Selling Banner Ads: Online Algorithms with Buyback

Moshe Babaioff
Microsoft Research
Mountain View, CA 94043
moshe@microsoft.com

Jason D. Hartline
Electrical Engineering and Computer Science
Northwestern University
Evanston, IL 60208
hartline@northwestern.edu

Robert D. Kleinberg
Computer Science
Cornell University
Ithaca, NY 14853
rdk@cs.cornell.edu

ABSTRACT

We initiate the study of online pricing problems in markets with "buyback," i.e., markets in which prior allocation decisions can be revoked, but at a cost. In our model, a seller receives requests online and chooses which requests to accept, subject to constraints on the subsets of requests which may be accepted simultaneously. A request, once accepted, can be canceled at a cost which is a fixed fraction of the request value. This scenario models a market for web advertising, in which the buyback cost represents the cost of canceling an existing contract.

We analyze a simple constant-competitive algorithm for a single-item auction in this model, and we prove that its competitive ratio is optimal among deterministic algorithms. Moreover, we prove that an extension of this algorithm achieves the same competitive ratio in any matroid domain, i.e., when the sets of requests which may be simultaneously satisfied constitute the independent sets of a matroid. This broad class of domains includes, for example, advertising markets in which each request is for a unit of supply coming from a specified subset of the available impressions. We also present algorithms and lower bounds for knapsack domains, i.e., when advertisers request varying quantities of a homogeneous but limited supply of impressions.

1. INTRODUCTION

The problem we study is motivated by the following real scenario in the Display Advertising industry. The large banner ads shown on a popular website such as MSN are sold by negotiated contract up to a year in advance. The impression inventory is diverse, the advertisers are looking to meet campaign limits in terms of demographics targets (who an ad is shown to), temporal targets (when an ad is shown), and location targets (where an ad is shown). Any particular advertiser request may be able to be met in a variety of ways. The ad seller has a complex online optimization task. As advertiser requests arrive the ad seller must decide whether the requests should be accepted or not. When formulated most naturally as an online optimization problem, two aspects make this a near impossible task to do with any reasonable quality guarantee. It is a small market; display advertisers on a major website number in the thousands and big advertisers number in the tens. It is an online market; inventory may be preallocated to advertisers at a low value early. It may not ever be learned that advertisers arriving later would have paid a premium for the same inventory. Absent this knowledge, there is little basis for accepting or declining ad orders placed by advertisers. This reality of the industry is mirrored by online algorithm theory which suggests that online algorithms for limited supply allocation problems suffer from the worst possible competitive ratios — linear in the number of requests.

This paper addresses both the theoretical problem in online algorithms analysis for allocation problems and the practical problem in settings such as Display Advertising, above, and gives algorithms with competitive ratios that are a constant, parameterized by a coefficient representing a required level of commitment. Notice that the scenarios that bring bad competitive ratios are ones where the algorithm commits to serve a request with low value, say \$1, and then an incompatible request arrives with high value, say \$1M. A committed algorithm is stuck. Practically speaking, we cannot think of one real scenario where it is impossible to bump out the \$1 request (perhaps at some cost) to take the \$1M request, unless the \$1 request has already been served.[1] Thus, we are motivated to explicitly model the revocation of a commitment into the model of online allocation and design algorithms that make use of it. Our resulting work gives a theory of online algorithms with buyback.

Of course this kind of contract revocation is not new and in many real settings a possibility of revocation is

[1]The situations modeled in this paper are those in which a seller makes a sequence of commitments to provide service at some later time after the end of the request sequence. All commitments are thus revocable in our model.

Workshop on Ad Auctions '08 Chicago, Illinois USA

explicitly written into agreed upon contracts. Notable instances of this are the airline industry that routinely over-sells economy seats when premium last minute bookings are made, bumps travelers on over-booked flights, and cancels under-booked flights. Even in our motivating example of Display Advertising, it is possible that the actual impression inventory does not meet forecasted estimates. In this case the actual inventory may be over-sold. There are clauses in the advertisers' contracts that specify that MSN will "make good" by substituting alternative, comparable inventory. Taking these kinds of contracts as given, our paper studies the effect of structural constraints, similar to those that arise in the Display Advertising setting, on optimal online admission and revocation policies.

We consider the following model for studying algorithms for online admission with costly revocation. An admitted request of value v can be revoked at any time at a cost proportional to v. Let f represent this *buyback factor*. Then the cost of revocation of a v-valued request is $f \cdot v$. Our algorithm now, faced with a sequence of requests arriving online, must make admission and revocation decisions satisfying the conditions that (a) at any time the set of admitted requests is feasible, (b) a request may only be admitted at the time it first arrives, and (c) a request may be revoked (a.k.a., bought-back) at a penalty of the buyback factor at any time, but once revoked it may not subsequently be readmitted. After the last request arrives and is processed, the set of admitted requests that have not been revoked are served. The total payoff of the algorithm is the cumulative value of these served requests less the cumulative buyback cost, i.e., the cumulative value of the revoked requests scaled by the buyback factor.

One type of structural constraint we may have in the Display Advertising setting is a matching constraint. Advertisers' requests could potentially be satisfied from different impression inventory. Supply constraints imply a matching structure on the set of served requests. We make the simplifying assumption that each request is for a unit of supply which implies that the sets of compatible requests are the *independent sets* of a *transversal matroid*. We give an optimal online algorithm for this problem and more generally for any matroid set system. We prove that our algorithm is $(1 + 2f + 2\sqrt{f(1+f)})$-competitive to the benchmark defined by the cumulative value of the set of requests served by an optimal offline algorithm; moreover, we prove that no online buyback algorithm achieves a better worst-case competitive ratio. Our algorithm's competitive ratio approaches one as as the buyback factor f goes to zero.

Another type of structural constraint that arises in Display Advertising is a knapsack-type constraint. Here advertisers' ad campaigns may require different quantities of inventory. An advertiser may want their request to be all fulfilled or all denied, as either they run their full campaign or no campaign. We make the simplifying assumption that there is a single commodity, e.g., one banner ad is being shown to web traffic that is not demographically or temporally segregated. The quantities and values of the request form an online knapsack problem. We give a deterministic online buyback algorithm that achieves a bi-criteria approximation result. With an assumption on the size of the largest demand relative to the total supply, we can turn this bi-criteria result into a full constant approximation (for constant f) that is not much worse than the best possible by any online algorithm with buyback. With no assumption, standard randomization approaches for knapsack can be used to convert our approximation algorithm to one that that has a constant approximation factor (for constant f) that is bounded away from one; however all online algorithms, even randomized ones, have a competitive ratio that is bounded away from one in this case.

Related Work. Many authors have studied Internet advertising in the context of online algorithms or in the context of knapsack problems. Most of this work deals with sponsored search auctions whereas our work is motivated primarily by problems that arise in selling banner advertisements. There are clear relations between the underlying algorithmic issues in both applications, but also some interesting differences: in sponsored search, the inventory (i.e., queries) arrives online, whereas in banner advertising the demand (i.e., requests from advertisers) arrives online. Here, we briefly review some of the most closely related work in this area. Aggarwal and Hartline [1] modeled advertising auctions as a knapsack problem with strategic bidders, and designed revenue-competitive offline auctions for this problem. Knapsack problems have also been used to model the bid optimization problem in sponsored search auctions (with the knapsack's capacity representing the advertiser's budget) in work by Borgs et al. [5] (which treats the problem of slot selection) and by Rusmevichientong and Williamson [18] (which treats keyword selection). Chakrabarty et al. [7] modeled the bidding problem using online knapsack and gave an online algorithm whose competitive ratio is logarithmic in the ratio of maximum to minimum value density. The same problem was studied, under a random-ordering assumption, by Babaioff et al. [3] in a paper on the "knapsack secretary problem" discussed further below. Ad auctions with non-strategic budget-constrained bidders have also been modeled as an online b-matching problem, the so-called AdWords problem. Randomized algorithms achieving the optimal competitive ratio of $e/(e-1)$ for this problem were given by Mehta et al. [15] and by Buchbinder et al. [6].

One approach that has been followed in the literature in order to overcome the linear lower bound for (randomized) online algorithms facing adversarial input is to assume that the requests come in a *random order*. The classical secretary problem [9] studies how to select online an element with maximum value in a randomly ordered sequence. Dynkin [9] presented an e-competitive algorithm for this problem. The extension to the case that k elements should be selected was studied in [12]; the paper presents an algorithm with competitive ratio tending to 1 as k grows to infinity. Matroid secretary

problems, i.e., those in which one would like to select an independent set of a matroid online, are considered in [3]. That paper presents an $O(\log(k))$-competitive algorithm for general matroids, where k is the rank of the matroid, and some better algorithms for some specific matroids. Constant competitive algorithm for the knapsack secretary problem are presented in [2].

All these papers assume that decisions are irrevocable yet the input comes in a random order. We, on the other hand, allow the algorithm to revoke acceptance decisions with some cost, and assume adversarial order. This sometimes allows for better results, for example, for a fixed buyback factor f we get a constant competitive algorithm for matroids, while the algorithm of [3] has only logarithmic competitive ratio.

We wish to emphasize another difference between the two approaches (random order vs. buyback). Secretary algorithms provide some guarantee on the *expected* payoff of the outcome, but have no guarantee on the quality of the outcome on a specific permutation (the variance might be high). For example, in the classical secretary solution there is a constant probability of failing to get any value. On the other hand, the buyback algorithm for matroids is deterministic and has a performance guarantee for *every* input sequence.

The problem of online knapsack (without removal) was introduced and studied by [13, 14]. Our results for the online buyback knapsack problem relate to papers that study online knapsack problems with the possibility of removing elements that were already accepted to the knapsack. All prior work assumes that such removal has no cost ($f = 0$ in our formulation). [10] presents tight competitive algorithms for online *unweighted* knapsack. Bi-criteria results for the case in which elements can be *fractionally* accepted are presented in [16], while bi-criteria results for the case of integral acceptance are presented in [11].

The idea of a buyback is related to the idea of "opportunistic cancellations" which was presented in [4]. That paper shows that if ones allow cancellation of some deals with a fixed penalty this results in higher efficiency. The paper makes some distributional assumptions, while we work in the framework of worst case competitive analysis. [19] proposes a leveled commitment contracting protocol that allows self-interested agents to efficiently accommodate future events by having the possibility of unilaterally decommitting from a contract by paying some penalty.

Concurrent Work. Constantin et al. [8] have independently and concurrently studied online buyback problems in a model quite similar to ours. To aid the reader consulting both papers we point out the key differences between the two papers. Both papers give buyback algorithms for matroid set systems, the algorithms are essentially the same. While we only consider the algorithmic problem, Constantin et al. focus on game theoretic properties of their algorithm. We prove a lower bound on any deterministic algorithm's performance that exactly matches the bound given by our algorithm, while Constantin et al. only have an implicit definition of a lower bound (as a solution to a set of equations). Finally, we extensively study the model of knapsack with buyback, a problem that is not considered in [8].

Organization. The paper is organized as follows. In Section 2 we start by a warmup example and study the single item case. In Section 3 we give our formal general model for considering constrained online optimization with buyback. In Section 4 we give an optimal algorithm for online buyback for matroid set systems. In Section 5 we consider knapsack set systems and give algorithms that approximate the optimal buyback algorithm.

2. WARMUP: THE SINGLE ITEM CASE

Consider n requests from customers coming online in an arbitrary order. There is a fixed buyback factor $f > 0$. Customer e has value v_e if she receives the item, and if we buy back the item from her later, we need to pay her $f \cdot v_e$ to compensate for revoking our commitment. We next present a $\left(1 + 2f + 2\sqrt{f(1+f)}\right)$-competitive algorithm and a matching lower bound.

2.1 The Algorithm

For the single item case our algorithm is very simple. The algorithm assumes we are given a parameter $r > 1$ whose value may depend on the buyback factor f. We will later see that the optimal competitive ratio is achieved by setting $r = 1 + f + \sqrt{f(1+f)}$. The algorithm operates as follows.

- Accept the first request.
- Consider the rest of the requests in order of arrival. If the value of the arriving request is more than r times the value of currently accepted request, buy back the item and accept the arriving request.

Observe that setting $r = 1 + f$ in the above algorithm corresponds to the very natural idea of accepting a new bid whenever it is profitable to do so. However, when $r = 1 + f$ the algorithm has an unbounded competitive ratio. This is easily seen, for example, by considering a bid sequence defined recursively by $v_1 = 1$, $v_{i+1} = (1 + f)v_i + \varepsilon$ for some arbitrarily small $\varepsilon > 0$. For every $n > 0$, the algorithm's profit after $n + 1$ bids is only $1 + \varepsilon n$, whereas the optimum profit is greater than $(1 + f)^n$.

THEOREM 1. *The single item buyback algorithm has a competitive ratio of $\frac{r(r-1)}{r-1-f}$. For a fixed buyback factor $f > 0$, this function is minimized by setting $r = 1 + f + \sqrt{f(1+f)}$, resulting in the competitive ratio $1 + 2f + 2\sqrt{f(1+f)}$.*

PROOF. We first prove that the competitive ratio is $\frac{r(r-1)}{r-1-f}$. Let m be the request with the highest value, that is $OPT = v_m$. Let w be the final request selected. We claim that $v_w \geq v_m/r$. If $w = m$ then this is trivially true (as $r > 1$). If on the other hand $w \neq m$, then at the time that m was considered and rejected, the item

was allocated to request u satisfying $v_m \leq r \cdot v_u$. As the winner's value monotonically increases, at the end it must be the case that $v_m \leq r \cdot v_w$ and the claim follows.

We next bound the total buyback payment, B. Let $\ell(1), \ell(2), \ldots, \ell(j)$ be the requests who are selected but subsequently bought back, numbered in *decreasing* order of arrival time. For notational convenience we will also define $\ell(0) = w$. For $i = 1, 2, \ldots, j$, the fact that the algorithm chose to buy back the item from $\ell(i)$ at the arrival time of $\ell(i-1)$ implies that $v_{\ell(i-1)} > r \cdot v_{\ell(i)}$. By induction on i we now obtain $v_{\ell(i)} < r^{-i} \cdot v_w$. Now,

$$B = f \cdot \sum_{i=1}^{j} v_{\ell(i)} \leq f v_w \cdot \sum_{i=1}^{j} r^{-i} < f v_w \cdot \frac{1}{r-1}.$$

Thus, the algorithm's profit $v_w - B$ satisfies

$$v_w - B > v_w \left(1 - \frac{f}{r-1}\right) \geq \frac{1}{r}\left(1 - \frac{f}{r-1}\right) v_m,$$

and the competitive ratio $\frac{r(r-1)}{r-1-f}$ follows by rearranging terms. It is now an exercise in calculus to verify that the optimal value of r and the resulting competitive ratio are as stated in the theorem. (See Appendix A.) □

2.2 A Lower Bound

In this section we prove that the competitive ratio $1 + 2f + 2\sqrt{f(1+f)}$ obtained by the algorithm in the preceding section is optimal for deterministic algorithms.

THEOREM 2. *For all $\beta < 1 + 2f + 2\sqrt{f(1+f)}$, there does not exist a β-competitive deterministic online buyback algorithm.*

PROOF. Let us call a finite non-decreasing sequence of numbers $y_1 \leq y_2 \leq \cdots \leq y_n$ an *all-sell sequence* if the algorithm sells to every buyer (after buying back from the preceding buyer) when presented with the input sequence $y_1, y_2, \ldots, y_n$. Denote the set of all such finite sequences by $\mathcal{AS}$. Define an infinite sequence $x_1, x_2, \ldots$ recursively, by stipulating that $x_1 = 1$ and that for $n > 1$,

$$x_n = \inf\{x \geq x_{n-1} | x_1, x_2, \ldots, x_{n-1}, x \in \mathcal{AS}\}. \quad (1)$$

Note that the set on the right side of (1) is non-empty, unless the algorithm has an unbounded competitive ratio.

When presented with the sequence $x_1, \ldots, x_n, (1-\varepsilon)x_{n+1}$, the algorithm sells to every costumer except the last one, and it buys back from every costumer except the last two. Thus its profit is $x_n - f\sum_{i=1}^{n-1} x_i$ — which is positive unless the algorithm has an unbounded competitive ratio — while the optimum profit is $(1-\varepsilon)x_{n+1}$. As this must hold for every $\varepsilon > 0$ and for every n, the β-competitiveness of the algorithm would imply

$$\forall n \quad x_{n+1} \leq \beta\left(x_n - f\sum_{i=1}^{n-1} x_i\right). \quad (2)$$

A non-decreasing sequence of positive real numbers can never satisfy (2) when $\beta < 1 + 2f + 2\sqrt{f(1+f)}$, as we prove in Appendix B. This completes the proof of Theorem 2. □

3. THE GENERAL MODEL

We consider a set $N = \{1, 2, ..., n\}$ of requests coming online, where request k from some customer arrives at time step k. Each request $k \in N$ has a value $v_k \geq 0$ associated with it. When presented with request k at time step k the online algorithm A needs to decide whether to accept request k. At time step k the algorithm holds a set $S^{(k-1)}$ of the requests accepted so far. The algorithm is constrained in the sets it picks. We assume that there is a set system $\mathcal{S}$ (a family of subsets of N) which represents the *feasible sets*. At each step the algorithm can hold the subset $S \subseteq N$ only if it is feasible, that is $S \in \mathcal{S}$. At time step k the algorithm can pick subsets, each satisfying $T \subseteq \{1, ..., k\}$, and check if $T \in \mathcal{S}$. This constraint on the queries models the fact that the algorithm has no knowledge about the future and cannot tell if future elements can be picked with the current elements or not.

Unlike many other online algorithms (such as algorithms for secretary problems, e.g. [9]) in which decisions of the algorithm are irrevocable, we allow the algorithm to revoke an acceptance decision with some cost, and we call it a "buyback". (Note that once a request is rejected it can never be accepted later.) That is, if the algorithm decides to reject a request that was already accepted it has to compensate the customer who issued that request. We assume that this compensation is proportional to the value of the request; the algorithm is given a buyback factor f and if it decides to revoke the acceptance decision on customer i it needs to compensate customer i with payment $f \cdot v_i$. At time k the algorithm has to pick a subset $S^{(k)} \subseteq S^{(k-1)} \cup \{k\}$ such that this subset is feasible, that is $S^{(k)} \in \mathcal{S}$. The algorithm pays $f \cdot v_i$ to each customer $i \in S^{(k-1)} \setminus S^{(k)}$ (a request that was previously accepted and is now going to be rejected).

The goal of the algorithm is to maximize the total value of the requests picked, minus the cost of the buybacks. For a set $S \subseteq N$ let $v(S) = \sum_{i \in S} v_i$ be the value of the set S. Let $F = S^{(n)}$ be the final set accepted by the algorithm and let R be the set of requests that were bought-back, that is, requests that were accepted and later rejected. Note that $R = \cup_{k=1}^{n} S^{(k)} \setminus F$ and we denote the total buyback cost by $B = f \cdot v(R)$. Thus the payoff of the algorithm is $\mathit{Payoff}(A) = v(F) - f \cdot v(R) = v(F) - B$. Our goal is to design algorithms that have good competitive ratio with respect to the optimal offline solution. An optimal offline solution is a set $OPT \subseteq N$ such that $OPT \in \mathcal{S}$ and $v(OPT) \geq v(S)$ for every other $S \in \mathcal{S}$. The competitive ratio of algorithm A with buyback is thus $v(OPT)/\mathit{Payoff}(A)$.

We would like to emphasize that an online algorithm with buyback needs to be competitive for **any** set of requests and **any** arrival order for these requests, i.e., an adversarial model. This is unlike the case of secretary problems [9] in which there is an assumption that requests come in a random order. The algorithm is given the power of costly revocation of its acceptance commitments, and this turns out to be very powerful.

Our paper focuses on two important set systems for

which buyback is natural. First we consider the case that the set system $\mathcal{S}$ constitutes the independent sets of a matroid structure[2] on ground set N. (Thus we sometimes identify requests with elements of the ground set of the matroid.)

Second, we consider online knapsack problems. In this case we are selling a divisible good of total supply C (corresponding to the capacity of the knapsack). Each customer i is a buyer with value v_i for s_i units of the good. A set of requests is feasible if the total demand of the set does not exceed the supply. Formally, for $S \subseteq N$ it holds that $S \in \mathcal{S}$ if and only if $\sum_{i \in S} s_i \leq C$.

4. AN ALGORITHM FOR MATROIDS

Recall that matroid set systems generalize many relevant allocation problems. The uniform matroid of rank k represents feasible sets of winners of a k-unit allocation problem. The transversal matroid represents feasible sets of winners when there is a matching constraint. Transversal matroids exemplify an important aspect of the display advertising problem: that certain advertisers are only interested in certain impressions (e.g., based on demographic, time of day, web page, etc.).

In this section we present a deterministic online buyback algorithm for matroids with competitive ratio of $1+2f+2\sqrt{f(1+f)}$. This matches the lower bound from Theorem 2, which holds even for the uniform matroid of rank 1, i.e. the matroid whose nonempty independent sets are all singletons.

In the special case where buyback factor $f = 0$, the online matroid buyback problem is trivial. Readers familiar with matroids will recall that the following very simple procedure selects an independent set with maximum total value. Iterate over the elements in an arbitrary order, in particular, the order of arrival. If the next arriving element is *independent* from the current set of selected elements then select it. Otherwise, if it can be *exchanged* with a selected element of lesser value then exchange it with such an element with the smallest value: remove the lesser valued element and select the arriving element.

When $f > 0$ we generalize the above algorithm by exchanging elements only if the value of the arriving element is at least r times the value of the smallest element it can be exchanged with. The resulting Matroid Buyback Algorithm, parameterized by r, is formalized below. (The best choice of r is identical to that in Section 2.)

We remind the reader with some notations. Recall that $S^{(k)}$ denotes the set S after k elements have been considered. OPT denotes a basis of maximal weight. Finally, recall that F denotes the final set picked by the algorithm, and B denotes the total buy-back payment.

We will begin with some lemmas culminating in a

[2]A *matroid* $(\mathcal{U}, \mathcal{I})$ is constructed from a ground set $\mathcal{U} \neq \emptyset$ and a nonempty family $\mathcal{I}$ of subsets of $\mathcal{U}$, called the *independent* subsets of $\mathcal{U}$, such that if $B \in \mathcal{I}$ and $A \subseteq B$ then $A \in \mathcal{I}$ ($\mathcal{I}$ is hereditary). Additionally, if $A, B \in \mathcal{I}$ and $|A| < |B|$, then there is some element $x \in B \setminus A$ such that $A \cup \{x\} \in \mathcal{I}$ (exchange property).

Algorithm 1 Matroid Buyback Algorithm

```
1: Given: a parameter r > 1.
2: Initialize S = ∅.
3: for all elements e, in order of arrival, do
4:    if S ∪ {e} ∈ I then
5:       Sell to e.
6:    else
7:       Let e' be the element of smallest value such that
         S ∪ {e} \ {e'} ∈ I.
8:       if v_e ≥ r · v_{e'} then
9:          Sell to e.
10:         S = S ∪ {e} \ {e'}.
11:         Pay back f · v_{e'} to the buyer e'.
12:      end if
13:   end if
14: end for
```

lower bound on the value of the final set picked by the algorithm (Lemma 7) and an *upper* bound the total buyback payment (Lemma 8). Theorem 9, establishing the competitive ratio of Algorithm 1, will follow immediately from these two lemmas.

We first recall some general definitions from matroid theory.

DEFINITION 3. *If $M = (\mathcal{U}, \mathcal{I})$ is a matroid and $T \subseteq \mathcal{U}$, then* $\mathrm{rank}(T)$ *denotes the cardinality of a maximal independent subset of T. We say that T* spans *an element $u \in \mathcal{U}$ if* $\mathrm{rank}(T \cup \{u\}) = \mathrm{rank}(T)$. *The* closure *of a set T is* $\mathrm{cl}(T) = \{u \in \mathcal{U} \mid T \text{ spans } u\}$.

Next we establish some notations needed for the analysis of Algorithm 1.

DEFINITION 4. *For a set of elements $T \subseteq \mathcal{U}$ and a threshold value $\theta \geq 0$, let $T(\theta)$ be the subset of T consisting of elements whose weight is at least θ, and let $\mathrm{cl}_\theta(T)$ be the set spanned by these elements. In other words, $T(\theta) = \{t \in T | v_t \geq \theta\}$ and $\mathrm{cl}_\theta(T) = \mathrm{cl}(T(\theta))$.*

The following two lemmas are crucial in comparing the value of F to the value of OPT.

LEMMA 5 (SPANNING LEMMA). *For every $\theta \geq 0$, if an element is spanned by elements of $S(\theta)$ at some stage of the algorithm, then it is spanned by elements of $S(\theta)$ at all later stages of the algorithm. In other words, for any $k \leq \ell$ it holds that $\mathrm{cl}_\theta(S^{(\ell)}) \supseteq \mathrm{cl}_\theta(S^{(k)})$.*

PROOF. The proof is by induction on ℓ, the base case $\ell = k$ being trivial. To prove the induction step we must show that $\mathrm{cl}_\theta(S^{(\ell+1)}) \supseteq \mathrm{cl}_\theta(S^{(\ell)})$. If $S^{(\ell+1)} \supseteq S^{(\ell)}$ then we are done. Otherwise, $S^{(\ell+1)} = S^{(\ell)} \cup \{e\} \setminus \{e'\}$ for two elements e, e'. If $e' \notin S^{(\ell)}(\theta)$ then $S^{(\ell+1)}(\theta) \supseteq S^{(\ell)}(\theta)$ and we are again done, so we may assume henceforth that $v_{e'} \geq \theta$. Because the algorithm reached Step 10, we know that the condition in Step 4 evaluated to "false", i.e. $S^{(\ell)} \cup \{e\} \notin \mathcal{I}$. This implies that $S^{(\ell)} \cup \{e\}$ contains a unique minimal *dependent* set $C \notin \mathcal{I}$, and that $e \in C$. (See, e.g., [17] Proposition 1.1.6.) For every element $e'' \in S^{(\ell)} \cup \{e\}$, the set $T(e'') = S^{(\ell)} \cup \{e\} \setminus \{e''\}$ is dependent if $e'' \notin C$, because $C \subseteq T(e'')$ in that case,

and $T(e'')$ is independent if $e'' \in C$ because every dependent subset of $S^{(\ell)} \cup \{e\}$ must contain C. Looking at the definition of e' in Step 7 of the algorithm, we see now that e' must be the minimum-weight element of C. Consequently $v_{e''} \geq \theta$ for every $e'' \in C$, so

$$e' \in \mathrm{cl}(C \setminus \{e'\}) = \mathrm{cl}_\theta(C \setminus \{e'\}) \subseteq \mathrm{cl}_\theta(S^{(\ell+1)}).$$

As e' is the unique element of $S^{(\ell)}$ which does not belong to $S^{(\ell+1)}$, we see now that $\mathrm{cl}_\theta(S^{(\ell+1)})$ contains $\mathrm{cl}_\theta(S^{(\ell)})$, as desired. □

The following Lemma is well known [20].

LEMMA 6 (MATCHING LEMMA). *Let B_1, B_2 by any two bases of the matroid M. There exists a perfect matching between elements of B_1 and B_2 such that if $b_1 \in B_1$ is matched to $b_2 \in B_2$ it holds that $B_2 \cup \{b_1\} \setminus \{b_2\}$ is a basis of M.*

Using the two preceding lemmas, we can relate the value of the final set picked by the algorithm to the value of OPT as follows.

LEMMA 7. $v(F) \geq \frac{1}{r} \cdot v(OPT)$

PROOF. Let $n = |\mathcal{U}|$. Recall that the final set is $F = S^{(n)}$. By the Matching Lemma (Lemma 6) there exists a perfect matching between OPT and F such that each element $e \in OPT$ is matched to an element $\tilde{e} \in F$ such that $e \notin \mathrm{cl}(F \setminus \{\tilde{e}\})$. (Note that this includes the possibility that $\tilde{e} = e$.) We show that $v_e \leq r \cdot v_{\tilde{e}}$, which is clearly sufficient to prove the lemma.

Let k be the arrival time of e, and let $\theta = v_e/r$. We consider two cases. If e was not picked by the algorithm it must be the case that $e \in \mathrm{cl}_\theta(S^{(k)})$. If, on the other hand, e was picked by the algorithm, it holds that $e \in \mathrm{cl}_\theta(S^{(k+1)})$. In both cases by the Spanning Lemma (Lemma 5) it must be the case that $e \in \mathrm{cl}_\theta(S^{(n)})$. Now, observe that e is matched to some element $\tilde{e}$ in F such that $e \notin \mathrm{cl}(S^{(n)} \setminus \{\tilde{e}\})$. This means that $e \notin \mathrm{cl}_\theta(S^{(n)} \setminus \{\tilde{e}\})$ but we know that $e \in \mathrm{cl}_\theta(S^{(n)})$.

Now assume in contradiction that $v_{\tilde{e}} < \theta$. This means that $\mathrm{cl}_\theta(S^{(n)} \setminus \{\tilde{e}\}) = \mathrm{cl}_\theta(S^{(n)})$. This yields a contradiction as on one hand $e \notin \mathrm{cl}_\theta(S^{(n)} \setminus \{\tilde{e}\})$ and on the other hand $e \in \mathrm{cl}_\theta(S^{(n)})$. □

To bound the total buy-back cost paid by the algorithm, we use the following lemma.

LEMMA 8. $B \leq \frac{f}{r-1} \cdot v(F)$

PROOF. For each element $e \in F$ we look at the chain of elements that were bought back in the process of getting e, and we charge all their buy-back costs to e. To make the argument more precise, let $e_0 = e$. For $i \geq 1$ let e_i be the element that was replaced by e_{i-1}. Let $v_i = v_{e_i}$. By induction on i, the buy-back cost of e_i is at most $f \cdot v/r^i$, thus the total buy-back cost for the set $\{e_i \mid i \geq 1\}$ is at most

$$\sum_{i=1}^{\infty} f \cdot \frac{v}{r^i} = f \cdot v \cdot \sum_{i=1}^{\infty} \frac{1}{r^i} = f \cdot \frac{v}{r-1}$$

The result follows directly from summing over all elements in F. □

THEOREM 9. *For any r it holds that $v(F) - B \geq \frac{r-1-f}{r(r-1)} \cdot v(OPT)$. That is, the algorithm achieves a competitive ratio of $\frac{r(r-1)}{r-1-f}$. For a fixed $f > 0$, this function is maximized by setting $r = 1 + f + \sqrt{f(1+f)}$, resulting in the competitive ratio $1 + 2f + 2\sqrt{f(1+f)}$.*

PROOF. By Lemma 8, $v(F) - B \geq \frac{r-1-f}{r-1} \cdot v(F)$. By Lemma 7, $v(F) \geq \frac{1}{r} \cdot v(OPT)$. Combining the two we get $v(F) - B \geq \frac{r-1-f}{r(r-1)} \cdot v(OPT)$. Now the optimal choice of r and the resulting competitive ratio follow from Lemma 17. □

5. RESULTS FOR KNAPSACKS

The online knapsack buyback problem exemplifies another very relevant aspect of display advertising: advertisers may want to either run all of their campaign or none of it. When there is a limited number of impressions this constraint yields a knapsack like problem. Unlike in the matroid setting, even with a buyback factor of $f = 0$, there is no online buyback algorithm achieves the optimal offline performance (proof to follow). We begin by presenting a deterministic algorithm for the knapsack buyback problem. We then present a lower bound for deterministic algorithms. Finally, we discuss a randomized algorithm and a lower bound for such algorithms.

5.1 Deterministic Upper Bounds

In this section we present a deterministic algorithm for the knapsack problem with buyback. Our first result is a bicriterion result. We show that if the largest element is of size at most γ times the knapsack capacity, where $0 < \gamma < 1$, then we get a competitive ratio of $1 + 2f + 2\sqrt{f(1+f)}$ with respect to the optimum solution for a knapsack whose capacity is scaled down by a factor of $(1 - 2\gamma)$. Then we derive a simple corollary about the competitive ratio when $\gamma < 1/2$.

Conventions. We consider a set $U = \{1, \ldots, n\}$ of requests presented online. Request $k \in U$ has size s_k, value v_k, and density (a.k.a., bang-per-buck) v_k/s_k (we sometimes identify the request with an item of the appropriate size and value). We will assume without loss of generality that the requests are totally ordered by density (e.g., by using request indices to break ties among requests of equal density). Let γ be the ratio of the largest request size to the capacity of the knapsack. Unless otherwise stated, we normalize the capacity of the knapsack to $C = 1$; therefore all requests have size at most γ. For a set $S \subseteq U$ let $\mathrm{OPT}_C(S)$ be the value of the optimal offline fractional knapsack solution with capacity C on requests S. Consider the greedy algorithm that sorts requests in S by density and accepts the densest requests until the next object exceeds the knapsack capacity. Let $\mathrm{dens}_C(S)$ denote the density of this request, or $\mathrm{dens}_C(S) = 0$ if the total size of requests in S does not exceed C. Let $\mathrm{greedy}_C(S)$ denote the requests accepted, and let $\mathrm{reject}_C(S)$ denote the remaining requests (i.e., $\mathrm{reject}_C(S) = S \setminus \mathrm{greedy}_C(S)$).

Algorithm. Our knapsack algorithm admits requests following the greedy fractional offline algorithm on a knapsack with restricted capacity $C = 1 - 2\gamma$, but penalizes requests which are not yet in the knapsack by a multiplicative factor of $r > 1$. When the knapsack's capacity is exceeded, it buys back items greedily, starting from the ones with the lowest bang-per-buck, until the remaining items fit the capacity constraint once again.

Algorithm 2 Knapsack Buyback Algorithm

1: **Given:** parameters $r > 1$, $0 < \gamma < 1/2$.
2: Initialize $S^{(0)} = \emptyset$.
3: **for all** requests k **do**
4: **if** the density of request k is at least $r \cdot \text{dens}_{1-2\gamma}(S^{(k-1)})$ **then**
5: /* (Integrally) greedily buyback cheapest requests so as to fit request k, i.e., buyback all requests of $\text{reject}_{1-s_k}(S^{(k-1)})$ to make room for request k. */
6: Add request k to knapsack.
7: Set $S^{(k)} \leftarrow \text{greedy}_1(S^{(k-1)} \cup \{k\})$.
8: Buy back all requests of $S^{(k)} \setminus S^{(k-1)}$
9: **else**
10: $S^{(k)} \leftarrow S^{(k-1)}$.
11: **end if**
12: **end for**
13: Output $F \leftarrow S^{(n)}$.

LEMMA 10. *When elements U are presented to the Knapsack Buyback Algorithm, the set of requests F selected satisfies* $\text{OPT}_{1-2\gamma}(F) \geq \text{OPT}_{1-2\gamma}(U)/r$.

PROOF. Let $U' = \cup_{k=1}^{n} S^{(k)} \subseteq U$ be the set of all requests that are ever admitted to the knapsack in Step 6, including the ones that get bought back in Step 7. Let S' be the largest prefix of U' that fits completely within the knapsack (i.e., $S' = \text{greedy}_1(U')$).

First, $S' = F$. To see this, note that the algorithm never buys back an element $j \in U'$ unless it has already bought back all elements of smaller density, and it never buys back an element which currently fits in the knapsack. Thus elements of S', once added into the knapsack, will never be bought back, implying that $S' \subseteq F$. The reverse inclusion follows from the fact that the most dense element $j \in F \setminus S'$ would have to satisfy (i) $S' \cup \{j\}$ fits the capacity constraint, (ii) j has higher density than any element of $U' \setminus S'$. If such an element existed, it would belong to $\text{greedy}_1(U')$, contradicting the definition of S'.

Consider the requests F^* (resp. U^*) that are selected by the optimal (offline) fractional knapsack solution with capacity $C = 1 - 2\gamma$ on F (resp. U). Each of these sets possibly contains one fractional request. For this analysis, replace the fractional request by an integral one with same density, whose size is scaled down so that the sum of all request sizes in F^* (resp. U^*) is exactly $1 - 2\gamma$. In other words, if F^* (resp. U^*) contains a fractional amount β of a request with size s and value v, then for this analysis we will instead treat the fractional request as an integral request of size βs and value βv. Thus, according to this convention, $\text{OPT}_{1-2\gamma}(U) = \sum_{k \in U^*} v_k$ and $\text{OPT}_{1-2\gamma}(F) = \sum_{k \in F^*} v_k$.

We now show that the total value in $F^* \setminus U^*$ is at least $1/r$ times that of $U^* \setminus F^*$. Using the fact that $F = S' = \text{greedy}_1(U')$, and that the largest request size is γ, we know that F^* is the optimal fractional knapsack solution with capacity $C = 1 - 2\gamma$ on U'. Thus, F^* contains all requests from U^* that are ever admitted to the knapsack, so the requests in $U^* \setminus F^*$ were not admitted to the knapsack and thus have density less than $r \cdot \text{dens}_{1-2\gamma}(F^*)$. Of course, the total capacity consumed by requests in $F^* \setminus U^*$ and $U^* \setminus F^*$ is identical. Thus the value of requests in $F^* \setminus U^*$ is at least $1/r$ times the value of the requests in $U^* \setminus F^*$. Thus the value of all requests in F^* is at least $1/r$ times the value of requests in U^*. This proves the lemma. □

LEMMA 11. *The total amount spent on buyback is at most* $\frac{f}{r-1} \text{OPT}_{1-2\gamma}(F)$

PROOF. We use an accounting scheme where when a request is admitted to the knapsack, we charge it with the fraction of any requests that henceforth exceed the capacity $1-\gamma$. By the time such a request is (integrally) bought back by the algorithm because some fraction of it exceeds the capacity 1 we will have already charged the entirety of its buyback cost to admitted requests.

To make the analysis simpler we initially fill the knapsack up to capacity $1-\gamma$ with requests of value zero (and buyback cost zero). Now, if request k of size s_k is admitted to the knapsack, the request receives a charge equal to the buyback cost of s_k fractional units of requests with the least density. This maintains the invariant that the capacity of requests whose buyback is uncharged remains constant at $1-\gamma$. When a fractional buyback of request j is charged to another request k, we also charge k for the sum of all the charges that j ever received, scaled by a factor equal to the fraction of request j that was bought back.

Notice that if request k is admitted to the knapsack then its density is at least $r \cdot \text{dens}_{1-2\gamma}(S^{(k-1)})$. Any requests with fractional buyback charged to k have density at most $\text{dens}_{1-2\gamma}(S^{(k-1)})$. Thus, the new buyback cost charged to item k is at most $\frac{v_k}{r}$. Request k inherits buyback charges previously assessed to these requests. The total buyback charged to k is thus at most

$$v_k f \sum_{i=1}^{\infty} \frac{1}{r^i} = \frac{v_k}{r-1}.$$

When the algorithm ends, the final (fractional) knapsack from F with capacity $1-2\gamma$ has charged buyback of all requests (integrally) bought back by the algorithm to requests in F (as discussed, we have actually overcharged some requests that have not been bought back yet). The total cost of these buybacks is at most

$$\frac{f}{r-1} \text{OPT}_{1-2\gamma}(F).$$

□

THEOREM 12. *The Knapsack Buyback Algorithm with parameter r has net payoff (value less buyback cost) of*

at least

$$\frac{r-1-f}{r(r-1)}\,\mathrm{OPT}_{1-2\gamma}(U).$$

For a fixed f the best way to pick r is to set $r = 1+f+\sqrt{f(1+f)}$. For such an r the net payoff is at least

$$\frac{1}{1+2f+2\sqrt{f(1+f)}}\,\mathrm{OPT}_{1-2\gamma}(U)$$

PROOF. By Lemma 10 it holds that $\mathrm{OPT}_{1-2\gamma}(F) \geq \mathrm{OPT}_{1-2\gamma}(U)/r$. By Lemma 11 the buyback cost B is at most $\frac{f}{r-1}\,\mathrm{OPT}_{1-2\gamma}(F)$ and as $F \subseteq U$ it holds that $\mathrm{OPT}_{1-2\gamma}(F) \leq \mathrm{OPT}_{1-2\gamma}(U)$ thus $B \leq \frac{f}{r-1}\,\mathrm{OPT}_{1-2\gamma}(U)$. We conclude that for the Knapsack Buyback Algorithm A it holds that

$$\begin{aligned} \mathit{Payoff}(A) &\geq \mathrm{OPT}_{1-2\gamma}(F) - B \\ &\geq \frac{\mathrm{OPT}_{1-2\gamma}(U)}{r} - \frac{f}{r-1}\,\mathrm{OPT}_{1-2\gamma}(U) \\ &= \frac{r-1-f}{r(r-1)}\,\mathrm{OPT}_{1-2\gamma}(U) \end{aligned}$$

Now the optimal choice of r and the resulting competitive ratio follow from Lemma 17. □

Note that the above result can be viewed as a bi-criterion result: the approximation is with respect to an optimal offline fractional solution for a knapsack of size $1-2\gamma$. With an additional loss of factor $1/(1-2\gamma)$ in the approximation ratio, we can clearly get a bound with respect to the optimal solution for the original knapsack size for the case that $\gamma < 1/2$. This is because $\mathrm{OPT}_{1-2\gamma}(U) \geq (1-2\gamma)\,\mathrm{OPT}_1(U)$, which follows from the definition of $\mathrm{OPT}_C(U)$ as the optimal **fractional** solution.

COROLLARY 13. *Assume $\gamma < 1/2$. The Knapsack Buyback Algorithm with $r = 1+f+\sqrt{f(1+f)}$ has net payoff (value less buyback cost) of at least*

$$\frac{1-2\gamma}{1+2f+2\sqrt{f(1+f)}}\,\mathrm{OPT}_1(U)$$

5.2 A Lower Bound for Deterministic Algorithms

We next present a simple lower bound for the problem: for deterministic algorithms (such as ours) there is no hope to get any competitive buyback algorithm when γ is large (say 1). For the proof see Appendix C.

THEOREM 14. *For any f when $\gamma = 1$ no deterministic buyback algorithm for knapsack problems has a bounded competitive ratio that is a function of f.*

5.3 Randomized Upper and Lower Bounds

We can overcome this negative result and achieve a competitive algorithm that works for any γ using randomization. Consider the following randomized algorithm A': with probability 1/3 we run the Knapsack Buyback Algorithm (Algorithm 2) and with probability 2/3 we run the buyback algorithm for a single item (Section 2.1).

THEOREM 15. *For any γ the randomized algorithm A' has an expected net payoff (value less buyback cost) of at least*

$$\frac{1}{3(1+2f+2\sqrt{f(1+f)})}\,\mathrm{OPT}_1(U).$$

PROOF. Define $O_{1/2} = \{u \in \mathrm{OPT}_1(U)\ s.t.\ s_u \geq 1/2\}$ to be the set of requests in $\mathrm{OPT}_1(U)$ with size at least 1/2. Let $O = \mathrm{OPT}_1(U) \setminus O_{1/2}$. Let $\delta(f) = 1+2f+2\sqrt{f(1+f)}$. Note that when A' runs the Knapsack Buyback Algorithm it gets at least $O/\delta(f)$ and when A' runs the Approximately Greedy Algorithm from Section 4 it gets at least $O_{1/2}/(2\delta(f))$ (the lost of factor 2 is due to the fact that $|O_{1/2}| \leq 2$ and the algorithm is competitive with the respect to the best request of $O_{1/2}$). We conclude the the expected payoff of A' is at least

$$\frac{1}{3}\cdot\frac{O}{\delta(f)} + \frac{2}{3}\cdot\frac{O_{1/2}}{2\delta(f)} = \frac{\mathrm{OPT}_1(U)}{3\delta(f)}$$

□

Note that our bi-criteria result shows that when f approaches 0, the Knapsack Buyback Algorithm (Algorithm 2) is close to being perfectly competitive (the competitive ratio goes to 1) with respect to the optimal knapsack solution with capacity $1-2\gamma$ (not 1). On the other hand the above randomized algorithm has competitive ratio of at least 3, even when f is 0. This raises the following question: can a randomized knapsack buyback algorithm have a competitive ratio (with respect to the optimal algorithm on the full size of the knapsack) that approaches 1 when f goes to 0? We next show that the answer for this question is no, and losing some constant in the competitive ratio is inevitable.

THEOREM 16. *Any randomized buyback algorithm for knapsack problems has a competitive ratio of at least 5/4. This is true for any $f \geq 0$, even $f = 0$.*

PROOF. We present two inputs such that for at least one of these inputs any randomized buyback algorithm has competitive ratio of at least 5/4. In both cases the capacity C is 1. The first input has $s_1 = 1, v_1 = 1$ and $s_2 = 1/2, v_2 = 1/2$. The second input is the same as the first with an additional request $s_3 = 1/2, v_3 = 1$. Let p be the probability that the algorithm picks the second request (clearly when faced with the second request the algorithm cannot distinguish the two inputs). The optimal offline algorithm gets a value of 1 on the first input (by picking the first request) and a value of 3/2 on the second input (by picking the second and third requests). The payoff of the algorithm on the first input is at most $(1-p)\cdot 1 + p\cdot 1/2 = 1-p/2$, while its payoff on the second input is at most $(1-p)\cdot 1 + p\cdot 3/2 = 1+p/2$. Thus the competitive ratio in the first case is at least $1/(1-p/2)$ and while in the second case it is at least $(3/2)/(1+p/2)$. It is easy to verify that the optimal way for the algorithm to pick p is to set $p = 2/5$ (in order to minimize the maximal of the two ratios). In this case the ratio is at least 5/4. Thus it is at least 5/4 for any p the algorithm picks. □

6. REFERENCES

[1] Gagan Aggarwal and Jason D. Hartline. Knapsack auctions. In *SODA*, pages 1083–1092, 2006.

[2] Moshe Babaioff, Nicole Immorlica, David Kempe, and Robert Kleinberg. A knapsack secretary problem with applications. In *APPROX-RANDOM*, pages 16–28, 2007.

[3] Moshe Babaioff, Nicole Immorlica, and Robert Kleinberg. Matroids, secretary problems, and online mechanisms. In *SODA*, pages 434–443, 2007.

[4] Eyal Biyalogorsky, Ziv Carmon, Gila E. Fruchter, and Eitan Gerstner. Research note: Overselling with opportunistic cancellations. *Marketing Science*, 18(4):605–610, 1999.

[5] C. Borgs, J. Chayes, O. Etesami, N. Immorlica, K. Jain, and M. Mahdian. Dynamics of bid optimization in online advertisement auctions. In *Proceedings of the 16th International World Wide Web Conference*, 2007.

[6] Niv Buchbinder, Kamal Jain, and Joseph (Seffi) Naor. Online primal-dual algorithms for maximizing ad-auctions revenue. In *ESA'07*, pages 253–264, 2007.

[7] D. Chakrabarty, Y. Zhou, and R. Lukose. Budget constrained bidding in keyword auctions and online knapsack problems. In *WWW2007, Workshop on Sponsored Search Auctions*, 2007.

[8] R. Constantin, J. Feldman, S. Muthukrishnan, and M. Pál. Online ad slotting with cancellations. In *Forth Workshop on Ad Auctions*, 2008.

[9] E. B. Dynkin. Optimal choice of the stopping moment of a Markov process. *Dokl. Akad. Nauk SSSR*, 150:238–240, 1963.

[10] Kazuo Iwama and Shiro Taketomi. Removable online knapsack problems. In *ICALP '02*, pages 293–305, London, UK, 2002. Springer-Verlag.

[11] Kazuo Iwama and Guochuan Zhang. Optimal resource augmentations for online knapsack. In *APPROX-RANDOM*, pages 180–188, 2007.

[12] Robert Kleinberg. A multiple-choice secretary algorithm with applications to online auctions. In *SODA*, pages 630–631 (electronic), New York, 2005. ACM.

[13] George S. Lueker. Average-case analysis of off-line and on-line knapsack problems. In *SODA*, pages 179–188, 1995.

[14] A. Marchetti-Spaccamela and C. Vercellis. Stochastic on-line knapsack problems. *Math. Program.*, 68(1):73–104, 1995.

[15] Aranyak Mehta, Amin Saberi, Umesh Vazirani, and Vijay Vazirani. AdWords and generalized online matching. *J. ACM*, 54(5), 2007.

[16] John Noga and Veerawan Sarbua. An online partially fractional knapsack problem. In *ISPAN '05: Proceedings of the 8th International Symposium on Parallel Architectures,Algorithms and Networks*, pages 108–112, Washington, DC, USA, 2005. IEEE Computer Society.

[17] James Oxley. *Matroid Theory*. Oxford University Press, 1992.

[18] P. Rusmevichientong and D.P. Williamson. An adaptive algorithm for selecting profitable keywords for search-based advertising services. In *EC'06*, 2006.

[19] Tuomas Sandholm and Victor R. Lesser. Advantages of a leveled commitment contracting protocol. In *AAAI/IAAI, Vol. 1*, pages 126–133, 1996.

[20] Alexander Schrijver. *Combinatorial Optimization*, volume B. Springer, 2003.

APPENDIX

A. TECHNICAL LEMMA

LEMMA 17. *For any given $f > 0$ the function $h(r) = \frac{r-1-f}{r(r-1)}$ for $r \in (1,\infty)$ is maximized at $r^* = 1 + f + \sqrt{f(1+f)}$ and its maximum is $h(r^*) = 1+2f+2\sqrt{f(1+f)}$.*

PROOF. We would like to pick r in order to maximize $h(r)$. Taking the derivative of $h(r)$ with respect to r, we obtain

$$h'(r) = \frac{r(r-1) - (r-1-f)(2r-1)}{r^2(r-1)^2}$$

This equals 0 if and only if r satisfies the quadratic equation $r(r-1)-(r-1-f)(2r-1) = 0$. The two roots of this equation are $r_1 = 1 + f + \sqrt{f(1+f)}$ and $r_2 = 1+f-\sqrt{f(1+f)}$. By checking the second derivative we find that $r = r_1 = 1 + f + \sqrt{f(1+f)}$ maximizes $h(r)$. Now let $s = \sqrt{f(1+f)}$. For $r = 1 + f + \sqrt{f(1+f)} = 1 + f + s$, the competitive ratio is

$$\begin{aligned} \frac{r-1}{r-1-f} &= \frac{(1+f+s)(f+s)}{s} \\ &= \frac{2s^2 + s(1+2f)}{s} \\ &= 2s + (1+2f) \\ &= 1 + 2f + 2\sqrt{f(1+f)} \end{aligned}$$

□

B. A LOWER BOUND FOR BUYBACK ALGORITHMS

LEMMA 18. *For every infinite non-decreasing sequence of positive real numbers $x_1 \le x_2 \le \ldots$, if $\beta < 1 + 2f + 2\sqrt{f(1+f)}$ then there exists some n such that*

$$x_{n+1} > \beta(x_n - f\sum_{i=1}^{n-1} x_i)$$

PROOF. The proof is by contradiction. Assume that there is a nonempty set S of sequences $x_1 \le x_2 \le \ldots$ of positive reals satisfying

$$x_{n+1} \le \beta x_n - \beta f \sum_{i=1}^{n-1} x_i \tag{3}$$

for all $n \ge 1$. For any such sequence $\mathbf{x} \in S$, let $n(\mathbf{x})$ be the least n such that the inequality (3) is strict, or $n(\mathbf{x}) = \infty$ if there is no such n. We claim that $n(\mathbf{x})$ takes unboundedly large values as $\mathbf{x}$ ranges over S. Indeed, assume to the contrary that $\mathbf{x} \in S$ is a sequence such that

$$n(\mathbf{x}) = N = \max_{\mathbf{w}\in S} n(\mathbf{w}).$$

Let

$$\lambda = \frac{x_{N+1}}{\beta x_N - \beta f \sum_{i<N} x_i},$$

which is less than 1 by assumption. Then the sequence

$$\mathbf{x}' = \lambda x_1, \lambda x_2, \ldots, \lambda x_N, x_{N+1}, x_{N+2}, \ldots$$

belongs to S, and it satisfies $n(\mathbf{x}') > N$, contradicting our choice of N.

Having thus established that $n(\mathbf{x})$ takes unboundedly large values as $\mathbf{x}$ ranges over the elements of S, we may conclude that the sequence defined by

$$y_1 = 1$$
$$y_{n+1} = \beta y_n - \beta f \sum_{i=1}^{n-1} y_i \qquad \text{for } n \geq 1$$

is non-decreasing. (To prove that $y_{m+1} \geq y_m$ for every m, let $\mathbf{x} \in S$ be a sequence such that $n(\mathbf{x}) > m$ and observe by induction that $y_i = x_i/x_1$ for $i = 1, 2, \ldots, m+1$. Since every sequence $\mathbf{x} \in S$ is non-decreasing, we obtain the inequality $y_{m+1} \geq y_m$ as claimed.) Now define $z_n = \sum_{i=1}^{n} y_i$, and observe that the recursion defining y_{n+1} implies that

$$z_{n+1} = (1+\beta) z_n - \beta(1+f) z_{n-1}.$$

This is a linear recursion, whose general solution is $z_n = as^n + bt^n$ where a, b are arbitrary constants and s, t are the roots of the quadratic equation

$$u^2 - (1+\beta)u + \beta(1+f) = 0. \tag{4}$$

The discriminant of this quadratic is $(1+\beta)^2 - 4\beta(1+f)$, which is strictly negative given our hypothesis that $1 < \beta < 1 + 2f + 2\sqrt{f(1+f)}$. Hence the two roots s, t are complex conjugates with nonzero imaginary part. We claim also that a, b are complex conjugates. To see this, note that

$$1 = z_1 = a + b$$
$$1 + \beta = z_2 = as + b\bar{s}$$

which implies that the linear system

$$w_1 + w_2 = 1$$
$$sw_1 + \bar{s}w_2 = 1 + \beta$$

is solved by $(w_1, w_2) = (a, b)$ and also by $(w_1, w_2) = (\bar{b}, \bar{a})$. Since the linear system is nonsingular, it has a unique solution. Hence $(a, b) = (\bar{b}, \bar{a})$, confirming our claim that a, b are complex conjugates.

At this point we have established that there are complex numbers a, s such that $z_n = as^n + \bar{a}\bar{s}^n = 2\Re(as^n)$ for all n, and s has nonzero imaginary part. Write $a = qe^{i\phi}, s = re^{i\theta}$ where $q, r > 0$ and θ is not an integer multiple of π. Interchanging a, s with $\bar{a}, \bar{s}$ if necessary, we may assume without loss of generality that $0 < \theta < \pi$. Now we have $as^n = qr^n e^{i(\phi + n\theta)}$ which has negative real part if $(2m + \frac{1}{2})\pi < \phi + n\theta < (2m + \frac{3}{2})\pi$ for some integer m. In particular, letting m be the least integer such that $(2m + \frac{1}{2})\pi > \phi$ and recalling that $0 < \theta < \pi$, we find that there must be some $n > 0$ such that $(2m + \frac{1}{2})\pi < \phi + n\theta < (2m + \frac{3}{2})\pi$, implying that $\Re(as^n) < 0$. This contradicts the fact that $z_n > 0$ for every n. □

C. A LOWER BOUND FOR DETERMINISTIC KNAPSACK ALGORITHMS: THE PROOF

In this section we prove Theorem 14.

PROOF. Assume in contradiction that there is an algorithm that is β-competitive for some $\beta = \beta(f) \geq 1$ (β is a function of f). We present a set of inputs and show that the algorithm cannot be β-competitive on at least one of the inputs. We pick an integer $m = \lceil 3\beta^2 \rceil$. We define $m+1$ inputs as follows. All inputs have the first request with $v_1 = 1, s_1 = 1$. For $k \in \{0, ..., m\}$, input k has k requests all with size $1/m$ and value $2\beta/m$. That is, input k has $s_i = 1/m$ and $v_i = 2\beta/m$ for $i \in \{2, ..., k+1\}$. Clearly for the algorithm to be β-competitive it must pick the first request, otherwise it fails of the first input (with only one request). Thus we can assume that the algorithm picks the first request. If it does not pick any other request on the $m+1$ input ($k = m$) it gets net profit of 1 while the optimal offline algorithm pick all requests of size $1/m$ and has net profit of $2\beta > \beta$, a contradiction. Thus there is a $k' \in \{2, ..., m+1\}$ such that the algorithm buys-back the first request and picks the request k'. Now consider the k' input. As the algorithm is deterministic it must behave the same on this input. On this input the algorithm has profit of at most $2\beta/m \leq (2\beta)/(3\beta^2) = 2/(3\beta) < 1/\beta$ while the optimal offline algorithm has profit of at least 1, by picking the first request. □