Recruitment Strategies That Take a Chance

Anonymous Author(s)
Affiliation
Address
email

Abstract

In academic recruitment settings, including faculty hiring and PhD admissions, committees aim to maximize the overall quality of recruited candidates, but there is uncertainty about whether a candidate would accept an offer if given one. Previous work has considered algorithms that make offers sequentially and are subject to a hard budget constraint. We argue that these modeling choices may be inconsistent with the practice of academic recruitment. Instead, we restrict ourselves to a single batch of offers, and we treat the target number of positions as a soft constraint, so we risk overshooting or undershooting the target. Specifically, our objective is to select a subset of candidates that maximizes the overall expected value associated with candidates who accept, minus an expected penalty for deviating from the target. We first analyze the guarantees provided by natural greedy heuristics, showing their desirable properties despite the simplicity. Depending on the structure of the penalty function, we further develop algorithms that provide fully polynomial-time approximation schemes and constant-factor approximations to this objective. Empirical evaluation of our algorithms corroborates these theoretical results.

1 Introduction

Anyone who has served on a faculty hiring committee or a PhD admissions committee knows that a successful outcome requires resolving the tension between two competing goals. On the one hand, some candidates are (perceived to be) better qualified than others, and the aim is to recruit the best candidates. On the other hand, there is a given number of positions to be filled, and while there is typically some flexibility, there is a real cost to recruiting too many or too few people. The tension arises because the stronger a candidate is, the more likely they are to receive multiple attractive offers and the less likely they are to accept any particular offer. In order to manage uncertainty, a good strategy may involve a mix of offers to stellar candidates and "safer" candidates.

To formalize this problem, we assume that we have access to two numbers for each candidate i: their value x_i and their probability p_i of accepting an offer. We acknowledge that in current practice, these numbers are not explicitly estimated. However, committees typically rank or assign numerical scores to candidates based on their strength or fit, and savvy committees roughly estimate recruitment chances by classifying candidates as, say, "high yield," "low yield" or "extremely low yield", for example, by past experience or assistive computational tools [12, 1]. Therefore, we believe that the gap between current practice and explicit value and probability estimates is not large.

Our approach builds on the work of Purohit et al. [11], who framed hiring under uncertainty as an optimization problem. In their basic model, there are n candidates (each associated with a value and probability), k positions, and t time steps. In each time step the algorithm may make an offer to a single candidate and receive a response, that is, at most t sequential offers can be made, and the budget of k cannot be exceeded. The goal is to maximize the expected value of candidates who accept offers. Purohit et al. also consider the setting where the algorithm may make parallel offers

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

in each round. For both problems, they develop polynomial-time, constant-factor approximation algorithms (with approximations ratios of 2 and 8, respectively).

This problem formulation captures key aspects of academic recruitment, but, in our view, it does have two shortcomings. First, in a sense it is overcomplicated, as computational hardness stems from the assumption that offers are made sequentially: Even with parallel offers, if there is a single time step $(t = 1)$ then it is optimal to make offers to the k candidates with largest $p_i \cdot x_i$. But, in our experience of faculty hiring and PhD admissions in several universities, offers are typically made in one batch. Indeed, delayed offers (in the case of faculty hiring) and waitlists (in the case of PhD admissions) are usually avoided as they negatively impact yield. Waitlists also increase the mental stress of the students that are placed on these waitlists.

The second, and more crucial, shortcoming is that Purohit et al. [11] consider the constraint of hiring k candidates as hard. Again, this is inconsistent with our experience: Offers are made so that the expected yield roughly matches a desired target, but some faculty hiring or PhD admission cycles are "too successful," in the sense that the number of candidates who accept their offers is much larger than expected. This has a real cost: in the case of PhD students, it creates difficulties in finding funding and advisors, and in the case of faculty hiring, it may precipitate a shortage of resources with long-term impacts on future hiring and even tenure. For example, in one of our institutions, a faculty hiring cycle with yield that was much higher than expected led to the cancellation of the subsequent year's search.

Let us, therefore, reformulate the problem of hiring under uncertainty in a way that avoids both issues, that is, offers are made in a single batch and the target number of positions k is treated as a soft constraint. Specifically, a penalty is incurred for deviating from the target number of positions; we consider several different options for the penalty function. The optimization problem, therefore, is this:

> *Select a subset S of candidates that maximizes overall expected reward $\sum_{i \in S} p_i \cdot x_i$ minus expected penalty for deviating from the target number of positions.*

Enumerating all possible subsets S may be practicable in the case of faculty hiring in small departments. However, a brute-force approach will not work for this purpose in larger departments, or at the scale of PhD admissions even in smaller programs, which motivates our search for good algorithms.

Our results. We first consider a simplified case where the goal is to solely minimize the penalty term of our objective (irrespective of the rewards), and show that the greedy algorithm selecting candidates in the decreasing order of probabilities is optimal (Section 3.1).

The full objective is considerably more complex, and we analyze it under two natural penalty functions. When the penalty function is the mean squared error from the target, we show that the optimization problem is weakly NP-hard and provide a fully polynomial-time approximation scheme (FPTAS). When the penalty is linear in the extent to which the target is exceeded (that is, a linear penalty is incurred for overshooting, not undershooting), we show that two greedy heuristics — picking in the decreasing order of the value x_i and the expected value $p_i x_i$ — provide approximations to the optimal solution that are polynomial in the minimum probability p_{min}. We then present a constant-factor approximation algorithm that runs in polynomial time for a fixed p_{min}, thereby improving upon the greedy heuristics.

Finally, we carry out experiments on synthetic data (Section 4), focusing on L_1^+ loss. We observe that the two greedy heuristics perform reasonably well, especially if the values and probabilities are positively correlated. At the same time, compared to the greedy heuristics, the constant-factor approximation algorithm adapts to the specific instances better, especially when the correlation is negative. These numerical experiments corroborate our theoretical results that the greedy heuristics provide reasonable guarantees, justifying their use in practice for both simplicity and good performance. However, in some regimes the additional flexibility of the constant-factor approximation algorithm may be worth the complexity overhead.

Related work. In practice, the challenge of uncertainty in admissions is mitigated by practices such as admitting students in multiple rounds, using a waitlist, and using a rolling process [9]. There are also assistive computational tools that predict student yield rate with machine learning [12, 1]. There are a few theoretical formulations that model and address the uncertainty in such problems.

Purohit et al. [11] considers an online setting where the acceptance of each candidate is revealed immediately if the candidate is given an offer at that timestep, and analyzes the optimal ordering to give the candidates offers to subject to a hard constraint on the total number of acceptances. Ganguly et al. [7] considers a setting with multiple rounds where the yield rate in each round is either q_L or q_H (with $q_L < q_H$). To model the negative correlation between the student quality and the probability of acceptance, the authors assume the probability of the yield rate being q_H is linear in the number of students admitted. They subsequently derive a decision tree that computes the number of admissions to make in each round. For one batch selection, Zhang and Pippins [13] analyze the optimal number of applicants to admit using tools from yield management, assuming that each applicant has the identical value and probability. A series of work models the admissions problem as a decentralized matching market [4, 5], where the uncertainty in acceptance is modeled by the students' stochastic preferences over multiple schools. An objective combining the utility and the accepted size is considered, but the penalty is on the expected size, not counting the variance involved as opposed to ours.

Our proposed formulation is closely related to the knapsack problem and its myriad variants. In the stochastic knapsack problem, each item has a deterministic value and an independent stochastic size; the actual size of an item is revealed only after it is selected [8, 6, 2]. For the "one-shot" version of this problem, the aim is to choose a subset maximizing the expected value of the realized items, such that the probability that these realizations violate the knapsack constraint is below some threshold. In contrast, our "items" have equal size, but we pay a penalty which is some function of our realized distance from our knapsack target size, exchanging the constraint for a mixed-sign objective. In this spirit, the one-sided loss functions we consider are similar to objectives which arise in the penalty method for solving constrained optimization problems [10], though our setting is stochastic and we do not aim to ultimately satisfy a hard constraint.

2 Problem Formulation

We begin with some n items (corresponding to candidates) with associated values $x_1, \ldots, x_n \in \mathbb{R}$. If we select an item $i \in [n]$, there is a probability $p_i \in [0, 1]$ that we receive this item (the candidate accepts the offer). We write these values and probabilities as vectors $x \in \mathbb{R}^n$ and $p_i \in [0, 1]^n$. Let $Z_i \in \{0, 1\}$ be the indicator variable that we receive item i if it is selected, sampled as $Z_i \sim \text{Ber}(p_i)$. We assume that the events that we receive each individual item are independent. Let $S_Z \subseteq S$ denote the random realization of received items, that is, $S_Z \coloneqq \{i \in S : Z_i = 1\}$. Our goal is to select a subset $S \subseteq [n]$. First, we consider the reward for a subset S as the expected total value obtained:

$$R(S) \coloneqq \mathbb{E}\left[\sum_{i \in S} Z_i x_i\right] = \sum_{i \in S} p_i x_i$$

At the same time, let $M \in \mathbb{N}_+$ denote a target size that we want to achieve. We want to control the variance on the deviation between the realized size of S_Z, which is $|S_Z| = \sum_{i \in S} Z_i$, and the target size M. We consider this penalty as

$$V(S) \coloneqq \mathbb{E}\left[\rho\left(|S_Z|,\ M\right)\right],$$

where $\rho \colon \mathbb{N} \times \mathbb{N}_+ \to \mathbb{R}_{\geq 0}$ is a loss function to be specified later. Combining the two parts, we define the overall objective as

$$U(S) \coloneqq R(S) - \lambda \cdot V(S), \tag{1}$$

where $\lambda \in \mathbb{R}_+$ is a hyperparameter that governs the importance of the penalty relative to the reward. Our goal is to find the subset that maximizes the overall expected utility:

$$S^* \in \operatorname*{arg\,max}_{S \subseteq [n]} U(S).$$

We consider a range of choices of loss function ρ. It is worth noting that since $U(S)$ is a mixed-sign objective, the optimal value of $U(S^*)$ may be negative depending on the instance and choice of ρ.

Given a target size M, it is natural for ρ to be a convex function minimized at M, which penalizes any deviation from M, or alternatively a monotone convex function that is nonzero above M, which can be seen as penalizing violation of the budget constraint. We focus in particular on one- and two-sided linear and quadratic losses, which are formally introduced in Section 3.2 below.

3 Theoretical Results

To begin we note that if we only consider the reward term and set the penalty term as $V(S) \equiv 0$, then the solution is to trivially select all items. In what follows, we first discuss the other extremal case, taking $R(S) \equiv 0$ and considering the penalty V in isolation. These may be viewed as the cases when $\lambda = 0$ and $\lambda \to \infty$. We then turn to the general objective and consider both terms jointly.

3.1 Warm-Up: Penalty Only

To gain intuition for this problem, we first consider the simplified case where the goal is only to minimize the penalty term. Note that in this case our objective is strictly non-positive. When there are multiple solutions that minimize the penalty term, we define the set of optimal solutions as the ones with the maximal reward.

Algorithm 1 PGREEDY

Require: $x \in \mathbb{R}^n, p \in [0,1]^n$
$S \leftarrow \emptyset$.
Sort $i \in [n]$ lexicographically in decreasing order according to (p_i, x_i)
for $i = 1, 2, \ldots, n$ **do**
  if $U(S \cup \{i\}) \geq U(S)$ **then**
    $S \leftarrow S \cup \{i\}$
  else
    break
return S

We consider PGREEDY, the greedy algorithm with respect to p_i (Algorithm 1). In words, PGREEDY selects items in their decreasing order of probabilities, and if there are multiple items with the same probability it adopts the tie-breaking strategy that selects items in the decreasing order of their value x_i. The algorithm keeps selecting the next item defined by this order, and terminates when adding the next item would decrease the objective. As usual this greedy algorithm is efficient, since the stopping criterion can be checked in polynomial time given access to ρ. Note that the item values x are only used for tie-breaking. Surprisingly, PGREEDY is optimal for minimizing $V(S)$ in isolation.

Proposition 1. *Let $M \in \mathbb{N}_+$ be any target size. For any convex loss $\rho(\cdot, M)$,* PGREEDY *(Algorithm 1) yields an optimal solution to minimizing the penalty* $\min_{S \subseteq [n]} V(S)$.

The proof of this proposition is provided in Appendix D.1. Note that this result is not obvious, as one may expect that as the sum of probabilities of all selected items so far gets closer to the target size, it may be better to select an item with lower probability than an item with higher probability to "fill the gap." This is not true. Intuitively, it is because the realization of each Z_i is binary, so the outcome of adding another item i into the selection is either we add this item (with probability p_i) or not (with probability $1 - p_i$). If adding this item gives lower penalty, then we desire to add the item with the largest probability possible.

3.2 The General Objective

We now turn to the general objective. At the outset it bears noting that $U(S)$ is submodular in S whenever ρ is convex, as is the case for the loss functions we consider.

Remark 1. *Suppose $\rho(\cdot, M)$ is convex, then $U(S)$ is submodular in S.*

The submodularity is used for analyzing $\rho = L_1^+$ (in Section 3.2.2 and the proofs in Appendix C). Unfortunately the existing body of work on (non-monotone) submodular maximization cannot be deployed to obtain a general-purpose approximation to $U(S)$, since U is mixed-sign and may be negative even at optimality, and applying an affine transformation in order to engineer nonnegativity in general destroys any approximation guarantees.

We focus on a few natural choices of the loss function. First, we consider ρ given by linear and quadratic losses, which we denote by L_1 and L_2 respectively. These yield penalty terms $V(S)$ which are equal to the mean average error (MAE) and mean squared error (MSE) for the realized size of the

subset S_Z. We also consider the corresponding one-sided losses, which are

$$L_1^+(|S_Z|, M) := \begin{cases} |S_Z| - M & \text{if } |S_Z| \geq M \\ 0 & \text{otherwise,} \end{cases} \qquad \text{and} \qquad L_2^+(|S_Z|, M) := L_1^+(|S_Z|, M)^2.$$

All of these losses considered penalize the case where the realized size is greater than the target size. In applications such as admissions and hiring, there is a limited, pre-specified amount of resources allocated to the newly admitted or hired people. Hence, having more people than the target size is not desired. At the same time, the two-side losses give explicit preference that the realized size should also not be smaller than the target size. This explicit penalty for undershooting could stem from a hit to morale (an unsuccessful recruitment cycle really is demoralizing) or insufficient manpower for required tasks such as teaching certain courses. The one-side loss functions also implicitly capture this consideration, as there is an implicit opportunity cost described by the reward term when fewer candidates accept.

3.2.1 An FPTAS for L_2 Loss

Recall that we have the solutions in both extremal cases (when only considering the reward term or the penalty term), so one might hope that some interpolation between them could solve the general case. However, the general case is more complicated. In particular XGREEDY and XPGREEDY are greedy algorithms analogous to Algorithm 1 which select $i \in [n]$ in decreasing order of their value x_i (with ties broken in the decreasing order of p_i) and expected reward $x_i p_i$, respectively. Despite these heuristics being natural and intuitive, in our setting they perform arbitrarily poorly even for the MSE-reminiscent loss L_2.

To see this, consider the limiting case above for which PGREEDY is optimal. When λ becomes large relative to the values x_i, the variance that i with smaller probabilities p_i confer to the loss term will become more important relative to their expected reward, and XGREEDY and XPGREEDY can be made to choose items with smaller p_i than they should. At the same time, PGREEDY may be induced to incur only arbitrarily small positive value, even when large positive value is attainable. For a more formal demonstration, see Appendix A.

Despite the failure of heuristic approaches, when the chosen loss function is $\rho = L_2$, our problem can in fact be approximated up to a negligible error. For this loss, our full objective (1) may be written as

$$U(S) = \sum_{i \in S} p_i x_i - \lambda \cdot \mathbb{E}\left[\left(\sum_{i \in S} Z_i - M\right)^2\right]. \tag{2}$$

Letting $b_i \in \{0, 1\}$ be the binary decision variable for whether $i \in S$, that is, $b_i := \mathbb{1}\{i \in S\}$, the optimization problem then becomes

$$\underset{S \in [n]}{\arg\max}\ U(S) = \underset{b \in \{0,1\}^n}{\arg\max}\left(\sum_{i \in [n]} b_i p_i x_i - \lambda \cdot \mathbb{E}\left[\left(\sum_{i \in [n]} b_i Z_i - M\right)^2\right]\right). \tag{3}$$

Expanding Eq. (3) yields a collection of terms which are constant, linear, and quadratic in the s_i, and so the objective can be reformulated as an unconstrained binary quadratic program (UBQP). Although UBQP is strongly NP-hard, Çela et al. [3] present a pseudo-polynomial algorithm for UBQP when the coefficient matrix for the quadratic form in the objective has constant rank. By proving that the objective (2) is sufficiently insensitive to small changes in our problem parameters, we may leverage this pseudo-polynomial time algorithm to derive a FPTAS for approximating the optimal objective value for our problem. A standard search-to-decision reduction then yields the following result.

Theorem 1. *For $\rho = L_2$, there exists an algorithm that identifies some $S \subseteq [n]$ satisfying $U(S) \geq U(S^*) - \epsilon$ in time $poly(1/\epsilon, n, \lambda, x_{max})$.*

Here $x_{\max}$ is the maximum value x_i over all $i \in [n]$. On the other hand, by a reduction from equipartition we have the following hardness:

Theorem 2. *For $\rho = L_2$, optimizing $U(S)$ is weakly NP-hard.*

The proofs are provided in Appendix B. The hardness landscape of our problem when $\rho = L_2$ is therefore similar to that of the knapsack problem, which is heartening since the knapsack problem is relatively tractable in practice.

By contrast, the one-sided loss L_2^+ is not so well-behaved. In this case the objective does not admit a quadratic factorization in terms of decision variables, and although the objective is nonnegative, it is difficult to analyze the performance of the greedy algorithm or contend with the nonlinearity of the loss function in a principled way. We instead turn to L_1^+ loss, where surprisingly these obstacles can be overcome.

3.2.2 Approximations for L_1^+ Loss

The loss $\rho = L_1^+$ enables the possibility of a multiplicative approximation because the optimum value $U(S^*)$ of the mixed-sign objective is non-negative. More generally, any ρ with $\rho(0, M) = 0$ has a non-negative optimal value, since in this case $U(\emptyset) = 0$; for $\rho = L_1^+$ any singleton item i with positive x_i and p_i has strictly positive objective, since $M \geq 1$ implies that $L_1^+(1, M) = 0$ and so $U(\{i\}) = p_i x_i$.

The L_1^+ loss also appears likely amenable to a greedy algorithmic approach, since early items incur no penalty and the marginal penalty of adding a later item i is simply proportional to the probability that the current solution exceeds the target M. However, we find that natural greedy algorithms, while efficient, perform arbitrarily badly compared to the optimal solution in the worst case. Recall that one approach, XGREEDY, is to choose items in order by value. But if $M = 1$, items are of type $(x_a, p_a) = (1, p)$ or $(x_b, p_b) = (1, 1)$ and λ is large, then XGREEDY is only a p-approximation to $U(S)$.

Another greedy algorithm mentioned earlier is XPGREEDY, which chooses items i in the order of their expected value $x_i p_i$. Note that this is the marginal contribution of i to the reward term of the objective. But the instance given by $M = 1$, $\lambda = 1/p$, and types $(x_a, p_a) = (1/p, p)$ and $(x_b, p_b) = (1, 1)$ shows that XPGREEDY is also at best a p-approximation; for details see Appendix A.

These upper bounds on the performance of XGREEDY and XPGREEDY are intriguing in that they both scale like p, where p is a probability in the instance which becomes small. It suggests that the minimum probability p_{min} is a natural parameter for measuring the complexity of an instance with respect to $\rho = L_1^+$ loss. Surprisingly, the performance of these greedy algorithms can be lower-bounded in terms of p_{min} as well, under an additional assumption about the value of items.

Theorem 3. *If $\rho = L_1^+$ and there is some c for which $0 \leq x_i \leq (1-c)\lambda$, then* XPGREEDY *is a $\Theta(p_{min})$ approximation to $U(S^*)$. Furthermore,* XGREEDY *is between an $O(p_{min})$ and $\Omega(p_{min}^2)$ approximation to $U(S^*)$ in this setting.*

The proof of this theorem is provided in Appendix A.2. This is in notable contrast to when $\rho = L_2$, where multiplicative guarantees are ill-defined and XGREEDY and XPGREEDY may incur arbitrarily large additive loss. Theorem 3 also raises a natural question: is this $1/p_{\text{min}}$ threshold tight, or is it possible to efficiently attain better approximations to $U(S^*)$ which do not depend on p_{min}? We address this by introducing ONESIDEDL_1^+, presented in Algorithm 2, which attains a constant-factor approximation to the optimal solution. In Algorithm 2, for any vector $v \in \mathbb{R}^n$ and set $S \subseteq [n]$, we use $v|_S$ to denote the $|S|$-dimensional vector by restricting v to the coordinates indexed by S. Its runtime is parameterized by the minimum probability p_{min} and the ratio of the maximum value x_{max} to the penalty parameter λ; in particular for fixed p_{min} and λ/x_{max} it is polynomial in n.

At a high level, this algorithm proceeds first by dividing the items into three groups according to their values x_i, and considering each group in turn. Since U is submodular (Remark 1), the optimal solution within at least one of these groups will be constant-competitive with $U(S^*)$. We obtain a constant-factor approximation for each group in a different way.

For the items with low values bounded away from λ, LOWVALUEL_1^+ (Algorithm 4 in Appendix C) checks all small subsets, which succeeds if the optimal subset for this group is small. It also computes rounded probabilities and values (q_i, y_i) for each item in this group, and then efficiently computes the optimal solution according to this rounded instance. If the optimal subset is large, we then prove that this search over rounded solutions necessarily identifies a subset with objective value comparable to that of the optimal subset. This is the technical crux of proving that ONESIDEDL_1 is a constant-factor approximation.

For the items with values just below λ, MEDIUMVALUEL_1^+ (Algorithm 5 in Appendix C) returns the optimal subset if the group is small. If the group is large, it tries to choose a subset such that the expected number of realizations is about M; if there are not enough it chooses a subset with

approximately half the expected realizations of the group overall. Finally for the group of items with values above λ, it is easy to see that choosing the entire group is optimal. The pseudocode and related proofs for these algorithms appear in Appendix C. The following result provides a theoretical guarantee for Algorithm 2.

Algorithm 2 ONESIDEDL$_1^+$

Require: $x \in \mathbb{R}^n$, $p \in [0,1]^n$, $M \in \mathbb{N}_+$, $\lambda \geq 0$
Ensure: $S \subseteq [n]$ for which $U(S) \geq c \cdot U(S^*)$ for universal constant c

```
N_L ← {i ∈ [n] : x_i ≤ (1 − p_min/4) · λ}
N_M ← {i ∈ [n] : (1 − p_min/4) · λ < x_i < λ}
N_H ← {i ∈ [n] : x_i ≥ λ}
S_L ← LowValueL_1^+(x|_{N_L}, p|_{N_L}, λ, M)
S_M ← MediumValueL_1^+(x|_{N_M}, p|_{N_M}, λ, M)
S_H ← N_H
Compute U(S_L), U(S_M), and U(S_H)
return S ∈ {S_L, S_M, S_H} maximizing U(S)
```

Theorem 4 (Constant-factor approximation for L_1^+)**.** *Algorithm 2 is a constant-factor approximation to $U(S^*)$ which runs in time $n^{O\left(1/p_{min}^2 \log \frac{\lambda}{p_{min} x_{max}}\right)}$.*

Intuitively, the reason Algorithm 2 divides the items into cases depending on their values is to handle the case when the reward portion of $U(S^*)$ is almost equal to the penalty portion. This presents an impediment to the efficient strategy of solving a rounded version of the instance, since in this case the magnitude and even sign of $U(S^*)$ is potentially quite sensitive to changes in the p_i and x_i. By restricting attention to the low-value case, we may prove that the expected number of realized items in S^* is not much more than the threshold M; this then allows us to argue that there exist good rounded solutions.

We conclude our theoretical results with a surprising equivalence between one- and two-sided linear losses L_1^+ and L_1. In what follows, let U_{L_1} and $U_{L_1^+}$ denote the objective with $\rho = L_1$ and $\rho = L_1^+$, respectively. Consider the affine map between problem instances $\psi : (x, p, \lambda, M) \mapsto (x', p', \lambda', M')$ which takes $x'_i := x_i + \lambda$ and $\lambda' := 2\lambda$, and which leaves M and p unchanged. If S' denotes $S \subseteq [n]$ in the instance transformed by ψ, then we have the following:

Theorem 5 (Equivalence between L_1 and L_1^+)**.** *For all instances (x, p, λ, M) and for all $S \subseteq [n]$,*

$$U_{L_1}(S) = U_{L_1^+}(S') - \lambda \cdot M.$$

In particular, since λ and M do not depend on S, this implies that S maximizes U_{L_1} on (x, p, λ, M) if and only if it maximizes $U_{L_1^+}$ on (x', p', λ', M').

Although Theorem 5 establishes a correspondence between the solutions to our problem for $\rho = L_1$ and $\rho = L_1^+$, it is not approximation preserving and so does not convert ONESIDEDL$_1$ into an approximation algorithm for the two-sided setting. Indeed, as noted above, the optimal value when $\rho = L_1$ may be strictly negative.

4 Numerical Experiments

Having established worst-case theoretical guarantees, we wish to test how well our algorithms perform empirically. We focus on L_1^+ loss because our result for L_2 is an FPTAS, so we know its performance is arbitrarily close to optimal. All error bars shown in the plots represent standard error of the mean.

4.1 Experimental Setting

In constructing instances we follow the approach of Purohit et al. [11] in their use of beta distributions to orchestrate correlation between x_i and p_i. We therefore first draw $x_i \sim \text{Unif}[0, 1]$, and then produce three types of correlation as follows:

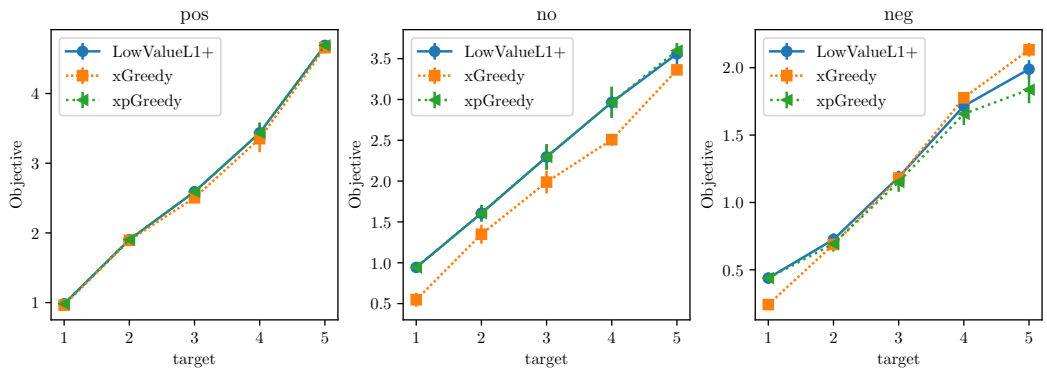

Figure 1: Sampling from the beta distribution with positive, no, and negative correlation. Here $n = 50$ and $\lambda = 3$.

- *Negative correlation:* $p_i \sim p_{\text{min}} + (1 - p_{\text{min}}) \cdot \text{Beta}(10(1 - x_i),\ 10x_i)$.
- *Positive correlation:* $p_i \sim p_{\text{min}} + (1 - p_{\text{min}}) \cdot \text{Beta}(10x_i\ 10(1 - x_i))$.
- *No correlation*: $p_i \sim \text{Unif}[p_{\text{min}}, 1]$.

This construction differs from the sampling paradigm of Purohit et al. [11] only in that we re-normalize the probabilities $\{p_i\}$, so that they are bounded in $[p_{\text{min}}, 1]$. We consider $n = 50$ and $p_{\text{min}} = 0.01$ throughout, and explore the greedy heuristics XGREEDY and XPGREEDY, as well as the constant-factor approximation algorithm ONESIDEDL$_1^+$ (Algorithm 2), for a range of M and λ. The lower bound on p_{min} ensures that the performance of the greedy heuristics and runtime of our algorithm are reasonable; a value of 0.01 is very realistic because in practice, if a candidate takes the time and effort to apply, it is reasonable to assume that they at least have some non-trivial probability to accept if they were given an offer. We also focus on the regime where $x_i < \lambda$, which is assumed by Theorem 3 and handled in Algorithm 2 by the subroutine LOWVALUEL$_1^+$. We believe that this is the main regime of practical interest: candidates with $x_i \geq \lambda$ are so exceptional that (under L_1^+ loss) we would want them to join regardless of how many candidates have already accepted offers, and such candidates are very rare indeed.

Note that our theoretical guarantees (Theorem 4) necessitate that all candidate solutions up to size $\tau = poly(1/p_{\text{min}})$ are checked by brute force. In this implementation of LOWVALUEL$_1^+$ we take $\tau = 0$ and isolate its search over rounded solutions. As we consider small target sizes M, this prevents LOWVALUEL$_1^+$ from outperforming the greedy algorithms simply by virtue of brute forcing over all relevant solutions. This only hinders the performance of LOWVALUEL$_1^+$. In fact, LOWVALUEL$_1^+$ only considers rounded solutions S satisfying $\sum_{i \in S} x_i p_i \leq 2M$. So long as $x_i < \lambda$, such a stopping condition may be deployed without loss of generality. For details see Appendix C. We believe this offers a favorable tradeoff between runtime and accuracy, and again serves as a lower bound to LOWVALUEL$_1^+$ as it is written.

4.2 Experimental Results

The objective values that our algorithms of interest attain for these distributions are shown in Figure 1. Note that positive correlation leads XGREEDY and XPGREEDY to pursue very similar (and optimal) strategies, as expected. This is intuitively the easier setting, and here LOWVALUEL$_1^+$ performs on par with the greedy heuristics. In the no-correlation and negative-correlation settings, there are regimes where one of the two greedy heuristics performs better than the other one, whereas LOWVALUEL$_1^+$ appears to perform as well as the better of two depending on the regimes, showing its better adaptivity across these instances in practice as well as in theory.

Negative correlation between x_i and p_i is of particular interest to us. It is also the regime that is the most relevant for the setting of faculty hiring and PhD admissions, and in fact hiring and admissions more broadly. In Figure 1, we also observe that negative correlation is the setting that displays the most heterogeneity in algorithm behavior. We therefore turn to this negative-correlation setting and explore the effect of increasing the penalty regularizer λ in Figure 2. In general, LOWVALUEL$_1^+$

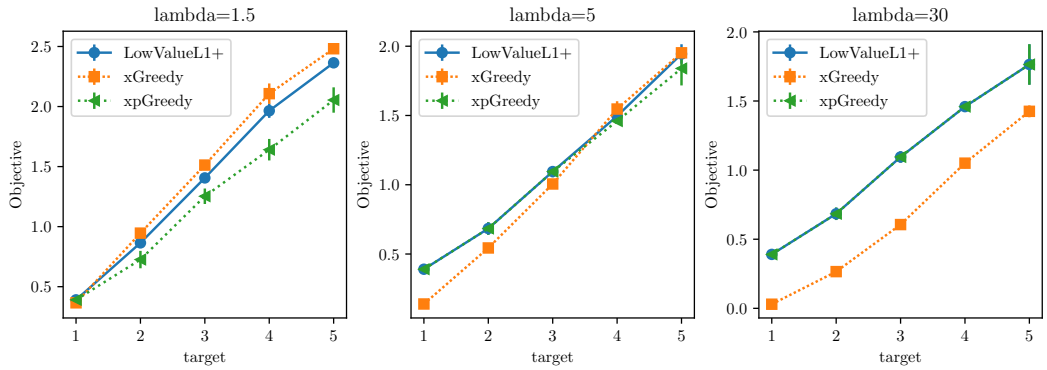

Figure 2: Performance for increasing penalty regularizer λ. Here $n = 50$ and sampling is via the negatively correlating beta distribution.

appears comparable to the better of the two greedy approaches across the values of M and λ that we examine, though there is a small gap between the objectives achieved by LowValueL$_1^+$ and xGreedy when $\lambda = 1.5$.

This is also good news for xGreedy and xpGreedy, because it suggests that the two of them together remain competitive across a wide range of instances. To the extent that LowValueL$_1^+$ falls short of $U(S)$, it is due to systematically rounding the probabilities p_i up by a constant factor when computing the prospective utility of solutions. Because its rounding preserves the reward term, such a systematic overestimate in the p_i leads it to overestimate the penalty term of any set under consideration. The presence of a large number of (p_i, x_i) with individually balanced but collectively large impact on the objective could therefore explain the extent to which LowValueL$_1^+$ lags behind xGreedy in Figure 2; the latter chooses many such individuals while the former judges their influence on the penalty to be too large. However, this can be mitigated by choosing smaller multiplicative bucket sizes for LowValueL$_1^+$, which is particularly effective in the case where the p_i of an instance fall in a small number of clusters or exhibit other structure.

5 Discussion

One of the takeaways from our theoretical and empirical results is that the greedy algorithm xGreedy, which makes offers to a subset of candidates with the highest values, is practicable for L_1^+ loss. This is intriguing because the algorithm is quite similar to how faculty hiring and PhD admissions committees typically think: they want to make offers to the best candidates. The difference is that xGreedy carefully selects the *number* of offers to be made, in a way that (greedily) maximizes the objective. Since xGreedy amounts to a relatively small tweak to current practice, we believe committees would find the algorithm to be especially palatable.

An issue our results do not address is which penalty function best matches the needs of a specific recruitment process. For example, is there a rigorous way to argue that a particular choice of penalty function is more broadly applicable than another? That said, the choice between one-sided and two-sided penalty is rather intuitive, depending on the application. And our results provide computational arguments in favor of L_2 when two-sided penalty is desired, and L_1^+ for one-sided penalty.

From an ethical viewpoint, a potential concern is that our proposal may ultimately have unintended negative consequences. For example, if many faculty hiring committees adopted our optimization-based approach, might faculty candidates have fewer opportunities? We believe, however, that the opposite is true. Currently the academic job market is strikingly inefficient, as committees converge on a few candidates who are inundated with interviews and offers, while comparably strong candidates are left with nothing. If our approach is adopted widely, it is likely to widen the pool of candidates who receive appealing offers. Granted, a centralized matching market (in the style of the National Resident Matching Program) may be an even better solution, but creating such a market requires a huge — and often impractical — degree of coordination; by contrast, our approach can be adopted independently by institutions and even by individual departments or committees.

References

[1] John Abelt, Daniel Browning, Celia Dyer, Michael Haines, Jalen Ross, Patrick Still, and Matthew Gerber. Predicting likelihood of enrollment among applicants to the UVa undergraduate program. In *Systems and Information Engineering Design Symposium*, pages 194–199, 2015.

[2] Anand Bhalgat, Ashish Goel, and Sanjeev Khanna. Improved approximation results for stochastic knapsack problems. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1647–1665, 2011.

[3] Eranda Çela, Bettina Klinz, and Christophe Meyer. Polynomially solvable cases of the constant rank unconstrained quadratic 0-1 programming problem. *J. Comb. Optim.*, 12(3):187–215, 2006.

[4] Yeon-Koo Che and Youngwoo Koh. Decentralized college admissions. *Journal of Political Economy*, 124(5):1295 – 1338, 2016.

[5] Xiaowu Dai and Michael I. Jordan. Learning strategies in decentralized matching markets under uncertain preferences. *Journal of Machine Learning Research*, 22(260):1–50, 2021.

[6] B.C. Dean, M.X. Goemans, and J. Vondrák. Approximating the stochastic knapsack problem: the benefit of adaptivity. In *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 208–217, 2004.

[7] Subhamoy Ganguly, Ranojoy Basu, and Viswanathan Nagarajan. A binomial decision tree to manage yield-uncertainty in multi-round academic admissions processes. *Naval Research Logistics (NRL)*, 69(2):303–319, 2022.

[8] Jon Kleinberg, Yuval Rabani, and Éva Tardos. Allocating Bandwidth for Bursty Connections. *SIAM Journal on Computing*, 30:191–217, January 2000.

[9] Yao Luo and Yu Wang. Dynamic decision making under rolling admissions: Evidence from U.S. law school applications. Technical report, 2020.

[10] Anne L. Olsen. Penalty functions and the knapsack problem. In *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence, Orlando, Florida, USA, June 27-29, 1994*, pages 554–558. IEEE, 1994.

[11] Manish Purohit, Sreenivas Gollapudi, and Manish Raghavan. Hiring under uncertainty. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5181–5189. PMLR, 09–15 Jun 2019.

[12] Linda Siefert and Fred Galloway. A new look at solving the undergraduate yield problem: The importance of estimating individual price sensitivities. *College and University*, 81(3):11–17, 2006.

[13] Li Zhang and John C. Pippins. Application of overbooking model to college admission at the citadel. *International Transactions in Operational Research*, 28(5):2402–2413, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]

(b) Did you include complete proofs of all theoretical results? [Yes]

3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]