Setting

In this section, we show that allowing the reward distribution to change allows the adversary to be too powerful, and can force an assistive dueling bandit algorithm to suffer linear regret. This will hold even when the human is maximally informative.

This means that for interesting algorithms to exist, we must model rewards as being drawn from a distribution fixed at the start of play.

Proposition 2. *Let $K > 2$ and $T > 1$. For every assistive dueling bandit, there exist reward vectors $r_1, \ldots, r_T \in [0,1]^K$ an oblivious adversary can fix before interaction and a deterministic human policy such that the human always pulls the higher reward arm, i.e.*

$$a_t \in \arg\max_{i \in S_t} r_t(i),$$

but the assistant incurs expected recommendation regret

$$\Omega(T).$$

Proof. By Yao's minimax principle, we only need to construct a distribution over reward sequences such that every deterministic assistant suffers $\Omega(T)$ regret.

Fix an arbitrary deterministic assistant, and let $M = \lfloor T/2 \rfloor$. Fix some arbitrarily small $\epsilon > 0$. Sample an arm $i^* \in [K]$ uniformly at random from $\{1,2,3\}$. We will make this the best arm in hindsight. Sample M uniformly random numbers $Z_1, \ldots, Z_M \sim \text{Unif}[\{1,2,3\}]$.

Every Z_i will induce two reward vectors. The first reward vector r_{2i-1} will be $Z_s/3$ for every arm. This will serve to signal to the human what the next best arm is. The second reward vector will depend on whether $Z_i = i^*$. If $Z_i = i^*$, then the reward vector r_{2i} gives arm i^* a reward of 1 and all other arms a reward of zero. If $Z_i \neq i^*$, then the reward vector gives arm Z_i a reward of ϵ and all other arms a reward of 0.

The reward vectors the oblivious adversary fixes is then $r_1, \ldots, r_T$. Consider the human that always picks the smaller arm on odd iterations, and uses the realized reward to find which arm is best in the next iteration. They use this information to pick the arm with higher reward if presented to them on even iteration, and otherwise picks the smaller arm.

Algorithm 2 A Tournament Max-Finding Algorithm

Require: A tournament hypergraph graph G on vertices V.

Initialize $V_1 \leftarrow V$ and $t \leftarrow 1$.
while $|V_t| > 1$ **do**
    Pick a cover of V_t by $\lceil |V_t| / r \rceil$ subsets of V of size r each, $T_1, \ldots, T_{\lceil |V_t|/r \rceil}$ that may intersect. These may include elements outside of V_t.
    Keep only nodes returned by running the query on each T_i,

$$V_{t+1} = \{\mathrm{MaxQuery}_r(T_i) : i \in [\lceil |V_r| / r \rceil]\}.$$

    Increment $t \leftarrow t + 1$.
end while
Return the remaining $i^* \in V_t$.

The human suffers no regret at all. And, their responses never depend on which underlying best arm i^* was sampled. Because i^* is chosen uniformly between three arms, and each recommendation cannot have more than two arms, for any $s \in [M]$,

$$\Pr(i^* \in S_{2s}) \leq \frac{2}{3},$$

where S_{2s} is the set of arms that the assistant recommends at iteration $2s$.

Since $Z_s = i^*$ with probability $1/3$, and the assistant's recommendations are independent of i^*, and because the human's arm pull is deterministic on odd rounds, Z_s must be independent of S_{2s}. Therefore,

$$\Pr(i^* \notin S_{2s} \text{ and } Z_s = i^*) \geq \frac{1}{9}.$$

Even if the assistant always picks the optimal arm on iterations when $Z_s \neq i^*$, this means that regret must be at least $\frac{M}{9} - \epsilon$, when compared with the optimal policy on average. Therefore, the assistant must suffer at least $\frac{M}{9} - \epsilon \in \Omega(T)$ recommendation regret. □

B L-Max Finding Oracles

The goal of this section is to restate and generalize the algorithms in the literature for performing imprecise max-finding. For the sake of showing the assistive dueling bandit setting is just as easy when the assistant must recommend $r \geq 2$ distinct items at a time, we consider the following model.

Throughout this section, let G be an r-ary tournament on a vertex set V: for every subset $S \subseteq V$ with $|S| = r$, the graph specifies a distinguished winner $\mathrm{MaxQuery}_r(S) \in S$. When $\mathrm{MaxQuery}_r(S) = i$, we say that i beats every other vertex in S. Thus each query to G reveals the winner of one r-set and the directed edges from this winner to the other vertices in the set.

A *beat path* from i to j is a sequence $i = v_0, v_1, \ldots, v_\ell = j$ such that v_{a-1} beats v_a for every $a \in [\ell]$. We say that i is an L-max of G when i has a beat path of length at most L to every vertex in V. When $r = 2$, this is exactly the usual notion of an L-max in a tournament graph. The goal in this section is to output vertices with short beat paths to all other vertices while using few queries.

A useful observation we will use throughout this section is that if a queried set S contains a k-max $i \in S$, then the returned winner of S must be a $(k+1)$-max.

Observation 2. *For any natural number k, let $S \subseteq V$ be a set of vertices containing a k-max and satisfying $|S| = r$. Then, $\mathrm{MaxQuery}_r(S)$ is a $(k+1)$-max.*

B.1 A $\mathcal{O}(\log K)$ Tournament Algorithm

Lemma 2 (Max-Finding In The Large r Regime)**.** *For any choice of r and any tournament graph G, Algorithm 2 returns a $\lceil \log_r(n) \rceil$-max in $\mathcal{O}(n/r)$ queries deterministically.*

Proof. The argument for this is simple. Notice that for every iteration t, and every node $i \in V_t$, either $i \in V_{t+1}$, or there exists some node in $j \in V_{t+1}$ such that j beats i. This proceeds for $\lceil \log_r(n) \rceil$ steps before we arrive at a single returned index i^*. We can see therefore that i^* must have a beat path of length $\lceil \log_r(n) \rceil$ to all indices in $V_1 = V$, and therefore is a $\lceil \log_r(n) \rceil$-max. □

Algorithm 3 Brute-Forcing a 5-max with $\text{poly}(n, r)$ generalized queries

1: Let $P \subseteq V$ be an arbitrary subset of $r-2$ vertices.
2: Let $U = V \setminus P$.
3: For every $i, j \in U$ with $i \neq j$, run $\text{MaxQuery}_r(P \cup \{i, j\})$.
4: Let $w \in U$ be the vertex outputted by MaxQuery_r most often.
5: **if** $r = 2$ **then**
6: return w.
7: **else**
8: Let $w' \in U$ be an arbitrary vertex such that $w \neq w'$.
9: Return $\text{MaxQuery}_r(P \cup \{w, w'\})$.
10: **end if**

A tournament is particularly effective in the very large r regime, when we can recommend $r = n^\alpha$ arms a time, for any fixed choice of $\alpha \in [0, 1]$. A $\mathcal{O}(1/\alpha)$-max returned in $\mathcal{O}\left(n^{1-\alpha}\right)$ queries.

B.2 Brute-Forcing in the Large r Regime

To be able to generalize algorithms in the literature with polynomial dependence on r, we first describe a simple generalize a brute-force algorithm that can deterministically find a winner without having to query all the $\mathcal{O}(n^r)$ possible subsets of V by simply appending a fixed padding.

Lemma 3. *For any choice of r, in $\mathcal{O}\left(n^2\right)$ queries, Algorithm 3 returns a 5-max. When $r = 2$, Algorithm 3 returns a 2-max.*

Proof. If P contains a 3-max. Then on line 6, by Observation 2, a 4-max must be returned.

Otherwise, P does not contain a 3-max. Let i^* be the index that beats most other nodes. By Lemma 1, it must be a 2-max. Notice that for all vertices $j \in U$ with $j \neq i^*$, it can never be the case that $\text{MaxQuery}_r(P \cup \{i^*, j\})$ is contained in P. This would imply that some vertex in P beats i^*. Because i^* is a 2-max, this would imply that P contains a 3-max, which is impossible by assumption.

Therefore, for all $j \in U$ with $j \neq i^*$, it must be that $\text{MaxQuery}_r(P \cup \{i^*, j\}) \in \{i^*, j\}$. Note further that none of the vertices in P can beat i^*. This is because if for some $j \in P$ j beat i^*, because i^* is a 2-max, j would be a 3-max, and P would contain a 3-max, contradiction!

Therefore, for any $j \in U$ with $j \neq i^*$, the query $\text{MaxQuery}_r(P \cup \{i^*, j\})$ acts exactly like $\text{MaxQuery}_2(\{i^*, j\})$.

Let w be the vertex that is outputted by $\text{MaxQuery}_r(P \cup \{i, j\})$ over all $i \neq j \in U$. We will show that w must have a path of length at most 2 to i^*. Let $B_w = \{i \in U : w \neq i \text{ and } \text{MaxQuery}_r(P \cup \{i, w\}) = w\}$ and $B_{i^*} = \{i \in U : i^* \neq i \text{ and } \text{MaxQuery}_r(P \cup \{i, i^*\}) = i^*\}$. We know that $|B_w| \geq |B_{i^*}|$.

We can split this into cases:

1. **Case 1:** $\text{MaxQuery}_r(P \cup \{i^*, w\}) = i^*$**.** Then $w \in B_{i^*}$. By definition, $i^* \notin B_w$. But because $|B_w| = |B_{i^*}|$, there must exist some $\ell \in U$ with $\ell \notin \{i^*, w\}$ such that $\ell \in B_w$ but $\ell \notin B_{i^*}$ (as otherwise, B_w would be strictly smaller than B_{i^*}).

   Because $\ell \notin B_{i^*}$, it must be that $\text{MaxQuery}_r(P \cup \{i^*, \ell\}) = \ell$ and so ℓ beats i^*. Because $\ell \in B_j$, it must be that there is an edge from j to ℓ. Composing these paths, j must be a path of length at most 2 from i^*. Because i^* is a 2-max, this implies that j is a 4-max.

2. **Case 2:** $\text{MaxQuery}_r(P \cup \{i^*, w\}) = w$**.** By the above, this means that w beats i^*. Therefore, because i^* is a 2-max, j must be a 3-max and hence also a 4-max.

Therefore, when $\text{MaxQuery}_r(P \cup \{w, w'\})$ is returned on the final line, because w is a 4-max, the node returned must be a 5-max, as claimed.

When $r = 2$, this algorithm simply returns the vertex with the most outgoing edges, which by Lemma 1, is a 2-max. □

In principle, by making all $\binom{n}{r}$ possible queries, one can recover the 2-max by taking the most outputted node. It is unclear how a 2-max can be recovered in time polynomial in r. This is okay

Algorithm 4 Round-robin five-cover

Require: A vertex set V and a block parameter $p \geq r$.
Partition V into parts $P_1, \ldots, P_m$, where all but possibly the last part have size exactly p.
Initialize $C \leftarrow \emptyset$.
for $a = 1, \ldots, m$ **do**
    if $|P_a| < r$ **then**
        Add all vertices in P_a to C.
    else
        Add the output of Algorithm 3 run on P_a to C.
    end if
end for
Return C.

Algorithm 5 r-ary recursive max-finding

Require: A vertex set V, an integer $\ell \geq 5$, and fixed query arity r.
if $\ell \leq 5$ or $|V| \leq 16r^2$ **then**
    Return the output of Algorithm 3 on V.
end if
Let $n = |V|$ and choose a block size $p = \max\{r, \lceil 2n^{\beta_\ell} \rceil\}$.
Let C be the 5-cover returned by Algorithm 4 with block parameter p.
Return the output of r-ary recursive max-finding on C with parameter $\ell - 5$.

for us, as it only worsens our regret by a constant factor. But the problem of finding a 2-max deterministically in $\mathrm{poly}(n, r)$ queries in this generalized setting remains open.

B.3 A $\mathcal{O}(\log \log K)$ Max Finding Algorithm

Now we describe how to generalize the deterministic $\mathcal{O}(\log \log K)$ max-finding algorithm in Ajtai et al. [2015] to the r arm regime, while keeping everything polynomial in r. First, it will be useful to define the notion of an h-cover.

Definition 9 (h-cover)**.** *A set $C \subseteq V$ is an* h-cover *of V if for every $v \in V$ there exists $c \in C$ with a beat path of length at most h from c to v.*

Lemma 4 (Round-robin five-cover)**.** *For every $r \geq 2$, Algorithm 4 returns a 5-cover of V of size*

$$\frac{n}{p} + r$$

using np queries.

Proof. On any block P_a with $|P_a| \geq r$, Algorithm 3 returns a 5-max of the subtournament induced by P_a. Hence the returned vertex has a beat path of length at most 5 to every vertex in P_a. If $|P_a| < r$, every vertex is kept and covers itself. Thus the returned set is a 5-cover.

The algorithm keeps one vertex from each block of size at least r, and at most $r - 1$ vertices from the final undersized block. This gives size $n/p + r$. Each block of size at most s costs s^2 queries by Lemma 3, so the total number of queries used is $\sum_{a=1}^{m} |P_a|^2 \leq p \sum_a |P_a| \leq np$. □

Lemma 5 (r-ary deterministic recursive max-finding)**.** *[Ajtai et al., 2015, Theorem 3.6] Fix $r \geq 2$. Define exponents $a_5 = 2$ and, for $\ell > 5$ a multiple of 5,*

$$a_\ell = \frac{2a_{\ell-5}}{a_{\ell-5} + 1}.$$

If Algorithm 5 chooses

$$\beta_\ell = \frac{a_{\ell-5} - 1}{a_{\ell-5} + 1},$$

then it returns an ℓ-max using $\mathcal{O}\left(r^2 n^{a_\ell}\right)$ generalized queries.

Proof. The proof is the exactly the same recursive cover argument as Ajtai et al. [2015, Theorem 3.6].

Let $Q_\ell(n)$ denote the number of queries that are used. We will prove that

$$Q_\ell(n) \le 16r^2 n^{a_\ell}.$$

The base case $\ell = 5$ is Lemma 3, using $n^2 = n^{a_5}$ queries. If instead $|V| \le 16r^2$, the same brute-force call returns a 5-max, and hence an ℓ-max, using at most $\mathcal{O}\left(r^2 n^{a_\ell}\right)$ queries since $a_\ell \ge 1$.

Otherwise, we know that $n > 16r^2$ and $\beta_\ell \le 1/3$. In particular, this implies that $r < \frac{1}{4}n^{1/2} = \frac{1}{4}n^{1-\beta_t} \cdot n^{\beta_t - 1/2} \le \frac{1}{4}n^{1-\beta_t} \cdot n^{-1/6} \le \frac{1}{8}n^{1-\beta_\ell}$.

Therefore, in the recursive step, we call the max-finding algorithm on C of size

$$|C| \le \frac{n}{p} + r \le \frac{1}{2}n^{1-\beta_\ell} + r \le \frac{5}{8}n^{1-\beta_\ell}.$$

Because $p = \max(r, \lceil 2n^{\beta_\ell} \rceil) \le 2rn^{\beta_\ell} \le r^2 n^{\beta_\ell}$, we know that constructing the cover costs

$$np \le r^2 n^{1+\beta_\ell} = r^2 n^{a_\ell}$$

queries.

By definition, $(1-\beta_\ell)a_{\ell-5} = a_\ell$. Therefore, by the induction hypothesis,

$$\begin{aligned} Q_\ell(n) &\le np + Q_{\ell-5}(|C|) \\ &\le r^2 n^{a_\ell} + 16r^2 \left(\frac{5}{8}n^{1-\beta_\ell}\right)^{a_{\ell-5}} \\ &\le \left(1 + 16 \cdot \frac{5}{8}\right) r^2 n^{a_\ell} \\ &\le 16r^2 n^{a_\ell}. \end{aligned}$$

□

Corollary 2. *[Ajtai et al., 2015, Corollary 3.7] For every $r \ge 2$, there is a deterministic generalized-query algorithm that returns a $\mathcal{O}(\log\log n)$-max using $\mathcal{O}\left(r^2 n\right)$ queries.*

Proof. Let $b_\ell = a_\ell - 1$. By Lemma 5,

$$b_\ell = \frac{b_{\ell-5}}{2 + b_{\ell-5}} \le \frac{b_{\ell-5}}{2}.$$

So after $\mathcal{O}(\log\log n)$ recursive levels, $b_\ell \le 1/\log n$, and the query bound becomes $\mathcal{O}\left(r^2 n^{1+1/\log n}\right) = \mathcal{O}\left(r^2 n\right)$. □

C Proof of the Main Result

It will be useful to have the following lemma:

Lemma 6. *For any function g satisfying the learning-human assumption, g is subadditive.*

Proof. Notice that because g is concave and $g(0, \delta_0) = 0$, for any $x, y \in \mathbb{R}$,

$$g(x, \delta_0) \ge \frac{x}{x+y} g(x+y, \delta_0) + \frac{y}{x+y} g(0, \delta_0) = \frac{x}{x+y} g(x+y, \delta_0),$$

and

$$g(y, \delta_0) \ge \frac{y}{x+y} g(x+y, \delta_0) + \frac{x}{x+y} g(0, \delta_0) = \frac{y}{x+y} g(x+y, \delta_0).$$

Summing these up,

$$g(x, \delta_0) + g(y, \delta_0) \ge g(x+y, \delta_0),$$

and so g is subadditive. □

Theorem 4.1 (Reduction to Imprecise Max Finding)**.** *Let $\mathcal{A}$ be an L-max oracle with query complexity Q and let $\delta > 0$ be arbitrary. Running Algorithm 1 with $\mathcal{A}$ with a g-rational human, for any choice of g, achieves a human regret of*

$$\mathcal{O}\left(Q + L(K+Q)\log({}^{T}/_{(Q+K)}) \cdot g\left(\frac{T}{(Q+K)\log({}^{T}/_{(Q+K)})}, \frac{\delta}{T\binom{K}{2}}\right)\right).$$

with probability at least $1-\delta$. When $g(t,\delta) \in \mathcal{O}\left(t^{\alpha} \cdot f(\delta)\right)$ for some constant $\alpha > 0$ and function $f : [0,1] \to [0,\infty)$, this can be improved to

$$\mathcal{O}\left(Q + Lf\left(\frac{\delta}{T\binom{K}{2}}\right) \cdot (K+Q)^{1-\alpha} \cdot T^{\alpha}\right).$$

Proof. Let

$$\delta_0 = \frac{\delta}{T\binom{K}{2}}.$$

We will prove the regret bound on a good event $\mathcal{E}$ on which comparisons are reliable. For a pair of arms $i, j \in [K]$, define

$$\Gamma_{i,j}(t) = \sum_{u \le t : S_u = \{i,j\}} \max_{i^* \in \{i,j\}} \mu_{i^*} - \mu_{i_u}$$

as being the regret incurred on pair $\{i,j\}$ up until time t. Define the event $\mathcal{E}$ to be wherefor all pairs of arms i, j, and all time steps t,

$$\Gamma_{i,j}(t) \le g(N_t(i,j), \delta_0).$$

By g-rationality, the probability that this inequality fails for a fixed pair of arms i, j and a fixed time step t is δ_0. Therefore, by the union bound,

$$\Pr(\mathcal{E}) \ge 1 - \binom{K}{2} T\delta_0 = 1 - \delta.$$

So from here on, we assume that $\mathcal{E}$ holds.

We now show the following: under $\mathcal{E}$, if after being compared at least m times, arm i is chosen more often than arm j, then $\mu_j - \mu_i \le \frac{g(m,\delta_0)}{m}$.

Suppose that a pair of arms $\{i,j\}$ is compared a batch of m times, where the final comparison is on time t. Then $N_t(i,j) \ge m$. Suppose that i was chosen at least as many times as j, i.e. $N_t(i \succ j) \ge N_t(j \succ i)$. Then, i was chosen at least $N_t(i,j)/2$ times, and so regret is upper bounded by

$$\frac{N_t(i,j)}{2} \cdot (\mu_j - \mu_i) \le \Gamma_{i,j}(t).$$

Because we are assuming $\mathcal{E}$ is true, this means

$$\frac{N_t(i,j)}{2} \cdot (\mu_j - \mu_i) \le g(N_t(i,j), \delta_0).$$

Therefore,

$$\mu_j - \mu_i \le \frac{2g(N_t(i,j), \delta_0)}{N_t(i,j)} \le \frac{2g(m, \delta_0)}{m},$$

where we use that $g(n, \delta_0)/n$ is non-increasing. Therefore, every batch of comparisons is an imprecise comparison with precision $2g(m, \delta_0)/m$.

To prove this result, we first prove that the winner of each phase must have high enough reward.

Claim 1. *Let i_s denote the winner of phase s.*

$$\Delta_{i_s} \le 2(L+1)\frac{g(2^s, \delta_0)}{2^s}.$$

Proof. First, recall that the winner of phase s is an L-max of the subset of arms W_s of strength 2^s. Therefore, for any $j \in W_s$,

$$\mu_j - \mu_{i_s} \leq 2L\frac{g(2^s, \delta_0)}{2^s}.$$

It remains to show this result for $j \notin W_s$. In phase 0, $W_0 = [K]$ is the full set of arms, so the inequality must hold for $j \notin W_0$ too.

In phases $s > 0$, recall that W_s is constructed by comparing all arms with i_{s-1} 2^s times and keeping any arm j that wins more often than not. Therefore, if $j \notin W_s$, there must be a edge from i_{s-1} to j of strength 2^s. Because $i_{s-1} \in W_s$ — i.e. we always place the winner of the previous phase in the next winning set — there must be a beat path of length L from i_s to i_{s-1} of length L of strength 2^s.

By combining these two paths — i_s to i_{s-1} of length L and i_{s-1} to j of length 1 — we get a beat path of length $L + 1$ of strength 2^s from i_s to any $j \notin W_s$. This means that for any $j \notin W_s$,

$$\mu_j - \mu_{i_s} \leq 2(L+1)\frac{g(2^s, \delta_0)}{2^s}.$$

When this holds for all arms $j \in [K]$, this holds for the arm with the highest mean i^*. So,

$$\Delta_{i_s} \leq 2(L+1)\frac{g(2^s, \delta_0)}{2^s}.$$ □

Notice that we can split up the regret at any time step t, $\max_{i^* \in [K]} \mu_{i^*} - \mu_{i_t}$ into two parts

$$\max_{i^* \in [K]} \mu_{i^*} - \mu_{i_t} = \left(\max_{i^* \in [K]} \mu_{i^*} - \max_{i \in S_t} \mu_i\right) + \left(\max_{i \in S_t} \mu_i - \mu_{i_t}\right),$$

where the first part is the regret due to our recommendation error and the second part is the regret due to the user's mistakes when given a pair.

First, we bound the second summand and show that the human cannot have picked the worse of the two arms recommended to them too often. Let $\mathcal{B}$ be the collection of batches of comparisons we show the user throughout the execution of the algorithm. For any $b \in \mathcal{B}$, let m_b be the number of times the human was shown the comparison for that batch. Because we assume $\mathcal{E}$,

$$\begin{aligned}
\sum_{t=1}^{T}\left(\max_{i \in S_t} \mu_i - \mu_{i_t}\right) &= \sum_{i<j} \Gamma_{i,j}(T) \\
&\leq \sum_{i<j} g(N_T(i,j), \delta_0) \\
&\leq \sum_{b \in \mathcal{B}} g(m_b, \delta_0)
\end{aligned}$$

where the last inequality follows by subadditivity.

Now we bound the first summand and show that the algorithm cannot have recommended too many bad pairs of arms. For each batch b during line 11 of the algorithm, comparing a winner i_s on phase s with another arm, we bound the first component of regret like so:

$$\begin{aligned}
\sum_{t \in b}\left(\max_{i^* \in [K]} \mu_{i^*} - \max_{i \in S_t} \mu_i\right) &\leq m_b \Delta_{i_s} && \text{(Because every comparison contains the winner } i_s\text{)} \\
&\leq 4(L+1)\frac{m_b}{2^{s+1}} g(2^{s+1}, \delta_0) && \text{(By Claim 1)} \\
&\leq 4(L+1) g(m_b, \delta_0) && \text{(Because } m_b \leq 2^{s+1}\text{, and the concavity of } g\text{)}
\end{aligned}$$

Now we bound the regret from calling the oracle. Because rewards are bounded between 0 and 1, every comparison incurs at most a regret of 1. Because the initial phase $s = 0$ performs at most Q comparisons, this means it suffers a regret of at most Q at the start.

On phases $s > 0$, we know that the winner of the previous phase i_{s-1}, by Claim 1,

$$\Delta_{i_{s-1}} \leq 2(L+1)\frac{g(2^{s-1}, \delta_0)}{2^{s-1}}.$$

By construction, every arm $j \in W_s$ such that $j \neq i_{s-1}$ must have beat i_{s-1} more often than not in the full history of comparisons at phase $s-1$. Therefore, by the $\mathcal{E}$,

$$\mu_j \geq \mu_{i_{s-1}} - 2\frac{g(2^s, \delta_0)}{2^s}.$$

Combining the two inequalities, and using that $g(t, \delta_0)/t$ is non-increasing, this means that for all $j \in W_s$ we have

$$\Delta_j \leq 4(L+2)\frac{g(2^s, \delta_0)}{2^s}.$$

For every batch b performed in line 6 of phase s, doing an oracle call, because oracle calls can only make comparisons in W_s,

$$\sum_{t \in b}\left(\max_{i^* \in [K]} \mu_{i^*} - \max_{i \in S_t} \mu_i\right) \leq m_b \max_{j \in W_s} \Delta_j = 4(L+2)\frac{m_b g(2^s, \delta_0)}{2^s}.$$

By concavity, because $m_b \leq 2^s$ for batches b in phase s, it follows that

$$\sum_{t \in b}\left(\max_{i^* \in [K]} \mu_{i^*} - \max_{i \in S_t} \mu_i\right) \leq 4(L+2)g(m_b, \delta_0).$$

Summing all these up, we finally get that the user's regret is

$$R_{\text{user}} \leq Q + (4L+9)\sum_{b \in \mathcal{B}} g(m_b, \delta_0).$$

Notice that each phase uses at least $B = Q + K$ batches, and there are $\log(T/B)$ phases, so $|\mathcal{B}| \leq B\log(T/B)$. By Jensen's inequality,

$$\begin{aligned} R_{\text{user}} &\leq Q + (4L+9)\sum_{b \in \mathcal{B}} g(m_b, \delta_0) \\ &\leq Q + (4L+9)\,|\mathcal{B}| \cdot g(T/\,|\mathcal{B}|\,, \delta_0) \\ &= Q + (4L+9)B\log(T/B)g\left(\frac{T}{B\log(T/B)}, \delta_0\right). \end{aligned}$$

Now suppose $g(n, \eta) \leq n^\alpha f(\eta)$ for $0 < \alpha \leq 1$. Let $x = T/B$. In phases s such that $2^s \leq x$, there are at most B batches per phase, each of length at most 2^{s+1}. Every batch in a later phase has $m_b > x$, so $m_b^\alpha \leq m_b x^{\alpha-1}$. Hence

$$\begin{aligned} \sum_{b \in \mathcal{B}} g(m_b, \delta_0) &\leq f(\delta)\sum_{b \in \mathcal{B}} m_b^\alpha \\ &\leq f(\delta_0) \cdot \left(B \sum_{s \geq 0:\ 2^s \leq x} 2^{\alpha(s+1)} + Tx^{\alpha-1}\right) \\ &= \mathcal{O}\left(f(\delta_0)B^{1-\alpha}T^\alpha\right). \end{aligned}$$

Substituting this above, we get

$$R_{\text{user}}(T) \leq Q + \mathcal{O}\big(Lf(\delta_0)B^{1-\alpha}T^\alpha\big). \qquad \square$$

D A $\tilde{\Theta}\left(T^{2/3}\right)$ Bound on Strong Regret

Another definition of regret we could have considered was, instead of looking at the regret of the arm that the human picked, considering the sum of the regrets of the two arms recommended by the assistant.

Definition 10 (Strong Regret)**.** *A sequence of recommendations by the assistant $S_1, \ldots, S_T$ achieves a **strong regret** of:*

$$R_{\text{strong}}(T) = 2T \cdot \max_{i^* \in [K]} \mu_{i^*} - \sum_{t=1}^{T}\sum_{i \in S_t} \mu_i.$$

We will show that this formulation of regret is fundamentally much harder than human regret, and, when $g(T) = \sqrt{T}$ the best possible strong regret any assistant can hope to achieve is $\tilde{\Theta}(T^{2/3})$.

For simplicity in this section, we allow the assistant to recommend the same arm twice. The results hold in the case where recommendations must be distinct too, where regret is measured in comparison to the sum of the top two arms $\max_{i_1^* \neq i_2^* \in [K]} \mu_{i_1^*} + \mu_{i_2^*}$.

D.1 An $\Omega(T^{2/3})$ Lower Bound on Strong Regret

In this section, we show the lower bound on strong regret in terms of T for two armed bandits.

Theorem 1 (A Strong Regret Lower Bound)**.** *For any choice of human regret rate g, there exists a bandit problem such that for any possible choice of assistant, there is some g-rational human that forces the human-AI team to suffer*

$$\Omega\left(\min_{1 \leq m \leq T} \max\left\{m, \frac{Tg(m)}{m}\right\}\right)$$

expected strong regret. In particular, when $g(t) = t^\alpha$, this is a lower bound of $\Omega\left(T^{1/(2-\alpha)}\right)$.

Proof. Fix an arbitrary choice of human regret rate g, and some $m \in [T]$. Let $\epsilon = \min\left\{1, \frac{g(m)}{m}\right\}$. Consider the bandit instance with two arms i_1, i_2 that, with probability $1/2$, is the instance

$$\mu_{i_1} = 1, \qquad \mu_{i_2} = 0,$$

and otherwise is the instance

$$\mu_{i_1} = 1 - \epsilon, \qquad \mu_{i_2} = 1.$$

Consider the human that always picks arm 1 on bandit instance 1, and on bandit instance 2 picks arm 1 for the first m steps then always picks arm 2 forever after. Such a human incurs 0 regret in the first bandit instance, and incurs regret

$$\epsilon \cdot m \leq g(m),$$

in the second bandit instance. Therefore, this human is g-rational with probability 1.

We will show there is no way of assisting this g-rational human without incurring either $\Omega\left(\frac{Tg(m)}{m}\right)$ or $\Omega(m)$ strong regret. The assistant has three possible actions: 1) show the human (i_1, i_1), 2) show the human (i_1, i_2), and 3) show the human (i_2, i_2).

If the assistant has shown the human (i_1, i_2) less than m times, then it cannot distinguish between the bandit instances at all, and can be in either with probability 1/2. Human behavior is the only input to the assistant, and the human behavior is exactly the same until (i_1, i_2) is recommended m times. This means we can divide any run with an assistant into two cases:

1. **Case 1: The assistant recommends the pair (i_1, i_2) at least m times in this run.** Half the time that the assistant does this, the bandit must be in world one. In this case, the joint reward of the arms is $m(1 + 0)$, whereas the optimal joint reward would have been to recommend (i_1, i_1) in those m steps and achieving a reward of $m(1 + 1) = 2m$. This means a strong regret of at least

   $$\frac{1}{2}(2m - m) \in \Omega(m).$$

2. **Case 2: The assistant recommends the pair (i_1, i_2) at most n times for in this run, for $n \leq m$.** Then human behavior is exactly the same throughout the run regardless of whether or not we are bandit instance 1 or 2.

   By the same argument as in case 1, on the n iterations where the assistant recommends (i_1, i_2), the strong regret is at least $n/2$.

   Suppose that the assistant plays (i_2, i_2) for at least half of the remaining iterations. Then, half the time it is in bandit instance 1, and so must suffer at least $(T - n)/2 = (T - n)/2$ regret on those iterations, meaning a total strong regret of at least $(T - n)/2 + n/2 = T/2$.

Suppose that, instead, the assistant plays (i_1, i_1) for at least half of the remaining iterations. Then, half the time it is in bandit instance 2, and so must suffer at least $\epsilon(T-n)/2$ strong regret. Summing this up, this means a regret of at least

$$\begin{aligned}\epsilon(T-n)/2 + n/2 &= \epsilon \cdot T/2 + n(1-\epsilon)/2 \\ &\geq \epsilon \cdot T/2 \\ &\geq T/2 \cdot \min\left\{1, \frac{g(m)}{m}\right\}.\end{aligned}$$

Therefore, regardless of whether the assistant chooses to play (i_1, i_1) or (i_2, i_2) more often when it has not yet fully determined which instance it is in, it must suffer a strong regret of at least $\frac{T}{2}\min(1, g(m)/m)$.

Since we must be in at least one of these cases, the strong regret suffered must be at least

$$\Omega\left(\min\left\{m, T, \frac{Tg(m)}{m}\right\}\right) = \Omega\left(\min\left\{m, \frac{Tg(m)}{m}\right\}\right).$$

Because we can pick m at the start of the proof to be anything and run this argument, this means strong regret is at least

$$\Omega\left(\max_{1\leq m\leq T} \min\left\{m, \frac{Tg(m)}{m}\right\}\right).$$

Notice that because m is increasing and $\frac{Tg(m)}{m}$ is decreasing (by the concavity of g), it follows that when the min and max are swapped, the solution is the same: regret is equally at least

$$\Omega\left(\min_{1\leq m\leq T} \max\left\{m, \frac{Tg(m)}{m}\right\}\right).$$

When $g(t) = t^\alpha$, because this minimum is reached at equality, the optimum is at $m = Tm^{\alpha-1}$, or equivalently when $m = T^{1/(2-\alpha)}$ □

This in particular implies that, when $g(T) = \sqrt{T}$, the human-AI team must suffer $\Omega(T^{2/3})$ strong regret.

D.2 Explore-Then-Commit Achieves a $\tilde{\mathcal{O}}\left(T^{2/3}\right)$ Upper Bound on Strong Regret

In terms of its dependence on the number of iterations, we can prove that this lower bound on strong regret is tight. The algorithm we run here is simple. Given an L-max oracle $\mathcal{A}$, we can choose to run it with some level of precision once on the arms in the bandit. Then, we simply commit to the winner.

We show that algorithm that does this will achieve the same asymptotic dependence on T in the strong regret.

Proposition 3 (Strong Regret Upper Bound)**.** *Let $\mathcal{A}$ be an L-max finding algorithm with query complexity Q. Let $g : \mathbb{R} \times [0,1] \to \mathbb{R}$ be an arbitrary function.*

For any choice of m, there exists an assistant that achieves a strong regret rate of

$$\mathcal{O}\left(NQ + LT\frac{g(N, \delta/\binom{K}{2})}{N}\right)$$

with probability at least $1-\delta$ when paired with any g-rational human.

In particular, when $g(t,\delta) = t^\alpha f(\delta)$ for some $\alpha \in [0,1]$ and function $f(\delta)$, it follows that strong regret is at most

$$\mathcal{O}\left(Q^{\frac{1-\alpha}{2-\alpha}} \cdot \left(f\left(\delta/\binom{K}{2}\right) LT\right)^{\frac{1}{2-\alpha}}\right).$$

Proof. The algorithm is simple. We run $\mathcal{A}$ on the set of arms $[K]$ and repeat every query N times, for some choice of N we will pick later. That is, when $\mathcal{A}$ queries arms (i,j), we show the human the (i,j) pair N times, and return $i \to j$ if i is chosen at least as much as j, and $j \to i$ otherwise.

Define the good event to be that every query is performed correctly, in that whenever $|\mu_i - \mu_j| \geq 2\frac{g(N,\delta/\binom{K}{2})}{N}$, i is chosen more often than j. By Observation 1, this happens with probability at least $1 - \delta/\binom{K}{2}$, so this happens for all possible pairs we could ask with probability at least $1 - \delta$.

Suppose we are in this good world. After QN queries, an L-max $\hat{i}$ is returned. The regret of $\hat{i}$ is at most

$$\Delta_{\hat{i}} \leq 2L \cdot \frac{g(N, \delta/\binom{K}{2})}{N}.$$

In the exploit phase after $\mathcal{A}$ has finished making queries, the assistant recommends $(\hat{i}, \hat{i})$ repeatedly until termination. This suffers a regret of at most $4LT \cdot \frac{g(N,\delta/\binom{K}{2})}{N}$. with probability at least $1 - \delta$.

Because rewards are bounded between 0 and 1, the max regret incurred by making the QN queries in the explore phase is at most $2QN$. Because N can be chosen arbitrarily, this means total regret suffered in the explore and exploit phases together is at most

$$\mathcal{O}\left(\min_{1 \leq N \leq T} QN + LT\frac{g(N, \delta/\binom{K}{2})}{N}\right). \qquad \square$$

Corollary 3 (Strong Regret Upper Bound)**.** *Let $g : \mathbb{R} \to \mathbb{R}$ be an arbitrary function. There exists an assistant that achieves a strong regret rate of*

$$\mathcal{O}\left(\min_{1 \leq N \leq T} NK + \log\log(K) \cdot T\frac{g(N, \delta/\binom{K}{2})}{N}\right)$$

when paired with any g-rational human with probability at least $1 - \delta$. When, for some $\alpha > 0$, we can write $g(t, \delta) \in \mathcal{O}\left(t^\alpha f(\delta)\right)$ for some function $f : [0, 1] \to [0, \infty)$, the assistant achieves a strong regret rate of

$$\mathcal{O}\left(K^{\frac{1-\alpha}{2-\alpha}} \cdot \left(f(\delta/\binom{K}{2}) \log\log(K)T\right)^{\frac{1}{2-\alpha}}\right).$$

Proof. Combine Proposition 3 with the max finding oracle in Appendix B. $\square$

When this is $g(t, \delta) = \sqrt{t \log(1/\delta)}$, we get an upper bound on strong regret of

$$\mathcal{O}\left(K^{1/3}T^{2/3} \cdot \left(\log(K/\delta) \log\log K\right)^{2/3}\right).$$

E Simultaneously Solving the Special Case of Dueling Bandits

The algorithm we propose for solving bandits with imprecise max-finding is much more general than it seems. In this section, we show that a large class of dueling bandit problems, namely minimizing weak regret on dueling bandits can simultaneously be solved by Algorithm 1 with no modification at all.

We consider the standard formulation with underlying utilities μ_i for each arm, where arm i is chosen over arm j with probability $\nu(\mu_i - \mu_j)$, for some fixed so-called 'link-function' ν.

Definition 11 (Dueling Bandits With ν-Link Functions)**.** *Let $\nu : [-1, 1] \to [0, 1]$ be an arbitrary non-decreasing function defined such that for all $x \in \mathbb{R}$, $\nu(x) + \nu(-x) = 1$.*

A dueling bandit with a ν-link function is defined by a set of arms $[K]$ and a set of rewards $\mu_i \in [0, 1]$ for each arm $i \in [K]$. A learner interacts with the dueling bandit for T iterations. At each iteration, the learner plays a pair of arms $\{i, j\}$, and receives that i beats j with probability $\nu(\mu_i - \mu_j)$ and that j beats i otherwise. Each comparison uses fresh randomness.

Because $\nu(x) + \nu(-x) = 1$ for all $x \in [-1, 1]$, it follows that $\nu(0) = 1/2$. A quantity we will care about is the slope of the best linear lower bound on ν, what we will call S_ν,

$$S_\nu := \inf_{x \in (0,1]} \frac{\nu(x) - \nu(0)}{x} = \inf_{x \in (0,1]} \frac{\nu(x) - 1/2}{x}.$$

Notice that when ν is concave on $(0, 1]$, it follows that $S_\nu = \nu(1) - 1/2$.

We define regret as the sum of the minimum regret of any arm in every batch. This corresponds to the notion of weak regret in the dueling bandit literature [Bengs et al., 2021].

Definition 12 (Weak Regret in Dueling Bandits with ν-Link Functions)**.** *Suppose in a dueling bandit with K arms and arm rewards $\mu_1, \ldots, \mu_K$, pairs of arms $S_1, \ldots, S_T$ are chosen over the T iterations. The weak regret of this choice of arms is*

$$R_{\text{weak}} = T \cdot \max_{i^* \in [K]} \mu_{i^*} - \sum_{t=1}^{T} \max_{i \in S_t} \mu_i.$$

Two standard choices of the link function are, for $\beta > 0$, the sigmoid link function $\nu_\beta(x) = 1/(1 + \exp(-\beta x))$ studied by work like Maystre and Grossglauser [2017] and is the standard modeling assumption for human feedback when performing RLHF [Christiano et al., 2017], and, for $\alpha > 0$, the linear link function $\nu_\alpha(x) = \max(\min(1/2 + \alpha x, 1), 0)$ used by work like Ailon et al. [2014].

By a simple application of the Chernoff bound, we can show that for both link functions, the key fact Observation 1 holds too.

Proposition 4 (Adaptation of Observation 1 to Dueling Bandits With ν-Link Functions)**.** *Let $\delta > 0$ be arbitrary, and for some function ν, consider an arbitrary dueling bandit with link function ν. For any two arms i, j with $\mu_i > \mu_j$, define $b_{i,j}$ to be*

$$b_{i,j} = \nu(\mu_i - \mu_j) - 1/2$$

times. Then, with probability at least $1 - \delta$, for all $t \in [T]$, if a pair was chosen enough times,

$$N_t(i, j) \geq \frac{\log(1/\delta)}{b_{i,j}^2},$$

then i wins a comparison more times than j: $N_t(i \succ j) \geq N_t(j \succ i)$.

In particular, with probability $1 - \delta$, if $S_\nu > 0$, for all $t \in [T]$, if

$$N_t(i, j) \geq \frac{\log(1/\delta)}{S_\nu^2(\mu_i - \mu_j)^2},$$

then $N_t(i \succ j) \geq N_t(j \succ i)$.

Proof. Fix the pair of arms i, j and let W_n be the number of times that arm i is chosen by a comparison after n comparisons of the pair (i, j). For every integer $m \geq 1$, we will show that

$$\Pr\left(\exists n \geq m : W_n \leq \frac{n}{2}\right) \leq \exp(-2m \cdot b_{i,j}^2).$$

Let $I_t^{i \succ j}$ be the indicator random variable denoting whether i beats j on the tth comparison between i and j. By Hoeffding's, for any $\lambda > 0$,

$$\mathbb{E}[\exp(-\lambda(I_t^{i \succ j} - \nu(\mu_i - \mu_j))] \leq \exp(\lambda^2/8).$$

Because each comparison uses fresh randomness, we can multiply these independent events and get that

$$\mathbb{E}[\exp(-\lambda(W_n - n\nu(\mu_i - \mu_j))] \leq \exp(n\lambda^2/8).$$

Thus,

$$M_n = \exp\left(-\lambda \cdot (W_n - n\nu(\mu_i - \mu_j)) - \frac{n\lambda^2}{8}\right)$$

is a supermartingale. Recall that Ville's inequality then states that for all $a > 0$,

$$\Pr\left(\sup_{n \geq 0} M_n \geq a\right) \leq \frac{1}{a}.$$

Setting $\lambda = 4b_{i,j}$ and $a = \exp(2mb_{i,j}^2)$ implies the claimed inequality.

Therefore, for any choice of $\delta > 0$ and any choice of m,

$$\Pr\left(N_T(i, j) \geq m \text{ and } W_{N_T(i,j)} \leq \frac{N_T(i, j)}{2}\right) \leq \frac{1}{a} = \exp(-2mb_{i,j}^2)$$

And when we take

$$m = \frac{\log(1/\delta)}{b_{i,j}^2},$$

we are done.

To see the second part of this observation, notice that by the definition of S_ν,

$$\nu(x) - 1/2 \geq x \cdot S_\nu.$$

This means we can upper bound

$$\frac{\log(1/\delta)}{(\nu(\mu_i - \mu_j) - 1/2)^2} \leq \frac{\log(1/\delta)}{S_\nu^2 \cdot (\mu_i - \mu_j)^2},$$

so when

$$N_T(i,j) \geq \frac{\log(1/\delta)}{S_\nu^2 \cdot (\mu_i - \mu_j)^2},$$

the same argument as above applies, proving the second claim. □

With this result in hand, we can derive the adaptation of the key observation that allows for our L-max approach to work for a wide range of link functions for dueling bandits.

Observation 3 (Adaptation of Observation 1 to Dueling Bandits With Linear Link Functions)**.** *Let $\delta, \alpha > 0$ be arbitrary. With probability $1 - \delta$, for all $t \in [T]$, if*

$$N_t(i,j) \geq \frac{\log(1/\delta)}{\min(1/2, \alpha)^2 \cdot (\mu_i - \mu_j)^2},$$

then

$$N_t(i \succ j) \geq N_t(j \succ i).$$

Proof. This follows by Proposition 4, because $\nu_\alpha(x)$ is concave for $x \geq 0$ and $\nu_\alpha(1) = \min(1, \alpha + 1/2)$. So $S_\nu = \min(1, \alpha + 1/2) - 1/2 = \min(1/2, \alpha)$. □

Observation 4 (Adaptation of Observation 1 to Dueling Bandits With Sigmoid Link Functions)**.** *Let $\delta, \beta > 0$ be arbitrary. Fix a pair of arms i, j with $\mu_i > \mu_j$. With probability $1 - \delta$, for all $t \in [T]$, if*

$$N_t(i,j) \geq \frac{\log(1/\delta)}{((1 + \exp(-\beta))^{-1} - 1/2)^2 \cdot (\mu_i - \mu_j)^2} \in \Omega\left(\frac{\log(1/\delta)}{(\mu_i - \mu_j)^2}\right)$$

times in a dueling bandit with sigmoid link function $\nu_\beta(x) = 1/(1 + \exp(-\beta x))$. Then, $N_t(i \succ j) \geq N_t(j \succ i)$.

Proof. This follows by Proposition 4, because $\nu_\beta(x)$ is concave for $x \geq 0$ and $\nu_\beta(1) = 1/(1 + \exp(-\beta))$. So $S_\nu = 1/(1 + \exp(-\beta)) - 1/2$. □

$\left(\frac{1}{S_\nu}\sqrt{T\log(1/\delta)}\right)$-rational humans act like dueling bandits in that it similarly takes $\frac{\log(1/\delta)}{S_\nu^2 \Delta^2}$ pulls for a pair of arms separated by Δ to be correctly oriented. This immediately implies that the algorithm works for minimizing dueling bandit regret with the same rate of regret as a $\left(\frac{1}{S_\nu}\sqrt{T\log(1/\delta)}\right)$-rational human.

Corollary 4 (Algorithm 1 is One-Size Fits All)**.** *Algorithm 1 simultaneously achieves*

$$\tilde{\mathcal{O}}(S_\nu^{-1} \cdot \sqrt{TK})$$

weak regret on all dueling bandits with any link function ν.

Algorithm 6 A g-Agnostic R-armed Assistant via Imprecise Max-Finding

Require: Max-finding algorithm $\mathcal{A}$ with R vertices at a time that finds an L-max in a graph using Q queries.

Initialize winning set $W_0 \leftarrow [K]$, and phase $s \leftarrow 0$.
while number of iterations is at most T **do**

Step 1: Run the Oracle With Precision 2^s
  If W_s is of size less than R, pad it with $R - |W_s|$ arbitrary elements from $[K] \setminus W_s$.
  Initialize $\mathcal{A}$ on the subgraph induced by W_s.
  while $\mathcal{A}$ has not terminated **do**
    Get query subset $S \subseteq [K]$ of size R from $\mathcal{A}$.
    Present S to human 2^s times.
    Return a vertex i to $\mathcal{A}$ such that i was pulled the most when presented S so far.
  end while
  Let i_s be the candidate returned by $\mathcal{A}$.

Step 2: Filter the Set of Arms
  Cover $[K] \setminus \{i_s\}$ by groups of size $R-1$, $S_1, \ldots, S_{\lceil (K-1)/(R-1) \rceil}$, that potentially intersect.
  Let $W_{s+1} \leftarrow \{i_s\}$.
  for each ℓ from 1 to $\lceil (K-1)/(R-1) \rceil$ **do**
    Present $\{i_s\} \cup S_\ell$ to human for 2^{s+1} rounds.
    Let $j \in S_\ell \cup \{i_s\}$ be the node that is chosen by the human most when presented $S_\ell \cup \{i_s\}$ so far.
    if $j \neq i_s$ **then**
      $W_{s+1} \leftarrow W_{s+1} \cup \{j\}$.
    end if
  end for
  Increment $s \leftarrow s + 1$.
end while

F R Arms Recommended Per Iteration

Another model we could have studied is to give the assistant more power, and allow it to recommend R arms at a time to the human. In this case, there are many possible ways one can extend the definition of g-rationality to R arm subgames. We consider the weakest definition, one that does not assume that the human is able to transfer knowledge gained from one subgame to another, and show that the natural generalization of our reduction still applies.

Definition 13 (g-rationality with R arms at a time)**.** *For a set of arms S of size R, let*

$$I_S^T = \{t \leq T : S_t = S\}, \qquad N_T(S) = |I_S^T|.$$

*A human policy is $g(\cdot,\cdot)$-**rational** if for every horizon T, every subset of arms S of size R, and every $\delta \in (0,1)$, with probability at least $1-\delta$,*

$$\max_{i^* \in S} \sum_{t \in I_S^T} (\mu_{i^*} - \mu_{i_t}) \leq g(N_T(S), \delta)\,.$$

The probability is over the human's choices, reward observations, and the assistant policy.

With this definition in hand, in Algorithm 6 we give the natural generalization of Algorithm 1 to the R arm case.

Theorem 2 (R-arm reduction to imprecise max finding)**.** *Let $\mathcal{A}$ be an R-arm L-max oracle with query complexity Q, let $\delta > 0$ be arbitrary, and let*

$$B_R = Q + \left\lceil \frac{K-1}{R-1} \right\rceil.$$

Running Algorithm 6 with $\mathcal{A}$ achieves a human regret of

$$\mathcal{O}\left(Q + RLB_R \log(T/B_R) \cdot g\left(\frac{T}{B_R \log(T/B_R)}, \frac{\delta}{T\binom{K}{R}} \right) \right).$$

with probability at least $1-\delta$*. When* $g(t,\delta) \in \mathcal{O}\left(t^\alpha f(\delta)\right)$ *for some constant* $\alpha > 0$ *and function* f*, this can be improved to*

$$\mathcal{O}\left(Q + RLf\left(\frac{\delta}{T\binom{K}{R}}\right) \cdot B_R^{1-\alpha} \cdot T^\alpha\right).$$

Substituting the generalized max-finding oracle in Corollary 2, which has $Q = \mathcal{O}\left(R^2K\right)$ and $L = \mathcal{O}\left(\log\log K\right)$, results in the following corollary.

Corollary 5 (R-armed Assistants)**.** *For every* $R \geq 2$ *and* $\delta > 0$*, there exist assistants that recommend* R *arms at a time and, with probability at least* $1-\delta$*, achieve a human regret of*

$$\mathcal{O}\left(K + R^3 \log\log(K) K \log\left({}^{T}/_{(R^2K)}\right) \cdot g\left(\frac{T}{R^2K\log\left({}^{T}/_{(R^2K)}\right)}, \frac{\delta}{T\binom{K}{R}}\right)\right).$$

In particular, when $g(t,\delta) \in \mathcal{O}\left(t^\alpha f(\delta)\right)$*, this is a regret of*

$$\mathcal{O}\left(K + R^{3-2\alpha} f\left(\frac{\delta}{T\binom{K}{R}}\right) \log\log(K) K^{1-\alpha} \cdot T^\alpha\right),$$

and when $g(t,\delta) = \sqrt{R \cdot t \log(1/\delta)}$*, this is a regret rate of*

$$\mathcal{O}\left(K + R^3 \log\log(K) \sqrt{TK \log\left(TK/\delta\right)}\right).$$

A weakness of this result is that as R increases, the regret guarantee worsens. Understanding the landscape of regret as R changes is an interesting open question.

G Detailed Setup and Further Experiments

G.1 Bandit instances and humans

Each experiment tests a Bernoulli bandit arms with means $\mu_1, \ldots, \mu_K$. Pulling arm a returns a Bernoulli reward of mean μ_a.

In the figure shown in the main body, the means are drawn i.i.d. from $\mathrm{Uniform}(0,1)$ and we run for $T = 50{,}000$ rounds.

We evaluate human regret against three kinds of human policies.

- **UCB1 human**: each unordered pair (i,j) runs an independent instance of UCB1 on the two arms, observing only its own pulls. The human is a no-regret online learner *within* each pair.
- **EXP3-IX human**: same construction but with EXP3-IX as the per-pair learner.
- **Maximally bad humans**: A family of deliberately adversarial humans designed to saturate the g-rationality bound. The human picks the incorrect arm for as long as possible before the regret bound is violated. More precisely, on the nth showing of any pair $\{i,j\}$, the human picks the worse arm if

  $$R_{\{i,j\}}(n-1) \;+\; \Delta_{\{i,j\}} \;<\; g(n),$$

  and otherwise picks the better arm. We instantiate the family with three growth rates:

  $$g(t) \in \left\{\log t,\ \sqrt{t},\ t^{3/4}\right\}.$$

Dueling-bandit baselines

We use a number of baselines in these experiments. These include Sparring [Ailon et al., 2014], Relative UCB (RUCB) [Zoghi et al., 2014], Copeland Confidence Bound (CCB) [Zoghi et al., 2015], Double Thompson Sampling (DTS) [Wu and Liu, 2016], Beat the Mean (BTM) [Yue and Joachims, 2011], Interleaved Filter 2 (IF2) [Yue et al., 2012], and D-EXP3 [Saha et al., 2021]. Each dueling bandit baseline recommends two arms and uses the human's choice as the comparison feedback. For Sparring, we use two copies of the UCB learner of Ailon et al. [2014], with the same hyperparameter α.

Table 1: Hyperparameter search grids. Parameter names follow the cited papers, except for the added UCB-on-K multiplier c. The search grids are our experimental choices. Here $\mathcal{M} = \{0.25, 0.5, 1, 2, 4\}$.

Algorithm	Parameter	Search grid
Sparring [Ailon et al., 2014]	α	$\mathcal{M}$
RUCB [Zoghi et al., 2014]	α	$\{0.51, 0.6, 0.8, 1, 1.5, 2\}$
CCB [Zoghi et al., 2015]	α	$\{0.51, 0.6, 0.8, 1, 1.5, 2\}$
DTS [Wu and Liu, 2016]	α	$\{0.51, 0.6, 0.8, 1, 1.5, 2\}$
BTM [Yue and Joachims, 2011]	γ	$\{1, 1.5, 2, 3\}$
IF2 [Yue et al., 2012]	δ	$\{0.001, 0.01, 0.05, 0.1\}$
D-EXP3 [Saha et al., 2021]	η	$\{m\eta_0 : m \in \mathcal{M}\}$
EXP3-on-K [Auer et al., 2002b]	γ	$\{\min\{1, m\gamma_0\} : m \in \mathcal{M}\}$
UCB-on-K [Auer et al., 2002a]	c	$\{0.1, 0.25, 0.5, 1, 2\}$

As controls, we use three algorithms: 1) randomly picking pairs of arms, 2) UCB1, and 3) EXP3 on the underlying bandit itself. The last two algorithms cheat and see the reward. These serve to lower bound the performance any assistive dueling bandit algorithm can achieve.

We use exactly the parameterizations of the cited papers. For UCB-on-K [Auer et al., 2002a], c represents the multiplier on the exploration bonus: $c\sqrt{2\log t/n_a}$, where n_a is the number of times arm a was pulled so far.

The reference rates are

$$\eta_0(K,T) = \left(\frac{\log K}{T\sqrt{K}}\right)^{2/3}, \qquad \gamma_0(K,T) = \sqrt{\frac{K\log K}{(e-1)T}},$$

where T is the horizon of the run. D-EXP3 uses exploration rate $\gamma = \min\{1, \sqrt{K\eta}\}$.

G.2 Hyperparameter sweep protocol

We use a standard tune-then-evaluate procedure with disjoint random seeds. Tuning is performed at $T_{\text{tune}} = 20{,}000$ rounds; the chosen hyperparameter for each (algorithm, human) pair is then reused at the evaluation horizon $T = 50{,}000$ reported in the main text.

Tuning phase. For every (algorithm, human, hyperparameter value) triple we run 25 independent simulations and record the mean cumulative regret at horizon T_{tune}. The hyperparameter value minimizing this tuning-mean final regret is selected for that (algorithm, human) pair. The two reward-observing controls (EXP3-on-K and UCB-on-K) bypass the human, so a single sweep is performed for each and the chosen hyperparameter is reused across all human types.

Evaluation phase. With the tuned hyperparameter fixed per (algorithm, human) pair, each algorithm is rerun on 25 new seeds. Reported regret curves are means over those seeds with ± 1 standard-error bands.

G.3 Computational resources

All experiments were run on a single laptop with an Intel Core i9-13900H (14 cores, 20 logical processors, 2.6 GHz base) and 32 GB of RAM, running Windows 11. The full experiments can be reproduced end-to-end in well under a day.

G.4 Further Experiments

We include in this section the same experiment run for various different number of arms. As the number of arms increases, the problem becomes harder and harder. See Figure 5 and Figure 6.

We also compare algorithms on bandits where the rewards are not chosen uniformly.

In Figure 7, we fix 10% of the arms to have a reward of 1, and we make the rest of the arms have a reward drawn uniformly at random from $[0, 1]$.

In Figure 8, we fix an arm to have a reward of 1 another arm to have a reward of 0.9, and make the rest of the arms have a reward chosen uniformly at random from $[0, 0.9)$.

In Figure 9, we draw rewards from a clipped Gaussian centered at 0.5 with standard deviation 0.2.

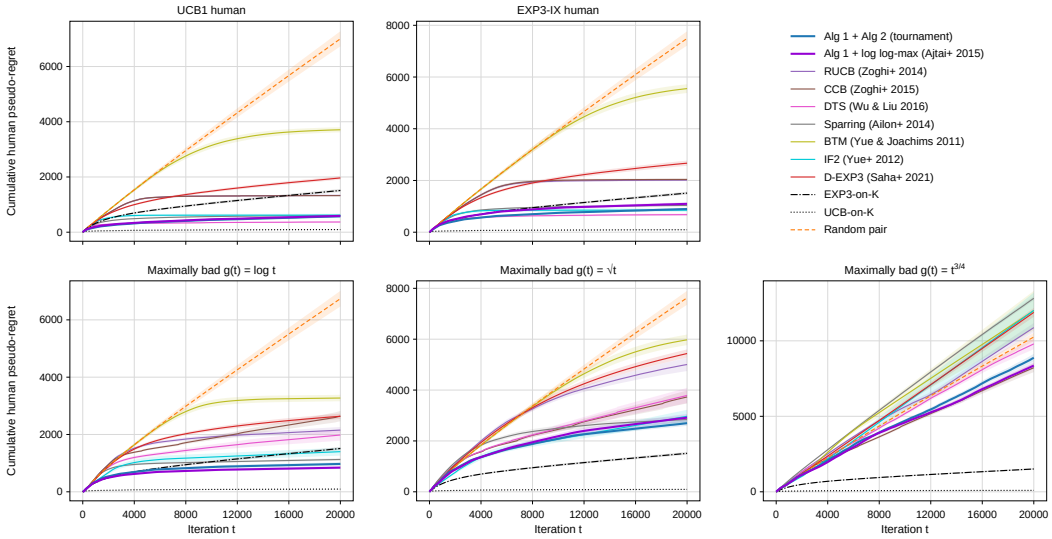


Figure 5: Figure 4, but run on $K = 20$ arms for 20,000 iterations.

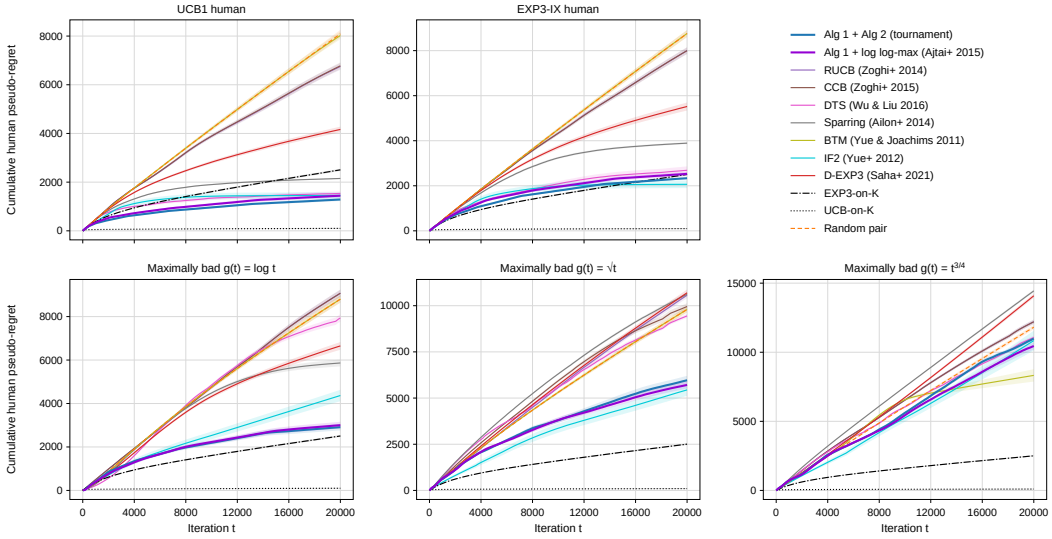


Figure 6: Figure 4, but run on $K = 50$ arms for 20,000 iterations.

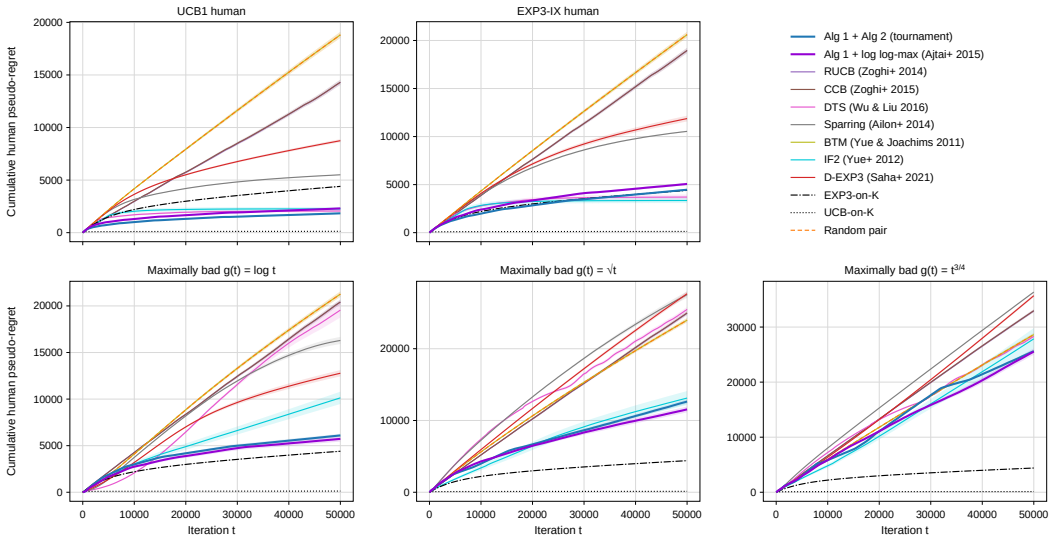


Figure 7: An experiment with $K = 100$ arms, 10% of which have a reward of 1, and the rest have an initial mean reward drawn uniformly between $[0, 1]$.

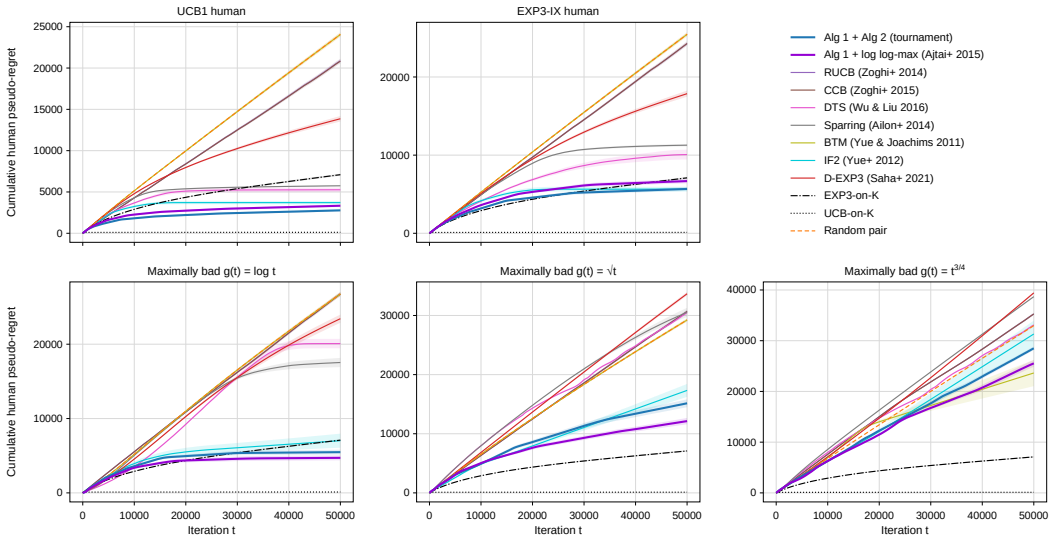


Figure 8: An experiment with $K = 100$ arms, with a fixed gap of 0.1 between the best and second best arm, and the rest of the arms with initial mean rewards drawn randomly from $[0, 0.9]$.

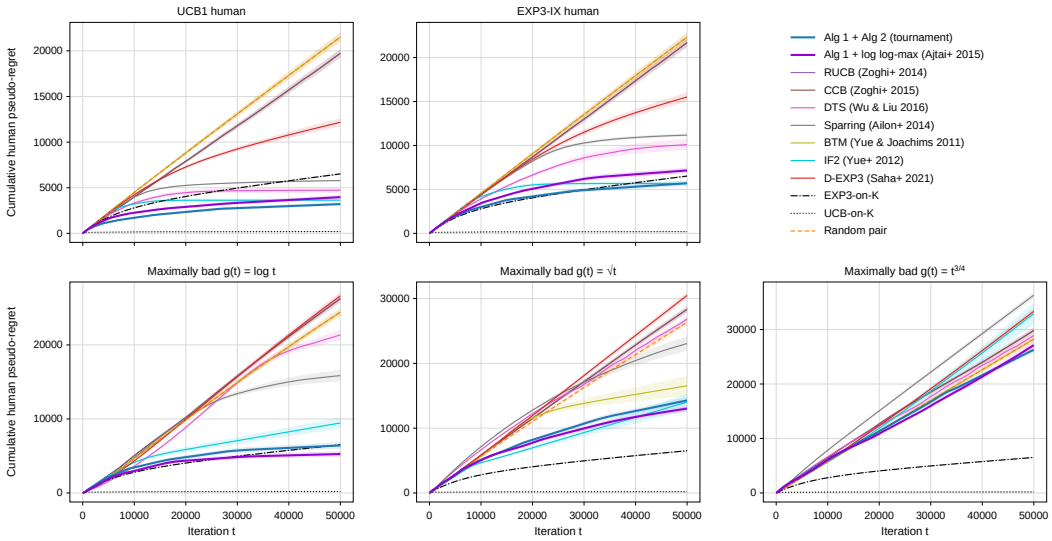


Figure 9: An experiment with $K = 100$ arms, the rewards of which are initially drawn from a Gaussian $\mathcal{N}(0.5, 0.2^2)$ clipped between 0 and 1.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: All claims are theoretically proven, and experimental evidence is given to corroborate the theory.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations of our results are discussed in the Discussion section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings,

model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.

- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Full proofs of all theorems are given in this work. All assumptions are given and discussed in the Preliminaries section.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: A section in the appendix is dedicated to describing the experiments in sufficient detail to be able to reproduce them. The code is included in the supplementary material.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code is included in the supplementary material.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.

- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: All experimental details are given in the appendix, including all hyperparameters and how they were tuned.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Runs of our experiments are repeated and standard error is plotted.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The computer resources used to run the experiments are given in the appendix.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: Our work conforms with the NeurIPS code of ethics.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [N/A]

Justification: As a primarily theoretical work aimed at introducing a bandit algorithm, the societal impact of this work is limited.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This work provides no trained model or data.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [N/A]

    Justification: This paper does not use existing assets.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Documentation is provided alongside the code.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method developed in this research does not involve LLMs.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.