Online Allocation with Traffic Spikes: Mixing Adversarial and Stochastic Models

HOSSEIN ESFANDIARI, University of Maryland.

NITISH KORULA, Google Research, New York.

VAHAB MIRROKNI, Google Research, New York.

Motivated by Internet advertising applications, online allocation problems have been studied extensively in various adversarial and stochastic models. While the adversarial arrival models are too pessimistic, many of the stochastic (such as i.i.d or random-order) arrival models do not realistically capture uncertainty in predictions. A significant cause for such uncertainty is the presence of unpredictable traffic spikes, often due to breaking news or similar events. To address this issue, a simultaneous approximation framework has been proposed to develop algorithms that work well both in the adversarial and stochastic models; however, this framework does not enable algorithms that make good use of partially accurate forecasts when making online decisions. In this paper, we propose a robust online stochastic model that captures the nature of traffic spikes in online advertising. In our model, in addition to the stochastic input for which we have good forecasting, an unknown number of impressions arrive that are adversarially chosen. We design algorithms that combine an stochastic algorithm with an online algorithm that adaptively reacts to inaccurate predictions. We provide provable bounds for our new algorithms in this framework. We accompany our positive results with a set of hardness results showing that that our algorithms are not far from optimal in this framework. As a byproduct of our results, we also present improved online algorithms for a slight variant of the simultaneous approximation framework.

1. INTRODUCTION

In the past decade, online budgeted allocation problems have been studied extensively due to their important applications in Internet Advertising. In such problems, we are given a bipartite graph $G = (X, Y, E)$ with a set of fixed nodes (also known as *agents*, or *advertisers*) Y , a set of online nodes (corresponding to *items* or *impressions*) X , and a set E of edges between them. Each agent / fixed node $j \in Y$ is associated with a total weighted capacity (or budget) c_j ; in the context of Internet advertising, each agent corresponds to an advertiser with a fixed budget to spend on showing their ad to users. The items / online nodes $i \in X$ arrive one at a time, along with their incident edges $(i, j) \in E(G)$ and the weights $w_{i,j}$ on these edges. These online nodes correspond to search queries, page-views, or in general, impressions of ads by users. Upon the arrival of an item $i \in X$, the algorithm can assign i to at most one agent $j \in Y$ where $(i, j) \in E(G)$ and the total weight of nodes assigned to j does not exceed c_j . The goal is to maximize the total weight of the allocation.

This problem is known as the *Budgeted Allocation* or *AdWords* problem, and it has been studied under the assumption that $\frac{\max_{i,j} w_{i,j}}{\min_j c_j} \rightarrow 0$, in [Mehta et al. 2007; Buchbinder et al. 2007; Devanur and Hayes 2009] (called the *large-degree* assumption). Many variants of this problem such as *the display ads* problem [Feldman et al. 2009a]

Authors email addresses: hossein@cs.umd.edu, nitish@google.com, mirrokni@google.com

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

EC'15, June 15–19, 2015, Portland, OR, USA. Copyright © 2015 ACM 978-1-4503-3410-5/15/06 ...\$15.00.
<http://dx.doi.org/10.1145/2764468.2764536>

have been studied, and techniques to solve the budgeted allocation problem have been generalized to solve those problems.

Traditionally, results have been developed for a worst-case arrival model in which the algorithm does not have any prior on the arrival model of online nodes. Under this most basic online model, known as the *adversarial model*, the online algorithm does not know anything about the items or $E(G)$ beforehand. In this model, the seminal result of Karp, Vazirani and Vazirani [Karp et al. 1990] gives an optimal online $1 - \frac{1}{e}$ -competitive algorithm to maximize the size of the matching for *unweighted graphs* where $w_{ij} = 1$ for each $(i, j) \in E(G)$. For weighted graphs, Mehta et al. [Mehta et al. 2007; Buchbinder et al. 2007] presented the first $1 - \frac{1}{e}$ -approximation algorithm to maximize the total weight of the allocation for the AdWords problem.

In practical settings motivated by placement of Internet ads, the incoming traffic of page-views may be predicted with a reasonable precision using a vast amount of historical data. Motivated by this ability to forecast traffic patterns, various stochastic online arrival models have been introduced. Such models include (i) the i.i.d. stochastic arrival model in which there is a (known or unknown) distribution over the types of items, and each item that arrives is drawn i.i.d. from this distribution [Feldman et al. 2009b; Vee et al. 2010], or (ii) the random order model [Devanur and Hayes 2009; Feldman et al. 2010; Agrawal et al. 2009], which makes the weaker assumption that individual items and edge weights can be selected by an adversary, but that they arrive in a random order. Several techniques have been developed to design asymptotically optimal online allocation algorithms for these stochastic arrival models (For example, these algorithms include a set of dual-based algorithms [Devanur and Hayes 2009; Feldman et al. 2010], and a set of primal-based algorithms discussed later).

These algorithms for the stochastic models are useful mainly if the incoming traffic of items (i.e. online impressions) can be predicted with high precision. In other words, such algorithms tend to rely heavily on a precise forecast of the online traffic patterns (or if the forecast is not explicitly provided in advance, that the pattern ‘learnt’ by the algorithm is accurate), and hence these algorithms may not react quickly to sudden changes in traffic. In fact, the slow reaction to such traffic spikes imposes a serious limitation in the real-world use of stochastic algorithms for online advertising, and more generally, this is a common issue in applying stochastic optimization techniques to online resource allocation problems (see e.g., [Wang et al. 2006]). To the best of our knowledge, no large Internet advertising systems deploy such stochastic allocation algorithms ‘as-is’ without modifications to deal with situations where the forecasts are inaccurate. Various techniques such as robust or control-based stochastic optimization have been described in the literature [Ben-Tal and Nemirovski 2002; Bertsimas et al. 2004; Wang et al. 2006; Tan and Srikant 2011] to deal with this shortcoming, but they do not provide theoretical guarantees when the input is near-adversarial.

One recent theoretical result in this direction is the simultaneous adversarial and stochastic framework [Mirrokni et al. 2011]. The main question of this recent work is whether there exists an algorithm which simultaneously achieves optimal approximation ratios in both the adversarial and random-order settings. More specifically, does there exist an algorithm achieving a $1 - \varepsilon$ approximation for the random-order model, and at the same time achieving a $1 - \frac{1}{e}$ -approximation for the adversarial model? Mirrokni et al. [2011] showed that the answer to this question is positive for unweighted bipartite graphs, but it is negative for the general budgeted allocation problem. Further, they show that the best $1 - \frac{1}{e}$ -competitive algorithm for the adversarial model achieves a 0.76-approximation in the random-order model. Though this shows that the adversarial algorithm has an improved competitive ratio in stochastic settings, it does not use forecast information explicitly, and hence it can be quite far from optimal

even when the forecast is perfectly accurate. Moreover, the simultaneous approximation framework is still trying to design an algorithm that is guaranteed to work well in extreme situations (where the input follows the forecast perfectly, or is completely adversarial). What if the forecast is mostly, but not entirely accurate? For instance, suppose traffic to a website largely follows the prediction, but there is a sudden spike due to a breaking news event? Treating this as entirely adversarial input is too pessimistic.

Our Model and Results. In this paper, we propose a model of online stochastic budgeted allocation with traffic spikes, referred to as *Robust Budgeted Allocation*, that goes beyond the worst-case analysis in the adversarial model, and develop algorithms that explicitly use the stochastic information available for arrival pattern. In our model, in addition to the stochastic input for which we have good forecasting, an unknown number of impressions arrive that are adversarially chosen. This model is motivated by the patterns of traffic spikes in online advertising in which part of the incoming traffic of users may be the result of a new event that we did not predict, corresponding to a new traffic pattern. We design an algorithm that adaptively checks if the traffic forecast is accurate, and reacts to flaws in traffic forecasting due to traffic spikes. We measure the accuracy of the forecast in terms of a parameter λ which, roughly speaking, captures the fraction of the value of an optimal solution that can be obtained from the stochastic input (as opposed to the adversarially chosen impressions). In general, the competitive ratio of the algorithm will naturally increase with λ . Furthermore, we accompany our results with a set of hardness results showing that our provable approximation guarantees are not far from the optimal achievable bounds. Interestingly, our techniques *also* result in new approaches for the simultaneous approximation framework of [Mirrokni et al. 2011]; though the models are slightly different (i.i.d. vs. random order, as well as the fact that we require a possibly inaccurate forecast of traffic), our algorithm gives improved performance for the weighted case under uncertain input compared to what was achieved in that paper. Section 2 describes the model precisely, allowing us to formally state our results.

Our technique is based on defining a notion of ε -closeness in distributions, and then understanding the behaviour of an online algorithm over sequences that are ε -close to a given distribution. Most notably, we can show how to modify any online stochastic algorithm to work for online adversarial input sequences that are ε -close to a known distribution. This technique is summarized in the next section. We then combine such a modified stochastic algorithm with an adversarial algorithm to guarantee robustness. Converting this idea to provable algorithms for the robust online allocation problem requires applying several combinatorial lemmas and proving invariants that can be converted to a factor-revealing mathematical program, which can then be analyzed numerically and analytically to prove desirable competitive ratios.

Other Related Work. Online Stochastic Allocation. Two general techniques have been applied to get improved approximation algorithms for online stochastic allocation problems: *primal-based* and *dual-based* techniques. The dual-based technique is based on solving a dual linear program on a sample instance, and using this dual solution in the online decisions. This method was pioneered by Devanur and Hayes [Devanur and Hayes 2009] for the AdWords problem and extended to more general problems [Feldman et al. 2010; Agrawal et al. 2009; Vee et al. 2010]. It gives a $1 - \varepsilon$ -approximations for the random order model if the number of items m is known to the algorithm in advance, and $\frac{OPT}{w_{ij}} \geq O(\frac{m \log n}{\varepsilon^3})$, where $n := |Y|$ is the number of agents. The primal-based technique is based on solving an offline primal instance, and applying this solution in an online manner. This method applies the idea of power-of-two choices, and gives

improved approximation algorithms for the iid model with known distributions. This technique was initiated by [Feldman et al. 2009b] for the online (unweighted) matching problem and has been improved [Bahmani and Kapralov 2010; Menshadi et al. 2011; Haeupler et al. 2011; Jaillet and Lu 2013]. All the above algorithms heavily rely on an accurate forecast of the traffic. An alternative technique that has been applied to on-line stochastic allocation problems is based on optimizing a potential function at each stage of the algorithm [Devanur et al. 2011, 2012]. This technique has been analyzed and proved to produce asymptotically optimal results under the i.i.d. model with unknown distributions. Although this technique does not rely on the accurate predictions as much, it does not combine stochastic and adversarial models, and the analysis techniques used are not applicable to our robust online allocation model. For unweighted graphs, it has been recently observed that the Karp-Vazirani-Vazirani $1 - \frac{1}{e}$ -competitive algorithm for the adversarial model also achieves an improved approximation ratio of 0.70 in the random arrival model [Karande et al. 2011; Mahdian and Yan 2011]. This holds even without the assumption of large degrees. It is known that without this assumption, one cannot achieve an approximation factor better than 0.82 for this problem (even in the case of i.i.d. draws from a known distribution) [Menshadi et al. 2011]. All the above results rely on stochastic assumptions and apply only to the random-order or the iid stochastic models.

Robust stochastic optimization. Dealing with traffic spikes and inaccuracy in forecasting the traffic patterns is a central issue in operations research and stochastic optimization. Methods including robust or control-based stochastic optimization [Ben-Tal and Nemirovski 2002; Bertsimas et al. 2004; Wang et al. 2006; Tan and Srikant 2011] have been proposed. These techniques either try to deal with a larger family of stochastic models at once [Ben-Tal and Nemirovski 2002; Bertsimas et al. 2004; Wang et al. 2006], try to handle a large class of demand matrices at the same time [Wang et al. 2006; Applegate and Cohen 2006; Azar et al. 2003], or aim to design asymptotically optimal algorithms that react more adaptively to traffic spikes [Tan and Srikant 2011]. These methods have been applied in particular for traffic engineering [Wang et al. 2006] and inter-domain routing [Applegate and Cohen 2006; Azar et al. 2003]. Although dealing with similar issues, our approach and results are quite different from the approaches taken in these papers. For example, none of these previous models give theoretical guarantees in the adversarial model while preserving an improved approximation ratio for the stochastic model. Finally, an interesting related model for combining stochastic and online solutions for the Adwords problem is considered in [Mahdian et al. 2007], however their approach does not give an improved approximation algorithm for the i.i.d. model.

Note: Due to space constraints, several proofs have been deferred to the full version. In particular, the proof of Lemma 3.4 for the unweighted case appears only in the full version, but we include the similar proof for the more general weighted case here.

2. PRELIMINARIES AND TECHNIQUES

2.1. Model

Let $\mathcal{I}$ denote a set of item ‘types’; in the Internet advertising applications, these represent queries / ad impressions with different properties that are relevant to advertiser targeting and bidding. A *forecast* $F = (D, f)$ has two components: A distribution D over $\mathcal{I}$, together with a number f ; this is interpreted as a prediction that f items will arrive, each of which is drawn independently from D .

In the *Stochastic Budgeted Allocation problem*, the input known to the algorithm in advance is a forecast $F = (D, f)$, and a set of agents Y , with a capacity c_j for agent j . A sequence of f items is drawn from the distribution D and sent to the algorithm

one at a time; the algorithm must allocate these items as they appear online. The total weight of items allocated to agent j must not exceed c_j , and the objective is to maximize the weight of the allocation. As discussed above, there has been considerable work on near-optimal algorithms for Stochastic Budgeted Allocation [Devanur and Hayes 2009; Feldman et al. 2010; Agrawal et al. 2009; Vee et al. 2010; Alaei et al. 2012].

In this paper, we define the new *Robust Budgeted Allocation problem*, for which our input model is the following: The adversary can create in advance an arbitrary forecast $F = (D, f)$, and a collection of agents Y . Further, at each time step, the adversary can either create a new arbitrary item (together with its incident edges and weights) and send it to the algorithm, or choose to send an item drawn from D . After at least f items have been drawn from D , the adversary can either send additional (arbitrary) items, or choose to terminate the input. The online algorithm knows in advance only the forecast F and the agents Y , and so it knows that it will receive f items corresponding to i.i.d. draws from D ; it does not know anything about the items created by the adversary, where in the sequence they arrive, or the total number m of items that will arrive. As usual, the competitive ratio of the algorithm is measured by the worst-case ratio (over all inputs) of the value of its allocation to the value of the optimal allocation on the sequence that arrives.

With the preceding description of the model, no algorithm can have a competitive ratio better than $1 - 1/e$, for the simple reason that we could set $f = 0$, allowing the adversary to control the entire input. (Or even for larger f , the edge weights for the adversarial items could be considerably larger than the weights for the forecast items in $\mathcal{I}$.) We have not quantified the accuracy of the forecast, or meaningfully limited the power of the adversary. Our goal is to design algorithms with a competitive ratio that improves with the accuracy of the forecast. We quantify this accuracy as follows:

Definition 2.1. For an instance I of the Robust Budgeted Allocation problem with forecast (D, f) , let $S(I)$ denote the set of f ‘stochastic’ items drawn from distribution D . Let $A(I)$ denote the set of $n - f$ ‘adversarial’ items. When I is clear from context, we simply use S and A to denote the stochastic and adversarial items respectively. We mildly abuse notation and, when clear from context, also use I to refer to the sequence of items in an instance. For a solution Sol to an instance I , let $\text{Val}_S(Sol)$ denote the value obtained by Sol from allocating the items of S to agents, and $\text{Val}_A(Sol)$ denote the value obtained by Sol from allocating the items of A .

Definition 2.2. An instance I of the Robust Budgeted Allocation problem is said to be λ -stochastic if $\lambda = \max_{Sol \in OPT(I)} \left\{ \frac{E[\text{Val}_S(Sol)]}{E[\text{Val}_S(Sol) + \text{Val}_A(Sol)]} \right\}$, where $OPT(I)$ is the set of all optimal solutions of I .

Note that when the forecast is completely accurate (there are no adversarial items), the instance is 1-stochastic, and when $f = 0$ (all items are adversarial), the input is 0-stochastic. Though λ is unknown to the algorithm, our goal is to design algorithms which, when restricted to λ -stochastic instances, have good competitive ratio (ideally, increasing in λ).

2.2. Algorithms for Working with Forecasts

In this section, we consider how to solve the Stochastic Budgeted Allocation problem. Similar problems have been applied before (see e.g. [Alaei et al. 2012]), but we describe a specific approach below that will be a useful tool for the Robust Budgeted Allocation problem. Further, our argument implies that this approach performs well even for an *adversarial* input sequence if it is sufficiently ‘close’ to the forecast.

Roughly speaking, given a forecast $F = (D, f)$, if the number of items f is sufficiently large, then we can work with an ‘expected instance’. In the expected instance, the set

of items is created by assuming each type $t \in \mathcal{I}$ arrives in proportion to $P_{D=t}$. We then run any (offline) algorithm Alg on the expected instance; when an item arrives in the real online instance, we assign it according to the allocation given by Alg on the expected instance. If the number of items f is sufficiently large, then only a small error is induced because we assign according to the expected instance instead of the random realization of the forecast.

We begin by defining the notion of a sequence being ε -close to a distribution. Indeed, we show that with high probability a ‘long’ sequence of draws from a distribution is ε -close to that distribution, where ε is an arbitrary small constant. We next prove that if we ignore inputs which are not ε -close to the input distribution, we lose only $\frac{1}{m}$ in the competitive ratio. Finally, we show that we can modify any online stochastic algorithm, or more strongly any offline algorithm for Budgeted Allocation to work for online adversarial input sequences which are guaranteed to be ε -close to a known distribution. Interestingly, this reduction loses only 4ε on the competitive ratio.

Definition 2.3. Let $S = \langle s_1, s_2, \dots, s_m \rangle$ be a sequence of items and let D be a distribution over a set of item types $\mathcal{I}$. For a type $t \in \mathcal{I}$, let $P_{D=t}$ be the probability that a draw from D is t . We say S is ε -close to distribution D , if for any continuous sub-sequence $S_{i,k} = s_i, s_{i+1}, \dots, s_k \subset S$ and any type t , the number of items of type t in $S_{i,k}$ is within the range $(k - i + 1 \pm \varepsilon m)P_{D=t}$. If S is not ε -close to distribution D we say it is ε -far from the distribution.

Consider that $(k - i + 1)P_{D=t}$ is the expected number of items of type t in a set of $k - i + 1$ draws from D . When $k - i + 1$ is large, using the Chernoff bound we can show that the number of items of type t is close to the expectation. On the other hand, when $k - i + 1$ is small, the term εm dominates $k - i + 1$, and thus the number of items of type t is in the range $(k - i + 1 \pm \varepsilon m)P_{D=t}$. Lemma 2.5 formalizes this intuition, showing that with high probability a sequence of drawn items from a distribution D is ε -close to D .

Definition 2.4. Given a distribution D , a sequence of f items is said to satisfy the long input condition for D if $\frac{f}{\log(f)} \geq \frac{15}{\varepsilon^2 \min_{t \in D} (P_{D=t})}$.

LEMMA 2.5. Let S be a sequence of m items drawn from a distribution D . Assuming the long input condition, the probability that S is ε -far from D is at most $\frac{1}{m^2}$.

In the rest of this section, we use the monotonicity and subadditivity properties of Budgeted Allocation, stated in Lemma 2.6 and Lemma 2.7 respectively.

LEMMA 2.6 (MONOTONICITY). Budgeted Allocation is monotone: Fixing the set of agents and their capacities, for any sequence of items S and any sequence $T \subseteq S$, we have $Opt(T) \leq Opt(S)$ where $Opt(S)$ and $Opt(T)$ are the values of the optimum solutions when the items that arrive are S and T respectively.

LEMMA 2.7 (SUBADDITIVITY). Budgeted Allocation is subadditive: Fixing the set of agents and their capacities, for any sequence of items S and any sequence of items T , we have $Opt(S \cup T) \leq Opt(S) + Opt(T)$ where $Opt(X)$ indicates the size of the optimum solution when the sequence of items that arrive is X .

Lemma 2.5 says that w.h.p, a sequence of items drawn from a distribution D is ε -close to D . That is, inputs which are ε -far from the input distribution are rare, but this does not immediately imply that the total value of such rare ε -far inputs is small as well. Lemma 2.8 says that we may ignore all inputs which are ε -far from the input distribution and only lose a small fraction in the competitive ratio.

LEMMA 2.8. *Let S be a sequence of m items drawn from a distribution D , satisfying the long input condition. Let Alg be an α -competitive algorithm for Stochastic Budgeted Allocation with forecast (D, m) . Let Alg_{close} be an algorithm that has the same outcome as Alg when the input is ε -close to D and 0 otherwise. Alg_{close} is $(\alpha - \frac{1}{m})$ -competitive.*

Note that any algorithm Alg for Budgeted Allocation has a random outcome when the items are drawn from a forecast, simply due to the randomness in the sequence of items. We now define a derandomization of such algorithms: Given an algorithm Alg , a forecast $F = (D, f)$ and a constant ε , algorithm $DeRand_\varepsilon^F(Alg)$ is defined as follows:

Let S' be a sequence of $(1 - \varepsilon)f$ impressions, with $(1 - \varepsilon)f \cdot P_{D=t}$ impressions of type t , for each type t . Run algorithm Alg on sequence S' . Let $Alg(S', t, i)$ be the agent to which Alg assigns the i th impression of type t in S' . Note that any sequence of f items which is ε -close to D contains at least $(1 - \varepsilon)f \cdot P_{D=t}$ impressions of each type t . We can now describe how $DeRand_\varepsilon^F(Alg)$ allocates items of a sequence S . For each type t , Algorithm $DeRand_\varepsilon^F(Alg)$ allocates the first $(1 - \varepsilon)f \cdot P_{D=t}$ impressions of type t in S in the same manner as Alg allocated S' . That is, we assign the i th impression of type t in S to $Alg(S', t, i)$. After the first $(1 - \varepsilon)f \cdot P_{D=t}$ impressions of type t have been allocated, we do not allocate any more items of this type. Finally, if at any time during the algorithm, we observe that the input sequence (so far) is not ε -close to distribution D , the algorithm stops and returns false. Otherwise, it returns true.

When it is clear from the context, we drop F and ε from the notation of $DeRand_\varepsilon^F(Alg)$.

Remark 2.9. Note that for any forecast $F = (D, f)$ and constant ε , the outcome of $DeRand_\varepsilon^F(Alg)$ on any item sequence of length f that is ε -close to D is a function purely of D and ε , but not the actual impressions in the sequence.

THEOREM 2.10. *Let $F = (D, f)$ be a distribution, and let A be an adversarial input with length f such that A satisfies the long input condition and A is ε -close to D . Let Alg be an α -competitive algorithm for Stochastic Budgeted Allocation. Though A is not explicitly drawn from D , $DeRand_\varepsilon^F(Alg)$ is $\alpha - 2\varepsilon$ competitive on sequence A .*

Consider an α -competitive online algorithm (or α -approximate offline algorithm), Alg , for Budgeted Allocation with stochastic input. By Theorem 2.10, for inputs which are ε -close to D , $DeRand_\varepsilon^F(Alg)$ has a competitive ratio that is only 2ε worse than that of Alg . Moreover, Lemma 2.8 says that if we ignore all inputs which are ε -far from the input distribution, we lose only $\frac{1}{m}$ on the competitive ratio. Together with the assumption that $\varepsilon \geq \frac{1}{m}$, we obtain the following corollary.

COROLLARY 2.11. *Let Alg be an α -competitive algorithm for Stochastic Budgeted Allocation (or α -approximate offline algorithm for Budgeted Allocation). For any small constant ε and any forecast F , $DeRand_\varepsilon^F(Alg)$ is an $(\alpha - 3\varepsilon)$ -competitive algorithm for Stochastic Budgeted Allocation.*

3. ROBUST ONLINE BUDGETED ALLOCATION

In this section, we consider the Robust Budgeted Allocation problem. As described above, the algorithm knows in advance the set of agents Y , together with a capacity c_j for each agent $j \in Y$. Further, it knows the forecast $F = (D, f)$, but not how many additional items will be sent by the adversary, nor how they are intermixed with the f items drawn from D .

Recall that a λ -stochastic instance is one where the ‘stochastic items’ (those drawn from D) provide λ -fraction of the value of an optimal solution. (Also, note that λ is not

Fig. 1. Summary of results, parametrized by λ . The blue curve denotes the upper bound on the competitive ratio of any algorithm, the green curve is our algorithm for the unweighted case, and the red curve is our algorithm for the weighted case.

known to the algorithm.) As λ increases (corresponding to an increase in the accuracy of our forecast, or a smaller traffic spike), we expect the performance of our algorithms to improve. However, we wish our algorithms to be robust, obtaining good performance compared to an optimal offline solution even when λ is close to 0 (corresponding to a very large traffic spike, when the typical ‘forecast’ traffic is only a small fraction of the total).

First, in Section 3.1, we consider the unweighted Robust Budgeted Allocation problem, in which w_{ij} is the same for all $(i, j) \in E$. As desired, we obtain an algorithm with competitive ratio tending to 1 as λ tends to 1, and $1 - 1/e$ when λ tends to 0. Then, in Section 3.2, we consider the general weighted Robust Budgeted Allocation problem, and finally, in Section 3.3, we give upper bounds on the competitive ratio of any algorithm. All our competitive ratios are parametrized by λ , and they are summarized in Figure 1.

For simplicity, throughout this section, we assume that the capacity of all the agents are the same (normalized to 1). This assumption can be removed by dividing agents with large capacities into multiple dummy agents.

3.1. Unweighted Robust Budgeted Allocation

Our general approach will be to *simulate* the following idea: Suppose we first receive the f stochastic items drawn from D . We could allocate these items optimally (assuming they are the entire input). If this uses up most of the budgets of the agents, we automatically have a good solution, even if we do not allocate any of the adversarial items. If some fraction of the budgets remain unused, use this remaining capacity to allocate any adversarial items that arrive. If there is a way to allocate these adversarial items using the remaining capacities, we obtain $1 - 1/e$ fraction of this value.

Unfortunately, in the real Robust Budgeted Allocation model, we do not know which items are stochastic and which are adversarial, so we cannot perform this clean separation perfectly. Still, we can approximate this separation as follows: Let A be an algorithm for Stochastic Budgeted Allocation. We have two algorithms running simultaneously. The first is a slight variant of $DeRand_\epsilon^F(A)$, and the second is Balance [Kalyana-

sundaram and Pruhs 2000]. More precisely, we send each item to $DeRand_\varepsilon^F(A)$; recall that this algorithm first checks if the input seen so far is ε -close to D . If it is, it allocates this item according to A ; otherwise, it returns false. Now, instead of returning false, we assume that this item must have been sent by the adversary. As such, we erase this item from the history of $DeRand_\varepsilon^F(A)$, and try to allocate it using Balance. That is, we ‘guess’ that all items that are matched by $DeRand_\varepsilon^F(A)$ are stochastic items and all other items are adversarial.

Note that $DeRand_\varepsilon^F(A)$ does not match more than $(1 - \varepsilon)f$ items, $(1 - \varepsilon)fP_{D=t}$ from each type t . From Lemmas 2.5 and 2.8, we know that w.h.p., the sequence of stochastic items is ε -close to D ; by ignoring the case when it is ε -far from D , we lose at most $\frac{1}{m}$ in the competitive ratio. Given the long input assumption, $1/m < \varepsilon$, and hence by losing ε in the competitive ratio, we assume that the stochastic items are always ε -close to D . Since the sequence of stochastic items is ε -close to D , we have at least $(1 - \varepsilon)fP_{D=t}$ items of type t . Therefore, the items that $DeRand_\varepsilon^F(A)$ leaves unallocated are a superset of all the adversarial items. (More precisely, there may be an adversarial item of type t that we guess is stochastic, but there must be a corresponding stochastic item of type t that we treat as adversarial instead.)

We now complete the description of our combined algorithm: Given an allocation, let x_j denote the fraction of agent j ’s budget used in this allocation. Let Alg_S be a $(1 - \varepsilon)$ -competitive algorithm for Stochastic Budgeted Allocation with the further property that it minimizes $\sum_{j=1}^n x_j^2$. (In other words, Alg_S reproduces an optimal offline solution such that each item is allocated to an eligible agent with the lowest x_j .) We run $DeRand_\varepsilon^F(Alg_S)$ as the first of our two algorithms. Recall from Remark 2.9 we know the exact allocation of $DeRand_\varepsilon^F(Alg_S)$, which is independent of the random draws. If x_j denotes the fraction of agent j ’s capacity used by this allocation, for items unallocated by $DeRand_\varepsilon^F(Alg_S)$, we run the Balance algorithm on the instance of Budgeted Allocation with the adversarial items, and in which the capacity of agent j is $1 - x_j$.

How does this combined algorithm perform? We use S to denote the stochastic items, and A the adversarial items. From Corollary 2.11 we know that $DeRand_\varepsilon^F(Alg_S)$ is $1 - O(\varepsilon)$ competitive against $Opt(S)$. We will prove in Lemma 3.5 that the optimum solution on the items of A using the remaining capacities $1 - x_j$ is at least $(1 - \lambda - O(\varepsilon))(Opt(A \cup S) - Opt(S))$. Since Balance is a $(1 - 1/e)$ -competitive algorithm, the value we derive from Balance is at least $(1 - \frac{1}{e})(1 - \lambda - O(\varepsilon)) \cdot (Opt(A \cup S) - Opt(S))$. Therefore, the competitive ratio of the combined algorithm is at least:

$$\begin{aligned} & \frac{(1 - O(\varepsilon))Opt(S) + (1 - \lambda - O(\varepsilon))(1 - \frac{1}{e})(Opt(A \cup S) - Opt(S))}{Opt(S \cup A)} \\ &= \frac{\lambda Opt(S \cup A) + (1 - \lambda)^2(1 - \frac{1}{e})Opt(S \cup A) - O(\varepsilon)}{Opt(S \cup A)} \\ &= \lambda + (1 - \lambda)^2(1 - \frac{1}{e}) - O(\varepsilon) \end{aligned}$$

where the first equality uses the definition of λ -stochastic to replace $Opt(S)$ with $\lambda Opt(S \cup A)$. This proves the following theorem.

THEOREM 3.1. *Assuming the long input condition, there exists a $\lambda + (1 - \lambda)^2(1 - \frac{1}{e}) - O(\varepsilon)$ competitive algorithm for Robust Budgeted Allocation when the input is λ -stochastic.*

In order to prove the key Lemma 3.5, we first write Mathematical Program 2, and show in Lemma 3.3 that this lower bounds the ratio of the optimum allocation of

$$\begin{array}{ll}
\text{Minimize:} & \frac{\sum_{j=1}^n (y_j - z_j)}{\sum_{j=1}^n y_j} \\
\text{Subject to:} & z_j \leq \max(0, x_j + y_j - 1) \quad \forall 1 \leq j \leq n \\
& \sum_{k=j+1}^n (x_k + y_k) \leq n - j \quad \forall 1 \leq j \leq n - 1 \text{ s.t. } x_j < x_{j+1} \\
& x_j \leq x_{j+1} \quad \forall 1 \leq j \leq n - 1 \\
& \lambda \left(\sum_{j=1}^n x_j + \sum_{j=1}^n y_j \right) \geq \sum_{j=1}^n x_j \\
& 0 \leq x_j, y_j, z_j \leq 1 \quad \forall 1 \leq j \leq n
\end{array}$$

Fig. 2. Mathematical Program 2 to bound the optimum allocation of adversarial items on the remaining capacities in uniform weight case.

the adversarial items on the remaining capacities to the overall optimum. Next, in Lemma 3.4, we show that the solution of this mathematical program is at least $1 - \lambda$. Lemma 3.5 is an immediate result of combining these two lemmas.

We now show that there exists an optimum allocation of all items such that the contribution of stochastic items in this allocation is exactly $Opt(S)$. We will need this for Lemma 3.3.

LEMMA 3.2. *For any disjoint sets of items A and S , there exist an optimum allocation of $S \cup A$ such that the contribution of items of S in this allocation is exactly $Opt(S)$.*

PROOF. We prove this lemma by contradiction. Assume that in all optimum allocations of $S \cup A$, the contribution of items of S is less than $Opt(S)$. Consider an optimum allocation $Opt'(S \cup A)$ of $S \cup A$ such that the items of S have the maximum contribution to the allocation. Denote the allocation of items of S in $Opt'(S \cup A)$ by $A'(S)$. By definition, $A'(S)$ is not an optimum allocation of S . Thus, there exists an augmenting path in $A'(S)$ which can increase the number of assigned items to one of the agents by exactly one, and keep the number of assigned items to all other agents the same. This change increases the number of assigned items in $A'(S)$ by one, and may decrease the number of assigned items from A by at most one. Therefore, it is an optimal solution of $S \cup A$ in which items from S have more contribution, and hence gives a contradiction. $\square$

LEMMA 3.3. *The optimum allocation of the adversarial items on the remaining capacities is at least $Opt(S \cup A) - Opt(S)$ times the solution of Mathematical Program 2.*

PROOF. Let x_j be the fraction of the capacity of the agent j which is filled by Alg_S . Without loss of generality, assume that x_j s are in increasing order. That is, for all $1 \leq j \leq n - 1$, we have $x_j \leq x_{j+1}$, which is the third constraint in Mathematical Program 2.

Consider the optimum solution of $S \cup A$ from Lemma 3.2. Let y_j be the fraction of the capacity of the agent j that is filled by the adversarial items in this solution. One can see that the fourth constraint says that the contribution of the stochastic items is at most λ fraction of the total value. (From Lemma 3.2, we could have equality in the fourth constraint, but for simplicity of analysis, we maintain the inequality.)

Note that we want to compare the optimum solution of the adversarial items on the remaining capacities $1 - x_j$ with the total value $\sum_{j=1}^n y_j$ of the adversarial items in the

optimum solution of $S \cup A$. For some agent j , if we have $y_j + x_j \leq 1$, we can assign the same adversarial items to the agent j as in the optimum allocation. On the other hand, if $y_j + x_j \geq 1$, we can only use $1 - x_j$ fraction of the capacity of agent j for the adversarial items. Thus, we can always assign $y_j - \max(0, x_j + y_j - 1)$ fraction of the capacity of agent i to the adversarial items. This quantity is denoted by z_j in the first constraint.

By assigning adversarial items as in the optimum solution of $S \cup A$ using the remaining capacities, we can obtain at least $\sum_{j=1}^n y_j - z_j$. The objective function of Mathematical Program 2 compares this to what the optimum solution of $S \cup A$ can get from adversarial items.

Finally, fix an index j such that $x_j < x_{j+1}$ and look at all agents with index greater than j . All stochastic items that we match to agents with index greater than j have no edge to the agents with index less than or equal to j . (By definition of Alg_S , which assigns items to the eligible agent with lowest value of x_j .) Thus, in any optimum solution of S , they cover at least $\sum_{k=j+1}^n x_k$ of the agents with index greater than j . Thus, this optimum solution of $S \cup A$ covers $\sum_{k=j+1}^n (x_k + y_k)$ of the agents with index greater than j . Consider that we have $n - j$ such agents, we get $\sum_{k=j+1}^n (x_k + y_k) \leq n - j$. This is the second constraint in Mathematical program 2. $\square$

LEMMA 3.4. *For any small number δ , the solution of the mathematical program 2 is at least $1 - \lambda - O(\delta)$.*

The following lemma is an immediate result of combining Lemma 3.3 and Lemma 3.4.

LEMMA 3.5. *The optimum allocation of adversarial items on the remaining capacities is at least $(1 - \lambda - O(\delta))(Opt(S \cup A) - Opt(S))$.*

3.2. Weighted Robust Budgeted Allocation

We can now describe our algorithm for general weighted case of Budgeted Allocation. As in the unweighted case, we combine the allocations of two algorithms: One that runs on the (stochastic) items that we guess are drawn from the forecast, and one on the items that we guess are constructed by the adversary.

For the stochastic items, we start with a base algorithm for Stochastic Budgeted Allocation that is *not necessarily optimal*. Instead of maximizing the weight of the allocation from the stochastic items, we start with an algorithm Alg_{pot} that *maximizes the potential* of the allocation, as defined below.

Definition 3.6. Let $X = (x_1, x_2, \dots, x_n)$ be a vector of numbers, such that for all $1 \leq j \leq n$, we have $0 \leq x_j \leq 1$. We define the potential of x_j , $Pot(x_j)$ to be $x_j - e^{(x_j-1)}$. We define the potential of the vector X , $Pot(X)$ to be $\sum_{j=1}^n Pot(x_j)$.

Let x_j denote the fraction of capacity c_j used by the potential-maximizing allocation of Alg_{pot} . Similarly to the unweighted case, when items arrive, we send them to $DeRand(Alg_{pot})$; for those items that are unmatched, by $DeRand(Alg_{pot})$, we send them to the Balance algorithm using the remaining capacities $1 - x_j$. Exactly the same argument that we provide for the unweighted case works here to show that by losing $O(\varepsilon)$ on the competitive ratio, we can assume that we match all stochastic items using $DeRand(Alg_{pot})$ and all adversarial items using the Balance algorithm. We use Alg to denote this combined algorithm. In order to analyze our algorithm Alg , we need to define another potential function based on $Pot(X)$.

Fig. 3. Competitive ratio of Alg parametrized by λ .

Definition 3.7. Let $X = (x_1, x_2, \dots, x_n)$ and $Y = (y_1, y_2, \dots, y_n)$ be two vectors of numbers between 0 and 1. For each $1 \leq j \leq n$, we define $Pot_X(y_j)$ as follow. If $x_j \leq y_j$, we have $Pot_X(y_j) = Pot(y_j)$. Otherwise, we have $Pot_X(y_j) = Pot(x_j) + (y_j - x_j)Pot'(x_j)$, where $Pot'(\cdot)$ is the first derivative of $Pot(\cdot)$. Thus, for $x_j < y_j$ we have:

$$Pot_X(y_j) = x_j - e^{x_j-1} + (y_j - x_j)(1 - e^{x_j-1})$$

We define $Pot_X(Y)$ to be $\sum_{j=1}^n Pot_X(y_j)$.

Note that the second derivative of $Pot(x_j)$ is $-e^{x_j-1}$ which is always negative. Thus, $Pot(x_j)$ is a concave function. Therefore, for a fixed x_j , $Pot_X(y_j)$ is concave in the range $[0, x_j]$. Moreover, $Pot_X(y_j)$ is linear in the range $(x_j, 1]$ and thus, $Pot_X(y_j)$ is a concave function.

Consider that, in the range $(x_j, 1]$, the function $Pot_X(y_j)$ is the degree 2 Taylor series of $Pot(y_j)$ at point x_j . In addition, the second derivative of $Pot(y_j)$ in the range $[0, 1]$ is lower-bounded by -1 , yielding the following lemma.

LEMMA 3.8. *For any constant ε and any vector of positive numbers X and any y_j such that $0 \leq y_j \leq x_j + \varepsilon \leq 1$ we have $|Pot_X(y_j) - Pot(y_j)| \leq \varepsilon^2$.*

We now have the tools to analyze Alg via means of a mathematical program: First, in Lemma 3.10, we show that the competitive ratio of our algorithm is lower bounded by the solution of the Mathematical program 4. Next, in Lemma 3.11, we lower bound the solution of the mathematical program; this lower bound is shown in Figure 3. Together, these lemmas allow us to prove the following theorem:

THEOREM 3.9. *There exists an algorithm for λ -stochastic weighted budgeted allocation with competitive ratio presented in Figure 3.*

LEMMA 3.10. *The competitive ratio of Alg is bounded by the solution of the Mathematical program 4.*

PROOF. For each agent j , let x_j be the expected fraction of c_j used by Alg_{pot} on the stochastic items. Consider the optimum assignment Opt_λ that maximizes $\frac{E[Stochastic(Opt_\lambda)]}{E[Opt_\lambda]}$. For each agent j , let t_j and y_j respectively denote the expected fraction

Minimize:	$\frac{\sum_{j=1}^n x_j + (1 - \frac{1}{e}) \sum_{j=1}^n (y_j - z_j)}{\sum_{j=1}^n y_j + \sum_{j=1}^n t_j}$	(1)
Subject to:	$z_j \leq \max(0, x_j + y_j - 1) \quad \forall 1 \leq j \leq n$	(2)
	$Pot_X(T) \leq Pot(X)$	(3)
	$t_j + y_j \leq 1 \quad \forall 1 \leq j \leq n$	(4)
	$\lambda \left(\sum_{j=1}^n t_j + \sum_{j=1}^n y_j \right) = \sum_{j=1}^n t_j$	(5)
	$0 \leq t_j, x_j, y_j, z_j \leq 1 \quad \forall 1 \leq j \leq n$	(6)

Fig. 4. Mathematical Program 4 to bound the competitive ratio of Alg .

of c_j used by stochastic items and adversarial items in Opt_λ . Therefore, the expected fraction of c_j used in Opt_λ is $t_j + y_j$, which is upper-bounded by 1. This gives us the Inequality 4.

By definition of λ , the contribution of the stochastic items to Opt_λ is λ fraction of the total value; equality 5 captures this fact.

For any agent j , if we have $y_j \leq 1 - x_j$, an offline algorithm can allocate all the same adversarial items to agent j as Opt_λ did by using the remaining $1 - x_j$ fraction of c_j . On the other hand, if we have $y_j > 1 - x_j$, the offline algorithm can use the entire remaining capacity $1 - x_j$ of agent j . Thus, an offline algorithm that assigns all the adversarial items using the $1 - x_j$ fraction of capacities that remain (after allocating the stochastic items according to Alg_{pot}) only loses at most $\max(0, x_j + y_j - 1)$ from agent j when compared to the allocation of adversarial items by Opt_λ . We denote $\max(0, x_j + y_j - 1)$ by z_j , as shown in Inequality 2.

One can see that the numerator of the objective function lower-bounds the expected outcome of Alg and the denominator is the expected outcome of Opt_λ . Thus, the objective function is a lower bound on the competitive ratio of Alg .

It remains only to verify inequality 3. By definition of Alg_{pot} as an algorithm maximizing potential, we know that $Pot(T) \leq Pot(X)$. However, this does not imply the inequality directly. We will show that if the inequality does not hold, we will be able to construct a new allocation with a larger potential, contradicting the definition of X . Suppose by way of contradiction that there exists some T , and some positive number δ such that $Pot_X(T) - Pot(X) \geq \delta$. For some arbitrary small constant η , consider the vector $\eta T + (1 - \eta)X$, which allocates η fraction of each stochastic item according to T and $1 - \eta$ fraction of each stochastic item according to X . By concavity of $Pot_X(\cdot)$ we have

$$Pot_X(\eta T + (1 - \eta)X) \geq \eta Pot_X(T) + (1 - \eta)Pot_X(X) = \eta Pot_X(T) + (1 - \eta)Pot(X).$$

Together with the assumption that $Pot_X(T) - Pot(X) \geq \delta$, this gives $Pot_X(\eta T + (1 - \eta)X) \geq Pot(x) + \eta\delta$. On the other hand, Lemma 3.8 implies that $Pot_X(\eta T + (1 - \eta)X) \leq Pot(x) + \eta\eta^2$. By setting η to be smaller than $\frac{\delta}{\eta}$, we obtain a contradiction. Thus, for any solution T we have $Pot_X(T) \leq Pot(X)$ which is the inequality 3. $\square$

LEMMA 3.11. *The solution of Mathematical Program 4 is lower-bounded by the curve in Figure 3.*

PROOF. Again, as in the unweighted case, given any solution to Mathematical program 4, we find a new solution with a restricted structure that only increases the

objective function by $O(\delta)$. Eventually, we lower bound the objective value of this new solution; this provides a lower bound on the original competitive ratio.

Our restricted solution will satisfy properties 3.12, 3.13, 3.14 and 3.15 below. Similar to the unweighted case, we can replace each variable with $O(1/\delta)$ copies with the same value in order to satisfy Property 3.12, which implies that if we change a constant number of the y_j and / or t_j variables, it changes the objective function by at most $O(\delta)$.

PROPERTY 3.12. *For any positive number δ , we can create an equivalent solution such that $\frac{1}{\sum_{j=1}^n y_j} \leq \delta$ and $\frac{1}{\sum_{j=1}^n t_j} \leq \delta$.*

Again as in the unweighted section, we call an agent j with positive z_j a *harmful* agent. We say that an agent j is a *source* if both z_j and t_j are zero. If an agent is neither harmful nor a source, we call it a *sink*.

PROPERTY 3.13. *All y_j s and t_j s are either 0 or 1.*

To satisfy this property, we add some new extra agents with $x_j = 0, y_j = 0, z_j = 0, t_j = 0$ initially. (Note that this requires slightly modifying the mathematical program to add constraints for the new agents, and including them in the summations. However, it is easy to see that this does not change the objective or feasibility of any constraint.)

Now, for any original agent j with positive y_j and such that j is either a sink or a source, we decrease y_j to 0 and increase y_k by the same amount for some newly added agent k . It is easy to see that this does not affect the sum of the y variables (and hence the objective function or the feasibility of any constraint), and that this can be done so that only a single agent has a fractional value of y .

Let Γ be an integer such that the sum of t_j s for all sinks is in the range $[\Gamma, \Gamma + 1)$. (Note that sources have $t_j = 0$.) We redistribute the t_j values to be 1 on the Γ sources and sinks with the maximum x_j s, leaving at most one sink with fractional t_j . Using the concavity of $Pot_X(T)$, this change only decreases $Pot_X(T)$. Since all sources and sinks now have $y_j = 0$, we still have $t_j + y_j \leq 1$ for each agent j . (Note that some agents may change status from source to sink or vice versa during this process.)

Now, y_j and t_j are either 0 or 1 for all except the harmful agents. (Further, y_j is 1 only on newly added agents, and t_j is 1 only on sink agents.)

For the harmful agents, let Y_H and T_H denote the sum of their y_j and t_j values respectively. From constraint (4), we know that $Y_H + T_H$ is less than the total number of harmful agents. We redistribute the y_j and t_j values as follows: Set $y_j = 1$ on the $\lfloor Y_H \rfloor$ harmful agents with the smallest values of x_j , and 0 otherwise; similarly, set $t_j = 1$ on the $\lfloor T_H \rfloor$ harmful agents with the largest values of x_j , and 0 otherwise. In this process, we lose the fractional y_j and t_j values of at most one agent, possibly increasing the competitive ratio by $O(\delta)$. Further, note that this process leaves some of these agents harmful (those with $y_j = 1$), while the others (all but $\lfloor Y_H \rfloor$ of them) are no longer harmful; $\lfloor T_H \rfloor$ of them now become sinks. There are two further effects of this move: First, as argued above for the sinks, by setting $t_j = 1$ on the agents with highest x_j , $Pot_X(T)$ only decreases. Second, the redistribution of y_j values affects the z_j values. To satisfy constraint (2), we must decrease z_j to 0 on each agent which is no longer harmful. However, we can increase the z_j values on the $\lfloor Y_H \rfloor$ agents which are still harmful, and it is easy to see that this only decreases the objective function (because the increase in these z_j values exceeds the decrease on the agents which are no longer harmful).

Therefore, we satisfy Property 3.13 while increasing the objective by at most $O(\delta)$.

For each harmful vertex i , Inequality (2) says that $z_j \leq x_j$ (since $y_j = 1$). Increasing z_j to x_j only decreases the objective function. Thus, in the rest of the proof, for any

harmful vertex j we assume $z_j = x_j$. Changing x_j of a harmful vertex j implicitly affects z_j as well.

PROPERTY 3.14. *All harmful agents have the same x_j , denoted by α_1 . All sink agents have the same x_j , denoted by α_2 .*

We replace the x_j of all harmful agents with their average. By concavity of the potential function, this does not decrease $Pot(X)$. (Note that all harmful agents j have $t_j = 0$.) It also does not change $\sum_{j=1}^n x_j$ and $\sum_{j=1}^n z_j$, and thus, keeps all the constraints feasible.

We also replace the x_j of sinks with their average. One can rewrite Inequality 3 as $\sum_{j=1}^n (Pot_X(t_j) - Pot(x_j)) \leq 0$. The left term for sink j is $Pot(x_j) + (1 - x_j)(1 - e^{x_j-1}) - Pot(x_j) = (1 - x_j)(1 - e^{x_j-1})$. Thus, for each sink, the left term is a convex function. This means that this change does not increase the left hand side of $\sum_{j=1}^n (Pot_X(t_j) - Pot(x_j)) \leq 0$. This change also does not affect $\sum_{j=1}^n x_j$ and $\sum_{j=1}^n z_j$, and hence keeps all the constraints feasible.

We use β_1 to denote the number of agents j with $y_j = 1$; this includes harmful agents and newly added agents. We use β_2 and β_3 to denote the number of sinks and sources respectively. We use γ to denote $\sum_j x_j$ for sources.

PROPERTY 3.15. $\sum_{i \text{ is source}} Pot(x_i)$ is bounded by $(1 - \frac{1}{e})\gamma - \frac{1}{e}\beta_3$

This follows from the fact that $Pot(x)$ is always less than $(1 - \frac{1}{e})x - \frac{1}{e}$.

Now, using Properties 3.13, 3.14 and 3.15, one can show that the objective function is lower bounded by

$$\frac{(\alpha_1\beta_1 + \alpha_2\beta_2 + \gamma) + (1 - \frac{1}{e})(1 - \alpha_1)\beta_1}{\beta_1 + \beta_2}. \quad (7)$$

(This is a lower bound because in the numerator, we have $\sum (y_j - z_j)$, which is 1 for the newly added agents included in β_1 , and $(1 - \alpha_1)$ for the harmful agents.)

We can write Equality (5) as

$$\lambda(\beta_1 + \beta_2) = \beta_2, \quad (8)$$

and write Inequality (3) as

$$\begin{aligned} &\beta_1(0 - e^{-1}) + \beta_2(\alpha_2 - e^{\alpha_2-1} + (1 - \alpha_2)(1 - e^{\alpha_2-1})) + \beta_3(0 - e^{-1}) \leq \\ &\beta_1(\alpha_1 - e^{\alpha_1-1}) + \beta_2(\alpha_2 - e^{\alpha_2-1}) + (1 - \frac{1}{e})\gamma + \beta_3(0 - e^{-1}). \end{aligned}$$

One can drop the terms $\beta_2(\alpha_2 - e^{\alpha_2-1})$ and $\beta_3(0 - e^{-1})$ from both sides of the inequality, and rearrange as follows:

$$\frac{e}{e-1}(\beta_1(-\frac{1}{e} - \alpha_1 + e^{\alpha_1-1}) + \beta_2(1 - \alpha_2)(1 - e^{\alpha_2-1})) \leq \gamma.$$

We replace γ in the objective function (Equation 7 with the max of zero and left hand side of the inequality. Then, we replace β_1 with $\frac{1-\lambda}{\lambda}\beta_2$ using Equality (8). Now we can cancel out β_2 and simplify the objective function as follows.

$$\begin{aligned} &\frac{\alpha_1}{e}(1 - \lambda) + (1 - \lambda)(1 - \frac{1}{e}) + \lambda\alpha_2 + \\ &\max(0, \frac{e}{e-1}((1 - \lambda)(-\frac{1}{e} - \alpha_1 + e^{\alpha_1-1}) + \lambda(1 - \alpha_2)(1 - e^{\alpha_2-1}))) \end{aligned}$$

Unfortunately, it is hard to solve this analytically to obtain a closed-form expression for the competitive ratio as a function of λ . When $\lambda \geq 0.6882$, we can lower bound the competitive ratio by the line $0.3714 + 0.4362\lambda$. However, this is not necessarily tight. Numerically solving, the curve lower bounding the competitive ratio as a function of λ can be seen in Figure 3 $\square$

3.3. Hardness of Robust Budgeted Allocation

THEOREM 3.16. *No online algorithm for Robust Budgeted Allocation (even in the unweighted case) has competitive ratio better than $1 - \frac{1-\lambda}{e^{1-\lambda}}$ for λ -stochastic inputs.*

Proof Sketch: We describe the instance that shows the desired hardness. For a full proof, please refer to the full version of the paper. Consider the following instance: We have n agents and n items. Initially, all of the agents are *unmarked*. Each item is connected to all agents which are unmarked upon its arrival. The first λn items are connected to all agents. After the arrival of the first λn items, we *mark* the λn agents with minimum [expected] load. Subsequently, after assigning each item we *mark* the agent with the minimum load among all unmarked agents.

Note that all of the first λn items are adjacent to all agents, and there is no uncertainty about them. Thus, one can consider the first λn items as being part of the forecast $F = (D, \lambda n)$, where the distribution D has a single type (items adjacent to all agents). The last $(1 - \lambda)n$ vertices as adversarially chosen items. Let v_j denote the j th agent that we mark. An optimum algorithm that knows the whole graph in advance can match the j th item to the j th agent and obtain a matching of size n . One can verify that no algorithm can obtain a competitive ratio better than $1 - \frac{1-\lambda}{e^{1-\lambda}}$ on this instance.

4. APPROXIMATING ADVERSARIAL AND STOCHASTIC BUDGETED ALLOCATION

In this section, we study a class of algorithms for Budgeted Allocation problem that provide good approximation ratios in both stochastic and adversarial settings. We say an algorithm is (α, β) -competitive if it is α -competitive in the adversarial setting and β -competitive in the stochastic setting.

The best algorithms for the stochastic setting have a competitive ratio of $1 - \varepsilon$ when the input is stochastic. However, these algorithms may have a competitive ratio close to zero when the input is adversarial. On the other hand, the Balance algorithm has the best possible competitive ratio of $1 - \frac{1}{e}$ when the input is adversarial, but, under some mild assumption, is 0.76-competitive for stochastic inputs [Mirrokni et al. 2011]. Indeed, this result holds when, 1) the capacities are large i.e. $\max_{i,j} \frac{w_{i,j}}{c_j} \leq \varepsilon^3$, and 2) the optimum solution is large i.e. $\varepsilon^{-6} \sum_j \max_{i: \text{opt}(i)=j} w_{i,j} \leq \text{Opt}$, where $\text{Opt}(i)$ is the agent the optimum matches to node i , and ε is a vanishingly small number. In this section, we refer to $\max_{i,j} \frac{w_{i,j}}{c_j} \leq \varepsilon^3$ as the large capacities condition, and refer to $\max \left(\varepsilon^{-6} \sum_j \max_{i: \text{opt}(i)=j} w_{i,j}, \varepsilon^{1/3} m \cdot \max_{i,j} w_{i,j} \right) \leq \text{Opt}$ as the large optimum condition.

Of course, one can design algorithms with ‘intermediate’ performance: Simply use the stochastic algorithm with probability p and Balance with probability $1 - p$; this yields an algorithm with competitive ratios of $((1-p)(1 - \frac{1}{e}))$ and $(p + (1-p) \times 0.76) - O(\varepsilon)$ in the adversarial and stochastic settings respectively. These competitive ratios for different values of p lie on the straight line with endpoints $(1 - \frac{1}{e}, 0.76)$ and $(0, 1 - \varepsilon)$. In this section, we use the tools that we developed in Section 2 to obtain competitive ratios those are ‘above’ this straight line.

We devise a mixed algorithm for this setting as follows: Divide the capacity of each agent / offline vertex into two parts. The first part has $1 - p$ fraction of the capacity, and the second part has the remaining p fraction of the capacity. Given a forecast $F =$

(D, f) , at the beginning, run the Balance algorithm on $1-p$ fraction of the forecast input (that is, on the first $(1-p) \cdot f$ items, using the first $(1-p)$ fraction of the capacities. Then, for any $1-\varepsilon$ -competitive stochastic algorithm Alg , run Alg on the rest of the forecast input (that is, the next pf items), using the remaining p fraction of the capacities. If at any time during the algorithm, we observe that the input sequence is ε -far from D , or the length of the input is more than f , we can detect that (w.h.p.) we are in the adversarial setting. We therefore flush out the items we assigned to the second part (the p fraction) of the capacities, if any, and return to the Balance algorithm. However, we now run two independent copies of Balance: The first using $1-p$ fraction of the capacities (with some items previously allocated by Balance), and the second using the remaining p fraction of the capacities (beginning with an empty allocation of items to this part of the capacities). For each item arriving online, with probability $1-p$ we send it to the Balance algorithm on the first part of the capacities, and with probability p we send it to the Balance algorithm on the second part of the capacities.

In the next two lemmas, we bound the competitive ratios of our mixed algorithm when the input is stochastic and adversarial respectively.

LEMMA 4.1. *Assuming the long input, the large capacities and the large optimum conditions, if the input is stochastic, the mixed algorithm has a competitive ratio of $(1-p) * \beta + p - O(\varepsilon)$, where β is 0.76.*

PROOF. From Theorem 2.10, we know that by ignoring the inputs which are ε -far from the distribution, we lose only $O(\varepsilon)$ fraction in the competitive ratio. We use $1-p$ fraction of the input sequence on the first $1-p$ fraction of the capacities. Therefore, the Balance algorithm gets $(1-p) * \beta$ fraction of the optimal solution from this part of the input, where $\beta = 0.76$ is the competitive ratio for Balance on stochastic inputs [Mirrokni et al. 2011]. On the other hand, we use p fraction of the input on the remaining p fraction of the capacities. Since we use a $1-\varepsilon$ competitive stochastic algorithm, we get $p - O(\varepsilon)$ fraction of the optimal solution from this part of the input sequence. $\square$

LEMMA 4.2. *Assuming the long input, the large capacities and the large optimum conditions, if the input is adversarial, the mixed algorithm has a competitive ratio of $\frac{0.76(1-p)}{1+0.202(1-p)} - O(\varepsilon^{1/3})$.*

The following theorem is the immediate result of combining Lemmas 4.1 and 4.2.

THEOREM 4.3. *Assuming the long input, the large capacities and the large optimum conditions, for any $p \in [0, 1]$ the mixed algorithm is $(1-p) * 0.76 + p - O(\varepsilon)$ competitive for stochastic input and $\frac{0.76(1-p)}{1+0.202(1-p)} - O(\varepsilon^{1/3})$ competitive for adversarial input.*

REFERENCES

- AGRAWAL, S., WANG, Z., AND YE, Y. 2009. A dynamic near-optimal algorithm for online linear programming. *Computing Research Repository*.
- ALAEI, S., HAJIAGHAYI, M., AND LIAGHAT, V. 2012. Online prophet-inequality matching with applications to ad allocation. In *Proceedings of the 13th ACM Conference on Electronic Commerce*. ACM, 18–35.
- APPLEGATE, D. AND COHEN, E. 2006. Making routing robust to changing traffic demands: algorithms and evaluation. *IEEE/ACM Transactions on Networking (TON)* 14, 6, 1193–1206.
- AZAR, Y., COHEN, E., FIAT, A., KAPLAN, H., AND RACKE, H. 2003. Optimal oblivious routing in polynomial time. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*. ACM, 383–388.

- BAHMANI, B. AND KAPRALOV, M. 2010. Improved bounds for online stochastic matching. In *European Symposium on Algorithms*. Springer, 170–181.
- BEN-TAL, A. AND NEMIROVSKI, A. 2002. Robust optimization—methodology and applications. *Mathematical Programming* 92, 3, 453–480.
- BERTSIMAS, D., PACHAMANOVA, D., AND SIM, M. 2004. Robust linear optimization under general norms. *Operations Research Letters* 32, 6, 510–516.
- BUCHBINDER, N., JAIN, K., AND NAOR, J. S. 2007. Online primal-dual algorithms for maximizing ad-auctions revenue. In *ESA*. Springer, 253–264.
- DEVANUR, N. AND HAYES, T. 2009. The adwords problem: Online keyword matching with budgeted bidders under random permutations. In *EC*. 71–78.
- DEVANUR, N. R., JAIN, K., SIVAN, B., AND WILKENS, C. A. 2011. Near optimal online algorithms and fast approximation algorithms for resource allocation problems. In *EC*. ACM, 29–38.
- DEVANUR, N. R., SIVAN, B., AND AZAR, Y. 2012. Asymptotically optimal algorithm for stochastic adwords. In *Proceedings of the 13th ACM Conference on Electronic Commerce*. ACM, 388–404.
- FELDMAN, J., HENZINGER, M., KORULA, N., MIRROKNI, V. S., AND STEIN, C. 2010. Online stochastic packing applied to display ad allocation. In *ESA*. Springer, 182–194.
- FELDMAN, J., KORULA, N., MIRROKNI, V., MUTHUKRISHNAN, S., AND PAL, M. 2009a. Online ad assignment with free disposal. In *WINE*.
- FELDMAN, J., MEHTA, A., MIRROKNI, V., AND MUTHUKRISHNAN, S. 2009b. Online stochastic matching: Beating $1 - 1/e$. In *FOCS*. 117–126.
- HAEUPLER, B., MIRROKNI, V., AND ZADIMOGHADDAM, M. 2011. Online stochastic weighted matching: Improved approximation algorithms. In *WINE*. 170–181.
- JAILLET, P. AND LU, X. 2013. Online stochastic matching: New algorithms with better bounds. *Mathematics of Operations Research* 39, 3, 624–646.
- KALYANASUNDARAM, B. AND PRUHS, K. R. 2000. An optimal deterministic algorithm for online b-matching. *Theoretical Computer Science* 233, 1–2, 319–325.
- KARANDE, C., MEHTA, A., AND TRIPATHI, P. 2011. Online bipartite matching with unknown distributions. In *STOC*. 587–596.
- KARP, R., VAZIRANI, U., AND VAZIRANI, V. 1990. An optimal algorithm for online bipartite matching. In *STOC*. 352–358.
- MAHDIAN, M., NAZERZADEH, H., AND SABERI, A. 2007. Allocating online advertisement space with unreliable estimates. In *Proceedings of the 8th ACM conference on Electronic commerce*. ACM, 288–294.
- MAHDIAN, M. AND YAN, Q. 2011. Online bipartite matching with random arrivals: A strongly factor revealing lp approach. In *STOC*. 597–606.
- MEHTA, A., SABERI, A., VAZIRANI, U., AND VAZIRANI, V. 2007. Adwords and generalized online matching. *J. ACM* 54, 5, 22.
- MENSHADI, V. H., OVEIS GHARAN, S., AND SABERI, A. 2011. Online stochastic matching: Online actions based on offline statistics. In *SODA*. 1285–1294.
- MIRROKNI, V. S., OVEIS GHARAN, S., AND ZADIMOGHADDAM, M. 2011. Simultaneous approximations of stochastic and adversarial budgeted allocation problems. In *SODA*. 1690–1701.
- TAN, B. AND SRIKANT, R. 2011. Online advertisement, optimization and stochastic networks. In *CDC-ECC*. IEEE, 4504–4509.
- VEE, E., VASSILVITSKII, S., AND SHANMUGASUNDARAM, J. 2010. Optimal online assignment with forecasts. In *EC*. 109–118.
- WANG, H., XIE, H., QIU, L., YANG, Y. R., ZHANG, Y., AND GREENBERG, A. 2006. COPE: traffic engineering in dynamic networks. *ACM SIGCOMM* 36, 4, 99–110.