The Matching Problem

- Objective: Minimize the expected cumulative regret over $n$ rounds

$$\inf_{\Pi} \mathbb{E}_{\Pi} \left[ \sum_{t=1}^{n} \sum_{j \in J} \max_{\mu_{ij}} \mu_{ij} - \mu_{ij} \left( \eta(t,k(t)) \right) n \right]$$

- $J$: the set of jobs available
- $\mu_{ij}$: the expected payoff for matching worker $i$ to job $j$
- $\eta(t,k(t))$: the available budget at time $t$
- $\Pi$: a policy

Lower Bounds on Achievable Performance

THEOREM [Information-theoretic lower bounds]: Naturally, the problem admits

- Instance-dependent lower bounds of $\Omega(n)$.
- Minimax lower bounds of $\Omega(\sqrt{n})$.

THEOREM [Worker-distribution dependent lower bounds]: Suppose that $|J|=1$ and jobs arrive one at a time. Let $\Delta < 1$ denote the proportion of the "optimal" worker-type in the population. Then, for all problem instances that have a difference of at least $\Delta > 0$ between the top two mean rewards, one has

$$\mathbb{L}_{\|\Pi\|,\eta} \left[ \inf_{\Pi} \mathbb{E}_{\Pi} \log n \right] \geq \frac{\Delta}{4 \alpha^2}$$

where $\Pi_{\alpha}$ denotes the class of "memoriless" policies, i.e., at any time, the decision to match an incoming job to an "unexplored" worker is history-independent.

Remark (II): $\Pi_{\alpha}$ is a rich policy class containing several natural approaches to the problem. For example, $\Pi_{\alpha}$ includes the policy that selects "unexplored" workers only in the initial rounds and subsequently matches incoming jobs only to "explored" workers.

Algorithm Design

1. MATCH: A meta-algorithm for the matching problem

In what follows, ALG is a job-type-specific worker assignment rule that prescribes one match per period.

- Input: $J$ and ALG.
- Initialization: Set $C_i = 0$ for each $i \in J$. (Shuffled $C_i$ are parallel threads for type $j$ jobs)
- For $t = 1,2,\ldots$ do:
- For $j \in J$ do:
- if $C_i = 0$ then:
- Update $C_i = \max_{j} \{ C_j \}$
- Otherwise:

2. CAB: A job-type-specific worker assignment rule

(Policy for a "Countable-armed" Bandit problem)

In what follows, an "arm" is synonymous to a "worker."