Averaging Techniques for Competitive Auctions*

Takayuki Ichiba† Kazuo Iwama‡

Abstract

We study digital-goods auctions for items in unlimited supply introduced by Goldberg, Hartline and Wright. Since no deterministic algorithms are competitive for this class of auctions, one of the central research issues is how to obtain a nice probabilistic distribution over truthful algorithms. In this paper, we introduce a rather systematic approach to this goal: Consider for example the Sampling Cost Share (SCS) auction. It is well known that SCS works well if the the current bid vector produces many winners against $\mathcal{F}^{(2)}$, the standard benchmark algorithm for competitive analysis. In fact, its competitive ratio is approaching to 2.0 as k (= the number of $\mathcal{F}^{(2)}$ winners) grows. On the other hand, its competitive ratio becomes as bad as 4.0 for $k = 2$. Our new approach is to develop a sequence of similar cost-share type algorithms, DCS_k, which work well for small k. Now we choose a sufficiently large constant N and run DCS_1, DCS_2, ..., DCS_N and SCS with probabilities p_1, p_2, ..., p_N and q, respectively. It should be noted that we can use LP to obtain optimal p_1, p_2,..., p_N and q. By this averaging method, we can improve the competitive ratio of SCS from 4.0 to 3.531 and that of the currently best Aggregated Υ3 algorithm due to Hartline and McGrew from 3.243 to 3.119.

1 Introduction

It is usually hard to design a single algorithm that works uniformly well for all possible inputs. More likely, suppose that input instances are divided into two groups, X and Y. We have two algorithms A and B and we know that algorithm A works well for X and B for Y. Unfortunately, however, we have no way of telling whether the current instance x is in X or in Y in advance, and furthermore we can run only one of A and B for x. In other words, we cannot use the obvious approach of testing both A and B simultaneously and taking a better outcome. Situations like this are quite common in several circumstances including those in online computation and our standard approach in such a case is to use randomization. Namely, we just apply A and B to x with some (fixed) probability distribution, which is one of the most popular paradigms in algorithm design.

This simple approach is particularly important in designing digital-goods auctions and there are many successful examples [4, 1, 2]. For instance, consider the algorithm due to Hartline and McGrew [7]: The basic idea of this algorithm is to optimize the auction algorithm for three bidders where the key issue is to carefully characterize the different types of three bids, such as the one that all three bids are similar, the one that only one bid is much higher than the others, etc. As a result, the probabilistic distribution of the price offered to one bidder, which is calculated from the bids of the other two in order to maintain truthfulness, is very sophisticated or even artistic.

In this paper, we are interested in more systematic approach to this probabilistic mixture of auction algorithms. Our model is a standard digital-goods auction introduced in [5]. Several different algorithms are known, including Deterministic Optimal Threshold (DOT) [5], Sampling Cost Share (SCS) [4], Consensus Revenue Estimate (CORE) [3] and Aggregated Upsilon-3 (AggΥ3) [7]. Thus the research has been quite active in the last decade, but we still have a relatively large gap between the upper and lower bounds of their competitive ratio; the current best upper bound (achieved by AggΥ3) is 3.243 and the best lower bound is 2.42 [3].

Our Contribution. Our approach includes two main issues, finding a parameter that characterizes each input and figuring out how to exploit that parameter. Consider for example the SCS auction. Its worst-case competitive ratio is 4.0, but this value is achieved for bid vectors in $\mathcal{B}_2$, where $\mathcal{B}_k$ is the set of bid vectors $\mathbf{b}$ such that $\mathcal{F}^{(2)}$ (=the benchmark auction for defining the competitive ratio of truthful auctions) has k winners for $\mathbf{b}$. The competitive ratio of SCS for $\mathcal{B}_k$ is then decreasing monotonically as k increases, approaching to 2.0. Thus this k appears to be a nice characteristic parameter.

How can we exploit this parameter? Notice that the reason for the SCS's relatively poor performance for $\mathcal{B}_2$ is this: When SCS divides all the bidders into two groups at random, the two bidders telling the highest

*Supported by KAKENHI 1920000 and 16092215

†School of Informatics, Kyoto University, Kyoto 606-8501, Japan. ichiba@kuis.kyoto-u.ac.jp

‡School of Informatics, Kyoto University, Kyoto 606-8501, Japan. iwama@kuis.kyoto-u.ac.jp

two values (who become winners in $\mathcal{F}^{(2)}$) are fallen into the same group with high ($= 1/2$) probability. In other words, it should be nice if we could always send these two bidders into different groups. It turns out that the resulting auction is equivalent to the Vickrey auction [11] whose competitive ratio for such bid vectors is 2.0. Similarly, we can apparently improve SCS by dividing the top $2k$ bidders in $\mathcal{B}_{2k}$ ($k \geq 2$) into the two groups *evenly*. We call such a modified SCS algorithm DCS_k. One might think that this would violate truthfulness, but fortunately we can prove it does not.

The problem is that we have to know the value of k *without looking at the bid vector*. (Otherwise it does violate truthfulness.) It is of course impossible to do so deterministically, but we can do so *probabilistically*. Namely, we first choose a sufficiently large constant N. Then for a given bid vector $\mathbf{b}$ such that $|\mathbf{b}| = n$, if $n \leq 2N$ we run DCS_1, ..., $DCS_{\lceil n/2 \rceil}$ with probabilities p_1, ..., $p_{\lceil n/2 \rceil}$, respectively. If $n > 2N$, we run DCS_1, ..., DCS_N and SCS with probabilities p_1, ..., p_N and q, respectively. If the current vector is in $\mathcal{B}_{2k}$, then DCS_k, which is run with probability p_k, works well. It turns out that we can use a simple LP to obtain optimal values for these probabilities, which is another merit of this approach.

We call such an algorithm *Averaged SCS* or *A-SCS* and prove that its competitive ratio is at most 3.531. Completely the same idea allows us to obtain Averaged AggΥ3 whose competitive ratio is turned out to be 3.119, thus improving the current best competitive ratio of digital-goods auctions (3.243 by AggΥ3).

Related Work. As mentioned before, there is also a strong need for averaging in online algorithms. Early and important work includes the optimal randomized algorithm for the ski-rental problem by Karlin, Manasse, McGeoch, and Owicki [8], where the number of ski tours is a natural characteristic parameter. LP plays an important role not only in its direct optimization purpose, but also in several occasions in design and analysis of algorithms and their complexity. Mechanism design is not exceptional; recently for instance, Guo and Conitzer use LP for optimizing redistribution payments in the VCG mechanism [6].

2 Auction Model

We study so-called *digital-goods auctions* which sell same items in unlimited supply (e.g. software). Note that in this class of auctions, more than one bidder can purchase the items simultaneously and pay their own (possibly different) prices offered to each bidder by the auctioneer. Our model is exactly the same as [5]:

Definition 1. (Digital-goods auctions [5])

1. *Each bidder i ($1 \leq i \leq n$) submits a bid $b_i > 0$. From the bid vector $\mathbf{b} = (b_1, ..., b_n)$, the auctioneer computes the output, each price t_i offered to bidder i, using her auction algorithm $\mathcal{A}$.*

2. *Each bidder i wins (gets an item) if $b_i \geq t_i$ at a winning price p_i which is equal to price t_i. Otherwise, she loses and $p_i = 0$ (gets nothing and pays nothing).*

3. *Each bidder has her own value for an item, called a utility, u_i, which she is willing to pay at most for the item. If she wins, she obtains a profit $u_i - p_i$. If she loses her profit is 0.*

4. *Bidders have full knowledge of the auctioneer's algorithm $\mathcal{A}$.*

5. *Each bidder bids so as to maximize her profit.*

6. *The auctioneer's profit (or auction's profit) $\mathcal{A}(\mathbf{b})$ is the total sales $\sum_i p_i$ and our goal in designing $\mathcal{A}$ is to maximize it.*

7. *Ties are broken arbitrarily if necessary.*

Thus an auction is a game between the bidders and the auctioneer: Each bidder attempts to maximize her profit, or determines her bid b_i in order to get as low t_i as possible under the current algorithm $\mathcal{A}$. On the other hand, the auctioneer should design her algorithm so that each bidder is forced to bid a high value that is as close to her utility as possible. Based on this, we usually restrict our attention to so-called *truthful auctions.*

8. *An auction algorithm $\mathcal{A}$ is said to be truthful (or also called incentive compatible) if each bidder i can maximize her profit by bidding her utility u_i.*

9. *$\mathcal{A}$ is said to be bid-independent if the price t_i does not depend on b_i, i.e., it can be written by a function f as $t_i = f(b_1, ..., b_{i-1}, b_{i+1}, ..., b_n)$.*

10. *The optimal single price omniscient auction, $\mathcal{F}$, is the following mechanism: For a bid vector $\mathbf{b}$, let v_i be the i-th largest bid. Then the price $t_j = v_k$ is all the same for $1 \leq j \leq n$ such that $kv_k \geq iv_i$ for any $i \neq k$. $\mathcal{F}^{(2)}$ is such an auction with at least two winners, i.e., $\mathcal{F}^{(2)}(\mathbf{b}) = \max_{2 \leq i \leq n} iv_i$.*

11. *$\mathcal{F}^{(2)}$ is used as a standard benchmark for competitive analysis, i.e., the competitive ratio of an algorithm $\mathcal{A}$ is defined to be minimum r such that*

$$E[\mathcal{A}(\mathbf{b})] \geq \mathcal{F}^{(2)}(\mathbf{b})/r$$

holds for any bid vector $\mathbf{b}$. We call a bid-independent algorithm $\mathcal{A}$ as a competitive auction if the competitive ratio of $\mathcal{A}$ is bounded. (End of Definition)

Recall that our general goal is to design *truthful* auctions with as small competitive ratio as possible. Goldberg, Hartline and Write [5] have shown that an auction $\mathcal{A}$ is truthful if and only if $\mathcal{A}$ is bid-independent. So we can restrict our attention to only bid-independent auctions. As mentioned before, SCS [4], whose competitive ratio is 4.0, was improved by CORE [1] with a competitive ratio of 3.39. The current best algorithm, AggΥ3 due to Hartline and McGrew [7] has an even better competitive ratio of 3.243. We will explain this algorithm more in detail in Section 4. As for lower bounds, [4] first proved a lower bound of 2.0 and it was then improved to 2.42 by [3].

3 Averaging SCS

3.1 SCS algorithm Seeking competitive algorithms was an obvious main goal of auction research in its early stage, but it was not that easy; SCS appeared in [4] after several unsuccessful attempts:

Auction 1 Sampling Cost Share (SCS)

Partition the set $\mathbf{b}$ of bids uniformly at random into $\mathbf{b}_1$ and $\mathbf{b}_2$, i.e., put each bid into $\mathbf{b}_1$ with probability 1/2 and into $\mathbf{b}_2$ with probability 1/2.
Calculate $\mathcal{F}_1 = \mathcal{F}(\mathbf{b}_1)$ and $\mathcal{F}_2 = \mathcal{F}(\mathbf{b}_2)$.
Run $\text{CostShare}_{\mathcal{F}_2}$ on $\mathbf{b}_1$ and $\text{CostShare}_{\mathcal{F}_1}$ on $\mathbf{b}_2$.

CostShare_C [9, 10] is the auction algorithm that finds the largest k such that the highest k bids are at least C/k for a given bid vector $\mathbf{b}$, and charges each bidder C/k.

CostShare_C is bid-independent and so is SCS. Notice that CostShare_C has profit C if $C \leq \mathcal{F}(\mathbf{b})$; otherwise it has no profit. Therefore SCS has profit $\min(\mathcal{F}_1, \mathcal{F}_2)$ except for the special case satisfying $\mathcal{F}_1 = \mathcal{F}_2$, where its profit is $2\mathcal{F}_1$.

LEMMA 3.1. [4] *Given a bid vector* $\mathbf{b}$*, let* k *be the number of winners in* $\mathcal{F}^{(2)}(\mathbf{b})$*. Then the profit of SCS satisfies*

$$\frac{E[SCS(\mathbf{b})]}{\mathcal{F}^{(2)}(\mathbf{b})} \geq \frac{1}{2} - \binom{k-1}{\lfloor \frac{k}{2} \rfloor} 2^{-k}.$$

This bound is tight, i.e., the equality holds for bid vector $\mathbf{b} = (1, ..., 1)$. Also the right hand side becomes minimum for $k = 2$ and its value then is 1/4. So the SCS algorithm has an worst-case competitive ratio of 4.0.

3.2 Basic Idea of Averaging Consider a bid vector $\mathbf{b} = (1 + \delta, 1, \varepsilon, ..., \varepsilon)$. If $\delta(> 0)$ and $\varepsilon(> 0)$ are very small, $\mathcal{F}^{(2)}(\mathbf{b})$ is approximately 2.0. SCS sends each bid into two groups $\mathbf{b}_1$ and $\mathbf{b}_2$ at random, so with probability 1/2, the highest two values go to different groups, say $1 + \delta$ to $\mathbf{b}_1$ and 1 to $\mathbf{b}_2$ (Case 1). Also with probability 1/2, the two values go to the same group (Case 2). Notice that SCS can get a profit of 1.0 (i.e., CostShare on $\mathbf{b}_1$ only succeeds) in Case 1 and almost no profit in Case 2. Thus its expected profit is 0.5 and its competitive ratio is no better than 4.0.

However, we can avoid Case 2, by using the new rule that we consider *only the highest two bids* and always put them into different groups. Note that this algorithm is exactly the same as the Vickrey auction and its competitive ratio is 2 for the above bid vector. This rule is generalized to the highest $2m$ bidders, whose bids are divided into two groups at random but so that each group accepts exactly m ones. Now we obtain the following algorithm (see Fig. 1):

Auction 2 Dividing Cost Share (DCS_m)

Pick up the highest $2m$ bids (if there are less than $2m$ bidders, add an appropriate number of '0' bids).
Divide them to two bins $\mathbf{b}_1$ and $\mathbf{b}_2$ randomly, such that each bin has exactly m bids.
Compute $B_1 = \mathcal{F}(\mathbf{b}_1)$, $B_2 = \mathcal{F}(\mathbf{b}_2)$.
Run CostShare_{B_2} on $\mathbf{b}_1$ and CostShare_{B_1} on $\mathbf{b}_2$.

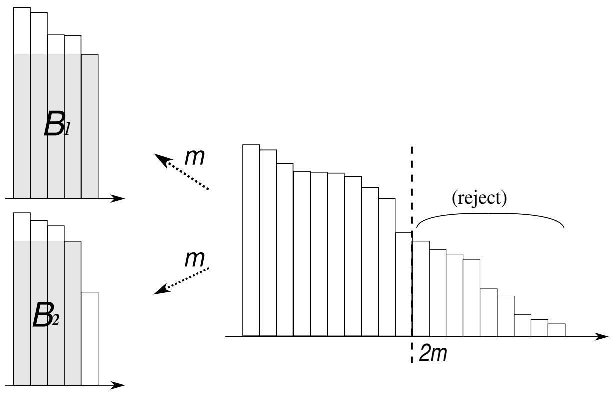

Fig. 1 The DCS_m mechanism

THEOREM 3.1. *DCS_m is bid-independent.*

Proof. Let b_i be the bid of the ith player, $\mathbf{b}_{-i} = (b_1, ..., b_{i-1}, b_{i+1}, ...b_n)$ and x be the $2m$th largest bid in $\mathbf{b}_{-i}$. It is enough to show that the winning price offered to the ith player does not depend on value b_i. (Case 1) If $b_i < x$, this bidder loses. (Case 2) Suppose $b_i > x$ and without loss of generality b_i is put into $\mathbf{b}_1$. Then the members in the other set $\mathbf{b}_2$ are obviously determined without depending on value b_i. So the value C of CostShare_C on $\mathbf{b}_1$ does not depend on value b_i. Note that CostShare_C is bid-independent [4] and therefore the winning price for the ith player is independent of b_i. (Case 3) $b_i = x$. In this case we break the ties by indices of bidders. Then if bidder i is in $\mathbf{b}_1$ or $\mathbf{b}_2$, then the argument is the same as Case 2. Otherwise she loses. □

As mentioned before, DCS_1 has a competitive ratio of 2.0 for bid vectors in $\mathcal{B}_2$. However, its competitive ratio for bid vectors in $\mathcal{B}_k$ gets worse quickly as k grows (3.0 for $k = 3$, 4.0 for $k = 4$ and so on). Even so, the fact that it works well for $\mathcal{B}_2$ is quite attractive and we can consider the algorithm that runs DCS_1 with probability 15/47 and SCS with 32/47. A simple calculation shows that this algorithm achieves a competitive ratio of 3.616, already much better than SCS.

A natural extension is to use not only DCS_1 but also many DCS_k for $k \geq 2$. Note that each DCS_k has a property similar to DCS_1, i.e., its competitive ratio is better than SCS for $\mathcal{B}_i$ ($i \leq 2k$), becomes minimum (=2.0) for $\mathcal{B}_{2k}$ and increases rapidly for $\mathcal{B}_j$ ($j > 2k$) (see Fig. 2 and formal proofs are given in Sec. 3.4). Suppose that $|\mathbf{b}| = 2m$. Then we can mix DCS_1, ..., DCS_m with probabilities p_1,...,p_m, respectively so that the expected profit will be maximum and we can calculate such optimal probabilities by LP. Thus so far so good, but unfortunately the competitive ratio of this mixed algorithm is getting worse as m grows and there is no obvious way of bounding it from above. So we use this algorithm only up to $|\mathbf{b}| \leq N$ for some constant N and if $|\mathbf{b}| > 2N$ we mix DCS_1, ..., DCS_N and SCS with optimum probabilities.

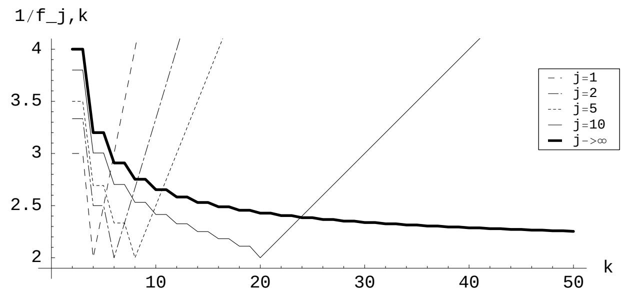

Fig. 2 Competitive ratio of DCS_j as a function of k

3.3 Averaged Version of SCS For a constant $N(\geq 2)$, we define A-SCS_N as follows. As will be seen later, the competitive ratio of A-SCS_N gets better for a larger N, but the size of LP we need to solve for defining A-SCS_N gets also larger (harder to solve) as N grows (we select $N = 50$ later). For exposition, we define A-SCS_N as a combination of A-SCS_N^1 for $n \leq 2N$ and A-SCS_N^2 for $n > 2N$. Here, n is the size of the bid vector $\mathbf{b}$. (Note that A-SCS_N works for any n regardless of the value of N.) Also, we can assume that n is even (otherwise just add a bidder whose bid is zero).

Auction 3 Algorithm A-SCS_N^1

Input: Bid vector $\mathbf{b}$ such that $|\mathbf{b}| = n \leq 2N$.

Optimize $p_1, ..., p_{n/2}$ by solving the LP given below.
Execute DCS_1, ..., $DCS_{n/2}$ with probability $p_1, ..., p_{n/2}$ respectively.

Probabilities $p_1, ..., p_{n/2}$ are obtained as a solution of the following LP:

$$\begin{aligned} \text{maximize}: \ & S_1 \\ \text{subject to}: \ & -\sum_{j=1}^{n/2} f_{j,k} \cdot p_j + S_1 \leq 0 \\ & \text{for } k = 2, ..., n, \\ & \sum_{j=1}^{n/2} p_j = 1, \\ & \text{and } p_1, ..., p_{n/2}, S_1 \geq 0, \end{aligned}$$

where $f_{j,k}$, given in Lemma 3.2 later, denotes the inverse competitive ratio of DCS_j, i.e., the minimum of $DCS_j(\mathbf{b})/\mathcal{F}^{(2)}(\mathbf{b})$ for any bid vector $\mathbf{b} \in \mathcal{B}_k$. Each of the first $n-1$ constraints means that the profit of A-SCS_N^1 is not less than $S_1 \cdot \mathcal{F}^{(2)}(\mathbf{b})$ for any $k \in \{2, ..., n\}$. Thus maximizing the objective function S_1 leads to minimizing the competitive ratio of the A-DCS_N^1 on n bidders and the competitive ratio is equal to $1/S_1$.

Auction 4 Algorithm A-SCS_N^2

Input: Bid vector $\mathbf{b}$ such that $|\mathbf{b}| = n > 2N$.

Optimize $p_1, ..., p_N, q$ by solving the LP given below.
Execute DCS_1, ..., DCS_N and SCS with probability $p_1, ..., p_N$ and q, respectively.

Probabilities $p_1, ..., p_N$, q are obtained as a solution of the following LP:

$$\begin{aligned} \text{maximize}: \ & S_2 \\ \text{subject to}: \ & -\sum_{j=1}^{N} f_{j,k} \cdot p_j - g_k \cdot q + S_2 \leq 0 \\ & \text{for } k = 2, ..., 20N, \\ & \sum_{j=1}^{N} p_j + q = 1, \\ & \text{and } p_1, ..., p_N, q, S_2 \geq 0, \end{aligned}$$

where $f_{j,k}$ is the same as before and $g_k = \frac{1}{2} - \binom{k-1}{\lfloor \frac{k}{2} \rfloor} 2^{-k}$

is the inverse competitive ratio of the SCS for $\mathcal{B}_k$ (see Lemma 3.1).

3.4 Competitive Analysis of A-SCS$_N$ For a competitive analysis of A-SCS_N, we first obtain an expected profit of DCS_j:

LEMMA 3.2. *For a given bid vector* $\mathbf{b}$, *suppose that* $\mathcal{F}^{(2)}(\mathbf{b})$ *has* k *winners. Then we have*

$$\frac{E[DCS_j(\mathbf{b})]}{\mathcal{F}^{(2)}(\mathbf{b})} \geq \frac{1}{k}\sum_{i=1}^{k} \min(i, k-i)\binom{k}{i}\binom{2j-k}{j-i}\Big/\binom{2j}{j} \tag{3.1}$$

if $k < 2j$, *and*

$$\frac{E[DCS_j(\mathbf{b})]}{\mathcal{F}^{(2)}(\mathbf{b})} \geq j/k, \tag{3.2}$$

otherwise.

Proof. Suppose that $\mathcal{F}^{(2)}(\mathbf{b})$ has k winners with price p, i.e., $\mathcal{F}^{(2)}(\mathbf{b}) = kp$. If $k \geq 2j$ then the values of all the highest $2j$ bids are at least p. Therefore, both B_1 ($= \mathcal{F}(\mathbf{b}_1)$) and B_2 ($= \mathcal{F}(\mathbf{b}_2)$) are at least jp, meaning the profit of DCS_j is at least jp. Thus (3.2) holds. If $k < 2j$, let $\mathbf{b}_1$ include k_1 bids out of the k bids (of the k winners). Then we have

$$\Pr[k_1 = i] = \binom{k}{i}\binom{2j-k}{j-i}\Big/\binom{2j}{j}. \tag{3.3}$$

Note that $E[DCS_j(\mathbf{b})] \geq \sum_{i=1}^{k-1} Pr[k_1 = i]\cdot\min(B_1, B_2)$ where B_1 and B_2 are at least ip and $(k-i)p$ respectively. Thus (3.1) follows from (3.3). □

This bound (3.1) is tight and it is not hard to see that such a worst-case profit happens for bid vector $\mathbf{b} = (1, ..., 1, 0, ..., 0)$ consisting of k 1's and $2j - k$ 0's. This fact makes the calculation of the minimum value of $E[DCS_j(\mathbf{b})]/\mathcal{F}^{(2)}(\mathbf{b})$ much easier for each j and k such that $k < 2j$ and this value is nothing other than $f_{j,k}$ used in the LP. If $k \geq 2j$, $f_{j,k}$ is a simple linear function in k. See Fig. 2 for some specific values of $f_{j,k}$ for small j and k.

LEMMA 3.3. *For any* $n = |\mathbf{b}|$, *the competitive ratio of A-SCS_N is at most* $\max(1/S_1, 1/S_2, 1/(q \cdot g_{20N}))$.

Proof. Recall A-SCS_N is the combination of A-SCS_N^1 and A-SCS_N^2. If $n \leq 2N$, then we run A-SCS_N^1 whose competitive ratio is at most $1/S_1$, as explained before. If $n > 2N$, then for all such n we have the same values for probabilities p_1, ..., p_N and q, or we run exactly the same algorithm. This means the competitive ratio is not decreasing as n grows (a bid vector $\mathbf{b}$ of length n is equivalent to the bid vector $\mathbf{b} \cup \{0\}$ for SCS). Therefore it is enough if we can bound the competitive ratio of A-SCS_N^2 for an arbitrarily large $n \geq 20N$. Suppose $\mathbf{b}$ is such a bid vector and consider the two cases: If $\mathbf{b} \in \mathcal{B}_k$ for $k \leq 20N$, then the competitive ratio of A-SCS_N^2 is at most $1/S_2$ by LP (note that Lemma 3.2 holds for any $n \geq k$). If $k > 20N$, then we count only the profit of SCS that is run with probability q, which still has a profit of $q \cdot g_k$, meaning the competitive ratio is at most $1/q \cdot g_{20N}$. □

THEOREM 3.2. *The competitive ratio of A-SCS_{50} is at most 3.531.*

Proof. We know all specific values for $f_{j,k}$ and g_k, which allows us to solve the two LP's. As a result, we have $S_1 = 0.28598$, $S_2 = 0.28321$, $q = 0.58459$ and $g_{1000} = 0.487387$. The theorem now follows by a simple calculation. □

Table 1 shows those values for $N = 2, 3, 4, 5, 10, 30, 50$. As one can see the competitive ratio gradually improves as N grows. Observe that for practical purposes, i.e, if we know a specific value, say a, of n, it is the best to use A-SCS_a^1, by which we can achieve a competitive ratio of $1/S_1$. When $n = 10$ for instance, it is about 3.37 due to the table. However, for theoretical purposes, we do need A-SCS_N^2 to obtain an upper bound of the competitive ratio.

N	$1/S_1$	$1/S_2$	q	g_{20N}	CR
2	3.0	3.53251	0.60299	0.437315	3.79224
3	3.0	3.53257	0.60304	0.448711	3.69562
4	3.125	3.53257	0.60304	0.455360	3.64166
5	3.20833	3.53257	0.60304	0.460205	3.60332
10	3.36550	3.53257	0.60304	0.471826	3.53257
30	3.47431	3.53135	0.59059	0.483720	3.53135
50	3.49670	3.53099	0.58459	0.487387	3.53099

Table 1: Result of the LPs for specific N.

4 Averaging Hartline and McGrew's Algorithm

4.1 Aggregated Υ3 Algorithm In order to explain our new algorithm, we first describe the original Υ3 algorithm which obtains an optimal outcome from any bid vector of size three and is used as a core of the main algorithm. Also we need to define an auction property which is necessary to introduce the Υ3 algorithm.

DEFINITION 2. (SCALE INVARIANCE) *A bid-independent auction is said to be scale-invariant if, for any* $c > 0$, $z > 0$ *and bidder* i, *it satisfies*

$Pr[t_i(\mathbf{b}_{-i}) \geq z] = Pr[t_i(c\mathbf{b}_{-i}) \geq cz]$. *(* $t_i(\mathbf{b}_{-i})$ *means a price to* i *as a function of the other bids* $\mathbf{b}_{-i}$ *).*

In scale-invariant auctions, we can assume that t_i is determined only from relative magnitudes of the other $n-1$ bids in $\mathbf{b}_{-i}$, i.e., the algorithm receives a normalized bid vector $\mathbf{b}'_{-i} = (b'_1, b'_2, ..., b'_{i-1}, b'_{i+1}, ...b'_n)$ where one of the bids is $b'_j = 1$ and all the other bids are at least 1. The following $\Upsilon 3$ is scale-invariant (and symmetric) and thus is defined by a function which receives $\mathbf{b}_{-i} = (1, x)$ $(x \geq 1)$:

Auction 5 $\Upsilon 3$ algorithm

Input: bid vector $\mathbf{b}$ of size 3

1: For each bidder i, assume that $\mathbf{b}_{-i} = (1, x)$ where $1 \leq x$.

2: $$t_i \leftarrow \begin{cases} \text{If } x \leq 3/2 \\ \begin{cases} 1 & \text{with probability } 9/13 \\ z & \text{with prob. density } g(z) \text{ for } z > 3/2 \end{cases} \\ \text{If } x > 3/2 \\ \begin{cases} 1 & \text{with probability } 9/13 - \int_{3/2}^{x} z g(z) dz \\ x & \text{with probability } \int_{3/2}^{x} (z+1) g(z) dz \\ z & \text{with prob. density } g(z) \text{ for } z > x \end{cases} \end{cases}$$

where $g(z) = \frac{2/13}{(z-1)^3}$. (Note that the probability is of discrete type for the prices of 1 and x, and of continuous type for the other.)

$\Upsilon 3$ is obviously bid-independent by its definition, and has 13/6 competitive ratio which is optimal for three bidders (proof is in [7]). Hartline and McGrew then introduced the *aggregation method*, a technique to convert an auction on m bidders into "aggregated" auction which accepts bid vectors of any size.

Auction 6 Aggregated auction Agg$\mathcal{A}_s$

Require: A bid-independent algorithm $\mathcal{A}_s$ (defined for s bidders)

1: Put each bidders $b_1, ..., b_n$ into one of s bins $\mathbf{b}_1, ..., \mathbf{b}_s$ uniformly at random.

2: Compute $B_j = \mathcal{F}(\mathbf{b}_j)$ for each bin.

3: Input $\mathbf{B} = (B_1, ..., B_s)$ to $\mathcal{A}_s$ and let its output be $(T_1, ..., T_s)$.

4: Run CostShare$_{T_j}$ on each $\mathbf{b}_j$.

It turns out that Agg$\mathcal{A}_s$ is bid-independent since both the aggregation method and $\mathcal{A}_s$ are bid-independent. If we use the Vickrey auction as $\mathcal{A}_2$, for example, the aggregated auction is identical to SCS. It is not so hard to show that Agg$\Upsilon 3$ has a competitive ratio of 13/4 by the following lemma.

Lemma 4.1. [7] *The expected profit of Agg*$\Upsilon 3$ *satisfies*

$$\frac{E[Agg\Upsilon 3(\mathbf{b})]}{\mathcal{F}^{(2)}(\mathbf{b})} \geq \sum_{i=0}^{k} \sum_{j=0}^{k-i} \frac{\mathcal{F}^{(2)}(i, j, k-i-j)\binom{k}{i,j,k-i-j}}{\beta k 3^k},$$

where k *is the number of winners in* $\mathcal{F}^{(2)}(\mathbf{b})$ *and* $\beta = 13/6$ *which is the competitive ratio of* $\Upsilon 3$ *for three bidders.*

4.2 Averaged Version of Agg$\Upsilon 3$

The bound of Lemma 4.1 is tight; equality holds for bid vector $\mathbf{b} = (1, ..., 1, 0, ..., 0)$ including k 1's and $n-k$ 0's. The right-hand side becomes minimum for $k = 2$ and its value then is 4/13, while the value increases and approaches to 6/13 as k grows. Thus Agg$\Upsilon 3$ has a worst-case competitive ratio of 13/4 and not surprisingly, the main reason for this bad performance is that the two highest bids can go to the same bin with relatively high (1/3) probability. This can be avoided by sending those important bids to different bins. Thus we can use exactly the same method as before to average Agg$\Upsilon 3$.

We first define, as an analogy of DCS_m, the following algorithm:

Auction 7 Algorithm $\mathcal{E}_m$

1: Pick up the highest $3m$ bids (if there are less than $3m$ bidders, add an appropriate number of '0' bids).

2: Divide them into three bins $\mathbf{b}_1$, $\mathbf{b}_2$ and $\mathbf{b}_3$ randomly such that each bin receives exactly m bids.

3: Calculate $B_1 = \mathcal{F}(\mathbf{b}_1)$, $B_2 = \mathcal{F}(\mathbf{b}_2)$ and $B_3 = \mathcal{F}(\mathbf{b}_3)$.

4: Simulate $\Upsilon 3$ on $\mathbf{B} = (B_1, B_2, B_3)$ and let the output be (T_1, T_2, T_3).

5: Run CostShare$_{T_1}$ on $\mathbf{b}_1$,CostShare$_{T_2}$ on $\mathbf{b}_2$ and CostShare$_{T_3}$ on $\mathbf{b}_3$.

Theorem 4.1. $\mathcal{E}_m$ *is bid-independent.*

Proof. The proof is similar to Theorem 3.1 and may be omitted. □

We then define A-Agg$\Upsilon 3$, again similarly as A-SCS_N, as a combination of A-Agg$\Upsilon 3^1_N$ for $n \leq 3N$ and A-Agg$\Upsilon 3^2_N$ for $n > 3N$.

Auction 8 Algorithm A-Agg$\Upsilon 3^1_N$

Input: Bid vector $\mathbf{b}$ such that $|\mathbf{b}| = n \leq 3N$

1: Optimize $p_1, ..., p_{n/3}$ by solving the LP.

2: Execute $\mathcal{E}_1$, ..., $\mathcal{E}_{n/3}$ with a probability p_1, ...,$p_{n/3}$ respectively.

Probabilities p_1, ..., $p_{n/3}$ are obtained as a solution of the LP similar to the previous one for A-SCS^1_N, where $f_{j,k}$, given in Lemma 4.2 later, denotes the inverse competitive ratio of $\mathcal{E}_j$ for any bid vector $\mathbf{b} \in \mathcal{B}_k$. See Fig. 3 for the specific values of $f_{j,k}$ for small j and k.

Auction 9 Algorithm A-Agg$\Upsilon 3^2_N$

Input: Bid vector $\mathbf{b}$ such that $|\mathbf{b}| = n > 3N$

1: Optimize $p_1, ..., p_N$ and q by solving the LP.
2: Execute $\mathcal{E}_1$, ..., $\mathcal{E}_N$ and SCS with a probability p_1, ...,p_N and q respectively.

Probabilities p_1, ..., p_N and q are obtained as a solution of the LP again as in A-SCS^1_N but it has constraints for $k = 2, ..., 21N$ this time. Also $f_{j,k}$ is the same as A-Agg$\Upsilon 3^1_N$ and g_k is the inverse competitive ratio of the AggΥ3 for $\mathcal{B}_k$ or the right-hand side of the formula in Lemma 4.1.

4.3 Competitive Analysis of A-AggΥ3 Algorithm

LEMMA 4.2. *For a given bid vector* $\mathbf{b}$*, suppose that* $\mathcal{F}^{(2)}(\mathbf{b})$ *has* k *winners. Then we have*

(4.4)

$$\frac{E[\mathcal{E}_j(\mathbf{b})]}{\mathcal{F}^{(2)}(\mathbf{b})} \geq \sum_{i_1=0}^{j}\sum_{i_2=0}^{j} \frac{\mathcal{F}^{(2)}(i_1, i_2, k-i_1-i_2)}{\beta k} \times \frac{\binom{k}{i_1,i_2,k-i_1-i_2}\binom{3j-k}{j-i_1,j-i_2,j-(k-i_1-i_2)}}{\binom{3j}{j,j,j}}$$

if $k < 3j$*. Otherwise, i.e., if* $k \geq 3j$*,*

$$\frac{E[\mathcal{E}_j(\mathbf{b})]}{\mathcal{F}^{(2)}(\mathbf{b})} \geq \frac{3j}{\beta k},$$

where $\beta = 13/6$ *that is the competitive ratio of* Υ3*.*

Proof. Suppose that $\mathcal{F}^{(2)}(\mathbf{b})$ has k winners with price p, i.e., $\mathcal{F}^{(2)}(\mathbf{b}) = kp$. If $k \geq 3j$, all the highest $3j$ bids are at least p. So B_1, B_2 and B_3 are at least jp, and therefore $\mathcal{F}^{(2)}(B_1, B_2, B_3) \geq 3jp$. Since Υ3 is β-competitive against $\mathcal{F}^{(2)}$, we have

$$\frac{E[\mathcal{E}_j(\mathbf{b})]}{\mathcal{F}^{(2)}(\mathbf{b})} \geq \frac{\mathcal{F}^{(2)}(B_1, B_2, B_3)}{\beta} \cdot \frac{1}{kp} \geq \frac{3j}{\beta k}.$$

If $k < 3j$, consider the case that $\mathbf{b}_1$, $\mathbf{b}_2$ and $\mathbf{b}_3$ include i_1, i_2 and $(k - i_1 - i_2)$ bids of the highest k bids, respectively. The probability that this occurs is

$$\frac{\binom{k}{i_1,i_2,k-i_1-i_2}\binom{3j-k}{j-i_1,j-i_2,j-(k-i_1-i_2)}}{\binom{3j}{j,j,j}}.$$

And in that case, the auction's profit is at least

$$\frac{\mathcal{F}^{(2)}(ip, jp, (k-i_1-i_2)p)}{\beta}.$$

Then we have the expected total profit by summing up the above profits multiplied by the corresponding probabilities, which implies (4.4). □

The above bound is tight for a vector $\mathbf{b} = (1, ..., 1, 0, ..., 0)$ (k 1's and the remaining $3j - k$ 0's), and $f_{j,k}$ used in the LP is equal to the right-hand side of (4).

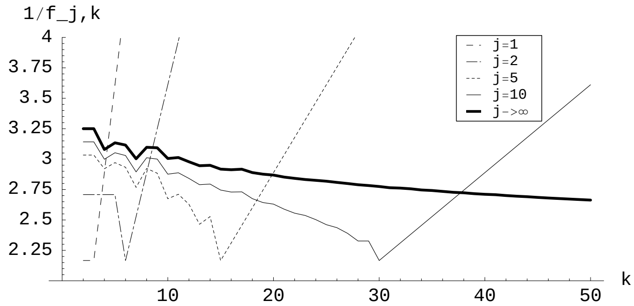

Fig. 3 Competitive ratio of $\mathcal{E}_j$ as a function of k

N	$1/S_1$	$1/S_2$	q	g_{21N}	CR
2	2.70833	3.14713	0.87115	0.370279	3.14713
3	2.88889	3.13369	0.83913	0.382539	3.13369
4	2.88592	3.13022	0.82518	0.391076	3.13022
5	2.91019	3.12815	0.82331	0.397505	3.12815
10	3.00827	3.12286	0.78659	0.415262	3.12286
30	3.08062	3.11945	0.74008	0.434689	3.11945
60	3.09956	3.11899	0.72511	0.442531	3.11899

Table 2 Result of the LPs for specific N

THEOREM 4.2. *The competitive ratio of A-Agg$\Upsilon 3_{60}$ is at most 3.119.*

Proof. Similarly as Lemma 3.3, we can show that the competitive ratio of A-Agg$\Upsilon 3_N$ is at most $\max(1/S_1, 1/S_2, 1/(q \cdot g_{21N}))$. We know all specific values of $f_{j,k}$ and g_k, and thus it is possible to solve the two LP's. As a result, for $N = 60$, we have $S_1 = 0.32262$, $S_2 = 0.32062$, $q = 0.72511$, $g_{1260} = 0.442531$. The theorem now follows by a simple calculation. □

Table 2 shows those values for $N = 2, 3, 4, 5, 10, 30, 60$. One can see that the competitive ratio gradually improves as n grows.

5 Concluding Remarks

See Table 2 again. The competitive ratio of our algorithm cannot be better than $1/S_1$ or $1/S_2$. Note that $1/S_1$ is increasing and $1/S_2$ decreasing (we can prove these properties). Thus the true ratio of the algorithm which will be achieved by a larger N is between 3.09956 and 3.11899.

References

[1] A. V. Goldberg and J. D. Hartline. Competitiveness via consensus. In *The 14th annual ACM-SIAM Symposium on Discrete Algorithms*, pages 215–222, 2003.

[2] A. V. Goldberg and J. D. Hartline. Collusion-resistant mechanisms for single-parameter agents. In *The 16th annual ACM-SIAM Symposium on Discrete Algorithms*, pages 620–629, 2005.

[3] A. V. Goldberg, J. D. Hartline, A. R. Karlin, and M. Saks. A lower bound on the competitive ratio of truthful auctions. In *The 21st International Symposium on Theoretical Aspects of Computer Science*, pages 644–655, 2004.

[4] A. V. Goldberg, J. D. Hartline, A. R. Karlin, and A. Wright. Competitive generalized auctions. In *The 34th annual ACM Symposium on Theory of Computing*, 2002.

[5] A. V. Goldberg, J. D. Hartline, and A. Wright. Competitive auctions and digital goods. In *The 12th annual ACM-SIAM Symposium on Discrete Algorithms*, 2001.

[6] M. Guo and V. Conitzer. Worst-case optimal redistribution of vcg payments. In *The 8th ACM Conference on Electronic Commerce*, pages 30–39, 2007.

[7] J. D. Hartline and R. McGrew. From optimal limited to unlimited supply auctions. In *The 6th ACM Conference on Electronic Commerce*, pages 175–182, 2005.

[8] A. R. Karlin, M. S. Manasse, L. A. McGeoch, and S. S. Owicki. Competitive randomized algorithms for nonuniform problems. *Algorithmica*, 11:542–571, 1994.

[9] H. Moulin and S. Shenker. Strategyproof sharing of submodular costs:budget balance versus efficiency. *Economic Theory*, 18(3):511–533, 2001.

[10] L. S. Shapley. Cores of convex games. *International Journal of Game Theory*, 1(1):11–26, 1971.

[11] W. Vickrey. Counterspeculation, auctions and competitive sealed tenders. *Journal of Finance*, 16(1):8–37, 1961.