Online Resource Allocation Problems*

Patrick Jaillet† Xin Lu§

January 2011

Abstract

In this paper, we consider an online resource allocation problem, where buyers with limited budgets are interested in purchasing items that consume a limited amount of resources. In general, there is no non-trivial performance guarantee for the worst cases. However, after imposing two reasonable restrictions, we can propose two online algorithms, a greedy algorithm and a primal dual algorithm, with non-trivial performance guarantees. We further show that these two algorithms are asymptotically the best among all algorithms, either deterministic or randomized, in terms of competitive ratios.

1 Introduction

Online optimization has attracted wide attention in computer science, operations research and management science communities, because it has various applications in many different practical problems, such as search engine auction problems [11]. In practice, data are not revealed at the beginning, but instead come in an online fashion. For example, in the Google AdWords problem, keywords arrive sequentially in real time. After observing a keyword, Google must decide immediately and irrevocably what advertisement to display to maximize its revenue.

Some special classes of resource allocation problems have been studied extensively, including b-matching problems [9] and AdWords problems [3][5]. The best possible algorithms have been proposed for these problems. For general linear programming problems with more restrictions on instances, near optimal algorithms have recently been proposed. In this paper, we will take an approach between these two: a more general class of problems will be considered; and fewer constraints will be imposed on instances.

Here, we consider an online resource allocation problem, whose introduction is motivated in part by the consideration of several applications arising within the Internet (e.g., sponsored search auctions, load balancing and distributed caching in content-delivery networks, and on-demand video requests) with the following inherent basic characteristics:

- a set of N buyers, who are indexed by i, are interested in purchasing items from a set of K object types, which are indexed by k;
- requests arrive one by one and their types become known upon their arrivals;
- buyer i is willing to pay (the operating entity) an amount f_{ik} to buy a request of type k;
- buyer i has a total available budget a_i;
- all items of object type k can use up to a total of b_k amount of dedicated resources from the operating entity;
- s_{ik} amount of resources is consumed away from b_k, when one unit of object type k is assigned to buyer i;
- the operating entity either turns down the request, or allocates it to one buyer upon the arrival of any request
- the operating entity receives revenue corresponding to the fee that buyer i was willing to pay, if one request is assigned to that bidder;

*Research funded in part by ONR, grant N00014-09-1-0326 and NSF, grant #1029603.

†Laboratory for Information and Decision Systems, Department of Electrical Engineering and Computer Science, and Operations Research Center, MIT, Cambridge, MA, 02139; jaillet@mit.edu

§Operations Research Center, MIT, Cambridge, MA, 02139; luxin@mit.edu

- the goal is to assign requests to buyers so as to maximize the total revenue of the operating entity, while respecting the various constraints.

These characteristics are relevant for modeling many other applications in various settings such as finance (e.g., execution over time of buying or selling very large orders in a volatile and unpredictable real-time market), yield management (e.g., room allocation in hotels, seat allocation in airlines, trains), and online auctions (e.g., many key processes underlying operations such as those from eBay).

1.1 Our Results and Techniques

In this paper, we will use competitive analysis to measure the performance of online algorithms. Let OPT(I) be the optimal value for a maximization problem with an instance I, the competitive ratio of an online algorithm is defined as

$$\min_{\mathrm{I}}\{\frac{\mathrm{Revenue}_{\mathrm{online}}(I)}{\mathrm{OPT}(I)}\}.$$

An online algorithm is called the best if no other algorithm has a strictly better competitive ratio.

In general, we can show by construction that no algorithm has a strictly positive competitive ratio. However, after imposing two reasonable restrictions, we propose two algorithms based on different ideas that have similar competitive ratios:

Theorem 1. *Our online algorithms are asymptotically* $1/2$*-competitive, if* $f_{ik} = s_{ik}, \forall i, k$ *and* $\max\{f_{ik}/a_i, f_{ik}/b_k\}$ *goes to 0.*

One algorithm that we propose is a greedy algorithm. Items are always assigned to the bidder who can pay the most. The other algorithm is based on the idea of a primal-dual method. It uses an integer programming formulation:

$$\begin{array}{lll}
\text{Maximize:} & \sum\limits_{i,j,k} f_{ik}x_{ijk} & \\
\text{Subject to:} & \sum\limits_{i,k} x_{ijk} \le 1 & \forall j \\
& \sum\limits_{j,k} f_{ik}x_{ijk} \le a_i & \forall i \\
& \sum\limits_{i,j} s_{ik}x_{ijk} \le b_k & \forall k \\
& x_{ijk} = 0 & \forall \text{ request } j \text{ not of type } k \\
& x_{ijk} \in \{0,1\} & \forall i, j
\end{array} \tag{1}$$

and its dual problem to make tradeoffs between the short term revenue and the long term revenue.

Furthermore, we argue that these two algorithms are the best possible ones in an asymptotical sense by constructing instances.

1.2 Related Work

Many papers have focused on assignment problems in an offline fashion. Shmoys and Tardos [13] presented an LP-rounding 2-approximation algorithm for a special case of the generalized assignment problem (GAP). Chekuri and Khanna [4] developed PTAS for a special case of GAP called the multiple knapsack problem. They classified the APX-hard special cases of GAP. Given an approximation algorithm for a single-bin problem, Fleischer et al. [7] proposed an LP-rounding algorithm and a simple local search algorithm for SAP.

Matching problems and assignment problems in an online setting have been extensively studied in the last two decades. Karp, Vazirani and Vazirani [16] studied the online bipartite matching problem. They proposed a randomized algorithm RANKING with a competitive ratio of $1 - 1/e$ and proved its optimality. Kalyanasundaram and Pruhs [9] studied an online b-matching problem, where each bidder has a budget of b, and each item has unit price. They proposed a deterministic algorithm BALANCE with a competitive ratio of $1 - \frac{1}{(1+\frac{1}{b})^b}$. They further showed that this algorithn is asymptotically optimal for deterministic algorithms. For the same problem, Goel and Mehta [8] analyzed the performance of a greedy algorithm with queries

arriving in a random permutation. They proved a tight competitive ratio of $1 - 1/e$. Mehta et al. [11] and Lahaie et al. [10] considered the AdWords problem, where each bidder has a budget and each i! tem has a price. By introducing a potential function to find the tradeoff between acting greedily and allowing bidders to bid in the future, they proposed an optimal deterministic algorithm with competitive ratio of $1 - 1/e$. For the same problem, Buchbinder et al. [3] gave a simple primal-dual algorithm by using weak duality, which is $(1 - 1/e)$-competitive, and thus optimal. The same technique was applied to other problems [2] as well.

Several different issues in Adwords auctions have also been studied. Aggarwal et al. [1], Edelman et al. [5] and Varian [15] studied properties of different auction mechanisms. Some papers focus on the optimization problems from advertisers' perspectives. Rusmevichientong and Williamson [12] considered how to change the set of interested keywords dynamically to maximize revenue. Feldman et al. [6] proposed an algorithm to maximize the number of bids won by an advertiser within his budget.

2 Upper Bounds

2.1 Inhomogeneous Cases

Intuitively, in order to maximize the revenue, a request tends to be assigned to a bidder who is willing to pay more so that more revenue will be obtained. On the other hand, if there are far more requests than the capacities, because of the capacity constraints, we would like to use capacities more efficiently, i.e. make more money per unit of capacity. In short, a good assignment should make both f_{ik} and f_{ik}/s_{ik} large. However, for problems in general where there is no consistency on the relation between f_{ik} and s_{ik}, it may be impossible to make both both f_{ik} and f_{ik}/s_{ik} large at the same time. In fact, we can show that any deterministic algorithm can be arbitrarily bad for problems in general:

Theorem 2. *$\forall m \in \mathbb{N}$, no deterministic algorithm has a ratio $c \geq 2/m$.*

Proof. Assume Algorithm A is a $2/m$-competitive deterministic algorithm.

Consider the following instance: there are $m + 3$ bidders and only 1 request type, $\forall i \in \{1, 2, \cdots, m + 3\}, a_i = \infty, f_{i1} = 1/m^{i-1}, s_{i1} = 1/m^{2i-2}$, $b_1 = m$. There are m^{2J-1} requests, where $J \in \{1, \cdots, m + 2\}$ will later be determined by the adversary depending on how the online algorithm behaves.

$\forall j \in \{1, 2, \cdots, m+2\}$, let $x_{i,j}$ be the numbers of requests assigned to bidder i after first m^{2j-1} requests, if applicable. Note that, after first m^{2j-1} requests, the optimal revenue is m^j (all requests are assigned to bidder j). Thus, to keep the algorithm $2/m$-competitive, the revenue generated by Algorithm A should be at least $2m^{j-1}$, i.e. $\sum_{i=1}^{j} \frac{x_{i,j}}{m^{i-1}} + m^{2j-1} \cdot \frac{1}{m^j} \geq 2m^{j-1}$; otherwise, we can find a contradiction by letting $J = j$. Since decisions are irrevocable, $\forall i, x_{i,1} \leq x_{i,2} \leq \cdots \leq x_{i,m+2}$, and thus $\sum_{i=1}^{j} \frac{x_{i,m+2}}{m^{i-1}} \geq \sum_{i=1}^{j} \frac{x_{i,j}}{m^{i-1}} \geq m^{j-1}$. From these inequalities, we can conclude

$$\sum_{i=1}^{m+2} \frac{x_{i,m+2}}{m^{2i-2}} \geq m + (m-1)/m,$$

which contradicts the capacity constraints. □

One solution to this issue is to make the revenue and the capacity usage consistent, i.e. $f_{ik} = s_{ik}, \forall i, k$. These equalities will be called the homogeneity assumption.

2.2 Large Bids

If the bid prices or the capacity usage are large compared to the budgets and the capacities, because of the knapsack-like capacity constraints, we can show that there is no non-trivial performance guarantee for any deterministic algorithm:

Theorem 3. *In general, any deterministic algorithm can be arbitrarily bad.*

Proof. Consider an instance with only 1 bidder and 2 request types, $a_1 = b_1 = b_2 = n$, $f_{11} = s_{11} = 1, f_{12} = s_{12} = n$. A request of type 1 comes first. If it is assigned (to bidder 1), a request of type 2 will be presented. Note that, at that time, the remaining budget is $n-1$ which is not enough to pay for the second request. Therefore, the online revenue is 1, while the optimal revenue is n. If the first request is not assigned, then no more requests come. In this case, the online revenue is 0 while the optimal revenue is 1. From the discussion above, we can conclude $\rho \leq 1/n$. Noting that n is an arbitrary integer, the algorithm can be arbitrarily bad. □

As Theorem 3 indicates, it is impossible to find an algorithm with a non-trivial competitive ratio for instances whose bid prices are large compared to the budgets and capacities. Thus, it is necessary to make the assumption that that the relative size of bid $c = \max_{i,k}\{\max\{\frac{f_{ik}}{a_i}, \frac{f_{ik}}{b_k}\}\}$ is small.

2.3 Assumptions on the Model

As Theorem 2 and Theorem 3 show, in general, no algorithm has a non-trivial competitive ratio. Thus, additional assumptions about the model are necessary to obtain meaningful results. As discussed in Section 2.1 and 2.2, the following homogeneity assumption and small bid assumption are necessary:

Homogeneity Assumption: $f_{ik} = s_{ik}, \forall i, k$.

Small Bid Assumption: the relative size of bid $c = \max_{i,k}\{\max\{\frac{f_{ik}}{a_i}, \frac{f_{ik}}{b_k}\}\}$ is small.

Worth noting is the fact that these two assumptions are independent: the instance constructed in the proof of Theorem 2 satisfies the small bid assumption, and the one in the proof of Theorem 3 satisfies the homogeneity assumption. Therefore, both assumptions are necessary.

2.4 Special Cases

Theorem 4. *No deterministic algorithm has a competitive ratio $\rho > 1/2$.*

In the instance we construct, different bidders have very different valuations on requests. The online algorithm has to make a tradeoff between the short term revenue and the long term revenue. For the short term revenue, the algorithm tends to act greedily to keep its revenue comparable to the optimal revenue. Thus, requests tend to be assigned to bidders who are willing to pay more. On the other hand, for the long term revenue, the algorithm tends to assign requests to bidders evenly to leave enough flexibility for the future. As a result of this dilemma, any deterministic algorithm cannot be better than 1/2-competitive. The formal proof is given as follows:

Proof. Consider the following instance: there are $2m$ bidders and $2m$ request types, $\forall i, j, a_i = b_j = n, f_{ij} = \epsilon^{i-1}$, where ϵ is a very small number such that $n\epsilon << 1$. M groups of requests will be sequentially presented (within a group, the order of requests does not matter), $M \in \{1, \cdots, 2m\}$ to be determined by the adversary depending on how the online algorithm behaves. In group k, there are requests of type $1, 2, \cdots, k$, and the number of type j requests is $n\epsilon^{j-k}$.

Assume there is a c-competitive algorithm. Assume in group k, the algorithm assigns $nx^j_{i,k}\epsilon^{1-i}$ requests of type j to bidder i ($i \leq k+1-j$), i.e. $nx^j_{i,k}$ units of revenue are obtained from these requests. Since for any $i > k+1-j$, at most $n\epsilon^{k-j}\epsilon^{i-1} \leq n\epsilon$ units of revenue can be gained from requests of type j in group k that are assigned to bidder i, which is neglectable according to our assumption; we can assume no revenue is obtained from assigning requests of type j in group k to bidder i.

We define following notations: $\forall j \leq k, \bar{x}^k_j = \sum_{l=k}^{2m} x^j_{k+1-j,l}$, $\bar{a}_i = \sum_{j=1}^{2m-i+1} \bar{x}^{i+j-1}_j$, $\bar{b}_j = \sum_{i=j}^{2m} \bar{x}^i_j$, and $\bar{r}_l = \sum_{k=1}^{l}\sum_{j=1}^{k} \bar{x}^k_j$. Here $n\bar{x}^k_j$ is the revenue gained from assigning requests of type j to bidder $k+1-j$, $n\bar{a}_i$ is the revenue gained from bidder i, $n\bar{b}_j$ is the space used of type j, and $n\bar{r}_l$ is a upper bound of the revenue gained from the first l groups. For technical simplicity, we define two more notations: $\forall 2m \geq k \geq 2i+1, \sigma^k_i = \bar{x}^k_1 + \cdots + \bar{x}^k_i + \bar{x}^k_{k-i+1} + \cdots + \bar{x}^k_k$, $\sigma_i = \sum_{k=2i+1}^{2m} \sigma^k_i$.

Because of the budget and capacity constraints, we have $\bar{a}_i \leq 1$ and $\bar{b}_j \leq 1$. It is easy to see, if only the first l groups are presented, the optimal revenue is nl, while the online revenue is at most $n\bar{r}_l$. Since the algorithm is c-competitive, we have $\bar{r}_l \geq lc$. Thus,

$$\forall l \leq m-1, \sigma_l \leq \sum_{i=1}^{l}(\bar{a}_i + \bar{b}_i) + \sum_{i=1}^{l-1}\sigma_i^{i+l} - \bar{r}_{2l-1} - \bar{r}_{2l} \leq 2l - (4l-1)c + \sum_{i=1}^{l-1}\sigma_i^{i+l}.$$

Adding all these $m-1$ inequalities together, we then have

$$\sum_{i=1}^{m-1}\sigma_i \leq (m-1)m - (m-1)(2m-1)c + \sum_{i=1}^{m-2}\sum_{k=2i+1}^{m-1+i}\sigma_i^k \leq (m-1)m - (m-1)(2m-1)c + \sum_{i=1}^{m-2}\sigma_i,$$

which implies $\sigma_{m-1} \leq (m-1)m - (m-1)(2m-1)c$. Given the fact that $\sigma_{m-1} \geq 0$, we can conclude $c \leq m/(2m-1)$. By letting $m \to \infty$, we have $c \leq 1/2$. □

3 Two Best Possible Algorithms

3.1 A Greedy Algorithm

The first algorithm we propose is a greedy algorithm. In specific, it always assigns a request to the bidder who can pay the most. Formally:

Greedy Algorithm

0. Initialization, j=1.
1. Let k be the type of request j.
   - 1a. Find $\text{argmax}_i\{f_{ik} > 0 | f_{ik} \leq \min\{$ remaining budget of i, remaining capacity of $k\}\}$. If there is no such i, then j is not assigned to anyone.
   - 1b. Charge i, k f_{ik} units of money and capacity.
2. j=j+1. Go to 1.

The homogeneous cases have a good property that, for every request, the money earned equals the capacity used. Thus, we can transform the revenue in terms of one to another. The idea to prove the competitive ratio of the greedy algorithm is similar to the analysis of applying the greedy algorithm to the simple bipartite matching problem. All bidders are divided into two groups: A_1, all those bidders who have little money available at the end; and A_2, all the other bidders. All request types are divided in a similar way: B_1, all the request types who have few capacity available at the end; and B_2, all the other request types. (We will give a more precise definition of these sets in the formal proof.) For a request type in B_2, given that the algorithm always assigns requests in a greedy way, either all requests of that type are won by bidders in A_2, or bidders in A_2 are not willing to pay a high price for that request type. In either way, the used capacity! associated with that request type can be lower bounded in terms of the money spend by bidders in A_2 of the optimal solution. On the other hand, the used capacity associated with a request type in B_1 is close to the total capacity of that type, which can easily be lower bounded by the used capacity associated with that type in the optimal solution. Through careful algebraic manipulation, we can conclude:

Theorem 5. *The greedy algorithm is $(1-c)/2$-competitive, where c is the relative bid size.*

Before giving the formal proof, we first define some notations. Let $f_{i\cdot} = \max_k f_{ik}$, $f_{\cdot k} = \max_i f_{ik}$, and $c = \max\{\max_i \frac{f_{i\cdot}}{a_i}, \max_k \frac{f_{\cdot k}}{b_k}\}$. $A_1 = \{i \in A | \text{the remaining budget of } i \text{ is less than } f_{i\cdot}\}$, $B_1 = \{k \in B | \text{the remaining capacity of } k \text{ is less than } f_{\cdot k}\}$, $A_2 = A \backslash A_1$, and $B_2 = B \backslash B_1$. Let $a_i^g, b_k^g, a_i^{\text{opt}}, b_k^{\text{opt}}$ be the money/capacity used by the greedy solution/optimal solution. Let $a_{ik}^{\text{opt}} = f_{ik} \cdot x_{ik}^{\text{opt}}$ be the revenue earned by assigning requests of type k to bidder i in the optimal solution.

Claim 1. $\forall i \in A_1, a_i^g \geq a_i^{\text{opt}} - f_{i.}$.

Proof. Obviously from definition of A_1. □

Claim 2. $\forall k \in B_1, b_k^g \geq \sum_{i \in A_2} a_{ik}^{\text{opt}} - f_{.k}$.

Proof. $b_k^g \geq b_k - f_{.k} \geq b_k^{\text{opt}} - f_{.k} \geq \sum_{i \in A_2} a_{ik}^{\text{opt}} - f_{.k}$ □

Lemma 1. $\forall k \in B_2, b_k^g \geq \sum_{i \in A_2} a_{ik}^{\text{opt}}$.

Proof. There are two different cases,

1. If every request in type k is assigned to someone, then anyone who wins a request of type k pays no less for the request than any bidder in A_2. Because no less requests of type k is assigned in the greedy algorithm than in the optimal solution, so, $b_k^g \geq \sum\limits_{i \in A_2} a_{ik}^{\text{opt}}$.

2. If some requests in type k are not assigned, no bidder in A_2 is interested in that type. So, $b_k^g \geq 0 = \sum\limits_{i \in A_2} a_{ik}^{\text{opt}}$.

□

Claim 3. $\forall i \in A_1, f_{i.} \leq \frac{c}{1-c} a_i^g$.

Proof. $f_{i.} \leq c \cdot a_i \leq c(a_i^g + f_{i.})$, which implies $f_{i.} \leq \frac{c}{1-c} a_i^g$. □

Claim 4. $\forall k \in B_1, f_{.k} \leq \frac{c}{1-c} b_k^g$.

Proof. $f_{.k} \leq c \cdot b_k \leq c(b_k^g + f_{.k})$, which implies $f_{.k} \leq \frac{c}{1-c} b_k^g$. □

Proof. After establishing all those lemmas, we can conclude Theorem 5:

$$\begin{aligned}
\tfrac{2}{1-c} R^g &\geq \sum_{i \in A_1} a_i^g + \tfrac{c}{1-c} \sum_{i \in A_1} a_i^g + \sum_{k \in B_2} b_k + \sum_{k \in B_1} b_k + \tfrac{c}{1-c} \sum_{k \in B_1} b_k \\
&\geq \sum_{i \in A_1} (a_i^{\text{opt}} - f_{i.}) + \sum_{i \in A_1} f_{i.} + \sum_{i \in A_2, k \in B_2} a_{ik}^{\text{opt}} + \sum_{i \in A_2, k \in B_1} (a_{ik}^{\text{opt}} - f_{.k}) + \sum_{k \in B_1} f_{.k} \\
&= \sum_{i \in A_1} a_i^{opt} + \sum_{i \in A_2, k \in B} a_{ij}^{\text{opt}} \\
&= R^{\text{opt}},
\end{aligned}$$

thus $R^g \geq (1-c)/2 \cdot R^{\text{opt}}$. □

Theorem 5 shows, under the small bid assumption, as c goes to 0, the competitive ratio goes to 1/2. Thus, the greedy algorithm is an asymptotic optimal algorithm.

3.2 A Primal-Dual Algorithm

Primal-dual algorithms are proven to be useful in many online optimization problems. For example, the classic Adwords problem, where there is no capacity constraint, can be solved by a primal-dual algorithm [3] with the best possible competitive ratio of that problem. The basic idea of primal-dual algorithms is to use the weak duality to bound the optimal solution. The key in the primal dual algorithm is to construct a dual feasible solution that is close to the primal feasible solution.

However, in this problem, the extra capacity constraints makes the construction of a good dual solution difficult. If using the same technique in [3], we will obtain a pair of primal and dual solutions with a duality gap of $2 + \frac{1}{e-1}$, worse than the greedy algorithm. Thus, to develop an algorithm with a better competitive ratio, we will project the problem into a smaller space that is similar to the one in [3], generate a pair of primal and dual solutions in that space, and link back to the original space. Next we view the capacity constraints

as side constraints, and consider the LP relaxation of the remaining problem and its dual problem:

	Primal Problem P		Dual Problem D
Maximize:	$\sum_{i,j,k} f_{ik}x_{ijk}$	Minimize:	$\sum_j z_j + \sum_i a_i r_i$
Subject to:		Subject to:	
$\forall j$:	$\sum_{i,k} x_{ijk} \le 1$	$\forall (i,j,k)$:	$f_{ik}r_i + z_j \ge f_{ik}$
$\forall i$:	$\sum_{j,k} f_{ik}x_{ijk} \le a_i$	$\forall i,j$:	$r_i, z_j \ge 0$
$\forall j$ not of type k:	$x_{ijk} = 0$		
$\forall (i,j,k)$:	$x_{ijk} \ge 0$		.

Based on this pair of LPs, we can develop a primal-dual algorithm:

Primal-Dual Algorithm

0. Initially, all $r, x, z = 0$. Let $j = 1$.
1. Let k be the type of request j.
   - 1a. Find i that maximizes $f_{ik}(1 - r_i)$. If $r_i \ge 1$ or there is no more capacity for type k, then reject that request and go to 2.
   - 1b. Charge i, k f_{ik} units of money and capacity.
   - 1c. $x_{ijk} = 1, z_j = f_{ik}(1 - r_i), r_i = r_i + \frac{f_{ik}}{a_i}$
2. $j = j + 1$, go to 1.

Claim 5. *x is almost feasible to the original problem (2), namely, every budget constraint and capacity constraint can be violated at most once.*

Proof. Note that, in the primal-dual algorithm, the dual variable $1 - r_i$ is the proportion of remaining budget of bidder i. If bidder i runs out of money, then $r_i \ge 1$, and no more requests will be assigned to him. Thus, bidder i will be overcharged at most once. Similarly, noting Step 1 of the algorithm, every capacity constraint can be violated at most once. Therefore, the primal solution generated by this algorithm is almost feasible to the original problem. □

Given the almost feasible solution, we can construct a feasible solution by removing the bid assignments that violate budget constraints or capacity constraints. Let R be the revenue that the feasible solution obtains, and R_p be the primal revenue from the algorithm. Because the almost feasible solution violates every budget constraint and capacity constraint at most once, transforming it to a feasible solution will only lose a small amount of revenue. To be more precise, let A be the set of all bidders, and let A' be the set of bidders who violate budget constraints; let B be the set of all request types, and let B' be the set of those types which violate capacity constraints. The lost revenue will be at most $\sum_{i \in A'} \max_k \{f_{ik}\} + \sum_{k \in B'} \max_i \{f_{ik}\} \le \sum_{i \in A'} ca_i + \sum_{k \in B'} cb_k \le 2cR_p$. Thus $R_p - 2cR_p \le R$, which implies,

Lemma 2. $R \ge R_p(1 - 2c)$.

However, the dual solution may not be feasible. When request j of type k comes, if there are some capacities available at that time, because i is the bidder that maximize $f_{ik}(1 - r_i)$, all dual constraints that involve request j are valid, even if $r_i \ge 1$. Nevertheless, if there is no more capacity for type k at that time, those dual constraints that involve request j may not hold. Thus, to have dual feasibility, we will remove some dual constraints. We will only keep request types in $B \backslash B'$. For ease of notations, we will index the requests types in $B \backslash B'$ by k', and the requests of types in $B \backslash B'$ by j'.

When request j' comes, if it is rejected, according to the definition of B', k' has some capacity left. Therefore, for all i, either $f_{ik'} = 0$ or $r_i \ge 1$. In either case, $f_{ik'}r_i + z_{j'} \ge f_{ik'}$ is satisfied:

Claim 6. *$\forall (i, j', k')$, r, z satisfies dual feasibility $f_{ik'}r_i + z_{j'} \ge f_{ik'}$.*

Let's remove all those request types in B': further relax the primal problem P by removing all requests of types in B', and consider its dual problem.

	Primal Problem P'		Dual Problem D'
Maximize:	$\sum_{i,j',k'} f_{ik'} x'_{ij'k'}$	Minimize:	$\sum_{j'} z_{j'} + \sum_i a_i r'_i$
Subject to:		Subject to:	
$\forall j'$:	$\sum_{i,k'} x'_{ij'k'} \leq 1$	$\forall (i,j',k')$:	$f_{ik'} r'_i + z'_{j'} \geq f_{ik'}$
$\forall i$:	$\sum_{j',k'} f_{ik'} x'_{ij'k'} \leq a_i$	$\forall i, j'$:	$r'_i, z'_{j'} \geq 0$
$\forall j'$ not of type k':	$x_{ij'k'} = 0$		
$\forall (i,j',k')$:	$x'_{ij'k'} \geq 0$		

Noting these two pairs of primal dual problems are very similar, except that the second pair has a lower dimension, we will construct (x', z', r') by projecting (x, z, r) into the smaller space. Mathematically, $x'_{k'} = x_{k'}, r'_{i'} = r_{i'}$, and $z'_{j'} = z_{j'}$. (x, z, r) is updated when a new request comes in the original problem; as a projection of (x, z, r), (x', z', r') will also be updated. To be more precise, the updating for (x', z', r') will be:

Update for(x', z', r')

0. Initially, let all $r', x', z' = 0$. Let $j = 1$.
1. Let k be the type of request j.
   - 1a. If it is not assigned to anyone, then nothing changes. Go to 2.
   - 1b. If it is assigned to bidder i, and j is of type k in $B \backslash B'$, $x'_{ij} = x_{ij}, z'_j = f_{ij}(1 - r'_i), r'_i = r'_i + \frac{f_{ij}}{a_i}$. Go to 2.
   - 1c. If it is assigned to bidder i, and j is of type k in B', $r'_i = r'_i + \frac{f_{ij}}{a_i}$ (note x'_{ij} and z'_j doesn't exist in P' and D'). Go to 2.
2. $j = j + 1$. Go to 1.

Given that (z', r') is just a projection of (z, r), Claim 6 implies,

Claim 7. *z', r' are dual feasible to dual problem D'.*

Let $R_{B'} = \sum_{k \in B'} b_k$, and let R_{opt} be the optimal value of the original problem (with capacity constraints), and let R'_{opt} be the optimal value of P'. Let $R'_p = \sum_{i,j',k'} x'_{ik'} f_{ij'k'}$ be the primal revenue of P'. Let $R'_d = \sum_{j'} z'_{j'} + \sum_i a_i r'_i$ be the dual cost of D'. Because the primal problems P and P' are closely related, their optimal solutions are not too different. Neither are the solutions constructed by the algorithms.

Lemma 3. $R_{\text{opt}} \leq R'_{\text{opt}} + R_{B'}$.

Proof. Projecting the optimal assignment to $B \backslash B'$ (i.e., withdraw those bids for requests in B', and keep the rest) is still a feasible assignment, and will lose at most all the revenue from B', i.e. $R_{B'}$. □

Lemma 4. $R_p \geq R'_p + R_{B'}$

Proof.

$$\begin{aligned} R_p &= \sum_{i,j,k} x_{ijk} f_{ik} = \sum_{i,j',k'} x'_{ij'k'} f_{ik'} + \sum_{i,j,k \in B'} x_{ijk} f_{ik} \\ &= R'_p + \sum_{k \in B'} \sum_{i,j,k} x_{ijk} f_{ik} \geq R'_p + \sum_{k \in B'} b_k = R'_p + R_{B'}. \end{aligned}$$

□

The next step is to find the gap between the primal solution and the dual feasible solution.

Lemma 5. $R'_d \leq 2R'_p + R_{B'} + cR$.

Proof. When request j comes:

1. If j belongs to types in B', then R'_p remains the same, and R'_d increases.
2. If j belongs to types not in B', and is assigned to bidder i, then R'_p increases by f'_{ik} and R'_d increases by $z'_j + a_i \Delta r'_i \leq 2f'_{ik}$.
3. If j belongs to types not in B', and is not assigned to anyone, then both R'_p and R'_d remain the same.

Note that the sum of the increase of R'_d in the first case is at most $R_{B'} + cR$, and the increase of R'_d is no more than twice of R'_p in the other two cases. Thus, $R'_d \leq 2R'_p + R_{B'} + cR$. □

Theorem 6. $R \geq \frac{1-2c}{2+c-2c^2} R_{\text{opt}}$

Proof.

$$\begin{aligned} \frac{2+c-2c^2}{1-2c} R &= \frac{2}{1-2c} R + cR \geq 2R_p + cR = 2R'_p + 2R_{B'} + cR \geq R'_d + R_{B'} \\ &\geq R'_{\text{opt}} + R_{B'} \geq R_{\text{opt}}, \end{aligned}$$

thus, $R \geq \frac{1-2c}{2+c-2c^2} R_{\text{opt}}$. □

Theorem 6 shows that the primal-dual algorithm is asymptotically 1/2-competitive as c goes to 0, and thus is optimal.

4 Numerical Results

4.1 Underlying Structure

In the instances for this subsection, there are 10 bidders and 10 types, all budgets and capacities are equal, i.e. $a_i = b_j = 20$. $f_{ij} = \begin{cases} 1 & \text{w.p. } 0.5 \\ x & \text{w.p. } 0.5 \end{cases}$. In each instance, there are 400 requests uniformly i.i.d. over all 10 types. We test 100 i.i.d. instances and find the average revenue.

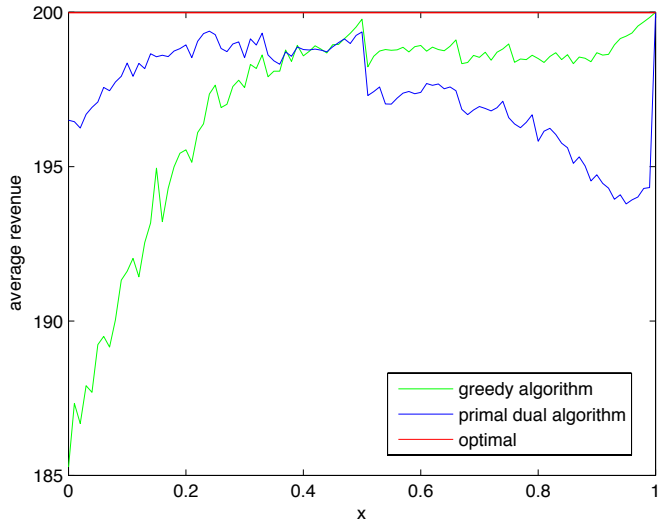

Figure 1: Performance under different graphs

From Figure 1, we can see that the primal-dual algorithm is better when the differences between bids are larger and there are sufficient requests. This result is quite intuitive. Since there are sufficient requests,

the short term revenue does not matter that much. Thus, the primal-dual algorithm, which takes the long term revenue into account when making an assignment, works better than the greedy algorithm.

4.2 Evolutionary Performance

Here we use the case with 10 bidders and 10 types. All budgets and capacities are equal, i.e. $a_i = b_j = 50$. $f_{ij} = \begin{cases} U_{ij} & \text{w.p. } 0.5 \\ 0 & \text{w.p. } 0.5 \end{cases}$, where U_{ij} is uniformly i.i.d. over $[0, 2]$ for all i, j. In each instance, there are 500 requests uniformly i.i.d. over all 10 types. We test 300 i.i.d. instances and find the average revenue.

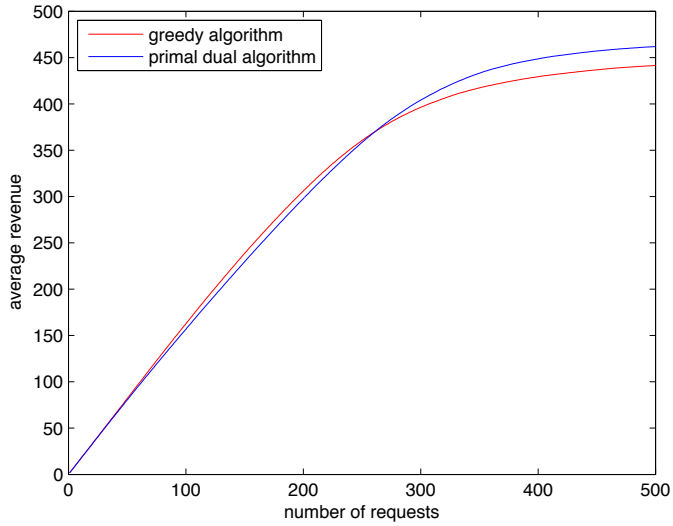

Figure 2: Evolutionary comparison between two algorithms

From Figure 2, we can see that the greedy algorithm works better when there are fewer requests while the primal-dual algorithm is better with more requests. It meets our expectation, because the primal-dual algorithm takes long term revenue into account while the greedy algorithm only maximizes revenue locally. If prior information on the number of requests is available, which algorithm to use to obtain a better performance can depend on that information.

4.3 Updating Rules

In general, it is not clear what is a better updating rule for the primal-dual algorithm. For example, in the Adwords problem, nonlinear updating will lead to an optimal algorithm, at the same time, in this problem, linear updating is the optimal one. We will compare these two different updating rules to find out which one works better empirically.

Here we use the case with 10 bidders and 10 types, all budgets and capacities are equal, i.e. $a_i = b_j = 20$. $f_{ij} = \begin{cases} U_{ij} & \text{w.p. } 0.5 \\ 0 & \text{w.p. } 0.5 \end{cases}$, where U_{ij} is uniformly i.i.d. over $[0, 2]$ for all i, j. In each instance, there are 400 requests uniformly i.i.d over all 10 types. We test 100 i.i.d. instances and find the average revenue.

From Figure 3, we can see that nonlinear updating does not work very well in this problem. It is even worse than the greedy algorithm. Thus, when implementing primal-dual algorithms for other problems, it is important to compare these two rules and other possible rules to find out which works better.

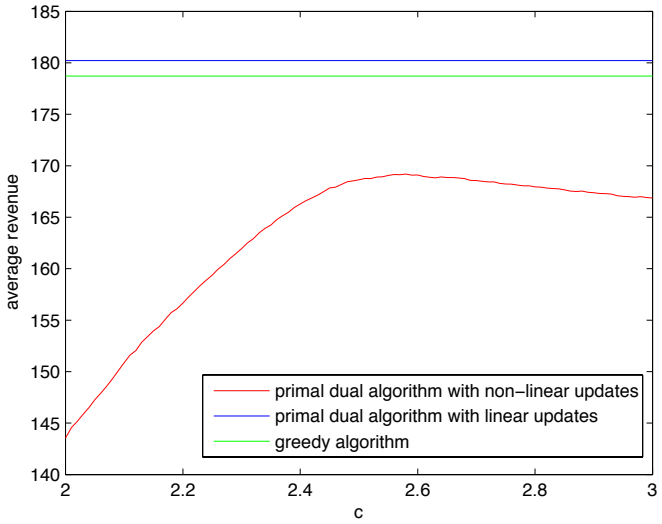

Figure 3: Different updating rules

5 Extensions

5.1 Randomized Algorithms

In this subsection, we will examine how randomized algorithms work for this problem. In general, randomized algorithms work better than deterministic algorithms for online optimization problems. However, as we will show, for this problem, randomized algorithms do not have better competitive ratios than the greedy algorithm and the primal-dual algorithm.

Randomized algorithms are usually much more difficult to characterize than deterministic algorithms. It is general practice to use the Yao principle [17] to derive bounds on competitive ratios of randomized algorithms. However, in many online problems, it is not straightforward. Instead, we will use a technique, introduced in [14], to derive a upper bound in a more straightforward way.

First, let's introduce some notations. For an online maximization problem, let $\mathcal{I}$ be a set of instances, and $\mathcal{A}$ be a set of deterministic algorithms. A randomized algorithm A_q can be defined as a probability distribution over $\mathcal{A}$ and a randomized instance I_q can be defined as a probability distribution over $\mathcal{I}$. Let $Z^A(I)$ and $Z^{\mathrm{OPT}}(I)$ be the objective value produced by algorithm A and optimal objective value on instance I. The competitive ratio of a randomized algorithm A_q is defined as $\min_{I\in\mathcal{I}} \frac{E_{A_q}[Z^{A_q}(I)]}{Z^{\mathrm{OPT}}(I)}$. Then, we have the following lemma on the upper bound of the competitive ratio of randomized algorithms:

Lemma 6. *[14] Given an online maximization problem, with possible input sequences $\mathcal{I}$, and possible deterministic algorithms $\mathcal{A}$, both possibly infinity. Then, for any random sequence I_p and any randomized algorithm A_q, we have*

$$\max_{A\in\mathcal{A}} \frac{E_{I_p}[Z^A(I_p)]}{E_{I_p}[Z^{\mathrm{OPT}}(I_p)]} \geq \min_{I\in\mathcal{I}} \frac{E_{A_q}[Z^{A_q}(I)]}{Z^{\mathrm{OPT}}(I)}.$$

Proof. Suppose there is a randomized algorithm A_q with competitive ratio ρ. Then $\forall I \in \mathcal{I}$, $E_{A_q}[Z^{A_q}(I)] \geq \rho Z^{\mathrm{OPT}}(I)$. Hence, $\rho E_{I_p}[Z^{\mathrm{OPT}}(I_p)] \leq E_{I_p}[E_{A_q}[Z^{A_q}(I_p)]] = E_{A_q}[E_{I_p}[Z^{A_q}(I_p)]] \leq \max_{A\in\mathcal{A}} E_{I_p}[Z^{A_q}(I_p)]$. □

To derive the upper bound on randomized algorithms, we only need to find a upper bound on applying deterministic algorithms on randomized algorithms:

Lemma 7. *There exists I_p such that* $\max_{A\in\mathcal{A}} \frac{E_{I_p}[Z^A(I_p)]}{E_{I_p}[Z^{\mathrm{OPT}}(I_p)]} \leq 1/2$.

The instance here is the same one used in Theorem 4 except for the number of requests. The number of requests is a random variable here rather than some number determined by the adversary in Theorem 4. The online algorithm still has to make a tradeoff between the short term revenue and the long term revenue. The proof is given as follows:

Proof. Consider the following instance: there are $2m$ bidders and $2m$ request types. $\forall i, j, a_i = b_j = n, f_{ij} = \epsilon^{i-1}$, where ϵ is a very small number such that $n\epsilon << 1$. M groups of requests will be sequentially presented: in group k, there are requests of type $1, 2, \cdots, k$, and the number of type j requests is $n\epsilon^{j-k}$. Here, M is a random variable, uniformly distributed over $\{1, 2, \cdots, 2m\}$.

For any deterministic algorithm, if all M groups of requests are presented, assume in group k, the algorithm assigns $nx^j_{i,k}\epsilon^{1-i}$ requests of type j to bidder i ($i \leq k+1-j$), i.e.,$nx^j_{i,k}$ units of revenue are gained from those requests. We can assume no revenue is gained from assigning requests of type j in group to bidder i, where $i > k+1-j$, since at most $n\epsilon^{k-j}\epsilon^{i-1} \leq n\epsilon$ units of revenue are gained, which is neglectable according to our assumption.

We define the following notations: $\forall j \leq k, \bar{x}^k_j = \sum_{l=k}^{2m} x^j_{k+1-j,l}$, $\bar{a}_i = \sum_{j=1}^{2m-i+1} \bar{x}^{i+j-1}_j$, $\bar{b}_j = \sum_{i=j}^{2m} \bar{x}^i_j$, and $\bar{r}_l = \sum_{k=1}^{l}\sum_{j=1}^{k} \bar{x}^k_j$. Here $n\bar{x}^k_j$ is the revenue gained from assigning requests of type j to bidder $k+1-j$, $n\bar{a}_i$ is the revenue gained from bidder i, $n\bar{b}_j$ is the space used of type j, and $n\bar{r}_l$ is a upper bound of the revenue gained from the first l groups. For technical simplicity, we define two more notations: $\forall 2m \geq k \geq 2i+1, \sigma^k_i = \bar{x}^k_1 + \cdots + \bar{x}^k_i + \bar{x}^k_{k-i+1} + \cdots + \bar{x}^k_k$, and $\sigma_i = \sum_{k=2i+1}^{2m} \sigma^k_i$.

Because of budget and capacity constraints, we have $\bar{a}_i \leq 1$ and $\bar{b}_j \leq 1$. It is easy to see, if there are only the first l groups presented, the optimal revenue is nl, while the online revenue is at most $n\bar{r}_l$.

$\forall l \leq m-1, \sigma_l \leq \sum_{i=1}^{l}(\bar{a}_i + \bar{b}_i) + \sum_{i=1}^{l-1}\sigma^{i+l}_i - \bar{r}_{2l-1} - \bar{r}_{2l} \leq 2l - \bar{r}_{2l-1} - \bar{r}_{2l} + \sum_{i=1}^{l-1}\sigma^{i+l}_i$. Adding all these $m-1$ inequalities together, we then have $\sum_{i=1}^{m-1}\sigma_i \leq (m-1)m - \sum_{i=1}^{2m-2}\bar{r}_i + \sum_{i=1}^{m-2}\sum_{k=2i+1}^{m-1+i}\sigma^k_i \leq (m-1)m - \sum_{i=1}^{2m-2}\bar{r}_i + \sum_{i=1}^{m-2}\sigma_i$, which implies $\sigma_{m-1} + \sum_{i=1}^{2m-2}\bar{r}_i \leq (m-1)m$. Obviously $\sigma_{m-1} \geq 0$, which implies $\sum_{i=1}^{2m-2}\bar{r}_i \leq (m-1)m$.

It is easy to see, the optimal revenue $R_{\text{opt}} = M$. Therefore, the expected optimal revenue $E[R_{\text{opt}}] = E[M] = m + 1/2$. Meanwhile, the expected online revenue $E[R] = \frac{1}{2m}\sum_{i=1}^{2m}\bar{r}_i \leq \frac{1}{2m}(2m + 2m + \sum_{i=1}^{2m-2}\bar{r}_i) \leq \frac{(m+3)m}{2m} = (m+3)/2$. By letting m goes to infinity, we can conclude the lemma. □

These two lemmas above conclude our main result here:

Theorem 7. *No randomized algorithm has a competitive ratio of $c \leq 1/2$.*

Theorem 7 shows that randomized algorithms do not have better competitive ratios. The greedy algorithm and the primal dual algorithm are the best algorithms among all possible algorithms, either deterministic or randomized.

5.2 Bounded Case

In this subsection, we will discuss how to solve the problem without the homogeneity assumption we previously made. Although, as Theorem 2 shows, any algorithm can be arbitrarily bad if the instance does not satisfy homogeneity assumption, its proof also indicates that the worst case happens in extreme situations where s_{ij}/f_{ij} varies widely. In fact, as we will show, if s_{ij}/f_{ij} is bounded on both sides and away from 0, there exist algorithms with positive competitive ratio.

If we are given a ρ-competitive algorithm for the homogeneous version (eg., greedy algorithm and primal-dual algorithm), then we can construct a $\frac{u\rho}{v}$-competitive algorithm, where $0 < u \leq \frac{s_{ij}}{f_{ij}} \leq v, \forall i, j$.

Modified Algorithm

1. Construct a new instance P2 based on the original one P1: the numbers of bidders and request types are exactly the same, and $b_j^2 = b_j^1, s_{ij}^2 = s_{ij}^1, f_{ij}^2 = s_{ij}^1/v, a_i^2 = u/v \cdot a_i^1$.
2. Whenever a new request comes in P1, present a request of the same type in P2. If the original algorithm assigns it to bidder i in P2, assign it to bidder i in P1. If the original one rejects it, reject it.

Theorem 8. *The modified algorithm is $\frac{u\rho}{v}$-competitive.*

Proof. Because $\frac{f_{ij}^2}{f_{ij}^1} = \frac{s_{ij}^1}{v f_{ij}^1} \geq \frac{u}{v}$ and $\frac{a_i^2}{a_i^1} = \frac{u}{v}$, the modified algorithm always gives a feasible solution. Noting that $f_{ij}^2 = s_{ij}^1/v \leq f_{ij}^1$, we have $R^1 \geq R^2 \geq \rho R_{\text{opt}}^2$.

To establish the relation between R_{opt}^1 and R_{opt}^2, we consider another problem P3, where the numbers of bidders and request types remain the same, and $b_j^3 = b_j^1, s_{ij}^3 = s_{ij}^1, f_{ij}^3 = s_{ij}^1/u = v/u \cdot f_{ij}^2, a_i^3 = a_i^1 = v/u \cdot a_i^2$. It is easy to see $R_{\text{opt}}^1 \leq R_{\text{opt}}^3 = v/u \cdot R_{\text{opt}}^2$.

From the discussion above, we can conclude $R^1 \geq R^2 \geq \rho R_{\text{opt}}^2 \geq \frac{u\rho}{v} R_{\text{opt}}^1$. □

5.3 Fractional Bids

As Theorem 3 and its proof indicate, if the relative bid size is large, it is impossible to obtain good results in the original settings. However, the situation will be different if fractional bids are allowed, where a request can be split and assigned to different bidders. Formally, the mathematical programming no longer has the integrality constraints:

$$\begin{array}{lll} \text{Maximize:} & \sum\limits_{i,j,k} f(i,j,k)x(i,j,k) & \\ \text{Subject to:} & \sum\limits_{i,k} x(i,j,k) \leq 1 & \forall j \\ & \sum\limits_{j,k} f(i,j,k)x(i,j,k) \leq a_i & \forall i \\ & \sum\limits_{i,j} s(i,j,k)x(i,j,k) \leq b_k & \forall k \\ & 0 \leq x(i,j,k) \leq 1 & \forall i,j,k \end{array} , \tag{2}$$

Given that fractional bids are allowed, we do not lose the revenue of the very last bids that bidders cannot previously afford, which counts for c in the competitive ratios of both the greedy algorithm and the primal-dual algorithm. In fact, both algorithms are 1/2-competitive, no matter how large the relative bid size is. We will propose the algorithms and show the results on the competitive ratios without proof:

Greedy Algorithm with Fractional Bids

0. Initialization, j=1.
1. Let k be the type of request j, let $n_j = 1$ be the number of request to be assigned.
   - 1a. Find i=argmax$_i\{f_{ik} > 0 | \min\{$ remaining budget of i, remaining capacity of $k\} \geq 0\}$. If there is no such i, Go to 2.
   - 1b. $n_{ij} = \min\{$ remaining budget of i, remaining capacity of $k\}/f_{ik}$. Assign n_{ij} unit of request j to i. Charge i,k $n_{ij}f_{ik}$ units of money and capacity. $n_j = n_j - n_{jk}$. If $n_j = 0$, go to 2; otherwise, go to 1a.
2. j=j+1. Go to 1.

Theorem 9. *The greedy algorithm with fractional bids is 1/2-competitive.*

Primal-Dual Algorithm with Fractional Bids

0. Initially, all $r, x, z = 0$. Let $j = 1$.

1. Let k be the type of request j. Let $n_j = 1$ be the number of request to be assigned.

   1a. Find i that maximizes $f_{ij}(1 - r_i)$. If $r_i \geq 1$ or there is no more capacity for type k, then reject that request and go to 2.

   1b. $n_{ij} = \min\{$ remaining budget of i, remaining capacity of $k\}/f_{ij}$. Assign n_{ij} unit of request j to i. Charge i, k $n_{ij}f_{ij}$ units of money and capacity. $n_j = n_j - n_{jk}$.

   1c. $x_{ij} = n_{ij}, z(j) = n_{ij}f_{ij}(1 - r_i), r_i = r_i + \frac{n_{ij}f_{ij}}{a_i}$. Update $n_j = n_j - n_{ij}$. If $n_j = 0$, go to 2; otherwise, go to 1a.

2. $j = j + 1$, go to 1.

Theorem 10. *The primal-dual algorithm with fractional bids is* 1/2*-competitive.*

6 Conclusions

In this paper, we investigate a generalized online assignment problem with budget and capacity constraints. The main focus of this paper lies in a special case with two more assumptions on the problem. We derive a upper bound on competitive ratios of any algorithm, either deterministic or randomized, and propose two algorithms which reach the upper bound asymptotically under assumptions. Then we discuss how to solve the problem without those assumptions.

We use worst case competitive analysis in this paper. In practice, the worst case is not likely to happen. It would be interesting to develop a better strategy to take advantage of some additional information, such as the distribution of coming requests. Like in the Adwords problem, the primal dual algorithm here is shown to be the best online algorithm, although with a totally different updating rule. It would be very interesting to know for what problems the primal dual algorithm works.

References

[1] G. Aggarwal, A. Goel, and R. Motwani. Truthful auctions for pricing search keywords. In *ACM Conference on Electronic Commerce*, pages 1–7, 2006.

[2] N. Buchbinder. *Designing competitive online algorithms via a primal-dual approach.* PhD thesis, Technion, 2008.

[3] N. Buchbinder, K. Jain, and S. Naor. Online primal-dual algorithms for maximizing ad auctions revenue. In *ESA*, 2007.

[4] C. Chekuri and S. Khanna. A PTAS for the multiple knapsack problem. In *SODA*, pages 213–222, 2000.

[5] B. Edelman, M. Ostrovsky, and M. Schwarz. Internet advertising and the generalized second price auction: Selling billions of dollars worth of keywords. *American Economic Review*, 97(1):242–259, 2007.

[6] J. Feldman, S. Muthukrishnan, M. Pal, and C. Stein. Budget optimization in search-based advertising auctions. In *ACM Conference on Electronic Commerce*, 2007.

[7] L. Fleischer, M. Goemans, V. Mirrokni, and M. Sviridenko. Tight approximation algorithms for maximum general assignment problems. In *SODA*, pages 611–620, 2006.

[8] G. Goel and A. Mehta. Online budgeted matching in random input models with applications to adwords. In *SODA*, pages 982–991, 2008.

[9] B. Kalyanasundaram and K. Pruhs. An optimal deterministic algorithm for on-line b-matching. *Theoretical Computer Science*, 233(1-2):319–325, 2000.

[10] S. Lahaie, D. Pennock, A. Saberi, and R. Vohra. Sponsored search. In Nisan et al., editor, *Algorithmic Game Theory*. Cambridge University Press, 2007.

[11] A. Mehta, U. Vazirani, A. Saberi, and V. Vazirani. AdWords and generalized online matching. In *FOCS*, pages 264–273, 2005.

[12] P. Rusmevichientong and D. Williamson. An adaptive algorithm for selecting profitable keywords for search-based advertising services. In *Seventh ACM Conference on Electronic Commerce*, 2006.

[13] D. Shmoys and E. Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical Programming*, 62(3):461–474, 1993.

[14] L. Stougie and A. Vestjens. Randomized algorithms for online scheduling problems: How low cant you go. *Operations Research Letters*, 30(2):89–96, 2002.

[15] H. Varian. Position auctions. *International Journal of Industrial Organization*, 25:1163–1178, 2007.

[16] U. Varirani, R. Karp, and V. Varirani. An optimal algorithm for online bipartite matching. In *STOC*, pages 352–358, 1990.

[17] A. Yao. Probabilistic computations: Toward a unified measure of complexity. In *FOCS*, pages 222–227, 1977.