Truthful Mechanism Design via Correlated Tree Rounding

YOSSI AZAR, Dept. of Computer Science, Tel-Aviv University
MARTIN HOEFER, Max-Planck-Institut für Informatik and Saarland University
IDAN MAOR, Dept. of Computer Science, Tel-Aviv University
REBECCA REIFFENHÄUSER, Dept. of Computer Science, RWTH Aachen University
BERTHOLD VÖCKING, Dept. of Computer Science, RWTH Aachen University

One of the most powerful algorithmic techniques for truthful mechanism design are maximal-in-distributional-range (MIDR) mechanisms. Unfortunately, many algorithms using this paradigm rely on heavy algorithmic machinery and require the ellipsoid method or (approximate) solution of convex programs. In this paper, we present a simple and natural correlated rounding technique for designing mechanisms that are truthful in expectation. Our technique is elementary and can be implemented quickly. The main property we rely on is that the domain offers fractional optimum solutions with a tree structure.

In auctions based on the generalized assignment problem, each bidder has a publicly known knapsack constraint that captures the subsets of items that are of value to him. He has a private valuation for each item and strives to maximize the value of assigned items minus payment. For this domain we design a mechanism for social welfare maximization. Our technique gives a truthful 2-approximate MIDR mechanism without using the ellipsoid method or convex programming. In contrast to some previous work, our mechanism achieves exact truthfulness.

In restricted-related scheduling with selfish machines, each job comes with a public weight, and it must be assigned to a machine from a public job-specific subset. Each machine has a private speed and strives to maximize payments minus workload of jobs assigned to it. For this domain we design a mechanism for makespan minimization. Although this is a single-parameter domain, the approximation status of the underlying optimization problem is similar to unrelated scheduling: The best known algorithm gives a (non-truthful) 2-approximation for unrelated machines, and there is 1.5-hardness. Our mechanism matches this bound and provides a truthful 2-approximation.

Categories and Subject Descriptors: F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems

General Terms: Algorithms, Economics, Theory

Additional Key Words and Phrases: Mechanism Design, Rounding, Combinatorial Auctions, Scheduling Mechanisms

1. INTRODUCTION

A major issue in algorithmic mechanism design is understanding the impact of truthfulness constraints on the computational complexity and approximation factors of optimization problems. This question has received considerable interest in recent years, most prominently in combinatorial auctions: The goal here is to assign a set of m items

This work is supported by DFG Cluster of Excellence MMCI, by ISF and I-CORE(4/11).
Author's addresses: Y. Azar, Tel-Aviv University, Tel-Aviv, Israel; azar@tau.ac.il. M. Hoefer, Max-Planck-Institut für Informatik, Saarbrücken, Germany; mhoefer@mpi-inf.mpg.de. I. Maor Tel-Aviv University, Tel-Aviv, Israel; R. Reiffenhäuser, RWTH Aachen University, Aachen, Germany; rebeccar@cs.rwth-aachen.de. B. Vöcking, RWTH Aachen University, Aachen, Germany.

EC'15, June 15–19, 2015, Portland, OR, USA. ACM 978-1-4503-3410-5/15/06 ...$15.00.

Copyright is held by the owner/author(s). Publication rights licensed to ACM.

http://dx.doi.org/10.1145/2764468.2764503

to a set of n bidders, where each bidder has a non-negative, monotone valuation function that assigns a value to each subset of items. The objective is to find an allocation that (approximately) maximizes social welfare given by the sum of bidder valuations.

Randomization is often helpful in designing polynomial-time truthful mechanisms with good approximation factors. A prominent and successful way to design randomized mechanisms is to formulate the allocation problem as an integer program, solve a fractional relaxation optimally in polynomial time, and then round it to a feasible allocation. In a celebrated result, Lavi and Swamy [2011] showed how to obtain truthful mechanisms with small approximation ratios for social welfare based on a technique called randomized meta-rounding. Subsequently, this approach was identified as an instance of a general technique termed maximum-in-distributional-range (MIDR) by Dobzinski and Dughmi [2013]. Here, we define a fixed and bounded range of distributions over allocations that allows to efficiently obtain and round a distribution maximizing a function of the bids. Using VCG arguments, we obtain a truthful-in-expectation mechanism that incentivizes truth-telling among risk-neutral bidders optimizing expected utility. For classes of submodular coverage valuations, convex rounding techniques can be used to design MIDR mechanisms with constant approximation factors [Dughmi et al. 2011]. These approaches, however, rely heavily on convex optimization techniques and the ellipsoid method, which represents a serious bottleneck for the intuitive understanding and the running time. While there exist some faster mechanisms for these domains, they only provide approximate truthfulness. In addition, recent work has shown several strong lower bounds for approximation factors of truthful mechanisms with different oracle access to bidder valuations. Finding simple truthful mechanisms with small, near-optimal approximation guarantees has become a serious challenge.

In this paper, we design a novel and elementary rounding technique for truthful-in-expectation mechanisms. Our technique is applicable to two-sided allocation problems, and we outline the application for multi-item auctions based on the generalized assignment problem (GAP). It yields truthful-in-expectation mechanisms with approximation ratio 2. We avoid any dependence on the ellipsoid method and achieve exact truthfulness. Our approach relies on the existence of a fractional optimum solution with tree structure, which we round using a simple and elementary construction. In addition, we indicate how to generalize the technique and apply it in different contexts to covering-style problems with tree structure. As an example, we outline an application to scheduling mechanisms with restricted-related machines, where we derive a truthful-in-expectation mechanism with approximation ratio 2. This restricted-related scenario is a single parameter domain but close in nature to the prominent unrelated machines model. There is a huge gap between a linear upper bound to a constant lower bound for the approximation ratio of truthful mechanisms for unrelated machines. Interestingly, our result shows that for the closely related problem of restricted-related machines one can achieve a truthful mechanism with a constant approximation ratio.

1.1. Related Work

Combinatorial Auctions and GAP. Combinatorial auctions are a central domain in (algorithmic) mechanism design, and the existing literature is too broad to review it here in detail. We only provide a context of our results, for a more general overview see, e.g., [Cramton et al. 2006; Milgrom 2004; Blumrosen and Nisan 2007].

Combinatorial auctions with general valuation functions are hard to approximate, there exist several lower bounds close to $m^{1/2-\epsilon}$ for constant $\epsilon > 0$, e.g., in terms of computational complexity [Lehmann et al. 2002], or for value oracle access [Mirrokni et al. 2008], where we are only allowed to make a polynomial number of value queries to the bidders. This motivated the study of relevant subclasses of valuations (for an overview see, e.g., [Blumrosen and Nisan 2007]). Lavi and Swamy [2011] developed a general framework for obtaining truthful-in-expectation mechanisms via randomized meta-rounding. Their framework yields mechanisms with approximation factors of $O(\sqrt{m})$ for combinatorial auctions, $(1+\epsilon)$ for multi-unit combinatorial auctions with multiplicity $\Omega(\log m)$, and 2 for multi-parameter knapsack problems.

A prominent class of valuations are submodular valuations, where social welfare maximization without truthfulness is essentially solved. Optimal $(1-1/\mathrm{e})$-approximation algorithms exist even for value oracle access [Vondrák 2008]. This factor cannot be improved assuming either polynomial communication in the value oracle model [Mirrokni et al. 2008] or polynomial-time complexity in general [Khot et al. 2008]. For the strategic setting and general submodular functions, there is a truthful-in-expectation mechanism with approximation factor $O(\log m/\log\log m)$ in the communication complexity model [Dobzinski et al. 2010] and a universally truthful mechanism with factor $O(\log m)$ using demand oracles [Krysta and Vöcking 2012]. For a subclass called matroid-rank-sum valuations, Dughmi et al. [2011] proposed a convex rounding technique to build MIDR mechanisms, and their approach yields an optimal $(1-1/\mathrm{e})$-approximation.

More recently, strong lower bounds for approximation factors of truthful mechanisms with submodular valuations have been established based on the structure of the mechanism and the query model to access valuations. Universally truthful mechanisms using randomization over maximal-in-range mechanisms (the deterministic analog to MIDR) that provide constant approximation factors have been ruled out for all oracle models [Dobzinski and Nisan 2011]. In the value oracle model, the lower bound on the approximation factor was strengthened to $m^{1/2-\epsilon}$ [Dobzinski 2011]. Even for truthful-in-expectation mechanisms, lower bounds of m^{ϵ} exist in the value oracle model [Dughmi and Vondrák 2011]. Moreover, there is a class of submodular valuations that allows a polynomial representation, but no truthful-in-expectation mechanism with approximation factor of m^{ϵ}, for some constant $\epsilon > 0$, unless $\mathsf{NP} \subseteq \mathsf{P/poly}$ [Dobzinski and Vondrák 2012a,b].

We here consider combinatorial auctions based on the generalized assignment problem (GAP). For GAP there are a variety of (non-truthful) approximation results. There exists a $(1-1/\mathrm{e})$-approximation algorithm [Fleischer et al. 2011], which improves on an earlier 2-approximation [Shmoys and Tardos 1993]. The best-known hardness is $11/10$ [Chakrabarty and Goel 2010], and the current best approximation ratio is $(1-1/e+\rho)$, $\rho \leq 10^{-5}$ [Feige and Vondrák 2006]. Fadaei and Bichler [2014] recently gave a $(1-\epsilon)$-truthful $(1-1/e)$-approximation for auctions based on GAP using a convex rounding approach. This mechanism, however, is only approximately truthful. For a very different variant of the problem where valuations are known and only the feasible bidder-item-pairs are private, an earlier logarithmic mechanism [Dughmi and Ghosh 2010] was recently improved to a universally truthful constant-factor-approximation [Chen et al. 2014].

Scheduling Mechanisms. For scheduling on unrelated machines, finding a truthful mechanism with sublinear approximation ratio for makespan optimization is a long-standing open problem since the seminal work of Nisan and Ronen [2001]. The main technical difficulty in designing scheduling mechanisms is that makespan is not utilitarian social welfare: The social objective is not to optimize the (expected) sum of times when each machine completes all assigned jobs, but the maximum of these times. Therefore standard approaches like VCG are not in line with the social objective. Another major complication is the multi-parameter domain: The private information of every machine are processing times for each single job. For the non-strategic variant, there exists a 2-approximation algorithm [Lenstra et al. 1990]. However, truthful mechanisms have only been obtained with ratios of $O(m)$ [Lu and Yu 2008a,b; Lu 2009], while only a lower bound of 2.61 has yet been proven [Christodoulou et al. 2009; Koutsoupias and Vidali 2013]. The upper bound of $O(m)$ is tight for the special case of anonymous mechanisms [Ashlagi et al. 2012]. For the fractional problem, a $(1+(n-1)/2)$-approximative mechanism exists [Christodoulou et al. 2010].

The paradigmatic problem in the single-parameter domain is scheduling on related machines, where each machine only holds its speed as private information. The classic approximation result is a PTAS for the non-strategic problem [Hochbaum and Shmoys 1988]. For truthful mechanisms, there exist a number of results that leverage the simpler monotonicity conditions required for single-parameter domains [Archer and Tardos 2001; Andelman et al. 2007; Kovács 2005]. The best mechanisms are PTAS'es that are truthful-in-expectation [Dhangwatnotai et al. 2008] and deterministically truthful [Christodoulou and Kovács 2013]. In addition, there exists a deterministic PTAS that is applicable to a variety of other objective functions [Epstein et al. 2013].

The problem we consider in our work lies in between unrelated and related machine scheduling: In restricted-related scheduling, although having only one single speed, machines can be unfit to do certain jobs at all – each task has only a subset of allowed machines it can be assigned to. Here – as for unrelated machines – the best-known hardness is 1.5, and the best-known approximation for makespan without truthfulness is 2 [Lenstra et al. 1990], which we match with our truthful-in-expectation mechanism.

1.2. Contribution and Overview

We present a general technique for truthful mechanism design in allocation problems and apply it in two domains. Our technique allows to derive truthful-in-expectation mechanisms when optimal solutions to a relaxation have a tree structure. This class covers a variety of resource allocation problems with covering and packing objectives, from which we consider generalized assignment problems and restricted-related scheduling. Our approach starts with a fractional optimum, which we then round in an iterative and correlated fashion.

From the packing domain, we apply our approach to combinatorial auctions based on GAP and derive a truthful-in-expectation mechanism via fractional VCG that approximates social welfare by a factor of 2. Note that a truthful 2-approximation for these domains can, in principle, also be obtained using randomized meta-rounding and the ellipsoid method. This contrast nicely outlines the strengths of our procedure – it offers a combinatorial understanding of the rounding step, avoids explicit composition of candidate solutions, and can thereby be implemented very quickly. GAP valuations allow a compact representation of m numbers, so there is no need to resort to oracles.

In our approach, item valuations are private, but the constraints that describe feasible item sets that can be given to each bidder are public knowledge. The latter property is crucial for truthfulness of our mechanisms, and we provide an example that our approach is not truthful if constraints are also private. Among other things, this implies that our mechanisms cannot be directly applied to combinatorial auctions with budget-additive valuations [Buchfuhrer et al. 2010; Chakrabarty and Goel 2010], which can be modeled via GAP with private constraints.

From the covering domain, we consider truthful makespan optimization on restricted-related machines. In this case, we rely on monotonicity arguments for the solution to a relaxation to guarantee truthfulness. The rounding technique is then applied to interpret the fractional relaxation as distribution and to sample one feasbile integral solution. Our mechanism achieves an approximation guarantee of 2 and thereby matches the best-known approximation guarantee of any algorithm solving the problem (with or without truthfulness).

2. NOTATION AND PRELIMINARIES

2.1. Combinatorial Auctions based on GAP

We consider auctions based on the well-known generalized assignment problem. In this setting there is a set I of n bidders and a set J of m items. The goal is to determine an allocation x with $x_{ij} \in \{0,1\}$ of items to bidders. Each item may only be assigned to at most one bidder. Each bidder has a valuation function $v_i(x)$ that returns the value for the set of items assigned to him. We consider valuations v_i that are additive subject to a public constraint: There are private item values $r_{ij} \geq 0$ for all $j \in J$. Also, there are public item weights $b_{ij} \geq 0$ for all $j \in J$ and a public capacity $B_i \geq 0$. The valuation for bidder i is a submodular function that captures the value of the best feasible subset of items allocated to i

$$v_i(x) = \max_{x' \leq x} \left\{ \sum_{j \in J} x'_{ij} r_{ij} \;\middle|\; \sum_{j \in J} x'_{ij} b_{ij} \leq B_i \right\}$$

Our aim is to design truthful mechanisms where each bidder reports his private information as a bid $r'_i = (r'_{ij})_{j \in J}$, and based on r' the mechanism computes a (possibly randomized) assignment of items to bidders $x(r')$ along with payments $p_i(r')$ for every bidder $i \in I$. Bidder i has a quasi-linear utility $u_i(r') = v_i(x(r')) - p_i(r')$, and the mechanism is *truthful in expectation* if

$$\mathbf{E}\left[u_i((r_{ij})_{j \in J}, r'_{-i})\right] \geq \mathbf{E}\left[u_i(r'_i, r'_{-i})\right]$$

for all bidders $i \in I$, possible bids r'_i and possible bids of other bidders r'_{-i}. The expectation is over internal randomization of the mechanism. We strive to design truthful mechanisms such that the allocation x maximizes social welfare $\sum_{i \in I} v_i(x)$. We will assume throughout that $b_{ij} \leq B_i$ for all $i \in I$ and $j \in J$.

2.2. Restricted-Related Scheduling

In restricted-related scheduling, the set of bidders is a set of machines I with speeds $v_i \in \mathbb{N}$ for all $i \in \{1, \ldots, n\}$. There is a set J of m jobs, and each job has a weight $w_j \in \mathbb{N}$ and a set of machines $M^{(j)} \subseteq I$ it may be assigned to. Our objective is to find an assignment $x = \{x_{ij}\}$ of jobs to machines that assigns each job j to a single machine $i \in M^{(j)}$. In our setting, we will assume that speeds v_i are private information of the machine owners, while everything else is public and therefore known to the mechanism designer. In a truthful mechanism, each machine reports the speed as a bid v'_i, and based on v' the mechanism computes a (possibly randomized) assignment of jobs to machines $x(v')$ along with payments $p_i(v')$ to every machine $i \in I$. Machine i has a quasi-linear utility $u_i(v') = p_i(v') - \sum_{j \in J} w_j x_{ij}(v')/v_i$, and the mechanism is *truthful in expectation* if revealing the private information truthfully is a dominant strategy in expectation, i.e.:

$$\mathbf{E}\left[p_i(v_i, v'_{-i}) - \sum_{j \in J} w_j x_{ij}(v_i, v'_{-i})/v_i\right] \geq \mathbf{E}\left[p_i(v'_i, v'_{-i}) - \sum_{j \in J} w_j x_{ij}(v'_i, v'_{-i})/v_i\right]$$

for all machines $i \in I$ and possible bids v'_i (where v'_{-i} denotes the bids of all machines except i). We strive to design truthful mechanisms that minimize the makespan $\max_{i \in I} \sum_{j \in J} w_j x_{ij}/v_i$, i.e., the earliest time after which every machine has completed all jobs assigned to it.

3. TREE ROUNDING

We assume that the underlying optimization problem is based on linear optimization with variables $x_{ij} \in \{0,1\}$ and models a two-sided budgeted allocation scenario with item set J and bidder set I. We consider this fundamental assignment problem in two variants, a packing and a covering variant. In the packing variant, we assign each item $j \in J$ to *at most one* bidder $i \in I$ in order to maximize a monotone function of x. In the covering variant, we assign each item $j \in J$ to *exactly one* bidder $i \in I$ in order to minimize a monotone function of x. The sets of items we can assign simultaneously to $i \in I$ can be described by a linear constraint of the form $\sum_{j \in J} w_{ij} x_{ij} \leq W_i$, where the weights w_{ij} and bound W_i are given by the problem. Furthermore, we assume that for a linear relaxation with $x_{ij} \in [0,1]$ there is a fractional optimal solution with a (near) tree structure.

Definition 1. *A vector $x = (x_{ij})_{i \in I, j \in J}$ with $x_{ij} \in [0,1]$ has* tree structure *iff the graph $G = (V,E)$ with $V = \{I \cup J\}$ and $E = \{\{i,j\} \mid i \in I,\, j \in J,\, x_{ij} > 0\}$ is a forest.*

We will use the tree structure in the fractional optima to apply a concise correlated rounding scheme, which allows to turn a scaled fractional optimum into a distribution over integral solutions. We then apply suitable payments such that the resulting mechanism becomes truthful in expectation. In addition, in our examples the rounding does not deteriorate the mechanism's objective significantly. We outline this approach in two application domains, one packing and one covering problem.

3.1. Packing Rounding in GAP-based Auctions

In this section, we present a correlated rounding scheme for packing problems that yields a 2-approximate truthful mechanism for combinatorial auctions based on GAP.

Theorem 1. *There is a randomized polynomial-time tree-rounding mechanism for combinatorial auctions based on GAP that is truthful in expectation and obtains an approximation ratio of 2 for social welfare.*

Consider the well-known linear relaxation for GAP:

$$\begin{aligned}
\max \sum_{i \in I, j \in J} & r_{ij} x_{ij} & & \\
\text{s.t.} \sum_{j \in J} b_{ij} x_{ij} & \leq B_i & & \forall i \in I \\
\sum_{i \in I} x_{ij} & \leq 1 & & \forall j \in J \\
x_{ij} & \geq 0 & & \forall i \in I, j \in J
\end{aligned} \tag{1}$$

This describes exactly a fractional relaxation of social welfare maximization with GAP valuations. We assume free disposal, thereby we can say any bidder is only given an amount of fractional items that does not exceed his capacity B_i. In what follows, we describe a framework to obtain a truthful-in-expectation 2-approximate mechanism.

We first solve (1) optimally. The graph G, whose edges are defined by non-zero variables in the optimum x^*, can be assumed to be a pseudo-forest: A collection of trees together with at most one additional edge. In general, there is at most one single cycle in G. We can assume for simplicity that G is a pseudo-tree. For proofs of these well-known properties and further polyhedral insights into GAP see, e.g., [Lenstra et al. 1990; Andelman 2006]. In the following, let us first assume G is a tree. We show below how to deal with pseudo-trees.

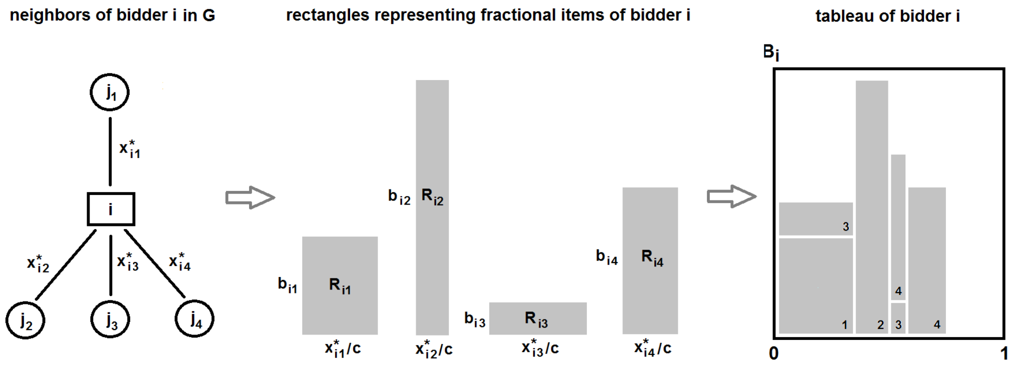

Fig. 1. Schematic view of a tableau for bidder i

3.1.1. Tableaus. Given a tree-structured optimal solution together with a suitable, constant scaling factor c, we can represent and round the fractional optimum x^* in a very convenient way. Intuitively, we represent each variable x^*_{ij} in the fractional optimal solution by a rectangle taking up a width of $x^*_{ij} \leq 1$ and a height of b_{ij}. We scale the rectangles down with a factor of c by width. Then, we pack all the rectangles for those items $j \in J$ with $x^*_{ij} > 0$ into a single rectangle R_i of width 1 and height B_i for bidder i. To do so, we allow a rectangle to be divided into several smaller ones by width, but not by height. Also, two rectangles belonging to the same $j \in J$ may never be placed above the same point on the x-axis. If an axis-parallel packing with these properties can be formed without overlap, we have a very nice property: For any random point $p \in [0, 1]$ along the x-axis of R_i, it is feasible to round those variables with rectangles above p to 1 and all others to 0. (see Figure 1 for a schematic outline).

Let us describe this approach more formally. We define a *tableau of bidder* $i \in I$ according to a scaling factor c as follows. For each $i \in I$, let R_i be a rectangle of width $w(R_i) = 1$ and height $h(R_i) = B_i$. We represent each variable x_{ij} by a rectangle R_{ij} of width $w(R_{ij}) = x^*_{ij}/c$ and height $h(R_{ij}) = b_{ij}$. R_{ij} is split into an arbitrary set of rectangles $(R^k_{ij})_{k=1,2,\ldots}$, such that for all k the width $w(R^k_{ij}) \geq 0$, height $h(R^k_{ij}) = b_{ij}$, and $\sum_k w(R^k_{ij}) = x^*_{ij}/c$.

Definition 2. *An axis-parallel packing of* $(R^k_{ij})_{k=1,2,\ldots}$ *for all* $j \in J$ *into* R_i *is called a* (c-)tableau *for* $i \in I$ *if and only if the following two properties hold:*

(*1*) *All rectangles* R^k_{ij} *are mutually non-intersecting*
(*2*) *For each* $j \in J$ *and each point* $p \in [0, 1]$ *on the x-axis of* R_i*, there is at most one* k *such that rectangle* R^k_{ij} *intersects the vertical line through* p*.*

For ease of formulation, let us define the following concepts.

Definition 3. *We say that at a point* $p \in [0, 1]$ *of a tableau,* $j \in J$ *is* active *iff some* R^k_{ij} *is situated above* p*. For* $p \in [0, 1]$ *let* $J(p)$ *denote the* active job set.

Obviously, $j \in J$ is active at all points along the x-axis above which any rectangle R^k_{ij} is placed. For convenience, we assume the R^k_{ij} don't include their right border, i.e., j is always active in a union of half-open intervals of the form $[l, r)$.

Definition 4. *The* height $h_i(p)$ *at a point* $p \in [0, 1)$ *of a tableau is the sum of heights of jobs that are active at* p*, i.e.,* $h_i(p) = \sum_{j \in J(p)} b_{ij}$.

Suppose we are given c-tableaus for each $i \in I$ based on an optimal fractional solution x^*. The rounding is done as outlined above: We choose some *evaluation point* $p \in [0, 1)$ on the x-axis of i's tableau uniformly at random. Then, we set $x'_{ij} = 1$ for those rectangles that are active at p, and zero for all others. Therefore, for every $x^*_{ij} > 0$ it follows that $\mathbf{Pr}\left[x'_{ij} = 1\right] = x^*_{ij}/c$. Hence, $\mathbf{E}\left[\sum_{j \in J} r_{ij} x'_{ij}\right] = \sum_{j \in J} r_{ij} x^*_{ij}/c$, i.e., each bidder obtains an expected welfare of half the one from x^*. In addition, no bidder ever exceeds his constraint capacity B_i because of the height of tableau R_i. Now, we have to determine a suitable choice for c, and properly correlate the rounding of the different tableaus.

Lemma 1. *For an optimal fractional solution* x^* *and* $c = 2$*, there is a 2-tableau for every* $i \in I$ *that can be computed efficiently. For* $c < 2$*, no* c*-tableau might exist.*

PROOF. Let us first observe that $c < 2$ will not be sufficient. Let $c = (2 - 3\epsilon)$ for some $\epsilon > 0$, and fix some $i \in I$. Let exactly two of the variables $\{x^*_{ij} \mid j \in J\}$ be non-zero, namely, $x^*_{i1} = (1 - \epsilon)$ and $x^*_{i2} = 1$. Assume further that $b_{i1} = (B_i - \epsilon')$, and $b_{i2} = (\epsilon' + \epsilon^*)$. Now, choose $\epsilon', \epsilon^* > 0$ such that $\epsilon = \frac{\epsilon^*}{B_i - \epsilon'}$. It follows that this choice for x^* is feasible, because

$$\sum_{j \in J} b_{ij} x^*_{ij} = (1 - \epsilon) b_{i1} + b_{i2} = \left(1 - \frac{\epsilon^*}{B_i - \epsilon}\right)(B_i - \epsilon) + \epsilon + \epsilon^* = B_i \ .$$

Consider now solution x^c, which we get by scaling down x^*. Here i is still assigned more than $1/2$ of items 1 and 2 each. Therefore, we can never place a fraction of R_{i1} on top of one R_{i2} (this would exceed B_i) nor can we place them all next to each other (this would exceed width 1).

It remains to show that for $c = 2$, all tableaus can be efficiently computed. We give a very simple procedure for constructing i's tableau: Pick item $j = 1$. As long as not all of R_{ij} is packed do the following. Find the leftmost interval on the x-axis of the tableau with current height at most $B_i - b_{ij}$ where j is not active. Split off a maximum-possible fraction of R_{ij} that can be packed directly on top of the rectangles in this interval. Place the part. After R_{ij} is fully packed, apply the procedure to next item j until all items are packed (see Figure 1).

So assume now that this strategy fails, and consider the iteration at which it does not find space for any (fractional) rectangle, and therefore does not compute a feasible tableau. At this iteration, we can choose some $p^* \in [0, 1)$ with minimum height $h_i(p^*)$. Assume for contradiction that the height at p^* is strictly positive. Let $J(p^*) = \{j_1, \ldots, j_l\}$ and consider some arbitrary $y \in [0, 1)$, $p^* \neq y$, with $h_i(y) \leq B_i - h_i(x^*)$. Assume there is item $j_s \in J(p^*) \setminus J(y)$ not active at y. Then, because of minimality of $h_i(p^*)$, there is some $j_r \in J(y) \setminus J(p^*)$. But now, j_s and j_r could have been placed by the algorithm at $\min(x^*, y)$. Thus, $J(p^*) \subseteq J(y)$. Because of $x^*_{ij}/2 \leq 1/2$, we have that for at least half the width of the tableau, the height is above $B_i - h_i(p^*)$, which makes the area covered by rectangles strictly larger than $B_i/2$. This contradicts the feasibility of x^*. Therefore, during the algorithm, there always exists some $x \in [0, 1)$ with $h_i(x) = 0$, and we will successfully assign all fractional rectangles. $\square$

3.2. Tree Structure and Correlated Rounding

Note that rounding the tableaus cannot be done independently for each bidder $i \in I$, since this infeasibly might assign an item $j \in J$ to more than one bidder. We therefore

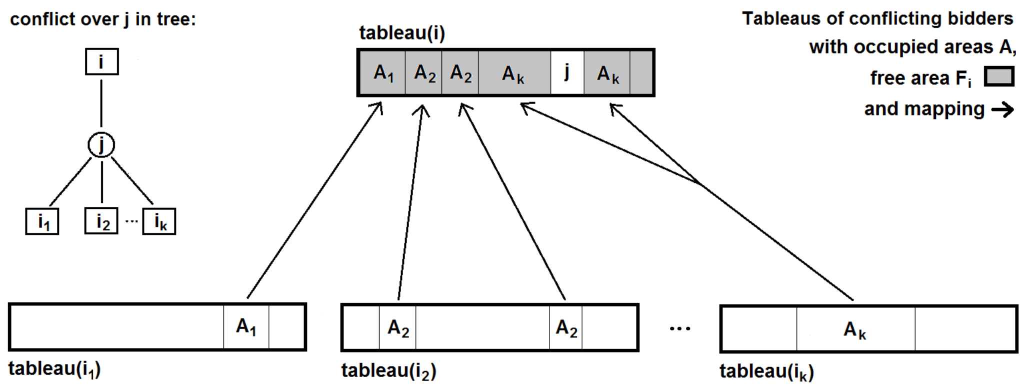

Fig. 2. Schematic view of bijective mappings for correlated rounding

employ the tree structure of x^* to correlate the potential rounding points on the x-axes of rectangles R_i for such bidders that might get the same item j.

Here, instead of randomly choosing independent evaluation points for every single tableau, we do so only for one bidder which we define as root of our tree G. Then, the evaluation points of all other tableaus are obtained by applying a bijective function to the already chosen evaluation point of the according bidder's grandparent in G, subsequently. While at the same time preserving $\mathbf{Pr}\left[x'_{ij} = 1\right] = x^*_{ij}/2$ for all $i \in I$, $j \in J$, our choice of the mapping functions will ensure feasibility of the resulting integral solution. To define these bijective mappings formally, we need the following definition:

Definition 5. *Every $p \in [0,1)$ where $J(p)$ changes in a bidder's tableau is a* breakpoint.

Observe that by our tree graph G, in our optimal solution every pair of bidders can share at most one item. We restrict our view to a conflict situation over an item j, which always looks as follows: Top-down from the root, we have an evaluation point chosen for j's father in G, and now need to choose those of j's children in a fashion that avoids assigning j more than once. We do this in the natural way:

Consider item j and denote by i the parent of j, and by $i_1, i_2, \ldots$ the children of j. An interval between consecutive breakpoints in $[0, 1)$ of i's tableau is *free* if j is not active in this interval. Denote the set of *free intervals* for i by

$$F_i = \{[a_1, b_1), \ldots, [a_r, b_r)\} \ ,$$

with $b_k \leq a_{k+1}$ for all $k = 1, \ldots, r-1$. If an interval is not free, we call it occupied. Clearly, we want to map such points p in the tableau R_{i_k} of some i_k where j is active only to points in F_i – otherwise, j could be rounded to both i and i_k. Also, we have to make sure that any point in F_i is mapped to a set of points in the tableaus of $i_1, i_2, \ldots$ such that j is active at *at most one* of these points.

We start with bidder i_1, and then compute the other bijections accordingly. We derive a suitable mapping between i_1's and i's tableau as follows: Consider the leftmost occupied interval in i_1's tableau. Choose the first interval $[a_1, b_1) \in F_i$. Going from left to right, we map the first occupied interval in the tableau of i_1 to this, until in one of the tableaus a breakpoint is reached. At this point, delete the used-up (partial) interval from F_1. Then, repeat for all other occupied intervals left in the tableau of i_1 (see Figure 2 for a schematic outline).

More formally, for the tableau of bidder i_1 consider the set of occupied intervals as

$$A_1 = \{[a_1^{i_1}, b_1^{i_1}), \ldots, [a_s^{i_1}, b_s^{i_1})\} \ ,$$

where we assume $b_k^{i_1} \leq a_{k+1}^{i_1}$ for $k = 1, \ldots, s-1$. We compute f^{i_1} that maps the x-axis of the tableau of i_1 to the one of the tableau of i as follows. As long as $A_1 \neq \emptyset$ do the following: For the leftmost interval $[x_1^{i_1}, y_1^{i_1})$ in A_1, map it to the leftmost interval in F_i by setting

$$f^{i_1}(x_1^{i_1}) = a_1 \text{ and } f^{i_1}(x_1^{i_1} + \alpha) = a_1 + \alpha$$

for all $\alpha \in (0, \min\{b_1 - a_1, y_1^{i_1} - x_1^{i_1}\})$. Then remove (sub-)intervals $[x_1^{i_1}, x_1^{i_1} + \alpha)$ from A_1 and $[a_1, a_1 + \alpha)$ from F_i, update the numbering in A_1 and F_i, and repeat. Finally, when A_1 becomes empty, choose $f^{i_1}(x)$ as some arbitrary, bijective mapping for all other $x \in [0, 1)$ in which it is not yet defined (this can be done similarly as before). This yields the bijective mapping for machine i_1. The inverse function is defined accordingly.

For bidder i_2, we subsequently apply the same idea, where we initially remove from F_i all subintervals to which an occupied interval was already mapped by f^{i_1}. The algorithm then proceeds similarly for the rest of the children, until the conflict over item j is completely resolved. For all other items in the tree, we apply the same procedure.

Clearly, the above construction ensures that no item will be assigned more than once in the according integral solution, as all occupied intervals from the i_k are mapped only without intersections to free intervals for i. Moreover, the algorithm terminates correctly, as the overall width of all intervals in F_i is always at least the sum of widths of all intervals in $A_1, A_2, \ldots$, since $\sum_{i \in I} x_{ij}^* \leq 1$. Observe that due to the choice of the bijections (no change in width of intervals, uniform choice of evaluation point in the root tableau), we maintain $\mathbf{Pr}\left[x'_{ij} = 1\right] = x_{ij}^*/2$ for all $i \in I$, $j \in J$.

Lemma 2. *If the optimal solution to* (1) *has tree structure, the bijective mappings for the tableaus can be computed in polynomial time.*

PROOF. Due to the simple structure of the bijections, evaluating them for each job will be of no greater complexity than their computation. We must therefore only consider the construction of one single bijection. Consider our greedy algorithm to construct tableaus. We observe that each single tableau has no more than $O(m)$ breakpoints: During the tableau's greedy construction, each R_{ij} can be split in a lot of fractions, but all of those splittings are already caused by an existing breakpoint. Only the last fraction will end at a point that possibly has not already been a breakpoint, and therefore create a new one. Thus, all tableaus contain at most $O(nm)$ breakpoints. Every time we need to compute a new, continuous fraction of one bijection, this is because a breakpoint is reached. Therefore, each bijection consists of at most $O(nm)$ continuous, simple parts and is obviously computable in polynomial time. □

This proves the following lemma.

Lemma 3. *The tree rounding procedure can be implemented in polynomial time and yields a distribution over feasible integral solutions x' such that for every $i \in I$ we have $\mathbf{E}\left[\sum_{j \in J} r_{ij} x'_{ij}\right] = \sum_{j \in J} r_{ij} x_{ij}^*/2$, and $\sum_{j \in J} b_{ij} x'_{ij} \leq B_i$.*

Truthfulness is now straightforward with fractional VCG payments, since we optimize over a range of feasible fractional solutions that is public knowledge and fixed independently of the bids. Hence, the mechanism is MIDR and truthful in expectation.

3.2.1. Correlated Rounding for Pseudo-Trees. The remaining difficulty is the existence of at most one (non-resolvable) cycle in x^*. The tableaus can be computed in exactly the same fashion as above. For correlated rounding via bijective mappings we must be more careful to preserve feasibility. The following lemma shows that the scaled fractional optimum can be rounded quickly. This proves our main theorem.

Lemma 4. *The bijective mappings for the tableaus in GAP can be computed in polynomial time.*

PROOF. If G contains a cycle $C = (i_1, j_1, \dots, i_s, j_s)$, we will pick a bidder node i_1 in the cycle as root for the computation of our allocation points. We first focus only on the nodes in C before applying our rounding anywhere else. If we map regions for tableaus of bidders iteratively along the cycle, the only problem occurs when we close the cycle, because the last bidder node is a grandchild of i_1. We have to ensure that their shared item j_s cannot be picked in i_s's tableau if it already is present in the tableau of i_1.

Intuitively, we need to account for possible allocation of item j_s already while we move along C: We can just, in each step from a bidder node i_k to i_{k+1}, record a set R_{k+1} of the points in $[0, 1)$ that were reached from an allocation point in i_1's tableau where j_s was active. (R_1 then is just the set of intervals where j_s is active in i_1's tableau.)

We will show that this can be implemented in polynomial time. R_1, as mentioned above, will consist of no more than $O(m)$ intervals because each tableau has only this many breakpoints. Let us analyze a step from R_k to R_{k+1} and consider some interval $I \in R_k$. If $f_{i_{k+1}}(I)$ is also an interval in $R_k + 1$, the number of intervals does not increase. If not so, I must have been split up by $f_{i_{k+1}}$. This can only happen at the borders between continuous parts of $f_{i_{k+1}}$, and each such point will only split up at most one interval in R_k into at most one additional part. The continuous parts of $f_{i_{k+1}}$, however, do all correspond to a breakpoint in one of the tableaus neighboring the corresponding shared item. Furthermore, because of the structure of G, each tableau's breakpoints will be taken into account no more than two times during the process. Overall, R_{s-1} cannot consist of more than $O(mn)$ intervals because each breakpoint in any considered tableau can cause only a constant number of additional intervals in R_{s-1}, and there exist at most $O(mn)$ such points.

Based on R_{s-1} we must define the last bijection f_{i_s} differently from the other ones: In addition to making sure no conflicts can occur regarding item j_{s-1}, it is also necessary to map all the intervals in R_{s-1} to such ones in i_s's tableau where item j_s is inactive.

Define $w_{j_s, j_{s-1}}$ as the overall width that item j_{s-1} and R_{s-1} take up in the tableau of bidder i_{s-1}. Then, because we are considering a scaled-down solution, it holds that in at most a width of $1 - w$ in i_s's tableau, any of the items j_{s-1} or j_s is active. Let L denote the set of these intervals. We consider the *non-critical points*, i.e., $x \notin R_{s-1}$ where j_{s-1} is not active in i_{s-1}'s tableau. Our goal is to map only non-critical points to L. We can do this similarly as in the other bijections, maintaining an overall number of at most $O(nm)$ continuous parts due to the above analysis of R_{s-1}.

In this fashion, we first compute tableaus and bijections for bidders in the cycle. Subsequently, we proceed along the tree as described above – with the exception of any bidders which are in conflict with item j_s, but not $\in C$. Here, the bijections to be chosen must take into account both the allocation points where j_s is active in i_1's tableau, and those that were mapped to a point with active j_s in the tableau of bidder i_s. To determine the set of these points, it will be necessary to track the according intervals backwards through all the bijections along the cycle, again causing a similar rise in the number of intervals to that for the set R_s before. Although, differently from R_1, our starting set may already consist of up to $O(mn)$ intervals, we still manage to keep polynomial-time computation and a resulting set of at most $O(mn)$ intervals: Just as in the other direction, moving backwards through the bijections around the cycle will cause no more than $O(nm)$ additional intervals to keep track of.

After that, also these missing bijections will be computable in the usual way, just starting with a slightly different F_s which contains both the critical points from the tableau of i_1, and the tracked-back ones from that of i_s. □

3.3. Private Constraints and Budget-Additive Valuations

In this section, we show an example that our mechanism is not truthful if constraints of GAP are also private knowledge. In our example, r_{ij}, b_{ij} are public and B_i is private. Similar examples can be constructed also if B_i is public and r_{ij}, b_{ij} are private. Note that in the example we assume $r_{ij} = b_{ij}$, which captures the case of budget-additive combinatorial auctions. Hence, our mechanisms do not guarantee truthfulness when applied directly to this domain.

Example 1. Consider an instance with two bidders and two items. The true values are $r_{11} = r_{12} = 4$, $r_{21} = r_{22} = 2$. For the constraints we have $b_{ij} = r_{ij}$ for all $i, j = 1, 2$ and budgets $B_1 = 4$, $B_2 = 2$. If both bidders report truthfully, an optimal solution is $x^*_{11} = x^*_{22} = 1$ and $x^*_{12} = x^*_{21} = 0$, giving item 1 to bidder 1 and item 2 to bidder 2. The mechanism takes $x^*/2$ and rounds it to integral solutions. Thus, bidder 1 has an expected valuation of 2. Fractional VCG payments imply a payment of $p_1 = 0$, since without bidder 1 we would get the same optimal fractional assignment for bidder 2. Thus, the utility for bidder 1 is 2.

Now suppose bidder 1 deviates and lies a budget of $B'_1 = 6$. An optimum with tree structure is $x'_{11} = 1$, $x'_{12} = x'_{22} = 0.5$ and $x'_{21} = 0$. The mechanism takes $x'/2$ and rounds it to integral solutions. However, since the heights of $b_{12} + b_{22} > B'_1$, it never assigns both items to bidder 1 simultaneously. Thus, the resulting integral solution also does not exceed the true budget of bidder 1, and he obtains an expected value of 3. Furthermore, bidder 2 has value $x'_{22}/2 = 0.5$, whereas without bidder 1 he would get value 1. This implies a payment of 0.5 for bidder 1 and a utility of $2.5 > 2$. ■

3.4. Covering Rounding in Restricted-Related Scheduling

In this section, we outline our correlated rounding scheme for covering problems. We adjust with very few changes the mechanism for packing problems to makespan scheduling with restricted-related machines and obtain the following result.

Theorem 2. *There is a randomized polynomial-time tree-rounding mechanism for makespan scheduling with restricted-related machines that is truthful in expectation and obtains an approximation ratio of 2.*

As above, we first solve a fractional relaxation of the problem, for which we obtain an optimal solution with forest structure. Essentially the same tree rounding serves then to round the fractional optimum x^* to an integral solution. Although it yields the same approximation factor, it is quite different from the tree-based rounding in [Lenstra et al. 1990]. Truthfulness requires us to maintain the fractional values as probabilities for assignment, and we cannot rely on an arbitrary matching between jobs and machines in the tree.

Since we have a single-parameter problem, we first show how to obtain truthfulness based on monotonicity.

3.4.1. Lexicographically-Optimal Fractional Solutions. To obtain the fractional relaxation, we allow each job to be assigned fractionally over all machines it can access. The objective here is makespan minimization, so there could be many solutions with optimal makespan that give different loads to machines which do not attain the makespan. This could potentially imply that when a machine lowers its speed, it still gets assigned more load - a contradiction to monotonicity and truthfulness. Instead, we compute a lexicographically-optimal fractional solution in polynomial time with a min-max vector of loads. Such a solution yields monotone load assignments for each machine.

Lemma 5. *A lexicographically optimal fractional solution x^* with forest structure can be computed in polynomial time. The load $\sum_{j \in J} w_j x^*_{ij}$ assigned to machine i is monotone in v_i.*

PROOF. Algorithms for computing lexicographically-optimal fractional solutions with forest structure are known, see, e.g., [Azar et al. 2004; Tamir 1995]. For completeness, we describe an approach for restricted-related machines in Appendix A.

Let us observe the monotonicity property. Fix some machine i and suppose it reports a speed $v'_i > v_i$. Consider the optimal values T and T' for the respective reports of i. As they are some rational numbers of the form $\sum_{j \in S} w_j / \sum_{k \in S'} v_k$, it is obvious that for $v'_i > v_i$ we have $T' \leq T$. We consider two cases.

If $T = T'$, then i can only move up in priority, since speed $v'_i > v_i$ allows to add more load on i until T is reached. Hence, i will only be fixed at an earlier iteration, where more fractional load is still available. This clearly results in at least the same load that i received when reporting v_i.

Otherwise if $T' < T$, this decrease must result from the increased speed of i. Hence, i attracts only more load from other machines, which allows them to potentially decrease their finishing time in the optimum. Note that this implies that the priority of machine i increases - it allows to accomodate at least as much load as for T, while the lower T' decreases the priority of other machines. Hence, it can only receive an increased load in the lexicographically optimal assignment when reporting $v'_i > v_i$. $\square$

3.4.2. Tableaus. Given an optimum solution x^* with tree structure, we now apply our tree rounding procedure. The outcome x' will have assignment probabilities that correspond exactly to $\mathbf{Pr}\left[x'_{ij} = 1\right] = x^*_{ij}$ and assigns all jobs, $\sum_{i \in I} x'_{ij} = 1$ for all $j \in J$. The final step is to show that the expected makespan of x' is at most $2T$.

Again, we represent each variable x^*_{ij} by a rectangle taking up a width of x^*_{ij}/c and a height of w_j – observe that this time, rectangles are scaled down by *height*. We pack all the rectangles for items $j \in J$ into a single rectangle R_i for bidder i with width 1 and height Tv_i. We allow splitting by width and avoid two rectangles belonging to the same job $j \in J$ placed above the same point on the x-axis. If such a packing can be done without any intersections, then at each point $p \in [0, 1]$ along the x-axis of R_i, rounding all variables with rectangles above p to 1 and all others to 0 will be feasible. By picking a point on the x-axis uniformly at random, this produces a random assignment x' with $\mathbf{Pr}\left[x'_{ij} = 1\right] = x^*_{ij}$ and $\sum_j w_j x'_{ij}/v_i \leq 2T$.

More formally, for each $i \in I$, let R_i be a rectangle of width $w(R_i) = 1$ and height $h(R_i) = T$. We represent each variable x_{ij} by a rectangle R_{ij} of width $w(R_{ij}) = x^*_{ij}$ and height $h(R_{ij}) = w_j/2$. R_{ij} is split into an arbitrary set of rectangles $(R^k_{ij})_{k=1,2,\ldots}$, such that for all k the width $w(R^k_{ij}) > 0$, height $h(R^k_{ij}) = w_j/2$, and $\sum_k w(R^k_{ij}) = x^*_{ij}$.

Based on this adjusted definition of R^k_{ij}, a c-tableau is now defined exactly the same as above in Definition 2. For each $p \in [0, 1)$, we again refer to *active* items $j \in J(p)$ as in Definition 3. Also, we define the height of a point in the tableau as in Definition 4.

Lemma 6. *For an optimal fractional solution x^* and $c = 2$, there is a 2-tableau for every $i \in I$ that can be computed efficiently. For $c < 2$, no c-tableau might exist.*

PROOF. Let us first observe that $c < 2$ will not be sufficient. Let $c = 2 - \epsilon$ for some $\epsilon > 0$. Let exactly two of the variables $\{x^*_{ij} \mid j \in J\}$ be non-zero, namely, $x^*_{i1} = \frac{1}{2}$ and $x^*_{i2} = (\frac{1}{2} + \epsilon')$, for some small $\epsilon' > 0$. Let $w_1 = w_2 = \frac{Tv_i}{2}$. Now R_{i1} and R_{i2} do not fit into the tableau next to each other, and so we must split them. Then, at some point $p \in [0,1]$ both must be active, and this point reaches a height of $\frac{Tv_i}{(2-\epsilon)} + \frac{Tv_i}{(2-\epsilon)(1+2\epsilon')} > Tv_i$ for a suitable $\epsilon' \ll \epsilon$.

It remains to show that for $c = 2$, all tableaus can be efficiently computed. We use a similar greedy packing strategy as outlined in Lemma 1: We pack each fraction of an item not into the leftmost place it fits, but into the leftmost one with lowest so-far overall height. If the algorithm succeeds, the running time is polynomial by similar arguments as outlined above.

To show that the algorithm works correctly, assume it fails to pack some fraction R_{ij}^k of R_{ij}, and for simplicity, let this have width $w^k \leq 1$ and height $h^k \leq \frac{Tv_i}{2}$. Then, in a fraction of more than $(1 - w^k)$ of the tableau's x-axis, which is given by a finite number of half-open intervals, every point p has already $h_i(p) > Tv_i - h^k$. Also, for the remaining fraction of less than w^k of the x-axis, again having the above form, every point p already has $h_i(p) > (Tv_i/2) - h^k$, since obviously $|h_i(x) - h_i(x')| < Tv_i/2$ due to our algorithm. Together with R_{ij}, the total area D_i of packed rectangles into i's tableau must be

$$\begin{aligned} D_i &> (1-w^k)(Tv_i - h^k) + w^k((Tv_i/2) - h^k) + w^k h^k & \iff \\ D_i &> Tv_i - h^k - Tv_i w^k + w^k h^k + (Tv_i/2)w^k - w^k h^k + w^k h^k & \iff \\ D_i &> Tv_i(1 - (w^k/2)) + h^k(w^k - 1) \end{aligned} \quad (2)$$

Now, with $D_i \leq Tv_i/2$, this gives us $2h^k(1/2 - w^k/2) > Tv_i(1/2 - w^k/2)$ and therefore $h^k > Tv_i/2$. This is a contradiction, since by construction there is no $j \in J$ with $x_{ij}^* > 0$ and $w_j > Tv_i$. □

Again, it is important to maintain the correct marginal probabilities while avoiding that a job gets assigned to several machines simultaneously. We correlate the tableaus by bijectively mapping the x-axes among tableaus in exactly the same manner as outlined in Section 3.2 for packing rounding. Since we have $\sum_{i \in I} x_{ij}^* = 1$, a feasible mapping exists and can be found by the greedy algorithm. Lemma 2 shows that this procedure can be implemented in polynomial time.

Truthfulness holds via monotonicity of the loads in x^* shown above. This concludes the proof of the main theorem.

4. CONCLUSIONS AND OPEN PROBLEMS

In this paper, we present a general correlated rounding technique that is applicable to resource allocation problems. We outlined the technique in a packing and a covering domain. For combinatorial auctions based on GAP our technique provides a truthful 2-approximation in polynomial time. It works with fractional VCG payments, offers a combinatorial understanding of the rounding step, avoids explicit composition of candidate solutions, and can thereby be implemented very quickly. It would be interesting to see to which extent the techniques here can be generalized to other problems, and in which way the tree assumption can be relaxed. For example, it is an open problem if our results can help to derive truthful mechanisms in budget-additive combinatorial auctions.

For restricted-related scheduling, the objective function is not social welfare. We show how to combine our technique with monotonicity arguments of the fractional optimum solution to obtain a truthful 2-approximation, which matches the best-known approximation guarantee for the problem. This result could potentially have consequences for unrelated scheduling, since using auxiliary machines with speeds as a power of 2, it is simple to apply any algorithm for restricted-related to unrelated scheduling. For m machines, this reduction increases the ratio only by an $O(\log m)$-factor. It is an open problem if this be implemented in a truthful manner to obtain a truthful mechanism with sublinear guarantees for unrelated scheduling.

Acknowledgement

The authors thank Piotr Krysta for insightful discussions and Salman Fadaei for pointing out an error in an earlier version of this work.

REFERENCES

ANDELMAN, N. 2006. Online and strategic aspects of network resource management algorithms. Ph.D. thesis, Tel Aviv University.

ANDELMAN, N., AZAR, Y., AND SORANI, M. 2007. Truthful approximation mechanisms for scheduling selfish related machines. *Theory Comput. Syst. 40,* 4, 423–436.

ARCHER, A. AND TARDOS, É. 2001. Truthful mechanisms for one-parameter agents. In *Proc. 42nd Symp. Foundations of Computer Science (FOCS)*. 482–491.

ASHLAGI, I., DOBZINSKI, S., AND LAVI, R. 2012. Optimal lower bounds for anonymous scheduling mechanisms. *Math. Oper. Res. 37,* 2, 244–258.

AZAR, Y., EPSTEIN, L., RICHTER, Y., AND WOEGINGER, G. J. 2004. All-norm approximation algorithms. *J. Algorithms 52,* 120–133.

BLUMROSEN, L. AND NISAN, N. 2007. Combinatorial auctions. In *Algorithmic Game Theory*, N. Nisan, É. Tardos, T. Roughgarden, and V. Vazirani, Eds. Cambridge University Press, Chapter 11.

BUCHFUHRER, D., DUGHMI, S., FU, H., KLEINBERG, R., MOSSEL, E., PAPADIMITRIOU, C., SCHAPIRA, M., SINGER, Y., AND UMANS, C. 2010. Inapproximability for VCG-based combinatorial auctions. In *Proc. 21st Symp. Discrete Algorithms (SODA)*. 518–536.

CHAKRABARTY, D. AND GOEL, G. 2010. On the approximability of budgeted allocations and improved lower bounds for submodular welfare maximization and gap. *SIAM J. Comput. 39,* 6, 2189–2211.

CHEN, N., GRAVIN, N., AND LU, P. 2014. Truthful generalized assignments via stable matching. *Math. Oper. Res. 39,* 3, 722–736.

CHRISTODOULOU, G., KOUTSOUPIAS, E., AND KOVÁCS, A. 2010. Mechanism design for fractional scheduling on unrelated machines. *ACM Trans. Algorithms 6,* 2.

CHRISTODOULOU, G., KOUTSOUPIAS, E., AND VIDALI, A. 2009. A lower bound for scheduling mechanisms. *Algorithmica 55,* 4, 729–740.

CHRISTODOULOU, G. AND KOVÁCS, A. 2013. A deterministic truthful PTAS for scheduling related machines. *SIAM J. Comput. 42,* 4, 1572–1595.

CRAMTON, P., SHOHAM, Y., AND STEINBERG, R., Eds. 2006. *Combinatorial Auctions*. MIT Press.

DHANGWATNOTAI, P., DOBZINSKI, S., DUGHMI, S., AND ROUGHGARDEN, T. 2008. Truthful approximation schemes for single-parameter agents. In *Proc. 49th Symp. Foundations of Computer Science (FOCS)*. 15–24.

DOBZINSKI, S. 2011. An impossibility result for truthful combinatorial auctions with submodular valuations. In *Proc. 43rd Symp. Theory of Computing (STOC)*. 139–148.

DOBZINSKI, S. AND DUGHMI, S. 2013. On the power of randomization in algorithmic mechanism design. *SIAM J. Comput. 42,* 6, 2287–2304.

DOBZINSKI, S., FU, H., AND KLEINBERG, R. 2010. Truthfulness via proxies. *CoRR abs/1011.3232*.

DOBZINSKI, S. AND NISAN, N. 2011. Limitations of VCG-based mechanisms. *Combinatorica 41,* 4, 379–396.

DOBZINSKI, S. AND VONDRÁK, J. 2012a. The computational complexity of truthfulness in combinatorial auctions. In *Proc. 13th Conf. Electronic Commerce (EC)*. 405–422.

DOBZINSKI, S. AND VONDRÁK, J. 2012b. From query complexity to computational complexity. In *Proc. 44th Symp. Theory of Computing (STOC)*. 1107–1116.

DUGHMI, S. AND GHOSH, A. 2010. Truthful assignment without money. In *Proc. 11th Conf. Economics and Computation (EC)*. 325–334.

DUGHMI, S., ROUGHGARDEN, T., AND YAN, Q. 2011. From convex optimization to randomized mechanims: Toward optimal combinatorial auctions. In *Proc. 43rd Symp. Theory of Computing (STOC)*. 149–158.

DUGHMI, S. AND VONDRÁK, J. 2011. Limitations of randomized mechanisms for combinatorial auctions. In *Proc. 52nd Symp. Foundations of Computer Science (FOCS)*. 502–511.

EPSTEIN, L., LEVIN, A., AND VAN STEE, R. 2013. A unified approach to truthful scheduling on related machines. In *Proc. 24th Symp. Discrete Algorithms (SODA)*. 1243–1252.

FADAEI, S. AND BICHLER, M. 2014. A truthful-in-expectation mechanism for the generalized assignment problem. In *Proc. 10th Intl. Conf. Web and Internet Economics (WINE)*. 247–248.

FEIGE, U. AND VONDRÁK, J. 2006. Approximation algorithms for allocation problems: Improving the factor of 1-1/e. In *Proc. 47th Symp. Foundations of Computer Science (FOCS)*. 667–676.

FLEISCHER, L., GOEMANS, M., MIRROKNI, V., AND SVIRIDENKO, M. 2011. Tight approximation algorithms for maximum separable assignment problems. *Math. Oper. Res. 36,* 3, 416–431.

HOCHBAUM, D. AND SHMOYS, D. 1988. A polynomial approximation scheme for scheduling on uniform processors: Using the dual approximation approach. *SIAM J. Comput. 17,* 3, 539–551.

KHOT, S., LIPTON, R., MARKAKIS, E., AND MEHTA, A. 2008. Inapproximability results for combinatorial auctions with submodular utility functions. *Algorithmica 52,* 1, 3–18.

KOUTSOUPIAS, E. AND VIDALI, A. 2013. A lower bound of $1+\phi$ for truthful scheduling mechanisms. *Algorithmica 66,* 1, 211–223.

KOVÁCS, A. 2005. Fast monotone 3-approximation algorithm for scheduling related machines. In *Proc. 13th European Symp. Algorithms (ESA)*. 616–627.

KRYSTA, P. AND VÖCKING, B. 2012. Online mechanism design (randomized rounding on the fly). In *Proc. 39th Intl. Coll. Automata, Languages and Programming (ICALP)*. 636–647.

LAVI, R. AND SWAMY, C. 2011. Truthful and near-optimal mechanism design via linear programming. *J. ACM 58,* 6, 25.

LEHMANN, D., O'CALLAGHAN, L., AND SHOHAM, Y. 2002. Truth revelation in approximately efficient combinatorial auctions. *J. ACM 49,* 5.

LENSTRA, J., SHMOYS, D., AND TARDOS, É. 1990. Approximation algorithms for scheduling unrelated parallel machines. *Math. Prog. 46,* 3, 259–271.

LU, P. 2009. On 2-player randomized mechanisms for scheduling. In *Proc. 5th Intl. Workshop Internet & Network Economics (WINE)*. 30–41.

LU, P. AND YU, C. 2008a. An improved randomized truthful mechanism for scheduling unrelated machines. In *Proc. 25th Symp. Theoret. Aspects of Computer Science (STACS)*. 527–538.

LU, P. AND YU, C. 2008b. Randomized truthful mechanisms for scheduling unrelated machines. In *Proc. 4th Intl. Workshop Internet & Network Economics (WINE)*. 402–

413.

MILGROM, P. 2004. *Putting Auction Theory to Work*. Cambridge University Press.

MIRROKNI, V., SCHAPIRA, M., AND VONDRÁK, J. 2008. Tight information-theoretic lower bounds for welfare maximization in combinatorial auctions. In *Proc. 9th Conf. Electronic Commerce (EC)*. 70–77.

NISAN, N. AND RONEN, A. 2001. Algorithmic mechanism design. *Games Econom. Behav. 35*, 166–196.

SHMOYS, D. AND TARDOS, É. 1993. An approximation algorithm for the generalized assignment problem. *Math. Prog. 62,* 3, 461–474.

TAMIR, A. 1995. Least majorized elements and generalized polymatroids. *Math. Oper. Res.* 20, 583–590.

VONDRÁK, J. 2008. Optimal approximation for the submodular welfare problem in the value oracle model. In *Proc. 40th Symp. Theory of Computing (STOC)*. 67–74.

A. FINDING A LEXICOGRAPHICALLY-OPTIMAL FRACTIONAL SOLUTION

We first determine an optimal fractional solution x_{ij} with smallest maximum finishing time T, where the finishing time of each machine i is assumed as the maximum of the set $\{\sum_j w_j x_{ij}/v_i\} \cup \{w_j/v_i \mid x_{ij} > 0\}$. Hence, the finishing time results either from the total fractional load assigned to i or the full load of the largest job that is assigned fractionally to i. If x_{ij} is integral, this coincides with the makespan. Hence, when optimizing over fractional x, the optimal T is a lower bound for the optimal makespan. We can find the optimal T by repeated solution of the following linear program LP(T):

$$\begin{aligned}
\max \sum_{i\in I, j\in J} & x_{ij} & & \\
\text{s.t.} \sum_{j\in J} w_j x_{ij} &\le T v_i & & \forall i \in I \\
\sum_{i\in I} x_{ij} &= 1 & & \forall j \in J \\
x_{ij} &= 0 & & \forall i, j \text{ s.t. } i \notin M^{(j)} \text{ or } w_j > T v_i \\
x_{ij} &\ge 0 & & \forall i \in I, j \in J
\end{aligned} \tag{3}$$

The objective function serves only as a feasibility check, as the value for every feasible solution will always be $|J|$. The program ensures that in each feasible solution (1) no machine has a makespan greater than T, (2) each job j is assigned exactly once in total, (3) a job j is never (fractionally) assigned to a machine outside its feasible set $M^{(j)}$, (4) if machine i gets a fraction of job j, then i would need at most time T to run the whole job j, and (5) each assigned fraction of any job is non-negative.

Lemma 7. *We can find the smallest value T for which LP(T) allows a feasible solution in polynomial time.*

PROOF. For a fixed T we can solve the LP in polynomial time. Using binary search, we can repeatedly solve the LP to find the smallest value T such that the resulting LP has a feasible solution. The value of T is obviously lower bounded by $\sum_{j\in J} w_j / \sum_{i\in I} v_i$ and upper bounded by $\min_{i\in I} \sum_{j\in J} w_j/v_i$. In addition, it is attained by an expression $\sum_{j\in S} w_j / \sum_{i\in S'} v_i$ for some subsets $S \subseteq J, S' \subseteq I$ with $S, S' \neq \emptyset$. Thus, the optimal T can be expressed as rational number with a denominator of $\prod_{i\in I} v_i$, and binary search will return the optimal bound on T in at most $O(n \log(\max_{j\in J} w_j) + m \log(\max_{i\in I} v_i))$ rounds. □

For known T, we now construct a fractional allocation x_{ij} that is feasible for T and is lexicographically optimal among all fractional feasible allocations for T. Thereby it guarantees the monotonicity property: Increasing speed v_i for any machine i will never result in a decreased load $\sum_{j \in J} w_j x_{ij}$. In the beginning, each machine i gets as priority the total load that can be assigned to i without exceeding T. Among those with highest priority we pick the first one according to an arbitrary fixed tie-breaking rule. More formally, for each single machine $i^* \in I$, solve the following $LP(1, i^*)$.

$$
\begin{aligned}
\max \sum_{j \in J} & w_j x_{i^* j} \\
\text{s.t.} \quad \sum_{j \in J} w_j x_{ij} &\le Tv_i \quad \forall i \in I \\
\sum_{i \in I} x_{ij} &= 1 \quad \forall j \in J \\
x_{ij} &= 0 \quad \forall i, j \text{ s.t. } i \notin M^{(j)} \text{ or } w_j > Tv_i \\
x_{ij} &\ge 0 \quad \forall i \in I, j \in J
\end{aligned}
\tag{4}
$$

This is an adjustment of $LP(T)$ with an objective to maximize load of machine i^*. Among those machines with maximum possible load, apply the tie-breaking rule and denote the resulting machine l_1. For l_1 we denote the maximum load by L_{l_1}. Next, we determine l_2 and L_{l_2} by solving the similar $LP(2, i^*)$, for each remaining machine $i^* \neq M_{l_1}$, with the constraint that we keep a load of L_{l_1} on l_1. More generally, for all $k \in \{2, ..., m\}$, we repeat these steps using $LP(k, i^*)$

$$
\begin{aligned}
\max \sum_{j \in J} & w_j x_{i^* j} \\
\text{s.t.} \quad \sum_{j \in J} w_j x_{ij} &\le Tv_i \quad \forall i \in I \\
\sum_{i \in I} x_{ij} &= 1 \quad \forall j \in J \\
x_{ij} &= 0 \quad \forall i, j \text{ s.t. } i \notin M^{(j)} \text{ or } w_j > Tv_i \\
\sum_{j \in J} w_j x_{l_i, j} &= L_{l_i} \quad \forall i < k \\
x_{ij} &\ge 0 \quad \forall i \in I, j \in J
\end{aligned}
\tag{5}
$$

This LP adjusts the preceding one by pinpointing the load of all fixed machines l_i during the following iterations. In this way, we obtain a lexicographically optimal fractional solution x^*.

Finally, we transform the solution into one with forest structure by applying a shifting idea. We shift weight along the cycles of the graph and make the smallest edge of the cycle disappear. By doing so, the overall load on every machine stays the same and the cycle breaks. Repeated application of this step resolves all cycles in polynomial time.