Coupled Bisection for Root Ordering

Stephen N. Pallone, Peter I. Frazier, Shane G. Henderson

School of Operations Research and Information Engineering
Cornell University, Ithaca, NY 14853

Abstract

We consider the problem of solving multiple "coupled" root-finding problems at once, in that we can evaluate every function at the same point simultaneously. Using a dynamic programming formulation, we show that a sequential bisection algorithm is a close-to-optimal method for finding a ranking with respect to the zeros of these functions. We show the ranking can be found in linear time, prove an asymptotic approximation guarantee of 1.44, and conjecture that this policy is near-optimal.

Keywords: bisection, dynamic programming, Lagrangian relaxation, index policies

1. Introduction

Consider a function $f : \mathcal{S} \times [0, 1) \to \mathbb{R}$, where $\mathcal{S}$ is finite but large, and for every s, $f(s, \cdot)$ is monotonic with unique root. If we are interested in finding the zero $x^*(s)$ of $f(s, \cdot)$ for all elements $s \in \mathcal{S}$, then for each $f(s, \cdot)$ we could employ the classical bisection method. However, if one evaluation of f for some x yields values of $f(s, x)$ for all $s \in \mathcal{S}$, then we could potentially solve multiple bisection problems at once. Furthermore, if we are only interested in the *ordering* of elements s with respect to their zeros $x^*(s)$, calculating the zeros to precision is computationally unnecessary.

The coupled root-ordering setting has applications in computing Gittins [1] and Whittle [2] index policies, respectively used in multi-arm bandit and restless bandit problems. We are then interested in ordering states in the state space according to their Gittins or Whittle indices, which correspond to the zero of a particular function. The ordering of the states is all that is required to implement the index policy. Methods for evaluating these indices to precision are prevalent in the literature (see [3] for a discussion on computational methods). In practical applications where Gittins indices are computed, the problems typically have additional structure, such as sparse transition kernels or the ability to compute indices in an online fashion. The most competitive algorithms for computing Gittins index policies exploit these kinds of structure. Coupled bisection does not take advantage of any additional structure, and therefore is not competitive with these algorithms. However, its generality allows it to compute index policies for a wide range of problems that be formulated as instances of coupled root-ordering [4][5][6][7].

Email addresses: snp32@cornell.edu (Stephen N. Pallone), pf98@cornell.edu (Peter I. Frazier), sgh9@cornell.edu (Shane G. Henderson)

Coupled bisection can also be used in solving coupled root-finding problems, because it can be more computationally efficient to sort the roots before further refining their respective locations. One such example is the estimation of phase diagrams during the evaluation of piezoelectric materials [8]. Given a pair of chemical compounds A and B, one must determine for each temperature s in some set a critical threshold $x^*(s)$ such that mixtures of A and B with a fraction $x > x^*(s)$ of A form one crystal phase, while mixtures with $x < x^*(s)$ form another phase. Here, $f(s, x)$ is the crystal phase at temperature s and mixing coefficient x, and can be observed through a physical experiment. Synthesizing a particular linear combination x is time consuming, but allows easy observation of $f(s, x)$ for all temperatures s. This is a coupled root-finding problem.

Coupled root-finding also arises in remote sensing, when finding the boundary of a forest or other geographical feature from an airborne laser scanner [9]. Here, an aircraft chooses a latitude x at which to fly and observes the presence or absence of the feature, $f(s, x) \in \{0, 1\}$, for all longitudes s in the flight path. The boundary at longitude s is given by the root $x^*(s)$.

Naively, we could discretize the interval $[0, 1)$ and calculate f with respect to all discretized values of x. Although this is easy to program and understand, the computational investment can be massive and unnecessary. We develop a coupled bisection method that can sort these elements in a more efficient fashion.

We solve this problem by sequentially evaluating f at different values of x. When we evaluate f at some value x, we find $f(s, x)$ for every element s, and we can deduce for every element whether $x^*(s) \geq x$ or $x^*(s) < x$. At every iteration, we know of a subinterval that contains $x^*(s)$. These subintervals form a disjoint partition of $[0, 1)$. By evaluating f for a different value of x at each iteration, we refine the previous partition, choosing one subinterval to split into two. This continues until for every element each subinterval contains at most one root.

In this process, we must find a way to sequentially select the next value of x at which we evaluate f. One might conjecture by analogy that the optimal decision is to choose some subinterval and select the midpoint of that interval to be the next value of x. This policy is *not* optimal, but we show it can sort the elements by their associated zeros in $O(|\mathcal{S}|)$ iterations in expectation, and calculate the asymptotic constant to be bounded above by 1.44. We also provide a lower bound of $|\mathcal{S}| - 1$ for the minimum number of iterations for any policy, implying an approximation guarantee of 1.44. Moreover, we give computational evidence suggesting our proposed policy is even closer to optimal than the 1.44 guarantee suggests.

2. Problem Specification

We first model the underlying decision process. Suppose $X = \left\{[x^{(i)}, x^{(i+1)}) : i = 0, \ldots, m\right\}$ denotes a partition of the interval $[x^{(0)}, x^{(m+1)})$. We assume $x^{(i)} < x^{(i+1)}$ for all i. Let $N = (n^{(0)}, n^{(1)}, \ldots, n^{(m)})$ represent the numbers of roots that lie in the corresponding subintervals in X. Together, the pair (X, N) determine the *computational state*. Suppose at decision epoch j, our current computational state is (X_j, N_j). We choose a refined partition

$$X_{j+1} = \left\{ \left[x_j^{(0)}, x_j^{(1)}\right), \ldots, \left[x_j^{(\ell)}, \bar{x}_{j+1}\right), \left[\bar{x}_{j+1}, x_j^{(\ell+1)}\right), \ldots, \left[x_j^{(j)}, x_j^{(j+1)}\right) \right\},$$

where $\ell \in \{0, \ldots, j\}$ and $\bar{x}_{j+1} \in (x_j^{(\ell)}, x_j^{(\ell+1)})$. Accordingly, let $\mathbb{X}(X_j)$ be the set containing all refined partitions of X_j containing $j+1$ subintervals. We then evaluate f at $\bar{x}_{j+1}$. For all elements $s \in \mathcal{S}$ we observe $f(s, \bar{x}_{j+1})$, and therefore we can determine whether $x^*(s)$ is less than or greater than $\bar{x}_{j+1}$. At this point, the $n_j^{(\ell)}$ roots in the original subinterval $\left[x_j^{(\ell)}, x_j^{(\ell+1)}\right)$ are split among the two newly-created subintervals. Hence, we have

$$N_{j+1} = \left(n_j^{(0)}, \ldots, n_j^{(\ell-1)}, \bar{N}_{j+1}, n_j^{(\ell)} - \bar{N}_{j+1}, n_j^{(\ell+1)}, \ldots, n_j^{(j)}\right),$$

where $n_{j+1}^{(\ell)} = \bar{N}_{j+1}$ and $n_{j+1}^{(\ell+1)} = n_j^{(\ell)} - \bar{N}_{j+1}$. All other components in N_j remain the same because we learn nothing new about roots in the other subintervals.

A priori, we assign a prior distribution to the location of $x^*(s)$ for every element $s \in \mathcal{S}$. For simplicity, we assume that for every s, independent of all else, $x^*(s) \sim \text{Unif}[0, 1)$. Otherwise, as long as under the prior distribution the root locations are i.i.d. and absolutely continuous with respect to Lebesgue measure, we can use an inverse mapping to appropriately stretch the real line to yield the above case. Therefore, *a priori*, $\bar{N}_{j+1} \sim \text{Binomial}\left(n_j^{(\ell)}, (\bar{x}_{j+1} - x_j^{(\ell)})/(x_j^{(\ell+1)} - x_j^{(\ell)})\right)$. Since we would like to find an ordering for $x^*(\cdot)$, we stop evaluating f when every subinterval in the partition X contains at most one root, i.e., we stop when $n^{(i)} \leq 1$ for all $i \in \{0, \ldots, |N| - 1\}$. Define the stopping time $\tau = \inf\{j \in \mathbb{N} : N_j^{(i)} \leq 1 \quad \forall i = 0, 1, \ldots, j\}$.

We would like to model this multiple root-finding problem as a dynamic program that finds the Bayes-optimal policy minimizing the expected number of evaluations of $f(\cdot, x)$ needed to find an ordering of all elements $s \in \mathcal{S}$ with respect to x^*. We define the value function for *computational effort* under policy π

$$W^\pi(X, N) = \mathbb{E}^\pi \left[\, \tau \,|\, X_0 = X,\, N_0 = N \right], \tag{1}$$

where π is a policy that maps computational states (X, N) to partitions $\bar{X} \in \mathbb{X}(X)$. The value function W^π counts the expected number of iterations needed to sort the elements $s \in \mathcal{S}$ with respect to x^* under policy π. We define the value function $W(X, N) = \inf_{\pi \in \Pi} W^\pi(X, N)$, where Π denotes the set of all policies π.

3. Recursion

Using the original definition of the value function in (1), we can derive a recursion for the computational dynamic program. For a computational state pair (X, N) that do not satisfy the stopping conditions, we have that

$$W^\pi(X, N) = 1 + \mathbb{E}^\pi \left[W^\pi(X_1, N_1) \,|\, (X_0, N_0) = (X, N) \right],$$

where $X_1 = \pi(X_0, N_0)$ indicates the next refinement of the partition, and N_1 is the subsequent spread of elements among subintervals.

By the principle of optimality, we can iteratively take the best partition X_1 over each step, which gives us a recursion for W, the value function under the optimal policy. Because the process $((X_j, N_j) : j \geq 0)$ is time-homogeneous, we can drop the subscripts and denote

$\bar{X}$ as the refinement of partition X, and $\bar{N}$ as the resulting distribution of elements among subintervals. Thus, we have

$$W(X,N) = 1 + \min_{\bar{X}\in\mathbb{X}(X)} \mathbb{E}\left[W(\bar{X},\bar{N}) \,\middle|\, X_0 = X,\ N_0 = N\right]. \tag{2}$$

3.1. Decomposition and Interval Invariance

We can greatly simplify this recursion by decomposing it by the different subintervals.

Theorem 1. *Under the optimal policy, the value function W has the following two properties.*

- *Decomposition:* $W(X,N) = \sum_{i=0}^{|N|-1} W\left(\left\{[x^{(i)}, x^{(i+1)})\right\}, n^{(i)}\right)$
- *Interval Invariance:* $W\left(\left\{[x^{(i)}, x^{(i+1)})\right\}, n^{(i)}\right) = W\left(\{[0,1)\}, n^{(i)}\right)$.

Proof. We will first prove the decomposition result. For any initial computational state (X,N), consider the following policy. For each subinterval $[x^{(i)}, x^{(i+1)})$, take an optimal policy for the computational state $\left(\left\{[x^{(i)}, x^{(i+1)})\right\}, n^{(i)}\right)$, and once we find a partition of the subinterval satisfying the stopping conditions, we do the same for another subinterval. Therefore, it must be $W(X,N) \le \sum_{i=0}^{|N|-1} W\left(\left\{[x^{(i)}, x^{(i+1)})\right\}, n^{(i)}\right)$.

Now we will prove the opposite inequality. First, note that the order we choose to further partition the subintervals is irrelevant, since we only seek to minimize the number of evaluations required, and each evaluation provides refinement only within its subinterval. Without loss of generality, consider only policies that evaluate the function with value within the leftmost subinterval that still does not satisfy the stopping conditions. Suppose this interval is $[x^{(i)}, x^{(i+1)})$ and contains the zeros of $n^{(i)}$ elements. Before we are allowed to move to the next subinterval, we must find a partition of $[x^{(i)}, x^{(i+1)})$ that satisfies the stopping conditions. By definition, this takes a minimum of $W\left(\left\{[x^{(i)}, x^{(i+1)})\right\}, n^{(i)}\right)$ steps. Since we only evaluate the function at one value at a time, we perform one evaluation on exactly one subinterval at each step. Therefore, repeating the same logic for every subinterval tells us $W(X,N) \ge \sum_{i=0}^{|N|-1} W\left(\left\{[x^{(i)}, x^{(i+1)})\right\}, n^{(i)}\right)$.

We will now prove the second claim of the theorem using a pathwise argument. Suppose we have initial computational state $(X_0, N_0) = \left([a,b), n^{(i)}\right)$. Define the operator $T((X,N)) = ((b-a)X + a, N)$. If we define $\left(\tilde{X}_j, \tilde{N}_j\right) = T^{-1}(X_j, N_j)$ for all time epochs j, there exists a one-to-one mapping between computational states. For any sample path of the process $((X_j, N_j) : j \ge 0)$ which reaches the stopping conditions at time epoch t, it must be that $(\tilde{X}_t, \tilde{N}_t)$ also satisfies the stopping conditions. Therefore, it must be that $W(\tilde{X}_0, \tilde{N}_0) \le W(X_0, N_0)$. Symmetry gives the opposite inequality, and hence the result. □

It may seem strange that the recursion relation does not depend on the size of the interval. In fact, it only depends on the number of elements in each sub-interval, because we are only concerned with finding the correct *ordering*.

The decomposition is helpful both when solving the dynamic program and describing the optimal policy. Since the value function is additive among subintervals, the order in which we refine the partition does not affect the optimal number of evaluations of f. Thus, we can focus our attention on solving a one-dimensional dynamic program and without loss of generality solely consider the subinterval $[0,1)$.

3.2. Simplified Recursion

Since we can write the value function W in terms of its smaller subintervals, we can just consider the special case where the partition $X = \{[0, 1)\}$. In a slight abuse of notation, we define $W(n) = W(\{[0, 1)\}, n)$ and have

$$W(n) = 1 + \min_{x \in (0,1)} \mathbb{E}\left[W(N_x) + W(n - N_x) \right], \tag{3}$$

where $N_x \sim \text{Binomial}(n, x)$, independent of all else. Intuitively, we choose a point $x \in (0, 1)$ to evaluate the original dynamic program, and the n elements get split among the two newly-created sub-intervals. As before, we have the stopping conditions $W(0) = W(1) = 0$. Computationally, we cannot use this recursion to solve for $W(\cdot)$ explicitly, since N_x can equal 0 or n with positive probability, causing $W(n)$ to appear on both sides of (3). Proposition 2 accounts for this.

Proposition 2.

$$W(n) = \min_{x \in (0,1)} \left\{ \frac{1}{1 - x^n - (1 - x)^n} + \mathbb{E}\left[W(N_x) + W(n - N_x) \,\middle|\, 1 \le N_x \le n - 1 \right] \right\}. \tag{4}$$

Proof. From (3), for any $x \in [0, 1]$,

$$\begin{aligned} W(n) \le 1 &+ \mathbb{E}\left[W(N_x) + W(n - N_x) \mid N_x \in [1, n - 1] \right] \cdot \mathbb{P}\left(N_x \in [1, n - 1] \right) \\ &+ \left(W(n) + W(0) \right) \mathbb{P}\left(N_x = n \right) + \left(W(0) + W(n) \right) \mathbb{P}\left(N_x = 0 \right), \end{aligned}$$

with equality for some $x \in [0, 1]$ (since the interval is compact and the right side is continuous in x). Since $W(0) = 0$, we get

$$\begin{aligned} W(n) \le 1 &+ \mathbb{E}\left[W(N_x) + W(n - N_x) \mid N_x \in [1, n - 1] \right] \cdot \mathbb{P}\left(N_x \in [1, n - 1] \right) \\ &+ W(n) \left(1 - \mathbb{P}\left(N_x \in [1, n - 1] \right) \right), \\ \text{i.e.,} \qquad W(n) \le &\frac{1}{\mathbb{P}\left(N_x \in [1, n - 1] \right)} + \mathbb{E}\left[W(N_x) + W(n - N_x) \,\middle|\, N_x \in [1, n - 1] \right]. \end{aligned}$$

This inequality is tight for the same x that made the previous inequality tight. Using $N_x \sim \text{Binomial}(n, x)$ gives the result. □

This recursion reveals the structure behind the coupled bisection algorithm. Suppose we have an interval that contains n elements. There are two possibilities when evaluating f at the next value of x: (i) splitting the interval into two subintervals, one of which contains all n elements, and (ii) splitting into two subintervals, both of which contain at least one element each. In case (i), we would have to perform the same procedure again on that smaller subinterval. The first term in (4) is exactly equal to the expected number of iterations it takes to break free of this loop. Case (ii) corresponds with the conditional expectation term in the same equation. Thus, the choice of x is a balancing act between limiting the iterations spent on the initial split and obtaining a desirable spread of elements in subintervals after that split occurs.

4. Bisection Policy

Consider the bisection policy $\beta \in \Pi$, where we choose to evaluate the function at the midpoint of the interval, i.e., choosing $x = 1/2$ for all n in the recursion (3). This yields

$$W^{\beta}(n) = 1 + 2\ \mathbb{E}[\, W^{\beta}(N_{1/2})\,], \tag{5}$$

where $N_{1/2} \sim \text{Binomial}(n, 1/2)$, with stopping conditions $W(0) = W(1) = 0$.

4.1. Greedy Shannon Entropy Reduction

The bisection policy β is a good candidate policy because it is intuitive and easy to implement. But there is also a metric where bisection is optimal. If our goal is to sort elements with respect to their zeros, we can view the problem as trying to find the correct permutation of elements among the $|\mathcal{S}|!$ possibilities. Since we assume *a priori* that the roots of all elements are i.i.d. uniformly distributed throughout $[0, 1)$, every permutation of elements in $\mathcal{S}$ is equally likely.

We define $H(X, N)$ to be the Shannon entropy of the distribution of possible permutations of $\mathcal{S}$ at computational state (X, N). In this case, since we are considering a uniformly discrete distribution, this is equivalent to

$$H(X, N) = \log_2 \left(\prod_{i=0}^{|N|-1} n^{(i)}! \right), \tag{6}$$

where the term inside the logarithm denotes the number of permutations of roots consistent with computational state (X, N). The first observation is that the entropy of a given computational state (X, N) does not depend on the partition X, but only on the number of zeros in each partition. Also, because of the convenient properties of the logarithm, the decomposition property comes for free, giving us

$$H(X, N) = \sum_{i=0}^{|N|-1} H(\{[x^{(i)}, x^{(i+1)})\}, n^{(i)}).$$

For the same reasons given for decomposing W, we only need to consider one subinterval at a time. Therefore, we can assume without loss of generality that we start with computational state $([0, 1), n)$. Similarly to W, we define $H(n) = H(\{[0, 1)\}, n)$. We would like to maximize the expected entropy reduction, meaning

$$\max_{x \in (0,1)} H(n) - \mathbb{E}\left[H(N_x) + H(n - N_x)\right], \tag{7}$$

where $N_x \sim \text{Binomial}(n, x)$.

Theorem 3. *An optimal solution for* (7) *is* $x^* = 1/2$ *for all* $n \geq 2$*, implying that the bisection policy maximally reduces entropy, given a single function evaluation, among possible permutations of elements in* $\mathcal{S}$*.*

Proof. The objective function in (7) is symmetric about $x = 1/2$, because $n - N_x \overset{d}{=} N_{1-x}$. If we can show that it is concave in x, then we are done. We know that $H(n) = \log_2 n!$, and therefore, we can compactly write the optimization problem in (7) as

$$\max_{x\in(0,1)} \mathbb{E}\left[\log_2 \binom{n}{N_x}\right]. \tag{8}$$

First, we use a property of binomial coefficients, namely that $\left(\binom{n}{k} : k = 0, \ldots, n\right)$ is a log-concave sequence [10], and hence, $\left(\log_2 \binom{n}{k} : k = 0, \ldots, n\right)$ is a concave sequence. Now we invoke a useful property of distributions in exponential families. A random variable Z_θ parameterized by θ is *convexly parameterized* if for any convex function f, $\mathbb{E}[f(Z_\theta)]$ is convex in θ. Mean-parameterized exponential family distributions are convexly parameterized [11][12], and since the binomial distribution is a member of that family, it follows that (8) is concave in x. □

We showed bisection is an optimal policy with respect to the one-step reduction of the Shannon entropy of possible orderings of roots. Now we explore how well the bisection policy can reduce computational effort in W.

4.2. Minimizing Computational Effort

It is reasonable to conjecture that bisection is also an optimal policy for minimizing computational effort. However, for $n = 6$, solving the dynamic program W computationally with (4) reveals that the optimal choice of x is not the midpoint of the interval, but rather is located at $x_6^* = 0.5 \pm 0.037$, with an optimality gap of 2.58×10^{-5}. This is the first of many disparities between the bisection policy and the optimal policy in this setting.

To bound the optimality gap, we derive upper bounds on $W^\beta(\cdot)$ and lower bounds on $W(\cdot)$. We can show a rather crude lower bound for $W(\cdot)$. If we want to sort n elements, we must perform at least $n - 1$ evaluations of f to separate them, and induction on n with (4) confirms that $W(n) \geq n - 1$.

In comparison, what upper bounds can we derive for $W^\beta(\cdot)$? We have computational evidence suggesting that W^β grows linearly for large n, so we focus on bounding the linear growth rate. For a function g defined on the integers, let $\Delta g(n) = g(n + 1) - g(n)$. We will prove for large n that $\Delta W^\beta(n) \leq \gamma$ for some constant γ.

4.3. What's the difference?

To derive upper bounds, we use a structural result regarding the growth rate of W^β, which is proved using a coupling argument.

Lemma 4.

$$\Delta W^\beta(n) = \mathbb{E}\left[\, \Delta W^\beta(N_{1/2}) \,\right], \tag{9}$$

where $N_{1/2} \sim$ *Binomial*$(n, 1/2)$*, and with stopping conditions* $\Delta W^\beta(0) = 0$ *and* $\Delta W^\beta(1) = 2$*.*

Proof. We start with the recursion from (5) and take a difference, giving us

$$\Delta W^\beta(n) = 2\,\mathbb{E}\left[W^\beta(\hat{N}_{1/2}) - W^\beta(N_{1/2})\right],$$

where $N_{1/2} \sim \text{Binomial}(n, 1/2)$ and $\hat{N}_{1/2} \sim \text{Binomial}(n+1, 1/2)$. We can couple these two random variables since the expression involves only their expectation. Let $B_{1/2} \sim \text{Bernoulli}(1/2)$, independent of all else, and since $\hat{N}_{1/2} \overset{\text{d}}{=} N_{1/2} + B_{1/2}$, we have that

$$\begin{aligned}
\Delta W^{\beta}(n) &= 2\,\mathbb{E}\left[W^{\beta}(N_{1/2}+B_{1/2}) - W^{\beta}(N_{1/2})\right], \\
&= 2\,\mathbb{E}\left[\frac{1}{2}\left(W^{\beta}(N_{1/2}+1) - W^{\beta}(N_{1/2})\right) + \frac{1}{2}\left(W^{\beta}(N_{1/2}) - W^{\beta}(N_{1/2})\right)\right] \\
&= \mathbb{E}\left[\,\Delta W^{\beta}(N_{1/2})\,\right].
\end{aligned}$$

For the stopping conditions, we use the original stopping conditions for (3), which gives us $W^{\beta}(0) = W^{\beta}(1) = 0$. By direct calculation using (4), we find $W^{\beta}(2) = 2$ (which happens to correspond with W at $n = 2$). □

Here the growth rate $\Delta W^{\beta}(n)$ is a weighted average of all previous growth rates, suggesting that the sequence should converge. We use this idea to derive a method for calculating upper bounds on $W^{\beta}(n)$ for large but finite values of n, and conjecture these bounds hold for all n.

4.4. Upper Bounds

Before we show a computational method for deriving upper bounds on $\Delta W^{\beta}(n)$, we first prove a property of the Binomial distribution.

Lemma 5. *For non-negative integers ℓ and m such that $\ell \in [0, m-2]$, and for $n \geq m$, let $p_{\ell,m}$ be defined as*

$$p_{\ell,m}(n) = \mathbb{P}\left(N_{1/2}(n) \geq \ell + 1 \;\middle|\; N_{1/2}(n) \leq m - 1\right),$$

where $N_{1/2}(n) \sim$ Binomial$(n, 1/2)$. Then $p_{\ell,m}(n)$ is non-decreasing in n.

Proof. For any non-negative integer n, $p_{\ell,m}(n) \leq p_{\ell,m}(n+1)$ is equivalent to $N_{1/2}(n) \leq_{rh} N_{1/2}(n+1)$, where $\leq_{rh}$ refers to the *reverse hazard rate ordering* [13, p. 37]. By definition, for two discrete random variables U and V, $U \leq_{rh} V$ if

$$\frac{\mathbb{P}(U = n)}{\mathbb{P}(U \leq n)} \leq \frac{\mathbb{P}(V = n)}{\mathbb{P}(V \leq n)}, \tag{10}$$

for all natural numbers n. It is clear from (10) that $N_{1/2}(n) \leq_{rh} N_{1/2}(n)$. We also have $0 \leq_{rh} B_{1/2}$, where $B_{1/2} \sim \text{Bernoulli}(1/2)$, independent of all else.

Now we use the fact that reverse hazard rate ordering is closed under convolutions [13, p. 38], i.e., if we have random variables U_1, U_2 and V_1, V_2 such that $U_1 \leq_{rh} V_1$ and $U_2 \leq_{rh} V_2$, then $U_1 + U_2 \leq_{rh} V_1 + V_2$. Because $N_{1/2}(n) \leq_{rh} N_{1/2}(n)$ and $0 \leq_{rh} B_{1/2}$, we deduce $N_{1/2}(n) + 0 \leq_{rh} N_{1/2}(n) + B_{1/2}$, and since $N_{1/2}(n) + B_{1/2} \overset{d}{=} N_{1/2}(n+1)$, we get $N_{1/2}(n) \leq_{rh} N_{1/2}(n+1)$. □

Using (9), we now present computationally tractable upper bounds on the value of the bisection policy.

Theorem 6. *For some non-negative integer m, we define $g_m(n) = \max_{\ell \in [n, m-1]} \Delta W^\beta(\ell)$, and define h as*

$$h_m(n) = \mathbb{E}\left[g_m\left(N_{1/2}\right) \,\middle|\, N_{1/2} \leq m-1 \right],$$

where $N_{1/2} \sim$ Binomial$(n, 1/2)$. Suppose we have $\gamma_m > 0$ so that the following condition holds:

$$\max\left\{\Delta W^\beta(m), h_m(m)\right\} \leq \gamma_m. \tag{11}$$

Then for all $n \geq m$, it must be that $\Delta W^\beta(n) \leq \gamma_m$.

Proof. We use induction on $n \geq m$. The condition gives us the base case $n = m$. Now suppose that for all $k \in [m, n-1]$ that $\Delta W^\beta(k) \leq \gamma_m$. From (9),

$$\begin{aligned}
\Delta W^\beta(n) &= \mathbb{E}\left[\Delta W^\beta\left(N_{1/2}\right)\right] \\
&= \Delta W^\beta(n)\ \mathbb{P}\left(N_{1/2} = n\right) + \mathbb{E}\left[\Delta W^\beta\left(N_{1/2}\right) \,\middle|\, N_{1/2} \leq m-1\right]\ \mathbb{P}\left(N_{1/2} \leq m-1\right) \\
&\quad + \mathbb{E}\left[\Delta W^\beta\left(N_{1/2}\right) \,\middle|\, m \leq N_{1/2} \leq n-1\right]\ \mathbb{P}\left(m \leq N_{1/2} \leq n-1\right) \\
&\leq \Delta W^\beta(n)\ \mathbb{P}\left(N_{1/2} = n\right) + \mathbb{E}\left[g\left(N_{1/2}\right) \,\middle|\, N_{1/2} \leq m-1\right]\ \mathbb{P}\left(N_{1/2} \leq m-1\right) \\
&\quad + \gamma_m\ \mathbb{P}\left(m \leq N_{1/2} \leq n-1\right) \\
&= \Delta W^\beta(n)\ \mathbb{P}\left(N_{1/2} = n\right) + h_m(n)\ \mathbb{P}\left(N_{1/2} \leq m-1\right) + \gamma_m\ \mathbb{P}\left(m \leq N_{1/2} \leq n-1\right),
\end{aligned}$$

where the inequality is due to the definition of g in the first term and the inductive hypothesis in the second term. Because we can solve for $\Delta W^\beta(n)$, all that remains is to show $h_m(n) \leq \gamma_m$. Consider

$$\begin{aligned}
h_m(n) &= \mathbb{E}\left[g\left(N_{1/2}\right) \,\middle|\, N_{1/2} \leq m-1\right] \\
&= g(0) + \sum_{\ell=0}^{m-2} \Delta g\left(\ell\right) \cdot \mathbb{P}\left(N_{1/2} \geq \ell + 1 \,\middle|\, N_{1/2} \leq m-1\right) \\
&= g(0) + \sum_{\ell=0}^{m-2} \Delta g\left(\ell\right) \cdot p_{\ell,m}(n).
\end{aligned}$$

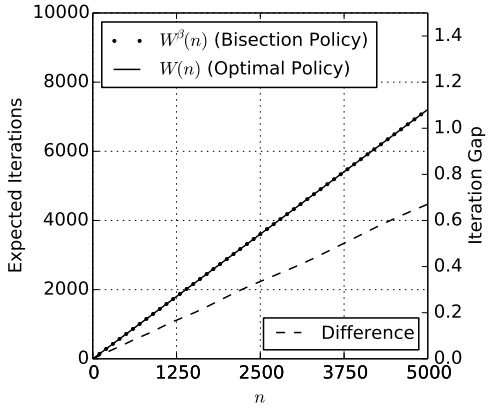
Since g is non-increasing, Δg is non-positive. Also, we proved in Lemma 5 that $p_{\ell,m}(n)$ is non-decreasing in n. Since $n \geq m$,

$$h_m(n) \leq g(0) + \sum_{\ell=0}^{m-2} \Delta g\left(\ell\right) \cdot p_m(m) = h_m(m) \leq \gamma_m,$$

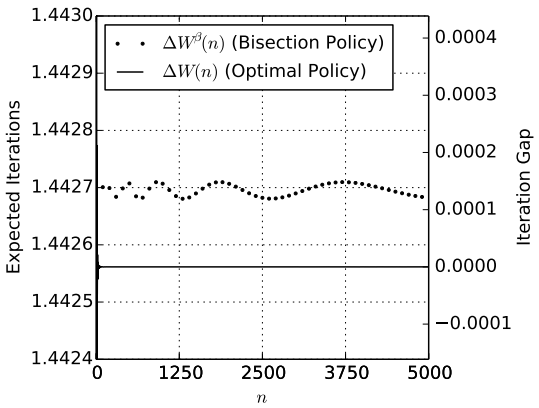
where the last inequality is true by the assumption. □

Theorem 6 is useful because we can compute ΔW^β for the first m terms using (9), then use the above result to find an arbitrarily tight bound on the policy for all $n \geq m$. The condition in (11) can be verified directly with the computed values of $W^\beta(n)$ for $n \in [0, m-1]$.

Figure 1: Performance of the Bisection Policy



(a) The values $W^\beta(n)$, $W(n)$ and their difference for $n \leq 5000$. The bisection policy is close to optimal in the number of expected iterations required, enough for W^β and W to coincide above, as further evidenced by the scale in panel (b).



(b) The rates $\Delta W(n)$ and $\Delta W^\beta(n)$ for $n \leq 5000$. Although we show theoretically that $\Delta W(n) \geq 1$, it appears that $\Delta W(n)$ is bounded below by a larger constant for large n, and the true gap in rates is closer to 0.0001.

5. Computational Results and Concluding Remarks

Using Theorem 6, and choosing appropriate values for m, we computationally derive upper bounds on the bisection policy. In general, the bounds improve as m increases. Choosing $m = 15$, we find that the linear growth rate of the expected number of iterations required under the bisection policy is bounded above by $\gamma_{15} = 1.4440$, compared to the lower bound of 1 shown earlier on the growth rate of the optimal policy. This implies that, as n approaches infinity, the ratio of the expected number of iterations required under the two policies is bounded above by 1.4440.

In fact, the gap in performance appears to even tighter. For $100 \leq n \leq 5000$, by using (4) to compute W directly, we empirically observe that $\Delta W(n) \geq 1.4425$, and the sequence seems to converge rather quickly, as shown in Figure 1b. Further, from Figure 1a, we see that the expected number of iterations required to sort n elements under the bisection policy is indistinguishably close to that under the optimal policy. This suggests that bisection is near-optimal, although proving this seems difficult. In any case, we have a linear-time algorithm for ordering elements by their associated zeros.

Acknowledgements

PIF was supported by NSF CMMI-1254298 and IIS-1247696, AFOSR FA9550-11-1-0083, FA9550-12-1-0200, and FA9550-15-1-0038, and the ACSF AVF.

SNP was supported by NSF CMMI-1254298 and AFOSR FA9550-11-1-0083.

SGH was supported by NSF CMMI-1200315.

References

[1] J. Gittins, D. Jones, A dynamic allocation index for the design of experiments, Progress in Statistics 2 (9).

[2] P. Whittle, Restless bandits: Activity allocation in a changing world, Journal of Applied Probability (1988) 287–298.

[3] J. Niño-Mora, Computing a classic index for finite-horizon bandits, INFORMS Journal on Computing 23 (2) (2011) 254–267.

[4] W. Hu, P. Frazier, J. Xie, et al., Parallel bayesian policies for finite-horizon multiple comparisons with a known standard, in: Simulation Conference (WSC), 2014 Winter, IEEE, 2014, pp. 3904–3915.

[5] X. Zhao, P. I. Frazier, A markov decision process analysis of the cold start problem in bayesian information filtering, arXiv preprint arXiv:1410.7852.

[6] S. Dayanik, W. Powell, K. Yamazaki, Index policies for discounted bandit problems with availability constraints, Advances in Applied Probability (2008) 377–400.

[7] K. D. Glazebrook, C. Kirkbride, J. Ouenniche, Index policies for the admission control and routing of impatient customers to heterogeneous service stations, Operations Research 57 (4) (2009) 975–989.

[8] T. R. Shrout, S. J. Zhang, Lead-free piezoelectric ceramics: Alternatives for pzt?, Journal of Electroceramics 19 (1) (2007) 113–126.

[9] R. Castro, R. Nowak, Active learning and sampling, in: Foundations and Applications of Sensor Management, Springer, 2008, pp. 177–200.

[10] R. P. Stanley, Log-concave and unimodal sequences in algebra, combinatorics, and geometrya, Annals of the New York Academy of Sciences 576 (1) (1989) 500–535.

[11] M. Shaked, On mixtures from exponential families, Journal of the Royal Statistical Society. Series B (Methodological) (1980) 192–198.

[12] T. Schweder, On the dispersion of mixtures, Scandinavian Journal of Statistics (1982) 165–169.

[13] M. Shaked, J. G. Shanthikumar, Stochastic orders, Springer Science & Business Media, 2007.