Catch Them if You Can: How to Serve Impatient Users

Marek Cygan
Institute of Informatics,
University of Warsaw,
Banacha 2, 02-097 Warszawa,
cygan@mimuw.edu.pl

Matthias Englert
Department of Computer
Science and DIMAP,
University of Warwick,
Coventry CV4 7AL,
englert@dcs.warwick.ac.uk

Anupam Gupta
Department of Computer
Science, Carnegie Mellon
University, Pittsburgh,
PA 15213
anupamg@cs.cmu.edu

Marcin Mucha
Institute of Informatics,
University of Warsaw,
Banacha 2, 02-097 Warszawa,
mucha@mimuw.edu.pl

Piotr Sankowski
Institute of Informatics,
University of Warsaw,
Banacha 2, 02-097 Warszawa,
sank@mimuw.edu.pl

ABSTRACT

Consider the following problem of *serving impatient users*: we are given a set of customers we would like to serve. We can serve at most one customer in each time step (getting value v_i for serving customer i). At the end of each time step, each as-yet-unserved customer i leaves the system independently with probability q_i, never to return. What strategy should we use to serve customers to maximize the expected value collected?

The standard model of competitive analysis can be applied to this problem: picking the customer with maximum value gives us half the value obtained by the optimal algorithm, and using a vertex weighted online matching algorithm gives us $1 - 1/e \approx 0.632$ fraction of the optimum. As is usual in competitive analysis, these approximations compare to the best value achievable by an clairvoyant adversary that knows all the coin tosses of the customers. Can we do better?

We show an upper bound of ≈ 0.648 if we compare our performance to such an clairvoyant algorithm, suggesting we cannot improve our performance substantially. However, these are pessimistic comparisons to a much stronger adversary: what if we compare ourselves to the optimal strategy for this problem, which does not have an unfair advantage? In this case, we can do much better: in particular, we give an algorithm whose expected value is at least 0.7 of that achievable by the optimal algorithm. This improvement is achieved via a novel rounding algorithm, and a non-local analysis.

Categories and Subject Descriptors

F.2.2 [**Theory of Computation**]: Nonnumerical Algorithms and Problems; G.2.2 [**Discrete Mathematics**]: Graph theory—*Graph algorithms*

ITCS'13, January 9–12, 2012, Berkeley, California, USA.

General Terms

Algorithms, Theory

Keywords

online algorithms, stochastic models, bipartite matching, impatience

1. INTRODUCTION

The presence of impatient users is a common phenomenon in many contexts, and in this paper we initiate an algorithmic investigation of impatience. In particular, we consider the following problem of *serving impatient users*: we are given a set of n customers we would like to serve. At each time step, we look at the customers still in the system and choose one of them to serve: if we serve customer i, we get a value of v_i. This customer, once served, departs from the system—we serve each customer only once. Moreover, the customers are impatient: at the end of each time step, each customer still in the system may "time out". We model this timing-out process in a stochastic way. After every timestep each customer i decides to stay in the system independently with probability p_i, and leaves, never to return, with the remaining probability $1 - p_i$. (Hence, customer i remains in the system $T_i \sim \text{Geom}(p_i)$ time steps, or fewer, if she was served earlier.) Our goal: to devise a strategy for serving customers, so that we (approximately) maximize the expected total value obtained.

Not only is this problem interesting in its own right, it also focuses attention on a new set of unexplored problems lying in the intersection of queueing theory/stochastic processes and approximation algorithms. It combines elements from both disciplines: the population evolves according to a natural stochastic process, but the values v_i and probabilities p_i are chosen in a worst-case fashion and not drawn from some underlying distribution themselves. To the best of our knowledge, this problem has not been studied in either literature. However, we believe that the problem is worth studying as it tries to answer a very basic question: what shall we do next when options can disappear? How shall we trade-off the probability of timing-out of the client with its value? Should clients with high value and low impatience be served before clients with low value and high impatience?

We show in this paper that this question does not have a straight forward answer, and even understanding the case when all clients are identical is non-trivial.

From an algorithmic point of view, such a problem naturally lends itself to investigation in the competitive analysis framework, where we compare ourselves to an all-knowing "clairvoyant" adversary who knows the values of the timeouts T_i. A simple online algorithm is to ignore the probabilities p_i, and serve the remaining customer with the highest value v_i, and it is easy to show this is $\frac{1}{2}$-competitive even against the clairvoyant adversary (see Section 2 for proofs of claims in this paragraph). An example with two equal clients shows that in the case of non-stochastic adversarial timeouts, this greedy approach cannot get more then $\frac{1}{2}$ of the optimum. However, even in our much weaker stochastic model greedy is not better—we give an upper bound example of $\frac{1}{2} + \epsilon$, for any $\epsilon > 0$. We can do something better: we can model this process as an online bipartite matching problem with customers on one side, and time steps on the other, where at each timestep we only need to know which customers are still in the system: using the optimal results of Aggarwal et al. [1] for vertex-weighted online matching, we get a $(1 - 1/e)$-competitive algorithm.

However, these results make an unfair comparison to a clairvoyant adversary. Instead, we would like to level the playing field, and compare our algorithm to the best algorithm of its class. This would not be a concern if we could get much better competitive ratios against clairvoyant adversaries. However, our first result shows that this is not the case, even for the simplest setting where all the customers are identical.

Theorem 1.1 *There exist instances with n identical customers, where the optimal algorithm gets at most $0.648\ldots+\varepsilon$ times the value obtained by the optimal clairvoyant algorithm w.p. $1 - e^{-\Omega(\varepsilon^2 n)}$.*

However, if we compare ourselves to the optimal non-clairvoyant algorithm, our next result shows how to obtain a better approximation.

Theorem 1.2 *There is a randomized algorithm that for any instance I, obtains an expected value of at least $(1 - 1/(2e - 2)) \geq 0.709$ times an LP relaxation, and hence at least 70.9% of the non-clairvoyant optimum.*

Our algorithm is based on rounding a linear-programming relaxation of the problem. The naïve rounding of the LP solution gives only a $(1 - 1/e)$-approximation. The improved approximation is based on a novel rounding algorithm that moves LP value between the rounds, and hence avoid lower bounds on a purely local analysis.

1.1 Some Important Notation

In the paper, we will use "items" instead of "customers" or "users", and "picking items" will represent "serving users". Again, the value of item i is v_i, its survival probability is p_i. *Time* begins with $t = 0$, and hence the probability of i being alive at time t (a.k.a. *round* t) is p_i^t. A *clairvoyant* algorithm is one that knows the coin tosses of all the items (and hence when the item dies/disappears), a *non-clairvoyant* or *online* algorithm does not know in time t which items will be survive to round $t + 1$.

1.2 Related Work

There has been a significant body of work on stochastic optimization in the operations research community (see, e.g., [3]). The work from a worst-case analysis and approximation algorithms viewpoint is more recent, less than a decade old, starting with the work on two-stage stochastic covering problems [8, 16, 11, 17]. A prominent class of problems deals with situations where the input is given a set of random variables, the distributions of which are known up-front, but the actual realizations are only revealed upon the algorithm probing them; examples of these are stochastic knapsack [7] and stochastic matching [4]. The challenge in these problems is the hidden stochastic information, whereas our problem is a full-information problem, where the challenge is to cope with the stochastic departure of the clients, and the associated loss in value.

A line of work relevant to our problem is that on online bipartite matching, where vertices on one side are given up front, and vertices on the other side (and the edges between them) are revealed online. This started with the paper of Karp et al. [14] showing that this problem admits a $(1 - 1/e)$-competitive randomized algorithm, and this is the best possible. Aggarwal et al. [1] show the same optimal approximation guarantee for the case where the known vertices have weights, and one wants to maximize the weight of matched nodes. (The connection between this problem and our work is sketched in Section 2.) Better results are known for the case where each arriving vertex is an i.i.d. draw from a known distribution [10] or when the unknown vertices are chosen by an adversary but appear in random order [13, 15].

Another extensively studied related problem is *online bounded-delay buffer management*, which can be cast as vertex-weighted bipartite matching where each vertex is connected to a contiguous set of vertices on the other side. In contrast to our model (a) items arrive over time instead of all in the first step and (b) the deadline of items are known instead of being modelled by a (known) probability distribution. In particular, Chin and Fung [6] prove an upper bound of 4/5 on the competitive ratio of any randomized algorithm for this related problem. On the positive side, Chin et al. [5] gave a $(1 - 1/e)$-competitive randomized online algorithm for this related problem. Jeż [12] also gives a $(1 - 1/e)$-competitive randomized algorithm using different techniques. This later algorithm and proof can be easily adapted to handle our problem, even if items are not all available from the start but arrive online.

Our problem also bears some similarities to problems of serving clients with fixed service times and with impatient customers, that have been studied in the queuing theory and stochastic processes literature [2, 18]. However, the problems studied there have a different flavor, i.e., clients wait in fixed-order queues, but leave the queue according to some distribution. In such models one is interested in analyzing the average queuing time in the steady state of the system, and not the value collected from customers. The question of whom to serve next is not studied in these models.

2. SIMPLE SOLUTIONS

Lemma 2.1 *The greedy algorithm that always picks the item with highest value is $\frac{1}{2}$-competitive against a clairvoyant adversary, and even against a non-clairvoyant adversary it does not do better.*

Proof. We first observe that the value of the items picked by the clairvoyant adversary and not picked by the greedy algorithm cannot be higher then the total value gained by the greedy algorithm. This is because in every step, in which the adversary picks such an item, the greedy algorithm picks an item with at least the same value. Hence, $v(\mathsf{opt}) \leq v(\mathsf{opt} \setminus \mathsf{alg}) + v(\mathsf{alg}) \leq 2v(\mathsf{alg})$, where alg is the set of items picked by the greedy algorithm, opt is the set of items picked by the adversary, and $v(S)$ denotes the total value of items in set S. For the upper bound, consider two items with $p_1 = 1$, $v_1 = 1 + \epsilon$ and $p_2 = 0$, $v_2 = 1$. The greedy algorithm collects $1 + \varepsilon$ value, while an optimal non-clairvoyant algorithm collects $2 + \varepsilon$. ■

As the above proof shows, the greedy algorithm is non-optimal with respect to a non-clairvoyant adversary, even for two items. The following is easy to show.

Lemma 2.2 *The optimal online strategy for two items with values v_1 and v_2 and survival probabilities p_1 andp_2 is to pick item i maximizing $q_i v_i = (1 - p_i)v_i$.*

What this lemma says is that for two items we should minimize the expected lost value, which seems to be a reasonable strategy. One could hope that it is actually the optimal one. However, as the following theorem shows, it can perform arbitrarily bad (proof in Appendix A).

Theorem 2.3 *The strategy of always picking an item maximizing $q_i v_i = (1 - p_i)v_i$ does not guarantee a constant competitive ratio with respect to a non-clairvoyant adversary.*

Call an algorithm *ordering-based* if, for every instance, it follows a fixed ordering of items, i.e. it picks items in a fixed order, skipping over the ones that are no longer alive. An interesting consequence of the proof of Theorem 2.3 is the following.

Corollary 2.4 *There is no ordering-based algorithm that is optimal with respect to a non-clairvoyant adversary.*

Proof. Consider any instance with all $p_i \neq 0$. For any pair of items $(v_1, p_1), (v_2, p_2)$ it is possible, that at some point they are the only two items left (unless one of them is always picked in the first round). Therefore, any optimal ordering-based algorithm needs to always pick the item with highest $(1 - p_i)v_i$ first. It follows that if there existed an ordering based optimal algorithm, it would need to pick items in order of decreasing $(1 - p_i)v_i$ (except for, perhaps, the first item). The example used in the proof of Theorem 2.3 shows that this strategy is not optimal. ■

As mentioned in the introduction, it is possible to be $(1 - 1/e)$-competitive in the case when items are allowed to arrive at arbitrary timesteps. Even the following stronger statement holds.

Lemma 2.5 *Against a clairvoyant adversary, there is a $(1 - 1/e)$-competitive algorithm even in the case of adversarial arrivals and departures of items.*

This follows by [1], when we observe that our model can be interpreted as a vertex weighted bipartite matching problem, where vertices on one side correspond to time rounds, whereas vertices on the other side correspond to items. Moreover, since the algorithm in [1] does not need to know the values of the items before timesteps in which it can be picked, this reduction also works when the items arrive online.

3. AN UPPER BOUND AGAINST CLAIRVOYANT ADVERSARIES

Let us consider the case where there are n items, each with the same survival probability $p = \frac{k-1}{k}$ for an integer value k to be chosen later. (This parameter k will lie in $[n/2, n]$.) We will show that the best clairvoyant algorithm can pick $\approx k(1 + \ln \frac{n}{k})$ items with high probability. Since all items are identical, there is only one reasonable non-clairvoyant algorithm, which is to pick one of the remaining items at each time, and we show that this algorithm gets $\approx k(\ln(1 + \frac{n}{k}))$ items, again with high probability. Hence, the competitive ratio against the clairvoyant adversary is $\ln(1 + \frac{n}{k})/(1 + \ln \frac{n}{k})$, which is minimized for $k \approx n/1.84$, taking on value $0.64842\ldots$ This shows that the $(1 - 1/e)$-competitiveness given by the reduction to vertex-weighted online matching cannot be significantly improved unless we compare to a non-clairvoyant optimum.

3.1 Clairvoyant Algorithms

First, let us give a lower bound on the performance of a clairvoyant algorithm, which knows when the items die, and hence can choose items accordingly.

Theorem 3.1 *For any $k \in [n/2, n]$, and any constant $\varepsilon \in [0, 1]$, there exists a clairvoyant adversary that can pick $(1 - \varepsilon)(k - 1)(1 + \ln(n/k))$ items with probability at least $1 - n\, e^{-\Omega(\varepsilon^2 k)}$.*

Proof. Let $e_1, \ldots, e_n$ be the items in the order in which they die, i.e., e_1 dies first and e_n dies last. Ties between items dying in the same step are broken arbitrarily. Define $\ell = n - k(1 - \varepsilon)p(1 + \ln(n/k))$. At step $t \in [0, n - \lceil \ell \rceil]$, pick item $e_{t+\lceil \ell \rceil}$ (if alive). If all these items are alive, we can collect at least $n - \ell$ items. It remains to show that, with high probability, all items we are trying to pick are indeed alive at the time we pick them.

Fix any step $1 \leq t \leq n - \ell$. We need an upper bound on the probability of the bad event that at least $t + \ell$ items died in steps $0, \ldots, t - 1$. Let X_i be the indicator r.v. for the i^{th} item dying in the first t steps, and let $X = \sum_{i=1}^{n} X_i$. It has expectation $\mathrm{E}[X] = n(1 - p^t)$, and we are interested in $\Pr[X \geq t + \ell]$. Equivalently, defining $\beta := \frac{t+\ell}{\mathrm{E}[X]} - 1$, we wish to upper bound $\Pr[X \geq (1 + \beta)\mathrm{E}[X]]$. To apply a Chernoff bound, we would like bounds for β and $\mathrm{E}[X]$.

Claim 3.2 *$\beta \cdot \mathrm{E}[X] \geq \varepsilon pk = \varepsilon(k - 1)$, for $k \leq n$.*

Proof. Substituting for β, ℓ and $\mathrm{E}[X] = n(1 - p^t)$ we get

$$\begin{aligned}
\beta \cdot \mathrm{E}[X] &= t + \ell - \mathrm{E}[X] \\
&= t + n - k(1 - \varepsilon)p(1 + \ln(n/k)) - n(1 - p^t) \\
&= t + np^t - kp(1 + \ln(n/k)) + \varepsilon kp(1 + \ln(n/k)) \\
&\geq \frac{\ln(-1/(n \ln p)) - 1}{\ln p} - kp(1 + \ln(n/k)) \\
&\quad + \varepsilon kp(1 + \ln(n/k)) \\
&\geq \varepsilon kp(1 + \ln(n/k)) \geq \varepsilon kp \ ,
\end{aligned}$$

where the third line follows from the fact that $t + np^t$ is minimized for $t = \ln(-1/(n \ln p))/\ln p$; the fourth line uses $-1/k = p - 1 \geq \ln p \geq 1 - 1/p = -1/(kp)$, where we replace $\ln p$ in the denominator by $-1/kp$ and the $\ln p$ in the numerator by $-1/k$. The last inequality follows from $k \leq n$. ■

Since $\mathrm{E}[X] = n(1 - p^t) \leq n$ the claim also implies $\beta \geq \varepsilon(k-1)/n \geq \varepsilon/4$ (for $n \geq 4$).

We can now apply a Chernoff bound to get the desired upper bound on $\Pr[X \geq t + \ell]$, which is the probability that, in step t, we can pick the designated item.

$$\Pr[X \geq t + \ell] = \Pr[X \geq (1+\beta)\mathrm{E}[X]] \leq \exp\left(-\frac{\beta^2 \mathrm{E}[X]}{2+\beta}\right)$$
$$\leq \exp\left(-\frac{\beta\varepsilon(k-1)}{2+\beta}\right) \leq \exp\left(-\frac{\varepsilon^2(k-1)}{8+\varepsilon}\right),$$

where the last inequality follows since $\beta \geq \varepsilon/4$. Taking a union bound over the failure probability of at most n steps, completes the proof of Theorem 3.1. ■

3.2 Non-Clairvoyant Algorithm

Since the items are identical and each dies in a memoryless way, there is only one non-clairvoyant algorithm: pick an arbitrary item at each step. We now give bounds on its performance. Recall that $p = \frac{k-1}{k}$ for some $k \in \mathbb{Z}$. Here, we no long require that $k \geq \frac{n}{2}$.

Theorem 3.3 *For small enough $\varepsilon > 0$*

$$\Pr\left[\mathsf{alg} \geq \log_p \frac{k-1}{n+k-1} + \varepsilon k\right] \leq e^{-\Omega(\varepsilon^2 k)}$$

and

$$\Pr\left[\mathsf{alg} \leq \log_p \frac{k-1}{n+k-1} - \varepsilon k\right] \leq e^{-\Omega(\varepsilon^2 k)}.$$

Proof. Consider the following process. We start with a bin of n balls marked *alive* and $\frac{p}{1-p} = k - 1$ balls marked *dead*. In each step, we first mark a single alive ball as dead. (If there are no alive balls to mark at the beginning of a step, we put an additional dead ball into the bin, and call it an *extra* dead ball.) After this, each ball in the bin (alive or dead) is removed from the bin with probability $1 - p$, independently of each other and of past events. Let A_t, D_t and E_t be the number of alive balls, (non-extra) dead balls and extra dead balls, respectively, after t steps. Note that $A_0 = n$, $D_0 = \frac{p}{p-1}$ and $E_0 = 0$.

Consider now two sequences of random variables: $B_t = A_t + D_t$ and $C_t = D_t + E_t$. We can use A_t to model the behaviour of the optimum online algorithm for identical items as follows: we have $\mathrm{E}[\mathsf{alg}] = \mathrm{E}[T]$, where $T = \min\{t \mid A_t = 0\}$. Note that whenever $A_t > 0$, we have $E_t = 0$, and so $B_t > C_t$. Therefore

$$T \leq \min\{t \mid E_t > 0\} = \min\{t \mid C_t > B_t\},$$

and it is this last expression that we are going to bound.

First consider B_t. This does not include the extra balls, so for each of the $B_0 = n + k - 1$ original balls, the probability of surviving t rounds is p^t independent of other balls, giving us $\mathrm{E}[B_t] = p^t B_0$. By a Chernoff bound, for any $0 < \varepsilon < 1$, and any value $\lambda \geq \mathrm{E}[B_t]$, we get

$$\Pr[B_t \geq (1+\varepsilon)\lambda] \leq e^{\frac{-\varepsilon^2\lambda}{3}}. \tag{3.1}$$

Now consider C_t. Here, a single new ball is added at the beginning of each step, and then each ball is removed with probability $1 - p$. Since $C_0 = D_0 = \frac{p}{1-p}$, we have $\mathrm{E}[C_t] = p(1 + \mathrm{E}[C_{t-1}]) = C_0$ by a simple induction. Moreover, each C_t is actually a sum of $C_0 + t$ independent (but not identical) $\{0,1\}$ r.v.s: C_0 for the balls initially in the bin, and a single variable for each new ball. Again using a Chernoff bound, we get for any $0 < \varepsilon < 1$

$$\Pr[C_t \leq (1-\varepsilon)\mathrm{E}[C_t]] = \Pr[C_t \leq (1-\varepsilon)C_0] \leq e^{\frac{-\varepsilon^2 C_0}{2}}. \tag{3.2}$$

To get an estimate of $\min\{t \mid C_t > B_t\}$, let us first identify t_0 such that $\mathrm{E}[C_{t_0}] = \mathrm{E}[B_{t_0}]$. This is just

$$\frac{p}{1-p} = \left(n + \frac{p}{1-p}\right) p^{t_0}.$$

Using $p = 1 - \frac{1}{k}$, we get $\frac{p}{1-p} = k - 1$ and consequently

$$t_0 = \log_p \frac{k-1}{n+k-1}.$$

Consider now $t \geq \lceil t_0 + 3k\varepsilon \rceil$, where $\varepsilon > 0$ is small. We have

$$\mathrm{E}[B_t] = B_0 p^t \leq (k-1)p^{3k\varepsilon} = (k-1)\left(1 - \frac{1}{k}\right)^{3k\varepsilon}$$
$$\leq (k-1)e^{-3\varepsilon} \leq (k-1)(1-2\varepsilon)$$

for small enough ε. We now use the concentration bound for B_t from (3.1) to get

$$\Pr[B_t \geq (1-\varepsilon)(k-1)] \leq \Pr[B_t \geq (1+\varepsilon)(1-2\varepsilon)(k-1)]$$
$$\leq e^{\frac{-\varepsilon^2(1-2\varepsilon)(k-1)}{3}}$$

and that for C_t from (3.2) with $C_0 = k - 1$ to get

$$\Pr[C_t \leq (1-\varepsilon)(k-1)] \leq e^{\frac{-\varepsilon^2(k-1)}{2}}.$$

This implies that

$$\Pr[\min\{t \mid C_t > B_t\} > t_0 + 3k\varepsilon] \leq e^{-\Omega(\varepsilon^2 k)}$$

and gives us the first claim.

An analogous reasoning leads to the second one. ■

Using the fact that $\ln p \leq -1/(kp) = -1/(k-1)$, we infer that the algorithm gets at most $(1+\varepsilon)(k-1)\ln(1+n/k)$ items with high probability. And combining with the result of Theorems 3.1, the competitive ratio of the algorithm versus the clairvoyant adversary is at most

$$\min_{k \in [n/2, n]} \frac{(1+\varepsilon)(k-1)\ln(1+n/k)}{(1-\varepsilon)(k-1)(1+\ln(n/k))}$$
$$= \min_{\alpha \in [1,2]} \frac{(1+\varepsilon)\ln(1+\alpha)}{(1-\varepsilon)(1+\ln\alpha)} \leq 0.64842\ldots + 3\varepsilon.$$

This is very close to the $(1 - 1/e) = 0.63212\ldots$ competitiveness, which shows we cannot improve the result substantially against clairvoyant adversaries. In the next section, however, we show that against the non-clairvoyant optimum, we can do significantly better.

4. AN LP-BASED ALGORITHM

Our algorithm is based on writing an LP relaxation for the problem which captures all non-clairvoyant algorithms, and then rounding it to guide the choices of the algorithm.

4.1 LP formulations and basic properties

Consider the following linear program $LP_x(I)$ for a given instance $I = (v, p)$.

$$\begin{aligned} \max & \sum_{i,t} x_{it} v_i && (LP_x(I)) \\ \text{subject to} & \sum_t x_{it}/p_i^t \le 1 && \forall i \\ & \sum_i x_{it} \le 1 && \forall t \\ & x_{it} \ge 0 \end{aligned}$$

To begin, we claim this linear program is a feasible relaxation of the problem.

Lemma 4.1 *For any instance I of the problem, we have $\mathsf{opt}(I) \le \mathsf{opt}_{LP}(I)$, where $\mathsf{opt}(I)$ is the expected value of the optimal non-clairvoyant algorithm, and $\mathsf{opt}_{LP}(I)$ is the optimal value of $LP_x(I)$.*

Proof. Given any non-clairvoyant algorithm $\mathcal{A}$ with expected total value of V^* on I, we can obtain a feasible solution to $LP_x(I)$ with the same objective value. To do this, set x_{it} to the probability that the algorithm $\mathcal{A}$ picks item i in round t. Since x_{it} are probabilities, $\sum_i x_{it} \le 1$. Also, define y_{it} to be the probability of picking item i in round t *conditioned on item i being alive in round t*: hence $x_{it} = y_{it} \cdot p_i^t$. Since the algorithm can pick i in at most one round, we get $\sum_t y_{it} \le 1$. Finally, the expected value we get is precisely $\sum_{it} x_{it} v_i = V^*$. ■

As in the above proof, it will often be convenient to think of the linear program in terms of the y_{it} variables, so let us record this equivalent LP (called $LP_y(I)$) here:

$$\begin{aligned} \max & \sum_{i,t} y_{it} p_i^t v_i && (LP_y(I)) \\ \text{subject to} & \sum_t y_{it} \le 1 && \forall i \\ & \sum_i y_{it} p_i^t \le 1 && \forall t \\ & y_{it} \ge 0 \end{aligned}$$

Lemma 4.2 *The integrality gap of this LP is at least $(1 - 1/2e)$: i.e., for every constant $\varepsilon > 0$, there are instances where $\mathsf{opt}(I) < (1 - \frac{1}{2e} + \varepsilon)\mathsf{opt}_{LP}(I)$.*

Proof. Consider the instance where we have n items, each with $v_i = 1$ and $p_i = \frac{1}{n}$. By setting $x_{i1} = \frac{1}{n}$, $x_{i2} = \frac{n-1}{n^2}$ and all other $x_{it} = 0$, we obtain a feasible solution to $LP_x(I)$ with value $2 - \frac{1}{n}$. However, any non-clairvoyant algorithm can get unit value in the first round, at most $1-(1-1/n)^n$ expected value in the second round, and at most $1/n$ expected value in all subsequent rounds. For large enough n, this is at most $2 - 1/e + \varepsilon$, which gives us the claimed integrality gap. ■

This integrality gap example is interesting because it shows that no "local" reasoning based on the LP relaxation (i.e., one that considers a single round at a time) can beat the ratio of $1 - \frac{1}{e}$ with respect to non-clairvoyant algorithms. Later on we show how to sidestep this problem. We show how one can introduce interactions between rounds, and impose additional structural constraints on the solutions to $LP_x(I)$, which are not satisfied by the solution considered in the above proof.

Finally, we note that $LP_x(I)$ does not capture clairvoyant algorithms, which is a positive thing, otherwise due to the results of Section 3 we would not be able to obtain competitive ratios better than 0.648 using this LP.

Theorem 4.3 *There exists an instance I, such that the expected total value of an optimal clairvoyant algorithm is strictly larger than $\mathsf{opt}_{LP}(I)$.*

Proof. Consider an instance with two items: one with $v_1 = 1, p_1 = 0$, and the other with $v_2 = 2, p_2 = 1/2$. Here, the optimal clairvoyant algorithm picks item 2 in the first round if and only if it dies before the second round, yielding a value of 2.5. On the other hand, any solution to $LP_x(I)$ has an objective value of at most 2. This is because we have $x_{21} = 0$, $x_{11} + x_{21} \le 1$ and $x_{21} + 2x_{22} \le 1$ and the objective function is the sum of the left hand sides of these two inequalities. ■

4.2 Randomized Rounding: the First Try

The basic approach to rounding $LP_x(I)$ could be to assign each item i to round t using $y_{it} := x_{it}/p_i^t$'s as probabilities. When the algorithm is run, in round t we choose the highest value item that was assigned to this round (and is still alive). The expected total value of such items is equal to $\sum_i y_{it} p_i^t v_i$, exactly the contribution of round t to the objective function of $LP_x(I)$. Of course, we may not be able to obtain this much value—we can only pick one item each round, and more than one item assigned to round t might still be alive in round t. The following lemma gives some insight into the performance of this strategy.

Lemma 4.4 *Let n items have values $v_1 \ge v_2 \ge \ldots \ge v_n \ge 0$ and let the i-th item be available with probability π_i, independently of other items, where $\sum_{i=1}^n \pi_i \le 1$. Let X be the value of the highest valued available item, and 0 if there are no items available. Then*

$$\mathrm{E}[X] \ge \Pr[\textit{at least one item is available}] \cdot \frac{\sum_{i=1}^n \pi_i v_i}{\sum_{i=1}^n \pi_i}.$$

We will prove this lemma (in fact, a generalization of it) shortly, but let us first point out its implications. Adding in a zero value item v_{n+1} with a probability of being available of $\pi_{n+1} := 1 - \sum_{i=1}^n \pi_i$, and using the fact that the probability of at least one item being available is $1 - \prod_{i=1}^{n+1}(1-\pi_i) \ge 1 - \exp(-\sum_{i=1}^{n+1} \pi_i) = 1 - 1/e$, Lemma 4.4 gives us the following corollary.

Corollary 4.5 *With v_i, π_i and X as defined in Lemma 4.4 we have $\mathrm{E}[X] \ge \left(1 - \frac{1}{e}\right) \sum_{i=1}^n \pi_i v_i$.*

Theorem 4.6 *The basic rounding scheme gives us expected value $\left(1 - \frac{1}{e}\right) \cdot \mathsf{opt}_{LP}(I)$, and hence $(1 - 1/e)$ times the expected value of the best non-clairvoyant algorithm. Moreover, this ratio is tight.*

Proof. To bound the competitive ratio, simply use Corollary 4.5 with availability probabilities set to $y_{it} p_i^t$. To show that it is tight, consider an instance I with n identical items and all $v_i = 1$ and all $p_i = 1$. One optimal solution to $LP_y(I)$

is $y_{it} = \frac{1}{n}$ for all $i = 1, \ldots, n$ and $t = 1, \ldots, n$. Rounding this solution using the basic rounding scheme puts us in a classic balls-and-bins scenario, where in expectation only a $\approx \left(1 - \frac{1}{e}\right)$ fraction of the bins are non-empty which proves the claim. ■

This means we need to work harder to get an improvement over $(1 - 1/e)$, which we will do in the next section. But first, let us give the proof of Lemma 4.4.

4.2.1 Proof of Lemma 4.4

For the improved analysis in Section 4.3, it will be useful to prove a slightly more general result. Let n items have values $v_1 \geq v_2 \geq \cdots \geq v_n \geq 0$. Let $S_j = \{1, 2, \ldots, j\}$. Fix a downwards closed family of subsets $\mathcal{I} \subset 2^{[n]}$. Consider the following process, which depends on the (v_i, π_i) values, and the family $\mathcal{I}$. Let the i-th item be *available* with probability π_i, independently of other items, and let A be the (random) set of available items. Pick a subset of the available items as follows: consider items in increasing order of indices, and *pick* item i if (a) it is available and (b) if $S_{i-1} \cap A$, the subset of available elements previously seen, lies in $\mathcal{I}$. (For instance, if $\mathcal{I}$ consists of all k-element subsets, then this process picks the top $k+1$ available elements with largest value—this is the context in which we will use the following lemma.)

Lemma 4.7 *Let V be the value and N be the cardinality of the set picked by the above process. Then*

$$\mathrm{E}[V] \geq \mathrm{E}[N] \cdot \frac{\sum_{i=1}^{n} \pi_i v_i}{\sum_{i=1}^{n} \pi_i}.$$

Proof. Define $\mathcal{B}_i$ as the event that $(S_{i-1} \cap A) \in \mathcal{I}$. Note that i is picked with probability $\pi_i \cdot \Pr[\mathcal{B}_i]$. Moreover, we have the monotonicity property that

$$\Pr[\mathcal{B}_i] \geq \Pr[\mathcal{B}_j] \qquad \forall i \leq j,$$

since $S_{j-1} \cap A \in \mathcal{I} \Rightarrow S_{i-1} \cap A \in \mathcal{I}$. Note that

$$\frac{\mathrm{E}[V]}{\mathrm{E}[N]} = \frac{\sum_{i=1}^{n} v_i \pi_i \Pr[\mathcal{B}_i]}{\sum_{i=1}^{n} \pi_i \Pr[\mathcal{B}_i]} \quad . \tag{4.3}$$

We first consider the case where, for some k, $v_1 = \cdots = v_k = 1$ and $v_{k+1} = \cdots = v_n = 0$. We call these sequences of values *basic*. In this case,

$$\begin{aligned}
\frac{\mathrm{E}[V]}{\mathrm{E}[N]} &= \frac{\sum_{i=1}^{k} \pi_i \Pr[\mathcal{B}_i]}{\sum_{i=1}^{n} \pi_i \Pr[\mathcal{B}_i]} \\
&= \frac{\sum_{i=1}^{k} \pi_i \Pr[\mathcal{B}_i]}{\sum_{i=1}^{k} \pi_i \Pr[\mathcal{B}_i] + \sum_{i'=k+1}^{n} \pi_{i'} \Pr[\mathcal{B}_{i'}]} \\
&\geq \frac{\sum_{i=1}^{k} \pi_i \Pr[\mathcal{B}_i]}{\sum_{i=1}^{k} \pi_i \Pr[\mathcal{B}_i] + \Pr[\mathcal{B}_k] \sum_{i'=k+1}^{n} \pi_{i'}} \\
&\geq \frac{\sum_{i=1}^{k} \pi_i \Pr[\mathcal{B}_k]}{\sum_{i=1}^{k} \pi_i \Pr[\mathcal{B}_k] + \Pr[\mathcal{B}_k] \sum_{i'=k+1}^{n} \pi_{i'}} \\
&= \frac{\sum_{i=1}^{k} \pi_i}{\sum_{i=1}^{k} \pi_i + \sum_{i'=k+1}^{n} \pi_{i'}},
\end{aligned}$$

where the first inequality uses that $\Pr[\mathcal{B}_k] \geq \Pr[\mathcal{B}_{i'}]$ for all $i' > k$, and the second inequality uses the two facts that (a) $\Pr[\mathcal{B}_i] \geq \Pr[\mathcal{B}_k]$ for all $i \leq k$, and (b) reducing the numerator and denominator by the same value causes the fraction to decrease.

Let us look again at the inequality we proved for all basic sequences v_i:

$$\mathrm{E}[V] \geq \left(\sum_{i=1}^{n} \pi_i v_i\right) \cdot \frac{\mathrm{E}[N]}{\sum_{i=1}^{n} \pi_i},$$

Note that if we fix the π_i, then the last term is a constant and the other two: $\mathrm{E}[V]$ and $\sum_{i=1}^{n} \pi_i v_i$ are linear in v_i and non-negative. Therefore, the inequality extends to all non-negative linear combinations of basic sequences, and these are exactly the non-increasing sequences of values. ■

Proof of Lemma 4.4: Use Lemma 4.7 with $\mathcal{I} = \{\emptyset\}$. Hence the process picks the most valuable item, and the expected size $\mathrm{E}[N]$ is equal the probability that at least one item appears. (We note that alternatively, Lemma 4.4 can be proven via the concept of fair contention resolution due to Feige and Vondrak [9].) ■

4.3 Randomized Rounding: Second Time's the Charm

To improve on the previous analysis, the main idea is the following. We want a rounding procedure that takes advantage of a situation where none of the items assigned to some round are picked because none survived. In this case we allow picking, say, two items *during the next round*. (We will actually pick one of these items in this round, "borrowing" from the future.) We show that if we are allowed to take two items in a round with constant probability, then we can obtain more than a $(1 - 1/e)$ fraction of the LP value in expectation.

Note that conditioning on the event that in round $t-1$ no item was picked changes the probabilities for items being assigned to round t. However, we show that these events are positively correlated. In other words, knowing that no item was picked in round $t-1$ can only increase the probability that an item is assigned to round t and survives.

Another technical problem we face is that for some rounds it may be the case that we always pick some item (e.g., this happens in round 0). We artificially decrease the probabilities of assigning items to such rounds to ensure that even in these rounds, with constant probability, no item is picked.

4.3.1 The Algorithm

We first change the LP formulation slightly as follows: we add the constraint that for a special item i', $x_{i'0} = 1$. By iterating over all items i, each time adding a constraint $x_{i0} = 1$ to the linear program $LP_x(I)$, we select the best LP solution among all the n solutions considered. This does not alter the fact that the (new) LP optimum value is still an upper bound for $\mathsf{opt}(I)$.

Before we present the rounding, here is a thought experiment. The natural rounding algorithm would be to assign item i to round t with probability y_{it}, independently of the other items. Let $Y_{i,t}$ be an indicator variable corresponding to an item i being available in step t, i.e., i being assigned to round t and not dying before round t. (Note that the underlying sample space is given by the random allocations of items to rounds, and by the randomness of the items coin flips.) Also, let $\mathcal{A}_t = \{\sum_i Y_{i,t} = 0\}$ be the event that there are no items available in round t (they all either die before this round, or are assigned elsewhere).

Now for the actual algorithm: we preprocess all the probabilities so that for each round t we have $\Pr[\mathcal{A}_t] \geq Q$, where

we set $Q = \frac{1}{2(e-1)} \approx 0.2909\ldots$ More precisely, for each round t we compute the greatest value of $\alpha_t \in (0,1]$ such that $\Pr[\mathcal{A}_t] = \prod_i (1 - \alpha_t\, y_{it}\, p_i^t) \geq Q$. Next, we randomly assign items to rounds so that item i is assigned to round t with probability $\alpha_t y_{it}$. (With probability $(1 - \sum_t \alpha_t y_{it})$, the item i remains unassigned).

When the algorithm is run, in round t we consider the available items in this round: those which were assigned to this round by the random process and are still alive. Among these, we pick the most valuable item. Moreover, for $t > 0$, if more than one item is available, and if no item was picked in round $t-1$, we also pick the second most valuable available item.

Note that this algorithm allows us to pick two items in some rounds, which is not allowed by our model. However, we can implement this as follows: if none of the items assigned to round t survived, we pick the most valuable item that was assigned to round $t+1$ and is still alive. Then, when round $t+1$ comes, we can pick an item again (if any survived), effectively picking two items in this round. Note that in this way we cannot do worse than actually picking two items in round $t+1$. We might do better though, if the item picked in round t were to die after that round. This brings us to a subtle technical issue: To know whether we can pick two items in round $t+2$, we need to know if any of the items assigned to round $t+1$ survived. However, we cannot know this if we picked an item assigned to round $t+1$ in round t, and then no other item survived until round $t+1$. When that happens, we flip a coin for this item and declare it alive/dead in round $t+1$ accordingly. We need to keep these technical issues in mind when analyzing the rounding algorithm.

4.3.2 Analysis

To analyze the algorithm, we need the following improvement over Lemma 4.4, where instead of picking the most valuable item in the round, we pick the two most valuable items (if possible). This is where we make use of the more general Lemma 4.7.

Lemma 4.8 *Let n items have values $v_1 \geq v_2 \geq \ldots \geq v_n \geq 0$ and let the i-th item be available with probability π_i, independently of other items, where $\sum_{i=1}^n \pi_i \leq 1$. Let X be the sum of the values of the two highest valued available item (or less if there are fewer items available). Then*

$$\mathrm{E}[X] \geq \left(2 - \frac{3}{e}\right) \sum_{i=1}^{n} \pi_i v_i.$$

Proof. To prove this, we use Lemma 4.7 with the family $\mathcal{I} = \{\emptyset, \{1\}, \{2\}, \ldots, \{n\}\}$, which ensures that we pick the two most valuable items. In this case $N = \min(Y, 2)$, where Y is the number of items available. It is easy to see that $\mathrm{E}[N]$ is maximized when all the π_i's are equal, in which case

$$\begin{aligned}\mathrm{E}[N] &= \Pr[Y \geq 1] + \Pr[Y \geq 2] \\ &= 2 - 2\Pr[Y = 0] - \Pr[Y = 1] \\ &= 2 - 2\left(1 - \frac{1}{n}\right)^n - \left(1 - \frac{1}{n}\right)^{n-1} \geq 2 - 3/e,\end{aligned}$$

where the inequality holds for all $n \geq 1$, which proves the lemma. The last step follows from Lemma B.1 (see Appendix B). ■

Recall that $Y_{i,t}$ is the probability that item i is available in round t (i.e., it is assigned to round t by the algorithm, and is actually alive in that round); $\mathcal{A}_t$ is the event that no items are available in round t.

Lemma 4.9 *The following facts hold true for the variables $Y_{i,t}$ and events $\mathcal{A}_t$:*

(a) $(Y_{i,t} \mid \mathcal{A}_{t-1}) = (Y_{i,t} \mid (Y_{i,t-1} = 0))$,

(b) $\Pr[Y_{i,t} = 1 \mid \mathcal{A}_{t-1}] = \frac{y_{i,t} p_i^t}{1 - y_{i,t-1} p_i^{t-1}} \geq \Pr[Y_{i,t} = 1]$,

(c) $Y_{i,t}$ for different items i are conditionally independent on $\mathcal{A}_{t-1}$.

Proof. Statement (a) is obvious, (b) follows from (a), and (c) is easy to verify using (b) and its joint version saying that the joint distribution of $Y_{i,t}$ and $Y_{j,t}$ conditioned on $\mathcal{A}_{t-1}$ is the same as their distribution conditioned on $Y_{i,t-1} + Y_{j,t-1} = 0$. ■

Theorem 4.10 *The rounding algorithm of Section 4.3.1 achieves a value of at least $(1 - \frac{1}{2(e-1)})\, LP_x(I)$, i.e., at least 70.9% of the optimum.*

Proof. By linearity of expectations we can analyze the contribution of each round separately and it is enough to prove that in each round we gain at least a $1 - Q$ fraction of its contribution to the objective function of the LP. Let $LP_t = \sum_i x_{it} v_i$. We consider two cases, $\alpha_t < 1$ and $\alpha_t = 1$.

Case I: $\alpha_t < 1$. In this case we know that $\Pr[\mathcal{A}_t] = Q$, and by Lemma 4.4 in round t we gain at least

$$(1 - \Pr[\mathcal{A}_t]) \cdot \frac{\sum_i v_i (\alpha_t\, y_{it}\, p_i^t)}{\sum_i (\alpha_t\, y_{it}\, p_i^t)} \geq (1-Q) \frac{\alpha_t\, LP_t}{\alpha_t} = (1-Q)\, LP_t.$$

Case II: $\alpha_t = 1$. We infer that $t > 0$, since we know that there exists an item i, such that $x_{i,0} = 1$, and consequently $\alpha_0 = 1 - Q$. Therefore with probability $\Pr[\mathcal{A}_{t-1}]$ we are eligible to pick two items in round t (since we picked no items in the previous timestep), and with probability $1 - \Pr[\mathcal{A}_{t-1}]$ we can pick only one item in round t. Let us denote by X_1 and X_2 the value of the most valuable and second most valuable item available in round t respectively. Note that the expected gain of the algorithm in round t equals $\mathrm{E}[X_1] + \mathrm{E}[X_2 | \mathcal{A}_{t-1}] \Pr[\mathcal{A}_{t-1}]$. As in our first approach, where we can always only pick one item, Corollary 4.5 implies that $\mathrm{E}[X_1] \geq (1 - 1/e) LP_t$. Moreover, by Lemma 4.8 we have $\mathrm{E}[X_1] + \mathrm{E}[X_2] = \mathrm{E}[X_1 + X_2] \geq (2 - 3/e) LP_t$. Observe, that it suffices to show $\mathrm{E}[X_2 | \mathcal{A}_{t-1}] \geq \mathrm{E}[X_2]$, to prove that the expected gain generated in round t is at least $(1 - Q) LP_t$, since then

$$\begin{aligned}&\mathrm{E}[X_1] + \mathrm{E}[X_2 | \mathcal{A}_{t-1}] \Pr[\mathcal{A}_{t-1}] \geq \mathrm{E}[X_1] + \mathrm{E}[X_2] \Pr[\mathcal{A}_{t-1}] \\ &= \Pr[\mathcal{A}_{t-1}] \mathrm{E}[X_1 + X_2] + (1 - \Pr[\mathcal{A}_{t-1}]) \mathrm{E}[X_1] \\ &\geq \left(\Pr[\mathcal{A}_{t-1}] \left(2 - \frac{3}{e}\right) + (1 - \Pr[\mathcal{A}_{t-1}]) \left(1 - \frac{1}{e}\right)\right) LP_t \\ &= \left(\Pr[\mathcal{A}_{t-1}] \left(1 - \frac{2}{e}\right) + \left(1 - \frac{1}{e}\right)\right) \\ &\geq LP_t \left(Q \left(1 - \frac{2}{e}\right) + \left(1 - \frac{1}{e}\right)\right) LP_t = (1 - Q) LP_t\,.\end{aligned}$$

Observe that by Lemma 4.9 when conditioning on $\mathcal{A}_{t-1}$ the probability that a given item is available in round t is increasing (property (b)) and the events of corresponding to different items being available in round t are still independent

(property (c)). Therefore in order to prove $\mathrm{E}[X_2|\mathcal{A}_{t-1}] \geq \mathrm{E}[X_2]$ we can use the coupling technique. By standard arguments we can define an auxiliary probability space, such that when a set of items S is available in round t and $\mathcal{A}_{t-1}$ holds, then in the second probability space the set of available items S' is a superset of S. Therefore the second most valuable item in S' is at least as valuable as the second most valuable item in S and consequently $\mathrm{E}[X_2|\mathcal{A}_{t-1}] \geq \mathrm{E}[X_2]$.

Since both cases give us at least $(1-Q)\,LP_t$, the theorem follows. ■

5. CONCLUSIONS

The presence of impatient users is an important aspect of many real-world processes, and we initiate an algorithmic investigation in this paper. In particular, we model the problem of serving impatient users to maximize the expected value accrued, where the impatience of the users is modeled by each user leaving the system based on an independent random Geometric random variable. The first approaches to solving this problem give only a $(1-1/e) \approx 0.63$ approximation, so our approach is to give a more careful rounding process that gets within a 0.709 factor of the non-clairvoyant optimum. We also show that any online algorithm can get only 0.64 factor of the clairvoyant optimum, so our algorithm's comparison to the non-clairvoyant optimum is not only fair but necessary. Several questions remain: can we close the gap between our approximation algorithms and the upper bounds? Can we give algorithms for the case of user arrivals? (For this last problem, it is not difficult to adapt the algorithm from [12] to give an improvement over $1-1/e$ when the survival probabilities are bounded away from 1.)

6. ACKNOWLEDGMENTS

We thank Marco Molinaro and Krzysztof Onak for many interesting discussions in the initial stages of this research. Moreover, the first, fourth and fifth author were partly supported by NCN grant N N206 567940. The fourth author was also supported by ERC StG project PAAl no. 259515. The second author was partly supported by EPSRC award EP/D063191/1 and EPSRC grant EP/F043333/1, while the third author was supported in part by NSF awards CCF-0964474 and CCF-1016799, and by a grant from the CMU-Microsoft Center for Computational Thinking.

7. REFERENCES

[1] G. Aggarwal, G. Goel, C. Karande, and A. Mehta. Online vertex-weighted bipartite matching and single-bid budgeted allocations. In *SODA*, pages 1253–1264, 2011.

[2] F. Baccelli and G. Hébuterne. *On Queues with Impatient Customers*. Rapports de recherche. Institut national de recherche en informatique et en automatique, 1981.

[3] J. R. Birge and F. Louveaux. *Introduction to stochastic programming*. Springer Series in Operations Research. Springer-Verlag, New York, 1997.

[4] N. Chen, N. Immorlica, A. R. Karlin, M. Mahdian, and A. Rudra. Approximating matches made in heaven. In *Proceedings of the 36th International Colloquium on Automata, Languages and Programming: Part I*, ICALP '09, pages 266–278, Berlin, Heidelberg, 2009. Springer-Verlag.

[5] F. Y. L. Chin, M. Chrobak, S. P. Y. Fung, W. Jawor, J. Sgall, and T. Tichý. Online competitive algorithms for maximizing weighted throughput of unit jobs. *Journal of Discrete Algorithms*, 4(2):255–276, 2006.

[6] F. Y. L. Chin and S. P. Y. Fung. Online scheduling with partial job values: Does timesharing or randomization help? *Algorithmica*, 37(3):149–164, 2003.

[7] B. C. Dean, M. X. Goemans, and J. Vondrák. Approximating the stochastic knapsack problem: The benefit of adaptivity. *Math. Oper. Res.*, 33(4):945–964, 2008.

[8] S. Dye, L. Stougie, and A. Tomasgard. The stochastic single resource service-provision problem. *Nav. Res. Logist.*, 50(8):869–887, 2003.

[9] U. Feige and J. Vondrák. The submodular welfare problem with demand queries. *Theory of Computing*, 6(1):247–290, 2010.

[10] J. Feldman, A. Mehta, V. S. Mirrokni, and S. Muthukrishnan. Online stochastic matching: Beating 1-1/e. In *FOCS*, pages 117–126, 2009.

[11] N. Immorlica, D. Karger, M. Minkoff, and V. Mirrokni. On the costs and benefits of procrastination: Approximation algorithms for stochastic combinatorial optimization problems. In *15th SODA*, pages 684–693, 2004.

[12] Ł. Jeż. One to rule them all: A general randomized algorithm for buffer management with bounded delay. In *ESA*, pages 239–250, 2011.

[13] C. Karande, A. Mehta, and P. Tripathi. Online bipartite matching with unknown distributions. In *STOC*, pages 587–596, 2011.

[14] R. M. Karp, U. V. Vazirani, and V. V. Vazirani. An optimal algorithm for on-line bipartite matching. In *STOC*, pages 352–358, 1990.

[15] M. Mahdian and Q. Yan. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing lps. In *STOC*, pages 597–606, 2011.

[16] R. Ravi and A. Sinha. Hedging uncertainty: Approximation algorithms for stochastic optimization problems. In *10th IPCO*, pages 101–115, 2004.

[17] D. B. Shmoys and C. Swamy. An approximation scheme for stochastic linear programming and its application to stochastic integer programs. *J. ACM*, 53(6):978–1012, 2006.

[18] S. Zeltyn and A. Mandelbaum. Call centers with impatient customers: Many-server asymptotics of the m/m/i+ g queue. *Queueing Systems*, 51:361–402, 2005. 10.1007/s11134-005-3699-8.

APPENDIX

A. PROOF OF THEOREM 2.3

Proof of Theorem 2.3: Fix a large constant c. We will define an instance, for which the algorithm that always picks an item with the highest value of $q_i v_i$, obtains in expectation at most $\frac{2}{c}$ of the value obtained by the optimum online algorithm.

Take two sets of items: S_1 contains $n_1 = n$ items, each with value $v_1 = 1$ and survival probability $p_1 = 1 - \frac{1}{k_1}$ where $k_1 = n^{1/c^2}$. S_2 contains $n_2 = n^{1/(2c)}$ items, each with

value $v_2 = c$ and survival probability $p_2 = 1 - \frac{1}{k_2}$ where $k_2 = ck_1 = cn^{1/c^2}$.

Consider first the algorithm which always picks an item with the highest value of $q_i v_i$. Note that this value is the same for both sets, so by perturbing it slightly, we can make the algorithm pick elements from set S_1 until none are left. How many elements will be picked from S_1. By Theorem 3.3 w.h.p. they will end after about

$$\log_{p_1} \frac{n_1}{k_1} \approx n^{1/c^2} \ln n$$

steps. Note that by this time, the expected number of elements of S_2 that are still alive is 1, so this algorithm does not profit from these elements.

Consider now the algorithm that picks elements from S_2 as long as possible. Using Theorem 3.3 again we get that it picks about

$$\log_{p_2} \frac{n_2}{k_2} \approx n^{1/c^2} \ln n \left(\frac{1}{2} - \frac{1}{c}\right)$$

elements. For large c this is almost half of the elements picked by the first algorithm, but the values of elements in S_2 are c, which gives the claim. ■

B. EXPLANATION FOR THE PROOF OF LEMMA 4.8

Lemma B.1 *For all positive integers n*

$$2\left(1 - \frac{1}{n}\right)^n + \left(1 - \frac{1}{n}\right)^{n-1} \leq \frac{3}{e}.$$

Proof. The claim can easily be verified for $n = 1, 2$. To prove it for larger values of n we show that the left-hand side of the claim is an increasing function of n for $n \geq 2$. The claim follows, since the right-hand side is the limit of the left-hand side as $n \to \infty$.

Let us define $x = \frac{1}{n}$ and let $f(x)$ be the value of the left-hand side of the claim. Then $x \in (0, 1]$ and

$$f(x) = 2(1-x)^{\frac{1}{x}} + (1-x)^{\frac{1}{x}-1} = (1-x)^{\frac{1}{x}}\left(2 + \frac{1}{1-x}\right).$$

We need to show that $f'(x) < 0$ for $x \in (0, \frac{1}{2}]$. By differentiating we obtain

$$f'(x) = (1-x)^{\frac{1}{x}} \frac{-3x - 3\ln(1-x) + 2x\ln(1-x)}{x^2(1-x)}.$$

By using the Taylor expansion

$$-\ln(1-x) = \sum_{n=1}^{\infty} \frac{x^n}{n}$$

we see that the numerator in the expression for $f'(x)$ is equal to

$$-\frac{x^2}{2} + \sum_{n=4}^{\infty} x^n \left(\frac{3}{n} - \frac{2}{n-1}\right).$$

Since for $n \geq 4$ we have

$$0 < \frac{3}{n} - \frac{2}{n-1} \leq 0.1$$

we can upper-bound this expression by

$$-\frac{x^2}{2} + \frac{x^4}{10(1-x)} = \frac{x^2(x^2+5x-5)}{10(1-x)},$$

which is negative in the interval $(0, \frac{3\sqrt{5}-5}{2}) \supseteq (0, \frac{1}{2}]$ and hence $f'(x)$ is negative as well, which ends the proof. ■