Improved Approximation Algorithms for Budgeted Allocations*

Yossi Azar[1], Benjamin Birnbaum[2], Anna R. Karlin[2], Claire Mathieu[3], and C. Thach Nguyen[2]

[1] Microsoft Research and Tel-Aviv University
azar@tau.ac.il
[2] University of Washington
<birnbaum,karlin,ncthach>@cs.washington.edu
[3] Brown University
claire@cs.brown.edu

Abstract. We provide a 3/2-approximation algorithm for an offline budgeted allocations problem with applications to sponsored search auctions. This an improvement over the $e/(e-1)$ approximation of Andelman and Mansour [1] and the $e/(e-1) - \epsilon$ approximation (for $\epsilon \approx 0.0001$) of Feige and Vondrak [2] for the more general Maximum Submodular Welfare (SMW) problem. For a special case of our problem, we improve this ratio to $\sqrt{2}$. We also show that the problem is APX-hard.

1 Introduction

The rising economic importance of online sponsored search advertising has led to a great deal of research focused on developing its theoretical underpinnings. (See e.g., [3] for a survey.) Since search engines such as Google, Yahoo! and MSN depend on sponsored search for a significant fraction of their revenue, a key problem is how to optimally allocate ads to keywords (user searches) so as to maximize search engine revenue [1, 4–7]. In this direction, Mehta et al. [7] studied a stylized version of the problem, which we call the *Online Budgeted Allocation* problem. In their model, there is a set of bidders U and a set of keywords V. Each bidder $i \in U$ has a known daily budget B_i and a non-negative bid b_{ij} for every keyword $j \in V$. The keywords arrive one-by-one in an online fashion, with the bids for keyword j revealed only when j arrives. At each keyword arrival, the algorithm (i.e., the search engine) allocates the keyword to one of the bidders (i.e., displays that bidder's ad as one of the sponsored search results the user sees). The total profit extracted by the algorithm from each bidder is the minimum of the budget B_i of that bidder and the sum of the b_{ij}'s for keywords j allocated to it. The goal is to find an allocation of keywords to bidders that maximizes the total profit extracted by the algorithm. Mehta et al. [7] presented an algorithm that achieves an optimal competitive ratio of $e/(e-1)$ for the

* This research was supported by the Israeli Science Foundation, NSF Grant CCF-0635147, and by an NSF Graduate Research Fellowship.

case when the bids are much smaller than the budgets, a result also proved by Buchbinder et al. [4]. When there is no restriction on the values of the bids relative to the budgets, the best known competitive ratio is 2 [8].

Surprisingly, the approximability of the *offline* version of Budgeted Allocation, in which the algorithm can see all of the bids before allocating keywords, is still not well understood. Lehmann et al. [8] showed that the problem is NP-hard, and Andelman and Mansour [1] provided the first non-trivial approximation ratio of $e/(e-1)$. Feige and Vondrak [2] improved this ratio to $e/(e-1)-\epsilon$ (for $\epsilon \approx 0.0001$) for the more general SMW problem.

In this paper, we give improved approximation algorithms for two versions of the offline Budgeted Allocation problem. In the *uniform* version, each keyword j has a single price b_j. If a bidder i is interested in j, its bid b_{ij} is equal to b_j. Otherwise, its bid b_{ij} is 0. The *non-uniform* version removes this restriction, so that the b_{ij} values can be arbitrary for each i and j.

1.1 Our Results

We provide a deterministic 3/2-approximation algorithm for the non-uniform Budgeted Allocation problem (Section 2). This improves the previous best-known approximation ratio of $e/(e-1)-\epsilon$ (for $\epsilon \approx 0.0001$) [2]. For the uniform version of the problem, we improve the approximation ratio to $\sqrt{2}$ (Section 3). In both these algorithms, we assume that the maximum bid is no larger than the smallest budget, i.e. $\max_{i,j} b_{ij} \leq \min_i B_i$.

We also show that the problem is APX-hard (Section 4).

1.2 Related Work

As discussed above, before this work, the first non-trivial approximation ratio for Budgeted Allocation was $e/(e-1)$, due to Andelman and Mansour [1]. For the special case in which the bidders all have the same budget, these authors were able to lower this ratio to approximately 1.39. Our algorithms apply to the more general case in which the budgets may be different for different bidders.

Two recent unpublished works also study the Budgeted Allocation problem and provide better approximation ratios than those obtained in this paper: Independently of our work, Chakrabarty and Goel [9] have provided two elegant algorithms, an iterative rounding algorithm and a primal-dual algorithm, both of which achieve an approximation ratio of 4/3, matching the integrality gap of the linear program used in this and other papers. Their paper also shows that it is NP-hard to approximate Budgeted Allocation to a factor better than 16/15, which subsumes the result in this paper on APX-hardness. In addition, building on our approach, Srinivasan [10] has recently provided another LP-rounding algorithm that achieves an approximation ratio of 4/3.

The Budgeted Allocation problem is an important special case of SMW, the problem of maximizing utility in a combinatorial auction in which the utility functions are submodular. In the combinatorial auction setting the keywords are items to be sold. SMW has been widely studied [2, 8, 11–14]. For submodular auctions using the value query model, the best approximation algorithm gives a factor $e/(e-1)$ [14]. This ratio has been shown to be the best possible for this model [12, 13]. For the stronger demand query model it is possible to do at least slightly better, that is $e/(e-1)-\epsilon$ (for $\epsilon \approx 0.0001$) [2]. For solving the Budgeted Allocation problem using the SMW demand query model one needs to provide a polynomial-time demand query oracle. As noted by [15], one can use a knapsack-type FPTAS algorithm to provide an approximate oracle that is good enough for solving the problem.

The Budgeted Allocation problem is also similar to the *generalized assignment problem* (GAP) [2, 15–17]. The main difference between Budgeted Allocation and GAP is that in GAP every keyword (or *item*, in GAP parlance) has a weight and each bidder (or *bin*) has a fixed capacity that cannot be exceeded, whereas in Budgeted Allocation, budgets can be exceeded, but no extra profit is obtained from doing so. The best approximation algorithm for GAP does slightly better than $e/(e-1)$ [2].

2 The 3/2-Approximation Algorithm

In this section, we describe the 3/2-approximation for the non-uniform version of the problem.

2.1 High-level Idea

Our algorithms use linear program rounding. We represent the Budgeted Allocation problem with the same natural integer program used in [1, 9, 10], in which the 0-1 variables x_{ij} represent whether keyword j is allocated to bidder i:

$$\max \sum_{i\in U} \min(B_i, \sum_{j\in V} b_{ij}x_{ij}) \quad \text{s.t.} \quad \begin{cases} \sum_{i\in U} x_{ij} \le 1 \ \forall j \in V \\ x_{ij} \in \{0,1\} \quad \forall i \in U, j \in V \end{cases} .$$

Let L be the linear relaxation of this integer program in which the second constraint is replaced by $x \ge 0$. (The upper-bound of 1 is guaranteed by the other constraint.) Rounding the optimal solution carefully is what allows us to beat the factor of $e/(e-1)$ of Andelman and Mansour [1].

For any fractional allocation $\mathbf{x}$, define the graph G *induced by* $\mathbf{x}$ to be the bipartite graph over $U \cup V$ with an edge $\{i, j\}$ for every $x_{ij} > 0$, with weight $w_{ij} = b_{ij}x_{ij}$.[1] Rounding $\mathbf{x}$ can be viewed as a transformation of G into another graph in which the degree of every keyword is 1. Our algorithms do this iteratively, at each step modifying local structures so that the degree of at least one new keyword is reduced to 1. Some weight in the objective function will be lost at

[1] Notice that in a fractional solution, we can assume without loss of generality that no bidder's budget is exceeded. Therefore, the value of $\mathbf{x}$ is equal to the sum of the edge weights in G.

each step, but we use a charging argument to bound this loss by 1/3 of the original value of $\mathbf{x}$.

The first observation for the proofs is that one can assume that the graph G induced by a feasible fractional solution to L has a special structure. This observation was made for optimal fractional solutions by [1], proved in [18], and used by [9, 10]. We use a slightly stronger version that holds for any feasible solution. This will allow us to assume that this special structure holds after each rounding step, not just at the beginning. Say that bidder i is *saturated* if $\min(B_i, \sum_{j\in V} b_{ij}x_{ij}) = B_i$ and is *unsaturated* otherwise.

Lemma 1. *Any feasible solution* $\mathbf{x}$ *of* L *with induced graph* G *can be transformed, in polynomial time, to another feasible solution* $\tilde{\mathbf{x}}$ *that has an induced graph that is a subgraph of* G *and that is a forest with at most one unsaturated bidder per tree.*

Proof. Consider a feasible solution $\mathbf{x}$ of L, and let G be its induced graph. We provide a polynomial-time procedure to first eliminate cycles in G, and then saturate all but one bidder in each component. The graph obtained from this procedure will correspond to a solution $\tilde{\mathbf{x}}$ that satisfies the statement of the lemma.

Consider a cycle in G. Let $i_0, j_0, \ldots, i_{k-1}, j_{k-1}$ be the vertices of the cycle in order, where the i_ℓ's are bidders and the j_ℓ's are keywords. For $1 \le \ell \le k-1$, let $\alpha_\ell = b_{i_\ell j_{\ell-1}}/b_{i_\ell j_\ell}$. We can assume without loss of generality that $\prod_{\ell=1}^{k-1}\alpha_\ell \le 1$. (If not, reverse the cycle, and all of the α's become their reciprocals.) Since an edge $\{i, j\}$ only appears in G if j is partially allocated to i in $\mathbf{x}$, it follows that both $x_{i_\ell j_\ell}$ and $x_{i_{\ell+1}, j_\ell}$ are in the open interval $(0, 1)$ for all ℓ.[2] Hence, we can choose an $\epsilon > 0$ small enough such that if, for all ℓ, $x_{i_\ell j_\ell}$ is increased by $\epsilon\prod_{\ell'=1}^{\ell}\alpha_{\ell'}$ and $x_{i_{\ell+1} j_\ell}$ is decreased by $\epsilon\prod_{\ell'=1}^{\ell}\alpha_{\ell'}$ then the modified solution is still feasible. Note that the amount allocated to each bidder except i_0 stays the same, and the amount allocated to i_0 increases by $\epsilon(1-\prod_{\ell=1}^{k-1}\alpha_\ell) \ge 0$. Hence, the modified solution does not have a smaller value. By performing this modification for the smallest $\epsilon > 0$ that changes the amount allocated on at least one edge to zero, we can break the cycle without decreasing the solution value. Repeating this until there are no more cycles, we can eliminate every cycle in polynomial time.

Now consider a connected component of G that contains two unsaturated bidders, say, i_0 and i_k. Let $i_0, j_1, i_1, \ldots, j_k, i_k$ be an arbitrary path in G that connects i_0 and i_k. For $1 \le \ell \le k-1$, let $\alpha_\ell = b_{i_\ell j_\ell}/b_{i_\ell j_{\ell+1}}$, and let $\alpha = \prod_{\ell=1}^{k-1}\alpha_\ell$. Now, either $b_{i_0 j_1} - \alpha b_{i_k j_k} \ge 0$ or $-b_{i_0 j_1} + \alpha b_{i_k j_k} \ge 0$ (or both). Suppose first that $b_{i_0 j_1} - \alpha b_{i_k j_k} \ge 0$. By a similar argument as before, there exists an $\epsilon > 0$ such that the following transformation results in a feasible solution: for $0 \le \ell \le k-1$, increase $x_{i_\ell j_{\ell+1}}$ by $\epsilon\prod_{\ell'=1}^{\ell}\alpha_{\ell'}$ and decrease $x_{i_{\ell+1} j_{\ell+1}}$ by $\epsilon\prod_{\ell'=1}^{\ell}\alpha_{\ell'}$. With this transformation, the amount allocated to each bidder except i_0 and i_k remains the same. The amount allocated to i_0 increases by $\epsilon b_{i_0 j_1}$ and the amount allocated to i_k decreases by $\epsilon\alpha b_{i_k j_k}$. Hence, by our assumption, the solution does not

[2] Here, and throughout the paragraph, indices are added modulo k.

decrease in value, and we can choose the smallest $\epsilon > 0$ that either saturates i_0 or breaks the path. In either case, we make progress towards saturating all but one bidder in each component. Now, suppose that $-b_{i_0 j_1} + \alpha b_{i_k j_k} \geq 0$. We proceed symmetrically: for $0 \leq \ell \leq k-1$, decrease $x_{i_\ell j_{\ell+1}}$ by $\epsilon \prod_{\ell'=1}^{\ell} \alpha_{\ell'}$ and increase $x_{i_{\ell+1} j_{\ell+1}}$ by $\epsilon \prod_{\ell'=1}^{\ell} \alpha_{\ell'}$. This transformation does not reduce the value of the solution, and we can choose the smallest $\epsilon > 0$ that either saturates i_k or breaks the path. Therefore, in either case, we can keep applying one of these procedures until, in polynomial time, each component has at most one unsaturated bidder. □

For a graph G induced by a fractional solution, say that a bidder is *active* if it has at least one neighbor of degree 2 or more in G, and is *inactive* otherwise. In our charging argument for the 3/2-algorithm, we charge to the weight allocated to bidders when they move from being active to inactive. (Since this happens at most once for each bidder, each unit of profit in the optimal fractional solution is charged at most once.)

We call the process of transforming a subgraph to reduce the degrees of the keywords *rounding* the subgraph. In general, this process will remove some of the edges and transfer some of the weight removed to the edges that remain, while respecting the constraints of the LP. For example, suppose that two keywords j_1 and j_2 are allocated fractionally to bidder i and fractionally to some other bidders i_1 and i_2, as shown in Fig. 1. One way to round this part of the graph would be to remove the edges $\{i, j_1\}$ and $\{i_2, j_2\}$. Once this is done, i has some unused budget that can be used to transfer an additional fraction of j_2 from i_2 to i. In general, transferring weight in this manner will be essential to obtaining our approximation ratio.

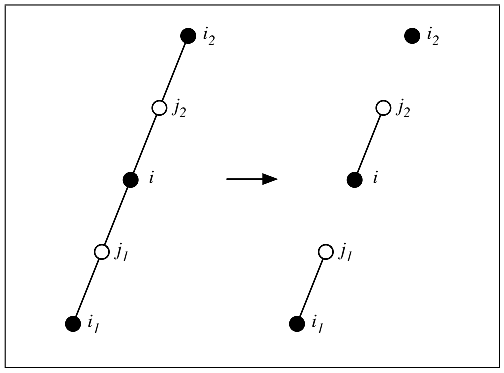

Fig. 1. One way to round a path that has three bidders. When edge $\{i, j_1\}$ is removed, this frees up some budget in i to accommodate some of the weight of j_2 that was originally allocated to i_2.

The main idea of our proof is that in every tree with active bidders, there is a small local structure involving only a constant number of nodes, such that if we round that structure so as to minimize the resulting loss in the objective function, the loss is at most 1/3 of the budget spent by the bidders that become inactive.

2.2 The Algorithm

Our 3/2-approximation algorithm is given below by Algorithm 1. At each iteration of the while loop, our algorithm looks for one of the *interesting nodes* (Definition 4) and rounds keywords in the associated structure. For each of the four types of interesting nodes, we describe a rounding subroutine used by the algorithm. We will show that the loss of these subroutines can be charged to the nodes that become inactive, which we will use to prove that Algorithm 1 is a 3/2-approximation.

Theorem 2. *Algorithm 1 is a polynomial-time 3/2-approximation for the Budgeted Allocation problem.*

Algorithm 1: 3/2-approximation algorithm for Budgeted Allocation

```
Input: Set of bidders $U$, set of keywords $V$; for each $i, j$, bid $b_{ij}$ of bidder $i$ for
       keyword $j$; and for each bidder $i$, budget $B_i$.
Output: Allocation of keywords to bidders.
solve the following LP to get an optimal solution $\mathbf{x}$ with induced graph $G$:

    $\max \sum_{i \in U} \min(B_i, \sum_{j \in V} b_{ij} x_{ij}) \quad \text{s.t.} \quad \begin{cases} \sum_{i \in U} x_{ij} \le 1 & \forall j \in V \\ 0 \le x_{ij} & \forall i \in U, j \in V \end{cases}$ .

transform $G$ into a forest with $\le 1$ unsaturated bidder per tree (Lemma 1).
while $G$ contains active bidders do
    Round the subgraph associated to an interesting node according to its type;
    Transform $G$ into a forest with $\le 1$ unsaturated bidder per tree (Lemma 1);
end
allocate each keyword to its unique adjacent bidder in $G$.
```

Root each tree of G at the unsaturated bidder if there is one, or at an arbitrary bidder if there is not.

Definition 3. *Consider a path $i_1, j_1, i_2, j_2, \ldots, i_{k-1}, j_{k-1}, i_k$ consisting of $2k-1$ nodes, starting and ending with a bidder, such that*

- *the $k-1$ keywords $j_1, j_2, \ldots, j_{k-1}$ all have degree exactly 2,*
- *bidder i_1 is the highest node on the path, called "root" of the path,*
- *for all other bidders $i_2, i_3, \ldots, i_k$, any keyword not on the path that is adjacent to the bidder has degree exactly 1.*

*We call the graph formed by this path and all the degree-1 keywords that are neighbors of $i_1, i_2, \ldots, i_k$ a k-*chain.

Definition 4. *In a tree of G, we say that a node is* interesting *if it has one of the following four types.*

1. *The root of the tree, if the tree consists of a 2-chain (Fig. 2(a)).*
2. *A bidder v whose subtree contains at least one 3-chain rooted at v (Fig. 2(b)).*
3. *A bidder v who is the root of more than one 2-chain and who is not the root of a k-chain, for $k > 2$ (Fig. 2(c)).*
4. *A keyword with at least 2 children, such that each child is a root of a 1-chain or a 2-chain (Fig. 2(d)).*

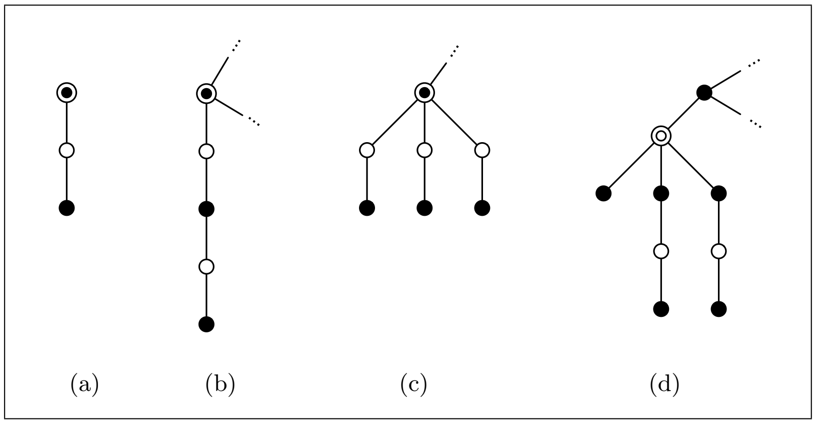

Fig. 2. Examples of the four types of interesting nodes with their associated subgraphs, as defined in Definition 4. For each type, the interesting node is shown with an extra circle.

Before we describe the rounding subroutines, we establish the correctness of Algorithm 1.

Lemma 5. *A tree that has a keyword of degree more than 1 must have at least one interesting node.*

Proof. In this proof, it will be convenient to work with the *trimmed graph* G' of G, which we define to be the graph obtained from G by removing all keywords of degree 1. Note that the statement of the lemma applies for G' if and only if it applies for G.

Consider a connected component T of G' that has at least one keyword with degree at least 2. If there are only two bidders in this component, then it must be a 2-chain, and hence there is an interesting node of type 1. Suppose now that there are at least three bidders. If there is a 3-chain, then clearly there is an interesting node of type 2. So suppose that there are no 3-chains. Let i_1 be a leaf bidder in T; let j_1 be its parent; and let i_2 be the parent of j_1. Suppose that j_1 has another child besides i_1. Then since there are no 3-chains in T, it must be that j_1 is an interesting node of type 4.

Now suppose that j_1 has no other child besides i_1. Suppose that i_2 has another child besides j_1, say j_2. Then, since T is in the trimmed graph, j_2 must have another child, say i_3. Since T has no 3-chains, i_2 must be an interesting node of type 3.

Finally, suppose that i_2 has no other child besides j_2. Let the parent of i_2 be j_2. The keyword j_2 must have another child, since otherwise, its parent would be the root of a 3-chain. Since no child of j_2 can be the root of a 3-chain, the keyword j_2 must be an interesting node of type 4. □

Hence, in the forest produced by Algorithm 1, every keyword has degree 1, and the output is an integer allocation.

This lemma, along with the analysis of the rounding subroutines described below, will give us all of the ingredients we need to prove that Algorithm 1 is a 3/2-approximation.

Proof of Theorem 2. Lemma 5 proves that when Algorithm 1 terminates, it returns a graph in which each keyword has degree 1. Each rounding step described below clearly takes polynomial time and makes at least one new bidder become inactive. Hence, the running time of the algorithm is polynomial.

For each rounding, we will associate each unit lost to 1/3 of the weight spent by a bidder that becomes inactive. Since each bidder becomes inactive only once, the total weight lost must be no larger than 1/3 of the weight of the optimal fractional solution. Therefore, the algorithm returns a solution with weight at least 2/3 of the optimal fractional solution and hence with weight at least two thirds of the optimal integral solution. □

2.3 The Rounding Subroutines

To simplify the exposition, we assume, without loss of generality, that all of the bids and budgets have been scaled so that the maximum bid is 1 (and hence the minimum budget is at least 1).

Type 1 Rounding

Rounding. Let i be the interesting node, j be its child of degree 2, and k be its grandchild. By Lemma 1, bidder k is saturated, while bidder i may have some unused budget $s \geq 0$.

Consider two ways to round the 2-chain rooted at i. In the first way, we remove the edge $\{i, j\}$. In the second way, we remove the edge $\{k, j\}$ and transfer as much as possible of the removed weight to the edge $\{i, j\}$ while maintaining feasibility. Of those two ways, we choose the one that incurs the smaller loss in the objective function.

Analysis. Since this rounding makes both i and k inactive, we can charge their total value, which is at least $\max(1, 2-s)$. The following lemma states the performance of this rounding.

Lemma 6. *Let L be the total weight lost by the rounding. Then $L \le \frac{1}{4}\max(1, 2-s)$.*

Proof. Let $\gamma = b_{ij}/b_{kj}$, $a = b_{kj}x_{ij}$ and $b = b_{kj}x_{kj}$. Then $w_{ij} = a\gamma$ and $w_{kj} = b$. We have $a + b = (x_{ij} + x_{kj})b_{kj} \le 1$ and $(a + b)\gamma = (x_{ij} + x_{kj})b_{ij} \le 1$. In the first way of rounding, we lose $a\gamma$. In the second way, if the the entire fraction of j previously allocated to k can be allocated to i, w_{ij} will increase by $x_{kj}b_{ij} = (b/b_{kj})b_{ij} = b\gamma$. So, we can transfer at least $\min(s, b\gamma)$, and the loss of this way is $b - \min(s, b\gamma)$. Hence, $L = \min(b - \min(s, b\gamma), a\gamma)$.

Consider the following two cases:

1. <u>Case $b\gamma \le s$.</u> If $\gamma > 1$, then the rounding actually increases the total weight, so assume that $\gamma \le 1$. We have $L = \min(b-b\gamma, a\gamma) \le \min(b-b\gamma, \gamma-b\gamma)$, and since b and γ are symmetric in this expression, we can assume that $b \le \gamma$. Thus,
$$L \le b - b\gamma \le b - b^2 \le \frac{1}{4} \le \frac{1}{4}\max(1, 2-s) \ .$$
2. <u>Case $b\gamma > s$.</u> Then $s < 1$. We claim that $a\gamma \le 1 - b$. This follows because if $\gamma \ge 1$, then $a\gamma \le 1 - b\gamma \le 1 - b$ and otherwise, $a\gamma < a \le 1 - b$. Thus,
$$\frac{L}{\max(1, 2-s)} \le \frac{\min(b-s, 1-b)}{2-s} \le \frac{1-s}{4-2s} = \frac{1}{2} - \frac{1}{4-2s} \le \frac{1}{4} \ .$$

□

Type 2 and Type 3 Roundings

Rounding. In type 2 rounding, we have a path with two keywords of degree 2; we consider all four possible ways of allocating each of those two keywords integrally to one of its two neighbors, transferring as much weight as possible in each allocation while respecting the LP constraints.

In type 3 rounding, we take a partial subtree rooted at the interesting node consisting of two of the paths below it. Together, these two paths define a path with two keywords of degree 2. We then proceed as in type 2 to define an integer allocation of those two keywords.

Analysis. In all cases, two saturated bidders become inactive. We show that the loss in the objective function is no more than 2/3 (Lemma 7), and charge it to the total value of the two bidders that become inactive, which is at least 2.

More precisely, consider a path i_1, j_1, i_2, j_2, i_3 where j_1 and j_2 are keywords. We show four ways to round this path so that one of the edges $\{i_1, j_1\}$, $\{i_2, j_1\}$ and one of the edges $\{i_2, j_2\}$, $\{i_3, j_2\}$ is removed in a way that loses no more than 2/3 (Lemma 7). If this path is a 3-chain rooted at i_1, then this procedure makes i_2 and i_3 inactive. Hence, we can charge the loss to the total value of i_2

and i_3, which is at least 2. On the other hand, if i_2 is the highest node on this path, j_1 and j_2 are degree-2 keywords and i_1 and i_3 do not have any degree-2 children, then this procedure makes i_1 and i_3 inactive. Hence, we can charge the loss to the total value of i_1 and i_3, which is at least 2.

Let $\gamma = b_{i_2j_1}/b_{i_1j_1}$, $\beta = b_{i_2j_2}/b_{i_3j_2}$, $a = x_{i_1j_1}b_{i_1j_1}$, $b = x_{i_2j_1}b_{i_1j_1}$, $c = x_{i_2j_2}b_{i_3j_2}$ and $d = x_{i_3j_2}b_{i_3j_2}$. Then $w_{i_1j_1} = a$, $w_{i_2j_1} = b\gamma$, $w_{i_2j_2} = c\beta$ and $w_{i_3j_2} = d$. This situation is illustrated in Fig. 3(a).

We consider four ways to round the path, illustrated in Figs. 3(b)-3(e). In the first way (Fig. 3(b)), we remove the edges $\{i_1, j_1\}$ and $\{i_3, j_2\}$, losing $a + d$. In the second way (Fig. 3(c)), we remove the edges $\{i_2, j_1\}$ and $\{i_2, j_2\}$, losing $b\gamma + c\beta$. In the third way (Fig. 3(d)), we remove the edges $\{i_1, j_1\}$ and $\{i_2, j_2\}$ and move part of the removed weight to $\{i_2, j_1\}$. If the entire amount of j_1 that was previously allocated to i_1 were allocated to i_2, then $w_{i_2j_1}$ would increase by $x_{i_1j_1}b_{i_2j_1} = (a/b_{i_1j_1})b_{i_2j_1} = a\gamma$. The budget freed up at i_2 from the removal of edge $\{i_2, j_2\}$ is $c\beta$. Thus, $w_{i_2j_1}$ can be increased to at least $b\gamma + \min(a\gamma, c\beta)$, causing a loss of $a + c\beta - \min(a\gamma, c\beta) = a + \max(0, c\beta - a\gamma)$. In the fourth way (Fig. 3(e)), we remove the edges $\{i_2, j_1\}$ and $\{i_3, j_2\}$ and transfer as much as possible of the removed weight to $\{i_2, j_2\}$, causing a loss of $d + \max(0, b\gamma - d\beta)$.

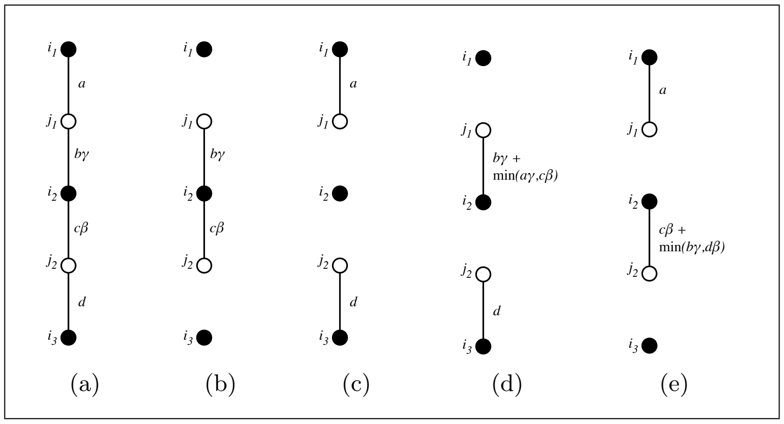
i_1
a
j_1
$b\gamma$
i_2
$c\beta$
j_2
d
i_3
$b\gamma$ + min($a\gamma$,$c\beta$)
$c\beta$ + min($b\gamma$,$d\beta$)
(a)
(b)
(c)
(d)
(e)

Fig. 3. A path with three bidders (a) and four ways to round that path (b)-(e).

Again, we choose the way that incurs the smallest loss. The following lemma states that this loss is never greater than 2/3.

Lemma 7. *Let*

$$L = \min\left(a + d, b\gamma + c\beta, a + \max(0, c\beta - a\gamma), d + \max(0, b\gamma - d\beta)\right) \quad .$$

Then $L \leq \frac{2}{3}$.

Proof. Since L does not decrease when b and c increase, we can assume that $(a+b)\max(1,\gamma)=1$ and $(c+d)\max(1,\beta)=1$. We also observe that for $\beta>1$, L is smaller than

$$\begin{aligned}L' &= \min(d\beta+\max(0,b\gamma-d\beta), a+\max(0,c\beta-a\gamma), b\gamma+c\beta, a+d\beta)\\ &= \min(d'+\max(0,b\gamma-d'), a+\max(0,c'-a\gamma), b\gamma+c', a+d')\ ,\end{aligned}$$

where $d'=d\beta$, $c'=c\beta$ with the condition $c'+d'\le 1$, i.e. L' is exactly the same as L with $\beta=1$. Hence, we can assume that $\beta\le 1$. Similarly we can assume that $\gamma\le 1$.

With these assumptions, we have $a+b=1$ and $c+d=1$. Thus, we can replace b by $1-a$ and c by $1-d$. We now need to bound the value of the following optimization problem:

$$\max L \quad \text{s.t.} \quad \begin{cases} 0\le a,d\le 1\\ L\le d+\max(0,\gamma-a\gamma-d\beta)=E_1\\ L\le a+\max(0,\beta-d\beta-a\gamma)=E_2\\ L\le \gamma+\beta-a\gamma-d\beta=E_3\\ L\le a+d=E_4\end{cases}\ ,$$

where $\gamma,\beta\le 1$ are parameters.

The rest of the proof is a case-by-case analysis. Let $A=\max(0,\gamma-a\gamma-d\beta)$ and $B=\max(0,\beta-d\beta-a\gamma)$. For each possibility for the values of A and B (expressed by a linear inequality), we have a linear program over 3 variables, hence at least three constraints are tight at the optimum.

We proceed with a case analysis. First, consider the cases where one of the first four constraints is tight.

1. Case $a=0$. Then

$$\begin{aligned}L &= \min(d+\max(0,\gamma-d\beta), \max(0,\beta-d\beta), \beta+\gamma-d\beta, d)\\ &= \min(\max(0,\beta-d\beta), d)\le \min(1-d,d)\le 1/2.\end{aligned}$$

2. Case $d=0$. Symmetric.
3. Case $a=1$. Then

$$L=\min(d, 1+\max(0,\beta-d\beta-\gamma), \beta-d\beta, 1+d)\le \min(1-d,d)\le 1/2.$$

4. Case $d=1$. Symmetric.

Next, consider the case where the first four constraints are not tight. Then, at least three of the four expressions E_i's are equal to L. Without loss of generality, assume that $\gamma\le\beta$. Since it is easy to check that $E_1+E_2\le E_3+E_4$, it must be that $L=E_1=E_2$ and there are only two more cases.

1. Case $L=E_1=E_2=E_4$. Since $E_1=E_4$, A is positive, and so $\gamma-a\gamma-d\beta=a$. Similarly, since $E_2=E_4$, B is positive, so we have $\beta-d\beta-a\gamma=d$. Solving

for a and d yields $a = (\gamma - \beta^2 + \gamma\beta)/(1+\gamma+\beta)$ and $d = (\beta - \gamma^2 + \gamma\beta)/(1+\gamma+\beta)$. Substituting the values of a and d into $L = E_4$ and using $\beta, \gamma \le 1$ yields

$$L = \frac{\gamma + \beta - \gamma^2 - \beta^2 + 2\gamma\beta}{1+\gamma+\beta} \le \frac{\gamma+\beta}{1+\gamma+\beta} \le \frac{2}{3} \ .$$

2. <u>Case $L = E_1 = E_2 = E_3$.</u> We consider three sub-cases according to the values A and B.
   - If both A and B are positive then $E_1 = E_3$ yields $d = \beta$, $E_2 = E_3$ yields $a = \gamma$, so we can replace a and d by their values and optimize over β and γ to get

     $$L = E_3 = \beta(1-\beta) + \gamma(1-\gamma) \le 1/4 + 1/4 = 1/2 \ .$$

   - If both A and B are 0 then $E_1 = E_2$ yields $a = d$. We replace d by a in $E_1 = E_3$ to solve for a and use $\beta, \gamma \le 1$ to obtain

     $$L = a = \frac{\gamma+\beta}{1+\gamma+\beta} \le 2/3 \ .$$

   - Finally, if only B is positive (since $\gamma \le \beta$) then $E_2 = E_3$ yields $a = \gamma$. We replace a by its value in $E_1 = E_3$ and solve for d, then optimize over γ and β to get

     $$L = d = \frac{\beta + \gamma - \gamma^2}{1+\beta} \le \frac{\beta + 1/4}{1+\beta} \le \frac{5/4}{2} = 5/8 \ .$$

Putting all the cases together gives $L \le \max(1/2, 2/3, 5/8) = 2/3$. □

Type 4 Rounding

Let v be the interesting node of type 4, and let u be its parent. Let h and k be the number of 1-chains and 2-chains rooted at children of v, respectively. We consider three cases based on the value of h and k.

Case 1: there are no 2-chains attached to v ($k = 0$). Then $h > 1$.

Rounding. Among the edges adjacent to v, retain the edge of largest weight and delete all others.

Analysis. We lose at most $h/(h+1)$ and make at least h saturated bidders inactive. Therefore, we can charge the loss to these nodes.

Case 2: there are no 1-chains attached to v ($h = 0$). Then $k > 1$. Let $p_1, p_2, \ldots p_k$ be v's children.

Rounding. We first round the path consisting of the edges $\{u, v\}$, $\{v, p_1\}$ and the 2-chain rooted at p_1, losing at most $2/3$ by Lemma 7. After this step, either p_1 or u is disconnected from v. We repeat the above step with the path containing the edge joining v and the other node (either u or p_1), $\{v, p_2\}$ and the 2-chain rooted at p_2. We repeat this k times.

Analysis. We lose at most $2k/3$ and make $2k$ saturated bidders inactive: $p_1, p_2, \ldots p_k$ and their grandchildren. Hence, we can charge the loss to these nodes.

Case 3: $h > 0$ and $k > 0$.

Rounding. We choose one 1-chain rooted at, say, q, and one 2-chain rooted at, say, p and round the path containing q, v, p and the 2-chain rooted at p.

Analysis. We lose at most $2/3$ by Lemma 7 and make two saturated bidders inactive: p's grandchild and either q or p. Hence, we can charge the loss to these nodes.

3 A $\sqrt{2}$-Approximation Algorithm for the Uniform Problem

In this section, we provide an algorithm that improves the approximation ratio to $\sqrt{2}$ for the uniform case of the problem. The main observation that leads to this improvement is that in the proof of Theorem 2, there was some weight that we could have charged to but that we did not use. For example, consider the type 2 rounding shown in Figure 3(b). We charged the loss of the rounding to the weight allocated to bidders i_2 and i_3, which must be at least 2 since these bidders are saturated. We can do better than this, however. In the rounding, a weight of a is deallocated from i_1. Because we rebalance according to Lemma 1 between every rounding, this is weight that will never be charged to again. Therefore, instead of charging to 2, we can actually charge to $2 + a$.

To make this more precise, we define an *active edge* to be an edge that is adjacent to an active bidder. During each rounding, the sum of the weights on active edges will decrease, both from active edges becoming inactive and from active edges being deleted or losing weight. Define the *accountable amount* of a rounding to be the amount by which this quantity decreases. We will show that the loss of each rounding can be charged to $(1 - 1/\sqrt{2})$ of the accountable amount of that rounding. Since each unit of accountable amount is charged at most once, this suffices to prove the approximation ratio.

The structure of Algorithm 2, our $\sqrt{2}$-approximation, is the same as that of Algorithm 1. The only difference is in the rounding subroutines and their analysis. Instead of choosing the rounding that minimizes the loss at each step, we choose the one that minimizes the ratio between the loss and the accountable amount. In the remainder of this section we prove the following.

Theorem 8. *Algorithm 2 is a polynomial-time $\sqrt{2}$-approximation for the uniform version of the Budgeted Allocation problem.*

Proof. As in Theorem 2, the algorithm terminates in polynomial time and outputs an integral solution. For each rounding subroutine, we show that we can charge the loss to $1 - 1/\sqrt{2}$ of the accountable amount. This implies that Algorithm 2 returns a solution of value at least $1/\sqrt{2}$ of optimal. □

We believe that Theorem 8 applies to the non-uniform version of the problem, but we have not been able to prove this, since it seems to involve calculations that are significantly more complicated than those in the proof of Lemma 7.

3.1 The Rounding Subroutines

For convenience, we define x to be $2 - \sqrt{2}$. For each rounding subroutine, we show that the ratio of the loss to the accountable amount is no greater than $x/2 = 1 - 1/\sqrt{2}$.

Type 1 Rounding

The rounding and analysis for type 1 is the same as for Algorithm 1. By Lemma 6, the ratio of the loss to the accountable amount is no greater than $1/4 < x/2$.

Type 2 Rounding

Rounding. Let i_1 be the interesting node and i_1, j_1, i_2, j_2, i_3 be the 3-chain associated with i_1, and let $a = w_{i_1 j_1}$, $b = w_{i_2 j_1}$, $c = w_{i_2 j_2}$ and $d = w_{i_3 j_2}$. Of the four ways to round this chain described in Fig. 3, we choose the rounding that minimizes the ratio of the loss over the accountable amount.

Analysis. The four ways to round the chain incur losses of $a+d$, $b+c$, $\max(a, c)$, and $\max(b, d)$, respectively. (Recall that in the uniform version $\beta = \gamma = 1$.) To derive the accountable amounts of the roundings, note that the first rounding makes i_2 and i_3 inactive, and thus makes all edges adjacent to these nodes inactive; the sum of these edges is at least 2, since these bidders are saturated. Furthermore, it also removes the active edge $\{i_1, j_1\}$. Thus, the accountable amount of the first rounding is $2 + a$. Similarly, the accountable amount of the second, third and fourth roundings are $2, 2 + a$ and 2, respectively. Hence, the following lemma shows that the we can always choose a rounding such that the ratio of the loss to the accountable amount is no greater than $x/2$.

Lemma 9. *Let*

$$R = \min\left(\frac{a+d}{2+a}, \frac{b+c}{2}, \frac{\max(a,c)}{2+a}, \frac{\max(b,d)}{2}\right) .$$

Then $R \leq x/2$.

Proof. The lemma is trivial if $\max(a,c) \leq x$ or $\max(b,d) \leq x$. Thus, assume that $\max(a,c) > x$ and $\max(b,d) > x$. Consider two cases:

1. <u>Case $b \leq d$.</u> Then $d > x$, and hence $c < 1-x$, $a > x$, and $b < 1-x$. We have

$$R \leq \min\left(\frac{a}{2+a}, \frac{\min(b+1-d,d)}{2}\right) \leq \min\left(\frac{a}{2+a}, \frac{b+1}{4}\right)$$
$$\leq \min\left(\frac{a}{2+a}, \frac{2-a}{4}\right) \quad .$$

   Since $a/(2+a)$ is increasing and $(2-a)/4$ is decreasing, the last expression gets its maximum when $a/(2+a) = (2-a)/4$, i.e. when $a = 2\sqrt{2}-2$. In that case, $R = x/2$.

2. <u>Case $b > d$.</u> Then $b > x$, and hence $a < 1-x$, $c > x$, and $d < 1-x$. Therefore

$$R \leq \frac{\min(c, a+d)}{2+a} \leq \frac{\min(1-d, a+d)}{2+a} \leq \frac{1+a}{4+2a} \leq \frac{2-x}{6-2x} = \frac{x}{2} \quad .$$

   The third inequality follows since $(1+a)/(4+2a)$ is increasing and $a < 1-x$.

□

Type 3 Rounding

Rounding. Let the interesting node be i_2; let j_1 and j_2 be two of its degree-2 children; and let i_1, i_3 be j_1's and j_2's children, respectively. Let $a = w_{i_1 j_1}$, $b = w_{i_2 j_1}$, $c = w_{i_2 j_2}$ and $d = w_{i_3 j_2}$. As in the previous section, there are four ways to round this path. We choose the rounding that minimizes the ratio between the loss and the accountable amount.

Analysis. The four ways to round this path incur losses of $a+d, b+c, \max(a,c)$ and $\max(b,d)$, respectively. The first rounding makes both i_1 and i_3 inactive, and thus has accountable amount at least 2. The second rounding makes, in addition to i_1 and i_3, the edges $\{i_2, j_1\}$ and $\{i_2, j_2\}$ inactive, and thus has an accountable amount of at least $2+b+c$. The third rounding makes i_1 and i_3 inactive. In addition, it makes $\{i_2, j_2\}$ inactive and increases the weight of the active edge $\{i_2, j_1\}$ by $\min(a,c)$. If $c > a$, this causes a loss of $c-a$ on the two active edges $\{i_2, j_1\}$ and $\{i_2, j_2\}$. Thus, the accountable amount of this rounding is at least $2+\max(c-a,0)$. Similarly, the accountable amount of the last rounding is at least $2+\max(b-d,0)$. Hence, the following lemma shows that we can always choose a rounding such that the ratio of the loss to the accountable amount is no greater than $x/2$.

Lemma 10. *Let*

$$R = \min\left(\frac{b+c}{2+b+c}, \frac{a+d}{2}, \frac{\max(b,d)}{2+\max(b-d,0)}, \frac{\max(a,c)}{2+\max(c-a,0)}\right) \quad .$$

Then $R \leq \frac{x}{2}$.

Proof. The lemma is trivial if $\max(a,c) \le x$ or $\max(b,d) \le x$. Thus, we assume that $\max(a,c) > x$ and $\max(b,d) > x$. Consider two cases:

1. Case $b \le d$. Then $d > x$, $c < 1-x$, $a > x$, and $b < 1-x$. Thus
$$R = \min\left(\frac{b+c}{2+b+c}, \frac{a+d}{2}, \frac{d}{2}, \frac{a}{2}\right) \le \min\left(\frac{b+c}{2+b+c}, \frac{a+d}{4}\right)$$
$$\le \min\left(\frac{b+c}{2+b+c}, \frac{2-b-c}{4}\right) .$$
As in the proof of Lemma 9, this function achieves its maximum when $b+c = 2\sqrt{2}-2$, in which case the ratio is $x/2$.
2. Case $b > d$. Then $b > x$, $a < 1-x$, $c > x$, and $d < 1-x$. Considering all ways to round except the first, we can round at cost
$$\min\left(\frac{a+d}{2}, \frac{b}{2+b-d}, \frac{c}{2+c-a}\right) \le \left(\frac{2-b-c}{2}, \frac{b}{1+b+c}, \frac{c}{1+b+c}\right)$$
$$\le \left(\frac{2-b-c}{2}, \frac{b+c}{2+2b+2c}\right) .$$
Let $y = b+c$. Note that $(2-y)/2$ is decreasing and $y/(2+2y)$ is increasing when $y \ge 0$. Therefore, the minimum is achieved when $(2-y)/2 = y/(2+2y)$, which happens when $y = \sqrt{2}$. In this case, the ratio is $x/2$.

$\square$

Type 4 Rounding

Let v be the interesting node of type 4, u be its parent, and h and k be the number of 1-chains and 2-chains rooted at children of v, respectively. As illustrated in Fig. 4, let a be the weight of $\{u,v\}$, $e_1, e_2, \ldots, e_h$ be the weights of the edges joining v and its children that are roots of the 1-chains, $b_1, b_2, \ldots, b_k$ be the weights of the edges joining v and its children that are roots of the 2-chains, $c_1, c_2, \ldots, c_k$ be the weights of the edges joining these children of v and their degree-2 children, and $d_1, d_2, \ldots, d_k$ be the weights of the edges joining these degree-2 grandchildren of v to their children. In the following, we will sometimes use these labels to refer to the edges themselves instead of their weights, but the meaning should be clear from the context. We consider three cases based on the values of h and k.

Case 1: there are no 2-chains attached to v ($k = 0$). Then $h > 1$.

Rounding. Without loss of generality, assume that $e_1 \ge e_\ell$ for all ℓ. Consider two ways to round: remove everything but e_1 and remove everything but a. Of these, choose the rounding that minimizes the ratio of the loss over the accountable amount.

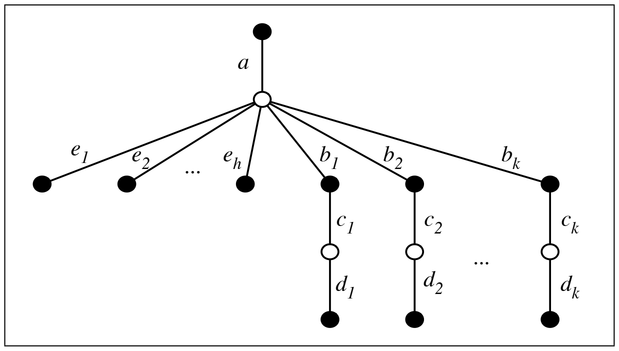

Fig. 4. An interesting node of type 4 and its associated structure.

Analysis. If the algorithm chooses the first way, the loss is at most $\delta - e_1 \leq \delta - (\delta - a)/h$, where $\delta = a + \sum_{\ell=1}^{k} e_\ell$. Since $\delta \leq 1$, the loss is at most $1 - (1-a)/h$. The accountable amount is at least $h + a$, since the rounding makes all children of v and the edge a inactive. If the algorithm chooses the second way, the loss is at most $1 - a$ and the accountable amount is at least h. Hence, the ratio is no greater than

$$\min\left(\frac{1-(1-a)/h}{h+a}, \frac{1-a}{h}\right) \leq \min\left(\frac{1-(1-a)/2}{2+a}, \frac{1-a}{2}\right) \quad ,$$

where the inequality follows because both functions are decreasing on h. The expression on the right is maximized when the two operands are equal, which happens when $a = \sqrt{2} - 1 = 1 - x$. Hence, the ratio is no greater than $x/2$.

Case 2: there are no 1-chains attached to v ($h = 0$). Then $k > 1$.

Rounding. Without loss of generality, assume that $b_1 \geq b_\ell$ for all ℓ. We round based on the following cases.

1. If there exists an ℓ such that $\max(b_\ell, d_\ell) \leq x$, we remove b_ℓ and d_ℓ and increase c_ℓ to $c_\ell + \min(b_\ell, d_\ell)$.
2. If there exists an ℓ such that $d_\ell \leq x/2$, we remove d_ℓ.
3. Otherwise, we remove a, remove the c_ℓ's, remove all of the b_ℓ's except b_1, and increase b_1 by $\min(c_1, \delta - b_1)$, where $\delta = \sum_{\ell=1}^{k} b_\ell + a$.

Analysis. The analysis for each case is as follows.

1. We lose $\max(b_\ell, d_\ell) \leq x$ and have an accountable amount of at least 2, since all of the edges attached to the saturated bidders of the 2-chain are made inactive. Hence, the ratio of the amount lost to the accountable amount is at most $x/2$.

2. We lose at most $x/2$ when we remove d_ℓ, and the accountable amount is at least 1.
3. The loss of this rounding is $L = \max(c_1, \delta - b_1) + \sum_{\ell=2}^{k} c_\ell$. The accountable amount is at least $2k + a$, since we make $2k$ bidders inactive and delete the active edge $\{u, v\}$. The following lemma shows that the ratio of the loss over the accountable amount is no greater than $x/2$.

   Lemma 11. *Let $R = L/(2k + a)$. Then $R \le x/2$.*

   Proof. Suppose first that $b_1 > x$. Then $b_\ell < 1 - x$ for all $\ell > 1$. Since $\max(b_\ell, d_\ell) > x$, we therefore have that $d_\ell > x$ and hence $c_\ell < 1 - x$ for all $\ell > 1$. Since $d_1 > x/2$, we have $c_1 < 1 - x/2$. Therefore, $\max(c_1, \delta - b_1) \le \max(c_1, 1 - b_1) < 1 - x/2$, and

   $$\frac{L}{2k+a} \le \frac{L}{2k} < \frac{1 - x/2 + (k-1)(1-x)}{2k} = \frac{x}{4k} + \frac{1-x}{2} \le \frac{4-3x}{8} < x/2 \ .$$

   Now suppose that $b_1 \le x$. Then $b_\ell \le x$ for all ℓ, and hence $d_\ell > x$ and $c_\ell < 1 - x$ for all ℓ. If $c_1 \ge \delta - b_1$, then $L \le k(1 - x)$, and the ratio is no greater than $(1 - x)/2 < x/2$. If $\delta - b_1 \ge c_1$, then

   $$\begin{aligned} L \le \delta - b_1 + \sum_{\ell=2}^{k} c_\ell \le \delta - (\delta - a)/k + \sum_{\ell=2}^{k} c_\ell \\ \le 1 - (1-a)/k + \sum_{\ell=2}^{k} c_\ell \ . \end{aligned}$$

   Hence,

   $$R \le \frac{1 - (1-a)/k + (k-1)(1-x)}{2k+a} = \frac{k - 1 + a + k(k-1)(1-x)}{k(2k+a)} \ .$$

   – If $k = 2$ then $R \le (3 - 2x + a)/(8 + 2a) \le (4 - 2x)/10 < x/2$.
   – If $k = 3$ then $R \le (2 + a + 6(1 - x))/(3(6 + a)) \le (9 - 6x)/21 < x/2$.
   – If $k \ge 4$ then $a < xka/2$. Since $k - 1 + k(k - 1)(1 - x) < xk^2$, we have $R < x/2$.

   □

Case 3: $h > 0$ and $k > 0$. Without loss of generality, assume that $e_1 \ge e_\ell$ for all ℓ.

Rounding. We round based on the following cases.

1. If there exists an ℓ and m such that $\max\{e_\ell, c_m\} \le x$, then remove e_ℓ and c_m and increase b_m by $\min\{e_\ell, c_m\}$.
2. If there exists an ℓ such that $\max\{b_\ell, d_\ell\} \le x$, we remove b_ℓ and d_ℓ and increase c_ℓ by $\min\{b_\ell, d_\ell\}$.
3. Otherwise, if $b_1 \ge d_1$, then remove all edges incident to the interesting node except b_1, and all of the d_ℓ's.
4. Otherwise, remove all edges incident to v except e_1, and all of the c_ℓ's.

Analysis. The analysis for each case is as follows.

1. The loss is at most x, and the accountable amount is at least 2.
2. Again, the loss is at most x, and the accountable amount is at least 2.
3. Since $b_1 \geq d_1$, we have $b_1 > x$, and hence $e_\ell < 1 - x$ for all ℓ. This implies that $c_\ell > x$, and hence $d_\ell < 1 - x$ for all ℓ. Hence, the loss is at most
$$1 - b_1 + \sum_{\ell=1}^{k} d_\ell \leq 1 - x + \sum_{\ell=1}^{k} d_\ell \leq (k+1)(1-x) \ ,$$
and the accountable amount is at least $2k + h$. Hence, the ratio is at most
$$\frac{(k+1)(1-x)}{2k+h} \leq \frac{2}{3}(1-x) < \frac{x}{2} \ .$$
4. Since $b_1 < d_1$, we have $d_1 > x$, and hence $c_1 < 1 - x$. Therefore, $e_\ell > x$ for all ℓ, and since $x > 1/2$, this implies that in fact $h = 1$. It also implies that $b_\ell < 1 - x$, $d_\ell > x$, and $c_\ell < 1 - x$ for all ℓ. Therefore, the loss is again at most $(k+1)(1-x)$, and the accountable amount is at least $2k+1$. As before, the ratio is thus at most $x/2$.

4 APX-hardness

In this section, we show that the Budgeted Allocation problem is APX-hard, which implies that there is no PTAS for the problem unless $P = NP$. Our proof uses a reduction from *Three Dimensional Matching* and the following known result.

Theorem 12 ([19]). *There exists a constant $c_1 > 0$ such that given an instance of maximum 3D-matching in which each element appears in no more than k triples, for some constant k, it is NP-hard to determine whether the maximum matching has size n or size less than $(1 - c_1)n$.*

Theorem 13. *Budgeted Allocation is APX-hard, even in the uniform version.*

Proof. We give the following reduction from 3D-matching to Budgeted Allocation. Given an instance (U, V, W, E) of 3D-matching, where U, V, W are sets of vertices such that $|U| = |V| = |W| = n$ and $|E| \subseteq U \times V \times W$ such that $|E| = m \leq kn$, we construct an instance of Budgeted Allocation as follows. First, we create $3n$ keywords corresponding to $3n$ vertices and m bidders corresponding to m triplets. Bidder i makes a bid of 1 for one of these keywords j if and only if i represents some triplet that contains the vertex represented by j. Next, we create $m - n$ new keywords and let every bidder make a bid of 3 for each of these keywords. We set the budget of each bidder to 3.

First, it is clear that if the 3D-matching instance has a matching of size x, then the instance of Budgeted Allocation has a solution of value $3x + 3(m - n)$. On the other hand, if the Budgeted Allocation instance has a solution of value

$3m - z$, then there exist at most z bidders whose budgets are not exhausted. Therefore, the 3D-matching instance has a matching of size at least $n - z$.

Now, consider two instances T_1 and T_2 of 3D-matching where T_1 has a matching of size n and T_2's maximum matching has size $n - c_1 n$. Let A_1 and A_2 be the corresponding instances of Budgeted Allocation. Then A_2 does not have a solution of value larger than $3m - c_1 n$. Therefore, if we have an approximation algorithm which outputs a solution of value larger than $3m - c_1 n$ for A_1 then we can distinguish between T_1 and T_2. Let c_2 be some constant such that $c_2 < c_1/3k$. Then $3m(1 - c_2) > 3m - c_1 n$, and thus Budgeted Allocation is NP-hard to approximate within $(1 - c_2)$. □

References

1. Andelman, N., Mansour, Y.: Auctions with budget constraints. In: SWAT 2004 (LNCS 3111), Springer 26–38
2. Feige, U., Vondrak, J.: Approximation algorithms for allocation problems: Improving the factor of 1 - 1/e. In: FOCS 2006. 667–676
3. Lahaie, S., Pennock, D., Saberi, A., Vohra, R.: Sponsored search auctions. In Nisan, N., Roughgarden, T., Tardos, E., Vazirani, V., eds.: Algorithmic Game Theory. Cambridge University Press (2007) 699–716
4. Buchbinder, N., Jain, K., Naor, J.: Online primal-dual algorithms for maximizing ad-auctions revenue. In: ESA 2007 (LNCS 4698), Springer 253–264
5. Goel, G., Mehta, A.: Online budgeted matching in random input models with applications to adwords. In: SODA 2008. 982–991
6. Mahdian, M., Nazerzadeh, H., Saberi, A.: Allocating online advertisement space with unreliable estimates. In: EC 2007. 288–294
7. Mehta, A., Saberi, A., Vazirani, U., Vazirani, V.: Adwords and generalized online matching. J. ACM **54**(5) (2007) 22
8. Lehmann, B., Lehmann, D., Nisan, N.: Combinatorial auctions with decreasing marginal utilities. Games and Economic Behavior **55**(2) (2006) 270–296
9. Chakrabarty, D., Goel, G.: On the approximability of budgeted allocations and improved lower bounds for submodular welfare maximization and GAP. Manuscript (2008)
10. Srinivasan, A.: Budgeted allocations in the full-information setting. Manuscript (2008)
11. Dobzinski, S., Schapira, M.: An improved approximation algorithm for combinatorial auctions with submodular bidders. In: SODA 2006. 1064–1073
12. Khot, S., Lipton, R., Markakis, E., Mehta, A.: Inapproximability results for combinatorial auctions with submodular utility functions. In: WINE 2005 (LNCS 3828). 92–101
13. Mirrokni, V., Schapira, M., Vondrak, J.: Tight information-theoretic lower bounds for welfare maximization in combinatorial auctions. Manuscript (2007)
14. Vondrak, J.: Optimal approximation for the submodular welfare problem in the value oracle model. In: STOC 2008 (to appear). (2008)
15. Fleischer, L., Goemans, M., Mirrokni, V., Sviridenko, M.: Tight approximation algorithms for maximum general assignment problems. In: SODA 2006. 611–620
16. Chekuri, C., Khanna, S.: A PTAS for the multiple knapsack problem. In: SODA 2000. 213–222

17. Shmoys, D., Tardos, E.: An approximation algorithm for the generalized assignment problem. Mathematical Programming **62** (1993) 461–474
18. Andelman, N.: Online and strategic aspects of network resource management algorithms. PhD thesis, Tel Aviv University (2006)
19. Kann, V.: Maximum bounded 3-dimensional matching is MAX SNP-complete. Inform. Process. Lett. **37** 27–35