Sequential Resource Allocation Under Uncertainty: An Index Policy Approach

Weici Hu
Advisor: Peter Frazier

Cornell ORIE

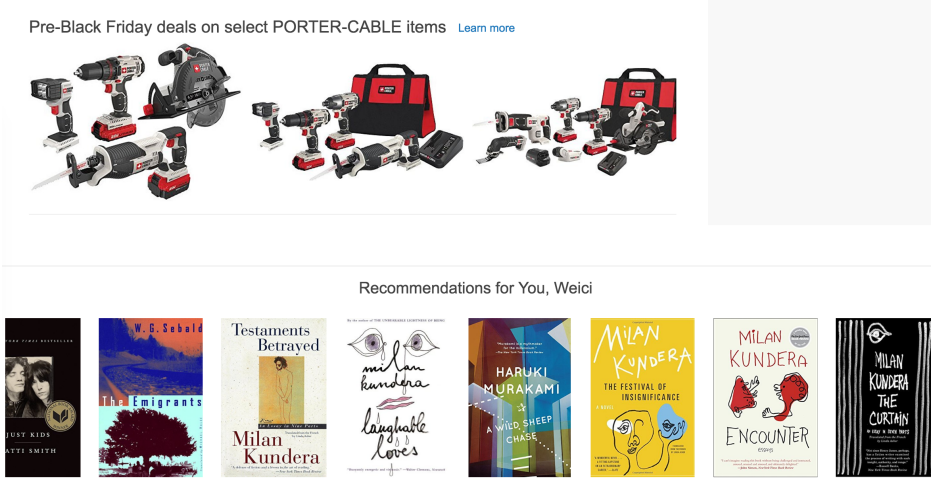
July 16, 2017

1. Index policy for Restless Multi-armed bandit (RMAB)
   - Introduction
   - Problem setup
   - Literature Review
   - Index policy
     - Computation of $\boldsymbol{\lambda}^*$
     - Compute ρ^*
     - Compute indices of states
     - Algorithm of Index policy
   - Asymptotic optimaity of the index policy
   - Numerical experiments
2. Index policy for other RMAB-like applications
   - MCS
   - crowdsourcing
3. Conclusion

RMAB problems have many applications in industry

Restless multi-armed bandit (RMAB) is a generalization of multi-armed bandit problem (MAB) by allowing:

- arms that are not pulled to change states
- multiple pulls per period



Personalized advertisement on Amazon

Project management

Problem Setup

We consider an MDP $(\mathbb{S}^K, \mathbb{A}^K, \mathbb{P}^{\cdot}, R)$ that consists of K identical sub-processes $(\mathbb{S}, \mathbb{A}, P^{\cdot}, r)$, specifically,

- Time horizon $T < \infty$.
- State space $\mathbb{S}^K$ is the cross-product of K $\mathbb{S}$. $\mathbb{S}$ is assumed finite.
- Action space $\mathbb{A}^K$ is the cross-product of K $\mathbb{A}$. $\mathbb{A} = \{0, 1\}$.
- Reward $R_t(\mathbf{s}, \mathbf{a}) = \sum_{x=1}^{K} r_t(s_x, a_x)$, $1 \leq t \leq T$, is additive of the reward of individual sub-processes.
- Transition probability $\mathbb{P}^{\mathbf{a}}(\mathbf{s}', \mathbf{s}) = \prod_{x=1}^{K} P^{a_x}(s'_x, s_x)$.
- Starts at an initial state $\mathbf{s}_1$.

I will use the term 'arm' and 'subprocesses' interchangeably.

Problem Setup Cont'd

- A Markov policy $\boldsymbol{\pi} : \mathbb{S}^K \times \mathbb{A}^K \times \{1, ..., T\} \rightarrow [0, 1]$, with $\boldsymbol{\pi}(\mathbf{s}, \mathbf{a}, t) = P(\mathbf{a}|\mathbf{S}_t = \mathbf{s})$ (Our decision).
  We require $\sum_{\mathbf{a} \in \mathbb{A}^K} \boldsymbol{\pi}(\mathbf{s}, \mathbf{a}, t) = 1. \ \forall \mathbf{s} \in \mathbb{S}^K, \forall 1 \leq t \leq T.$
- Objective

$$\begin{aligned} \underset{\boldsymbol{\pi} \in \boldsymbol{\Pi}}{\text{maximize}} \quad & \mathbb{E}^{\boldsymbol{\pi}} \left[\sum_{t=1}^{T} R_t(\mathbf{S}_t, \mathbf{A}_t) \right] \\ \text{subject to} \quad & P^{\boldsymbol{\pi}}(||\mathbf{A}_t|| = m_t) = 1, \quad \forall 1 \leq t \leq T, \end{aligned} \tag{1}$$

where $||\mathbf{v}|| = \sum_{i=1} |v_i|$.

Difficulty: Optimal solutions are computationally infeasible

Optimal solutions of (1) can be obtained with Bellman optimality equations.

But it requires $O(|\mathbb{S}|^K |\mathbb{A}|^K T)$ time complexity and $O(|\mathbb{S}|^K |\mathbb{A}|^K T)$ storage complexity.

The complexity grow exponentially with the number of sub-processes K, and becomes computationally infeasible for large K.

Past attempts

- **[Gittins 1979]**: Proposes Gittins index policy that is optimal for infinite horizon MAB with single pull per period.
- **[Whittle 1988]**: Proposes Whittle's policy that is derived from a Lagrangian relaxation of the RMAB, but only for the infinite time horizon case.
- **[Weiss and Weber 1990]** Whittle's index policy is asymtotically optimal when both the number of arms and the constraints go to infinity, but this is only true under restricted condition.
- **[Adelman & Mersereau 2008]**: Provides a heuristic policy for Weakly Coupled Dynamic Programs (WCDP) based upon value function approximation that decomposes across sub-problems.

Computing Optimal Lagrange Multiplier of (2)

We relax the original problem (1) to

$$\begin{aligned} \underset{\boldsymbol{\pi}\in\boldsymbol{\Pi}}{\text{maximize}} \quad & \mathbb{E}^{\boldsymbol{\pi}}\left[\sum_{t=1}^{T} R_t\left(\mathbf{S}_t, \mathbf{A}_t\right)\right] \\ \text{subject to} \quad & \mathbb{E}^{\boldsymbol{\pi}}(||\mathbf{A}_t||) = m_t, \quad \forall 1 \leq t \leq T. \end{aligned} \tag{2}$$

We relax (2) further to

$$P(\boldsymbol{\lambda}) = \max_{\boldsymbol{\pi}\in\boldsymbol{\Pi}} \mathbb{E}^{\boldsymbol{\pi}}\left[\sum_{t=1}^{T} R_t\left(\mathbf{S}_t, \mathbf{A}_t\right)\right] - \sum_{t=1}^{T} \lambda_t \left(\mathbb{E}^{\boldsymbol{\pi}}[||\mathbf{A}_t||] - m_t\right). \tag{3}$$

Computing Optimal Lagrange Multiplier of (2)

We decompose (3) to

$$P(\boldsymbol{\lambda}) = KQ(\boldsymbol{\lambda}) + \sum_t \lambda_t m_t, \tag{4}$$

where

$$Q(\boldsymbol{\lambda}) = \max_{\pi \in \Pi} \mathbb{E}^{\pi} \left[\sum_{t=1}^{T} r_t(S_t, A_t) - \lambda_t A_t \right], \tag{5}$$

is the objective function for sub-process $(\mathbb{S}, \mathbb{A}, P^{\cdot}, r)$.
π is defined similarly to $\boldsymbol{\pi}$, with $\pi(s, a, t) = P(a|S_t = s)$.

Computing Optimal Lagrange Multiplier of (2)

For index policy we compute an optimal Lagrange Multiplier $\boldsymbol{\lambda}^*$

$$\arg\min_{\boldsymbol{\lambda}} P(\boldsymbol{\lambda}) = \arg\min_{\boldsymbol{\lambda}} \left(Q(\boldsymbol{\lambda}) + \sum_t \lambda_t \frac{m_t}{K} \right), \tag{6}$$

which can be solved by the following linear program (LP):

$$\begin{aligned} \min_{\{V(s,t),\lambda_t : s\in\mathbb{S}, t\in\{1,\ldots,T\}\}} \quad & V(s_1,1) + \frac{1}{K}\sum_t \lambda_t m_t \\ \text{subject to} \quad & V(s,t) - \sum_{s'\in\mathbb{S}} P^a(s,s')V(s',t+1) \geq r_t(s,a) - \lambda_t a, \\ & \forall\, s\in\mathbb{S}, a\in\mathbb{A}, 1\leq t\leq T-1 \\ & V(s,T) \geq r_T(s,a) - \lambda_1 a \quad \forall\, s\in\mathbb{S}, a\in\mathbb{A} \end{aligned} \tag{7}$$

$V^*(s_1,1) = Q(\boldsymbol{\lambda}^*)$.

Computing Optimal Lagrange Multiplier of (2)

Remarks:

- Problem (7) has $O(|\mathbb{S}|T)$ variables and $O(|\mathbb{S}||\mathbb{A}|T)$ constraints, which is manageable.
- $P(\lambda)$ is an point-wise maximum of a group of affine functions of $\boldsymbol{\lambda}$, hence convex in $\boldsymbol{\lambda}$. $\boldsymbol{\lambda}^*$ can also be solved by sub-gradient descent.

Compute occupation measure ρ^* of an optimal policy of $Q(\lambda^*)$

Occupation measure of a policy π: the fraction of the time a process spends in each state-action pair at a time step under π. $\rho(s, a, t) = \pi(s, a, t)P(S_t = s)$.

To compute ρ^*, we solve the following linear program (LP):

$$
\begin{aligned}
\max_{\rho} \quad & \sum_{t=1}^{T} \sum_{a \in \mathbb{A}} \sum_{s \in \mathbb{S}} \rho(s, a, t) r_t(s, a) \\
\text{subject to} \quad & \sum_{s \in \mathbb{S}} \rho(s, 1, t) = \frac{m_t}{K}, \forall t = 1 \ldots, T \\
& \sum_{a \in \mathbb{A}} \rho(s, a, t) - \sum_{a \in \mathbb{A}} \sum_{s' \in \mathbb{S}} \rho(s', a, t-1) P^a(s', s) = 0, \forall s \in \mathbb{S}, 2 \leq t \leq T \\
& \sum_{a \in \mathbb{A}} \rho(s, a, 1) = \mathbb{1}(s = s_1) \ \ \forall s \in \mathbb{S} \\
& \rho(s, a, t) \geq 0, \ \forall s \in \mathbb{S}, a \in \mathbb{A}, t = 1, \ldots, T.
\end{aligned}
\tag{8}
$$

Compute occupation measure ρ^* of an optimal policy of $Q(\boldsymbol{\lambda}^*)$

Lemma

If we construct a policy π^* *by*

$$\pi^*(s,a,t) = \begin{cases} \frac{\rho^*(s,a,t)}{\sum_{a\in\mathbb{A}}\rho^*(s,a,t)}, & \text{if } \sum_{a\in\mathbb{A}}\rho^*(s,a,t) > 0 \\ \frac{1}{|\mathbb{A}|}, & \text{if } \sum_{a\in\mathbb{A}}\rho^*(s,a,t) = 0 \end{cases} \tag{9}$$

for all $s \in \mathbb{S}, a \in \mathbb{A}, 1 \le t \le T$, *then* π^* *is an optimal policy for* $Q(\boldsymbol{\lambda}^*)$, *and satisfies*

$$\mathbb{E}^{\pi^*}[A_t] = \frac{m_t}{K}. \tag{10}$$

Quick justification:

- π^* meets the definition of a policy.
- $\mathbb{E}^{\pi^*}[\sum_{t=1}^{T} r_t(S_t, A_t)] = Q(\boldsymbol{\lambda}^*) + \sum_{t=1}^{T} \lambda_t^* \frac{m_t}{K}$ by strong duality. Hence $\mathbb{E}^{\pi^*}[\sum_{t=1}^{T} r_t(S_t, A_t) - \boldsymbol{\lambda}^* A_t] = Q(\boldsymbol{\lambda}^*)$.

Pre-computations: 2. Occupation Measure ρ^* of an optimal policy of $Q(\boldsymbol{\lambda}^*)$

Remark:

- Solving for ρ^* requires solving an LP with $|\mathbb{S}||\mathbb{A}|T$ variables and at most $T|\mathbb{S}|$ constraints.

Pre-computations: 3. Indices of states

Notation:

- Let $\pi^{\boldsymbol{\lambda}}$ be the optimal policy of $Q(\boldsymbol{\lambda})$that always chooses to play when the forward value of playing is tied with the forward value of not playing.
- Use $\mathbf{v}[c, t]$ to denote $\mathbf{v} + (c - v_t) * \mathbf{e}_t$.

We compute the index of a state $s \in \mathbb{S}$ at time t as

$$\beta_t(s) = \sup\{\beta : \pi^{\boldsymbol{\lambda}^*[\beta,t]}(s, 1, t) = 1\} \tag{11}$$

Algorithm of Index policy $\hat{\pi}$

Pre-compute: $\boldsymbol{\lambda}^*$; $\boldsymbol{\beta}$; ρ^*.

for $t = 1, ..., T$ **do**

Let $\beta_{t,[i]}$ be the i^{th} largest element in the list $\beta_t(\mathbf{S}_{t,1}), ..., \beta_t(\mathbf{S}_{t,K})$, so $\beta_{t,[1]} \geq \ldots \geq \beta_{t,[K]}$.

Let $\bar{\beta}_t = \beta_{t,[m_t]}$

Let $I_t = \{s : \beta_t(s) = \bar{\beta}_t \text{ and } s = \mathbf{S}_{t,x} \text{ for some x}\}$

Let $N_t(s) = |\{x : \mathbf{S}_{t,x} = s\}|$, for all s.

For $s \in I_t$, let

$$q(s) = \begin{cases} \frac{\rho^*(s,1,t)}{\sum_{s' \in I_t} \rho^*(s',1,t)}, & \text{if } \sum_{s' \in I_t} \rho^*(s',1,t) > 0 \\ \frac{N_t(s)}{\sum_{s' \in I_t} N_t(s')}, & \text{otherwise} \end{cases}$$

Let $b = \text{Rounding}(m_t - \sum_{s':\beta_t(s') > \bar{\beta}_t} N_t(s'), (q(s) : s \in I_t), (N_t(s) : s \in I_t))$

for all s **do**

If $\beta_t(s) > \bar{\beta}_t$, set all $N_t(s)$ sub-processes in s active.

If $\beta_t(s) = \bar{\beta}_t$, set $b(s)$ sub-processes in s active.

If $\beta_t(s) < \bar{\beta}_t$, set 0 sub-processes in s active.

end for

end for

Algorithm of Index Policy $\hat{\pi}$

Remark:

- Computational complexity of $\hat{\pi}$ (exluding pre-computations) is $O(TKlog(K))$.

Asymptotic optimality of the index policy $\hat{\pi}$

Notation:

- For $\boldsymbol{\alpha} \in \mathbb{R}^T$, and $c \in \mathbb{R}$, define $\lfloor \boldsymbol{\alpha} c \rfloor = (\lfloor \alpha_1 c \rfloor, ..., \lfloor \alpha_T c \rfloor)$.
- Let $Z(\boldsymbol{\pi}, \mathbf{m}, K)$ denote the expected reward of the original MDP (1), with constraint $\mathbf{m} = (m_1, ..., m_T)$ and K sub-processes.
- Let $\boldsymbol{\Pi}_{\mathbf{m},K}$ denote the set of all feasible Markov policies of the original MDP (1), with constraint $\mathbf{m} = (m_1, ..., m_T)$ and K sub-processes.

Theorem (1)

For any $\boldsymbol{\alpha} \in [0,1]^T$,

$$\lim_{K\to\infty} \frac{1}{K} \left(\max_{\boldsymbol{\pi} \in \boldsymbol{\Pi}_{\lfloor \boldsymbol{\alpha} K \rfloor, K}} Z(\boldsymbol{\pi}, \lfloor \boldsymbol{\alpha} K \rfloor, K) - Z(\hat{\pi}, \lfloor \boldsymbol{\alpha} K \rfloor, K) \right) = 0. \quad (12)$$

Asymptotic optimality of the index policy $\hat{\pi}$

Notations:

- Define $N_t(s)$ as the number of sub-processes in state s at time t, and $M_t(s)$ the number of sub-processes taking active actions in state s at time t, both under $\hat{\pi}$.
- Let π^* be a policy constructed using ρ^*.
- Let $P_t(s)$ denote the probability of a sub-process being in state s at time t under π^*.

Theorem (2)

For every $s \in \mathbb{S}$ *and* $1 \leq t \leq T$,

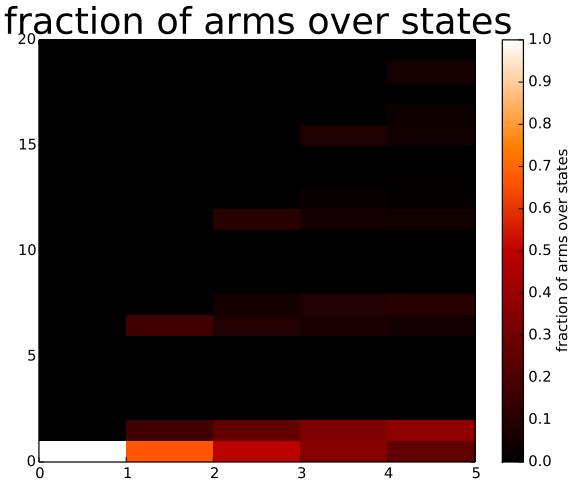
$$\lim_{K\to\infty} \frac{N_t(s)}{K} = P_t(s), \quad P^{\hat{\pi}} - a.s., \tag{13}$$

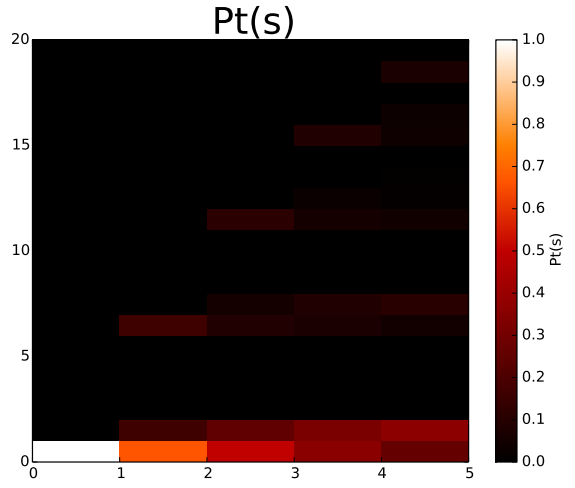
and

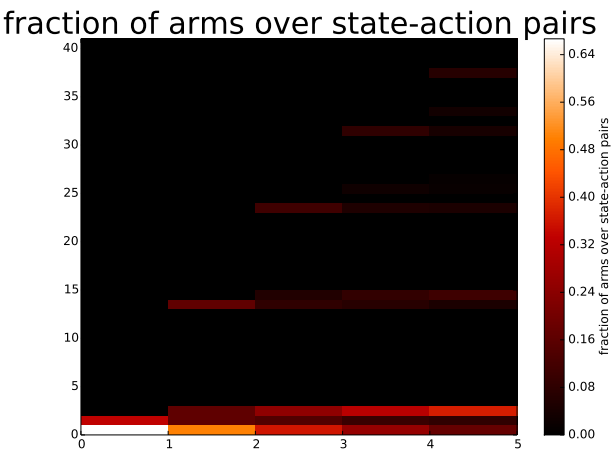
$$\lim_{K\to\infty} \frac{M_t(s)}{K} = P_t(s) * \pi^*(s, 1, t), \quad P^{\hat{\pi}} - a.s., \tag{14}$$

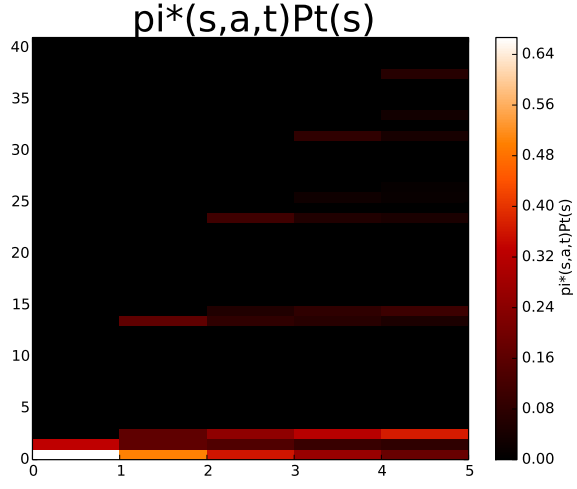
Theorem 2 is proven by induction over time

A numerical illustration of Theorem 2 using an instance of MAB with $K = 10000$









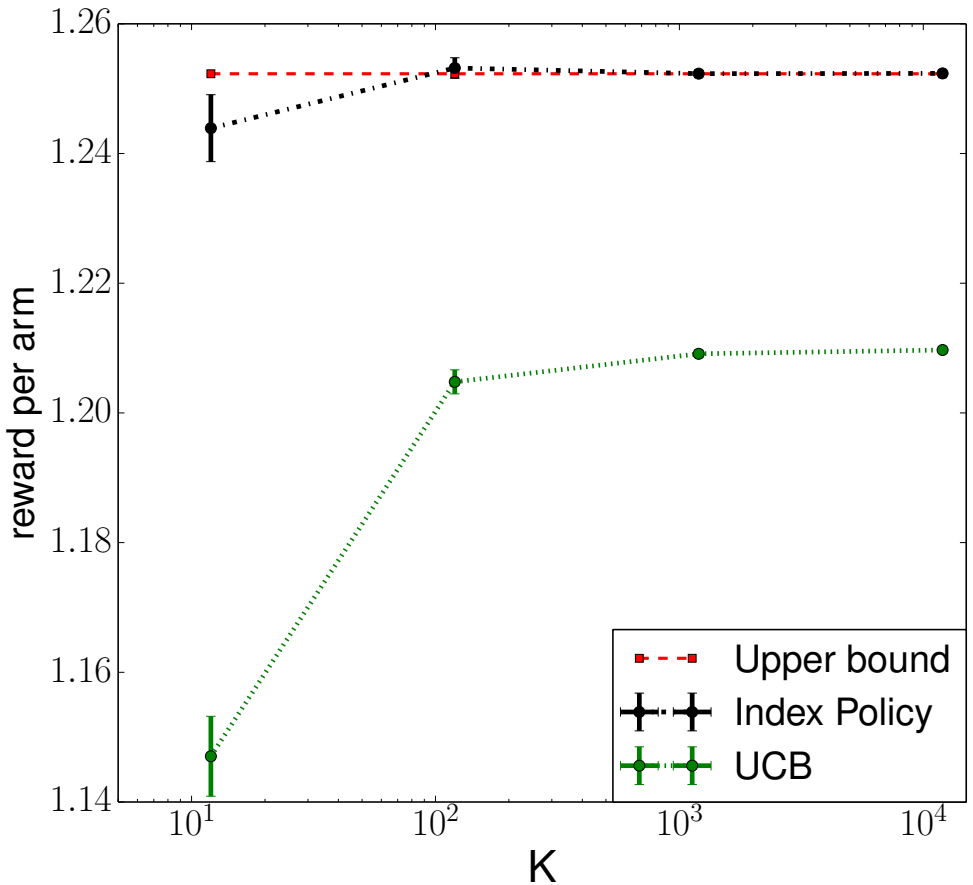
Theorem 2 is proven by induction over time

Proof Intuition of Theorem 2 (Warning: lots of hand-waves):
Assume Theorem 2 holds for time $t-1$.

- Proof that the first result holds at t:
  Same distribution of arms over state, same fraction of arms get to set active in each state $\Rightarrow$ same distribution of arms over state in the next time period.
- Proof that the second result holds at t:
  - 1. $\{s \in \mathbb{S} : \beta_t(s) > \lambda_t^*\} \approx \{s \in \mathbb{S} : \pi^* \text{ chooses } a = 1 \text{ in } s\}$.
    2. π^* chooses $a = 1$ α_t of the time.
  - $1 + 2 \Rightarrow \sum_{s \in \{s \in \mathbb{S} : \beta_t(s) > \lambda_t^*\}} P_t(s) \approx \alpha_t$
    $\Rightarrow \sum_{s \in \{s \in \mathbb{S} : \beta_t(s) > \lambda_t^*\}} \frac{N_t(s)}{K} \approx \alpha_t$.
  - The index policy $\hat{\pi}$ sets $\alpha_t K$ arms active, hence

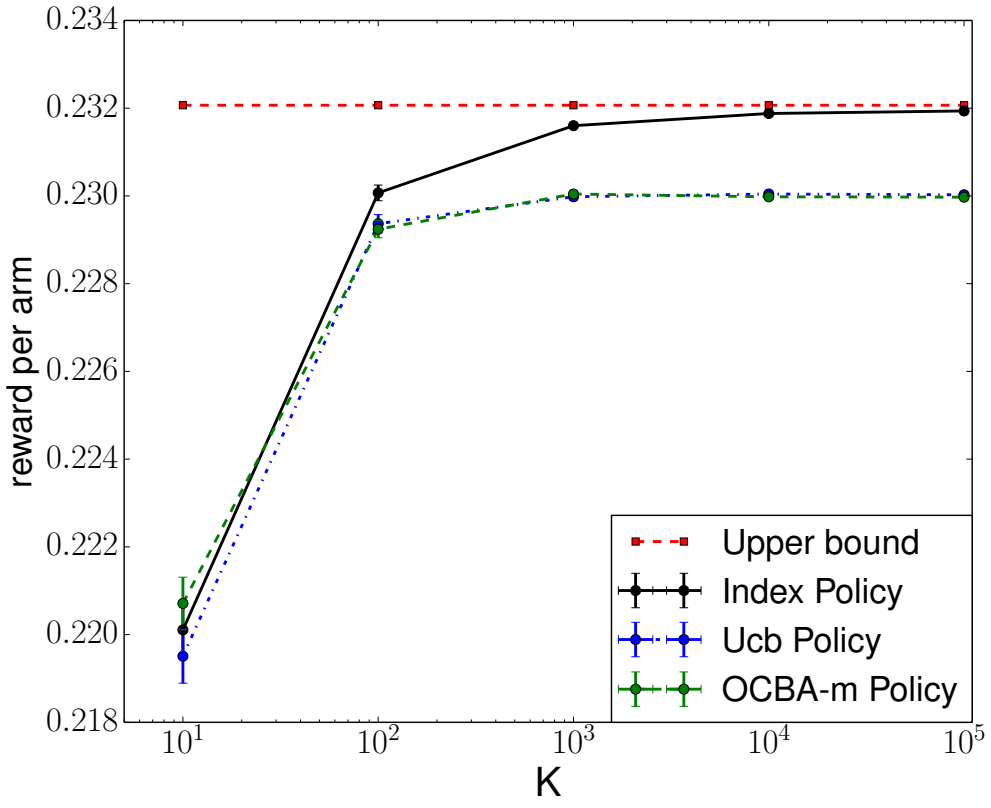
Numerical Experiments: Bernoulli MAB

K arms, each returns a random reward of 0 or 1 when pulled.
$T = 6$, $\alpha_t = \frac{1}{3}$, for all t.
The problem aims to maximize the total expected reward.



Numerical Experiments: Subset Selection Problem

K designs, m parallel computing resources. $T = 5$
$\alpha_t = 0.5$ for $1 \leq t \leq 4$. $\alpha_5 = 0.3$
The problem aims to select the best $\bar{m}$ designs out of K.



Application 1: Multiple Selection with Standard

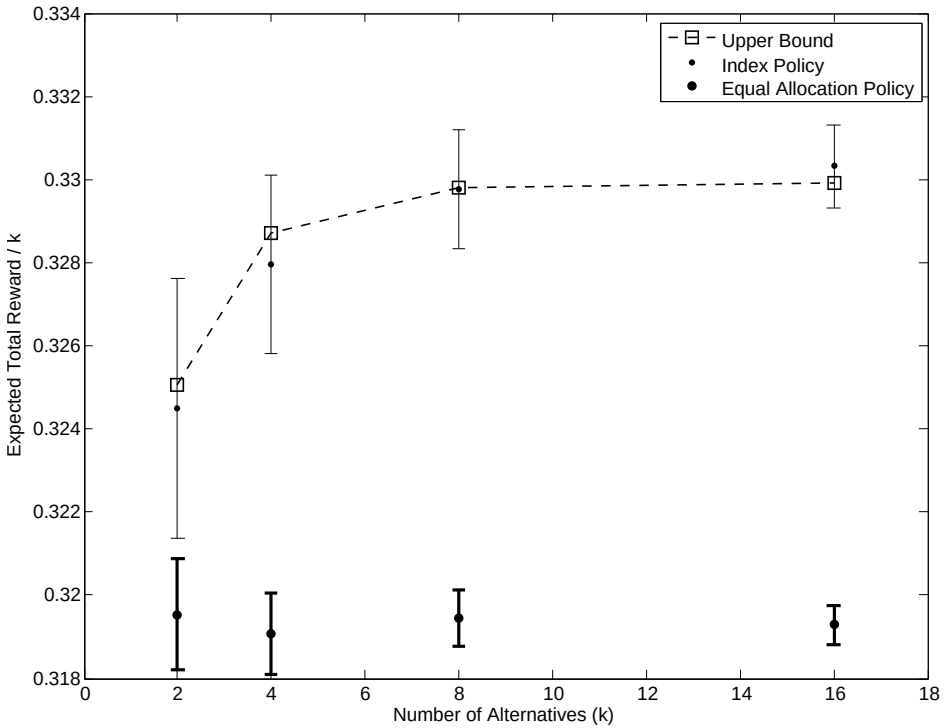
- k alternatives
- m parallel simulating resources per time step
- T time horizon
- θ_x is the underlying true performance of alternative x, $x \in \{1, ..., k\}$
- d_x the known threshold for alternative x
- $z_{n,x}$ is the number of simulation resources to use on alternative x at time step n. This is our allocation decision. $\sum_x z_{n,x} \leq m$
- After time step T, for each alternative x, we decide whether $\theta_x > d_x$ based on the past results of simulation
- We obtain a reward $R = \sum_x R_x$. R_x can be a 0-1 reward or linear reward.

Goal:Find an allocation of simulation resources to best support the decision at time T x

Application 1: Multiple Comparison with Known Standard

For each $k = 2, 4, 8, 16$

- $m = k$
- $d_x = 0.2, \forall x \in \{1, ..., k\}$
- $\mathbf{S}_0 = (\boldsymbol{\alpha}_0, \boldsymbol{\beta}_0) = (1, 1)^k$



Conjecture: The index policy has asymptotic optimality as m and K goes to infinity at the same rate.

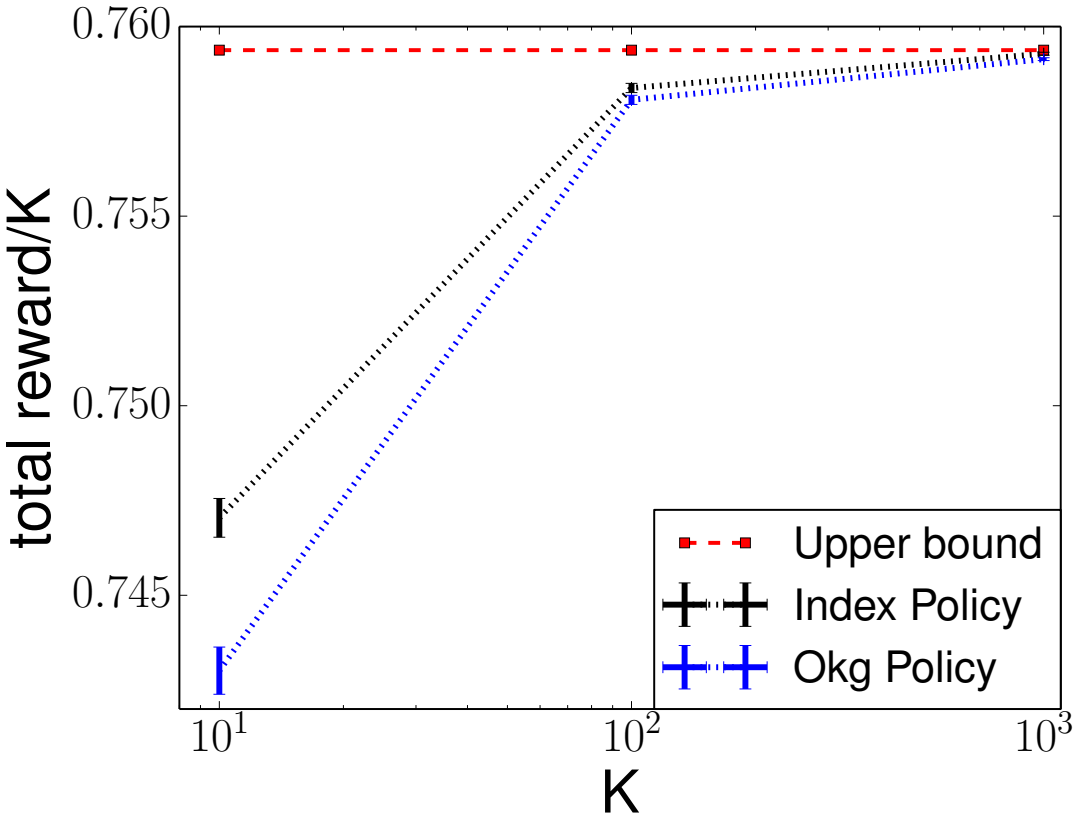
Application 2: Classification with Crowdsourcing

- k labeling tasks
- U total workers
- M/M/k queue: workers come in with rate r, and complete their job with rate μ.
- θ_x is the underlying likelihood for a task to have a positive label.
- d_x the known threshold for alternative x
- After worker budget U has been exhausted, for each alternative x, we decide whether the true label is positive or negative based on a 1-0 reward.

Goal:Find an allocation of workers to best support the final decision on true labels

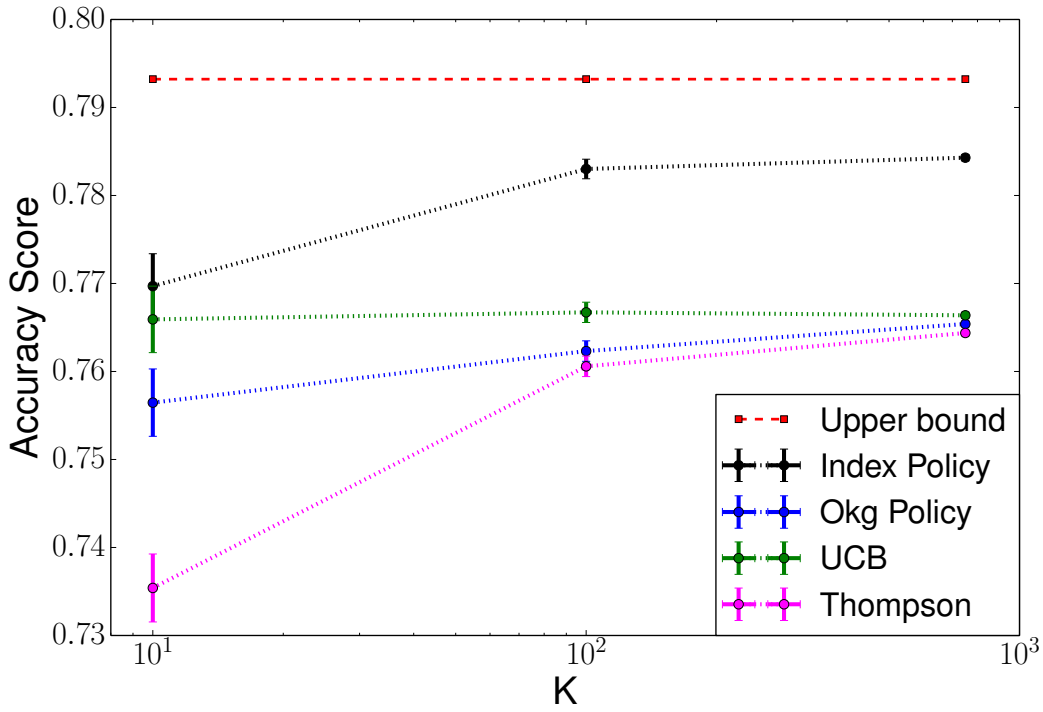
Application 2: Classification with Crowdsourcing

Queue: $r = 0.1$, $\mu = 0.4$, $K = 10, 100, 1000$, $U = 1.2K$.
We use a non-informative prior with $\boldsymbol{\alpha} = \mathbf{1}$ and $\boldsymbol{\beta} = \mathbf{1}$, and a threshold $d_x = 0.5$ for all the tasks.



Application 2: Classification with Crowdsourcing

We used dataset PASCAL RTE-1[**Snow2008**], which consists of 800 tasks, each comes with 10 labels obtained from crowdworkers and a gold standard label



Conjecture: The index policy has asymptotic optimality as U and K goes to infinity at the same rate.

Conclusion

Index policies under different setting show asymptotically optimal behaviors while significantly reduce the computational complexity. Possible extension of the framework:

- To include infinite state space
- To allow a total budget over all time period
- To allow multiple actions
- To allow a more general weakly coupled dynamic program setting