Selecting Among Heuristics by Solving Thresholded k-Armed Bandit Problems

Matthew J. Streeter[1] and **Stephen F. Smith**[2]
Computer Science Department
and Center for the Neural Basis of Cognition[1] and
The Robotics Institute[2]
Carnegie Mellon University
Pittsburgh, PA 15213
{matts, sfs}@cs.cmu.edu

Abstract

Suppose we are given k randomized heuristics to use in solving a combinatorial problem. Each heuristic, when run, produces a solution with an associated quality or value. Given a budget of n runs, our goal is to allocate runs to the heuristics so as to maximize the number of sampled solutions whose value exceeds a specified threshold. For this special case of the classical k-armed bandit problem, we present a strategy with $O(\sqrt{np^*k\ln n})$ additive regret, where p^* is the probability of sampling an above-threshold solution using the best single heuristic. We demonstrate the usefulness of our algorithm by using it to select among priority dispatching rules for the resource-constrained project scheduling problem with maximal time lags (RCPSP/max).

1. Introduction

Suppose we are given a set of k randomized heuristics to solve a combinatorial optimization problem. Running the i^{th} heuristic produces a solution with an associated quality or value. With (unknown) probability $p_i(t)$, the value is $> t$. The value of $p_i(t)$ is instance-dependent, and the heuristics are black boxes whose only observable behavior is the value of the solutions they return. We would like to solve the following problem:

Problem 1: Given a budget of n runs, allocate runs among the heuristics so as to maximize the probability that a solution with value $> t_1$ is obtained.

Unfortunately, when solving Problem 1 we cannot do better than to select heuristics at random (no information about the probabilities p_i is gained until an acceptable solution has been found, at which point the information is useless). We will instead focus on the following related problem:

Problem 2: Given a budget of n runs, allocate runs among the heuristics so as to maximize the expected number of solutions with value $> t_2$.

The idea is that $t_2 < t_1$, that $n \cdot \max_i p_i(t_1)$ is prohibitively small, and that $n \cdot \max_i p_i(t_2)$ is small but not prohibitively so. If $\arg\max_i p_i(t_1) = \arg\max_i p_i(t_2)$ (i.e., the heuristic that is most likely to generate a nearly-acceptable solution is also most likely to generate an acceptable solution), then in solving Problem 2 we will also solve Problem 1. In practice, we have found that sets of heuristics for real combinatorial problems often have this property.

1.1. A Taxonomy of k-Armed Bandit Problems

In a k-armed bandit problem, we are faced with a set of k slot machines ("one-armed bandits"), each with a single arm. Each arm, when pulled, returns a payoff drawn from a fixed distribution over the interval $[0, 1]$. Given a budget of n pulls, we wish to allocate pulls so as to maximize some objective. We consider three variants of the problem, as summarized in the following table. The "thresholded" variant is new to this paper.

Problem	**Objective to maximize**
Classical	Total payoff
Max	Maximum payoff (from any single pull)
Thresholded	Number of payoffs that exceed a fixed threshold t.

Note that the thresholded k-armed bandit problem is a special case of the classical k-armed bandit problem where payoffs are drawn from $\{0, 1\}$.

We denote the mean payoff of the i^{th} arm by μ_i, and define $\mu^* = \max_i \mu_i$. For thresholded k-armed bandit problems, we denote by p_i the probability that a payoff from the i^{th} arm exceed the specified threshold, and define $p^* = \max_i p_i$.

1.2. Contributions

The contributions of this paper are twofold. First, we present an algorithm for the classical k-armed bandit problem with additive regret $O(\sqrt{n\mu^*k\ln n})$. When applied to the thresholded k armed bandit problem, our algorithm has additive regret $O(\sqrt{np^*k\ln n})$. Our algorithm has better regret than the algorithm of Auer, Cesa-Bianchi, and Fischer (2002b), which has additive regret $O(\sqrt{nk\ln n})$. Regrettably, our bound is slightly worse than that of the algorithm of Auer

et al. (2002a), which addresses a more general problem that the one considered here. We hope to address this discrepancy in future work.

Second, we demonstrate that an algorithm for the thresholded k-armed bandit problem can be profitably applied to the problem of selecting among priority dispatching rules for the RCPSP/max.

1.3. Related Work

The classical k-armed bandit problem was first studied by Robbins (1952) and has since been the subject of numerous papers; see Berry and Fristedt (1986) and Kaelbling (1993) for overviews. As discussed in §1.2, the algorithms of Auer et al. (2002a) and Auer, Cesa-Bianchi, and Fischer (2002b) are the most relevant to this paper.

Studies of the max k-armed bandit problem have much the same objectives as ours (Cicirello & Smith 2004; 2005; Streeter & Smith 2006). These works use ideas from *extreme value theory* to justify assumptions about the payoff distributions of each arm. In constrast, our work takes a non-parametric approach. We compare our algorithm to the max k-armed bandit algorithm of Cicirello and Smith (2005) in §4.

3. An Interval Estimation Algorithm

We will analyze the following procedure for solving (thresholded) k-armed bandit problems.

Procedure **IntervalEstimation** (n, δ):
1. Initialize $x_i \leftarrow 0$, $n_i \leftarrow 0$, and $u^i \leftarrow \infty$ for all $i \in \{1, 2, \ldots, k\}$.
2. Repeat n times:
   (a) $i^* \leftarrow \arg\max_i u^i$.
   (b) Pull arm i^*; increment x_i by the payoff received; and increment n_i.
   (c) $u^i \leftarrow U(x_i, n_i, \delta)$.

The function $U(x_i, n_i, \delta)$ returns a $1 - \delta$ upper confidence interval for p_i. More formally, for any parameter p_i (as well as any n_i, and δ), we are guaranteed that

$$\mathbb{P}_{p_i}[U(x_i, n_i, \delta) < p_i] \leq \delta \,. \tag{1}$$

The tightest possible upper bound U can be computed exactly using the binomial distribution. For the purposes of our analysis, it is easier to consider a weaker upper bound defined using Chernoff's inequality.

Chernoff's inequality. *Let $X = \sum_{i=1}^{n} X_i$ be the sum of n i.i.d. variables with $X_i \in \{0, 1\}$ and $\mathbb{P}[X_i = 1] = p$. Then for $\beta > 0$,*

$$\mathbb{P}\left[\frac{X}{n} > (1+\beta)p\right] < \exp\left(-\frac{\mu\beta^2}{3}\right)$$

and

$$\mathbb{P}\left[\frac{X}{n} < (1-\beta)p\right] < \exp\left(-\frac{\mu\beta^2}{2}\right).$$

Lemma 1. *The function U defined by*

$$U(x_i, n_i, \delta) = \max\{p_i | f(p_i, x_i, n_i) > \delta\}$$

where

$$f(p_i, x_i, n_i) = \exp\left(-\frac{1}{2} n_i p_i \left(1 - \frac{x_i}{n_i p_i}\right)^2\right)$$

satisfies condition (1).

Proof. Omitted. □

We first establish a bound on the number of times a suboptimal arm will be sampled.

Lemma 2. *With probability at least $1 - n\delta$, each arm $i \in \{1, 2, 3, \ldots, k\}$ will be sampled at most $\frac{6}{p^*}(1 - \sqrt{\alpha_i})^{-2} \ln(\frac{1}{\delta})$ times, where $\alpha_i = \frac{p_i}{p^*}$.*

Proof. Omitted. □

Theorem 1. *Running interval estimation for n trials with parameter $\delta = \frac{1}{n^3}$ yields at least*

$$s - 6\sqrt{2s(k-1)\ln(n)} - \frac{1}{n}$$

above-threshold payoffs in expectation, where $s = np^*$.

Proof. We consider only the special case $k = 2$. The proof for general k is similar.

Assume $p_1 = p^*$ and let $p_2 = \alpha p_1$, where $\alpha < 1$. By Lemma 2, we sample arm 2 at most $\min\{n, \frac{6}{p_1}(1 - \sqrt{\alpha})^{-2} \ln(\frac{1}{\delta})\}$ times, so (with probability at least $1 - n\delta$) expected regret is at most

$$p_1(1-\alpha) \min\left\{n, \frac{6}{p_1}(1-\sqrt{\alpha})^{-2} \ln\left(\frac{1}{\delta}\right)\right\} .$$

For $\alpha < 1$, we have

$$\begin{aligned} \tfrac{1-\alpha}{(1-\sqrt{\alpha})^2} &= \tfrac{1-\alpha}{(1-\sqrt{\alpha})^2} \cdot \tfrac{(1+\sqrt{\alpha})^2}{(1+\sqrt{\alpha})^2} \\ &= \tfrac{(1+\sqrt{\alpha})^2}{1-\alpha} \\ &< \tfrac{4}{1-\alpha} \,. \end{aligned}$$

Thus the expected regret is at most

$$\min\left\{s\Delta, \frac{24}{\Delta} \ln\left(\frac{1}{\delta}\right)\right\}$$

where we define $\Delta = 1 - \alpha$. Solving the equation $s\Delta = \frac{24}{\Delta} \ln(\frac{1}{\delta})$ gives $2\sqrt{\frac{6}{s} \ln(\frac{1}{\delta})}$ as the value of Δ that maximizes expected regret. So the expected regret is at most $2\sqrt{6s \ln(\frac{1}{\delta})} = 6\sqrt{2s \ln(n)}$.

With probability $n\delta = \frac{1}{n^2}$, Lemma 2 cannot be applied. Because regret can never exceed n, this increases expected regret by at most $\frac{1}{n}$. □

4. Experimental Evaluation

To evaluate the practical value of our interval estimation algorithm, we use it to select among randomized priority dispatching rules for the resource-constrained project scheduling problem with maximal time lags (RCPSP/max). Briefly, in the RCPSP/max one must assign start times to each of a number of activities in such a way that certain temporal and resource constraints are satisfied. Such an assignment of start times is called a *feasible schedule*. The objective is to find a feasible schedule whose makespan is minimal, where makespan is defined as the maximum completion time of any activity.

Even without maximal time lags (which make the problem more difficult), the RCPSP is NP-hard and is "one of the most intractable problems in operations research" (Möhring *et al.* 2003). When maximal time lags are included, the feasibility problem (i.e., deciding whether a feasible schedule exists) as well as the optimization problem is NP-hard.

4.2. Heuristics

We consider six randomized priority dispatching rules for the RCPSP/max. An approach that selects among randomized priority dispatching rules has been shown to give competitive performance on benchmark instances of the problem (Cicirello & Smith 2005). We consider the six randomized priority dispatching rules in the set $\mathcal{H} = \{LPF, LST, MST, MTS, RMS, Random\}$; see Cicirello and Smith (2004; 2005) for a more complete description of these heuristics.

4.3. Methodology

We evaluate our approach on a set $\mathcal{I}$ of 540 RCPSP/max instances from the ProGen/max library (Schwindt 1996). For each RCPSP/max instance $I \in \mathcal{I}$, we ran each heuristic $h \in \mathcal{H}$ 10,000 times, storing the results in a file. Using this data, we created a set $\mathcal{K}$ of 540 6-armed bandit problems (each of the six heuristics $h \in \mathcal{H}$ represents an arm). For each instance $K \in \mathcal{K}$, we ran three algorithms with a budget of $n = 10,000$ pulls: our interval estimation algorithm, the QD-BEACON algorithm of Cicirello and Smith (2005), and a straw man algorithm that simply sampled the arms in a round-robin fashion. When running our interval estimation algorithm, we use a thresholded version of K. We calculated (offline) the highest threshold such that, for some heuristic $h \in \mathcal{H}$, at least 5% of the schedules had quality (equal to -1 times makespan) in excess of the threshold.

4.5. Results

We first evaluate the three algorithms in terms of the number of above-threshold schedules that were obtained. For each algorithm $\mathcal{A}$ and each instance $\mathcal{K}$, we computed the ratio of the number of above-threshold schedules sampled by $\mathcal{A}$ to the number that would have been sampled using the single best heuristic. The table below shows the minimum and average value of this ratio over all 540 instances. In addition, the table shows p_{best}, the probability that the arm sampled most often by $\mathcal{A}$ was an arm that would yield a maximum-quality solution if sampled for all n trials.

Heuristic	**Min. ratio**	**Avg. ratio**	p_{best}
Interval estimation	0.80	0.95	0.93
QD-BEACON	0	0.81	0.88
Round-robin	0.16	0.49	NA

The above table shows that interval estimation outperforms the other two algorithms in terms of its ability to obtain above-threshold schedules.

We additionally computed the fraction of instances where the best schedule generated by interval estimation was better than the best schedule generated by QD-BEACON (resp. round-robin). Ignoring ties, the best schedule from interval estimation was superior to that from QD-BEACON in 84% of the time, and superior to that from round-robin 85% of the time.

Future Work

Below we outline two areas for potential future work.

Threshold selection. In the experiments reported in §4, we calculated (offline) the highest threshold such that, for some heuristic $h \in \mathcal{H}$, at least 5% of the schedules had quality in excess of the threshold. In a real application, the thresholds instead must be determined online, and may be adjusted dynamically over time. We are currently investigating approaches to this problem.

Variable run lengths and restarts. In this work, we have assumed that each run of each heuristic has (approximately) the same computational cost. It is desirable to effectively handle a set of heuristics where the run time varies significantly across heuristics and across multiple runs of a single heuristic. In this scenario it is also desirable to restart a heuristic if it appears unlikely to produce an acceptable solution within a reasonable amount of time. Specifically, it is desirable to learn online, for each heuristic, a time bound after which the heuristic should be restarted.

References

Auer, P.; Cesa-Bianchi, N.; Freund, Y.; and Schapire, R. E. 2002a. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing* 32(1):48–77.

Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002b. Finite-time analysis of the multiarmed bandit problem. *Machine Learning* 47:235–256.

Berry, D. A., and Fristedt, B. 1986. *Bandit Problems: Sequential Allocation of Experiments*. London: Chapman and Hall.

Cicirello, V. A., and Smith, S. F. 2004. Heuristic selection for stochastic search optimization: Modeling solution quality by extreme value theory. In *Proceedings of the 10th International Conference on Principles and Practice of Constraint Programming*, 197–211.

Cicirello, V. A., and Smith, S. F. 2005. The max k-armed bandit: A new model of exploration applied to search

heuristic selection. In *Proceedings of AAAI 2005*, 1355–1361.

Kaelbling, L. P. 1993. *Learning in Embedded Systems*. Cambridge, MA: The MIT Press.

Möhring, R. H.; Schulz, A. S.; Stork, F.; and Uetz, M. 2003. Solving project scheduling problems by minimum cut computations. *Management Science* 49(3):330–350.

Neumann, K.; Schwindt, C.; and Zimmerman, J. 2002. *Project Scheduling with Time Windows and Scarce Resources*. Springer-Verlag.

Robbins, H. 1952. Some aspects of sequential design of experiments. *Bulletin of the American Mathematical Society* 58:527–535.

Schwindt, C. 1996. Generation of resource–constrained project scheduling problems with minimal and maximal time lags. Technical Report WIOR-489, Universität Karlsruhe.

Streeter, M. J., and Smith, S. F. 2006. An asymptotically optimal algorithm for the max k-armed bandit problem. Technical Report CMU-CS-06-110, Department of Computer Science, Carnegie Mellon University.