On the Nisan-Ronen conjecture for submodular valuations

George Christodoulou[*] Elias Koutsoupias[†] Annamária Kovács[‡]

Abstract

We consider incentive compatible mechanisms for a domain that is very close to the domain of scheduling n unrelated machines: the single exception is that the valuation of just one machine is submodular. For the scheduling problem with such cost functions, we give a lower bound of $\Omega(\sqrt{n})$ on the approximation ratio of incentive compatible deterministic mechanisms. This is a strong information-theoretic impossibility result on the approximation ratio of mechanisms that provides strong evidence for the Nisan-Ronen conjecture. This is the first non-constant lower bound that assumes no restriction on the mechanism side; in contrast, all previous general results hold for only special classes of mechanisms such as local, strongly monotone, and anonymous mechanisms. Our approach is based on a novel multi-player characterization of appropriately selected instances that allows us to focus on particular type of algorithms, linear mechanisms, and it is a potential stepping stone towards the full resolution of the conjecture.

1 Introduction

The design of protocols that provide appropriate incentives to participants, entice them to cooperate, and behave in a way which is socially beneficial has a long and celebrated history. It is the realm of mechanism design which is one of the most researched branches of Game Theory and Microeconomics. It studies the design of algorithms, called *mechanisms*, and it has numerous applications to many situations in modern societies, whenever a protocol of conduct of selfish participants is required. The mechanism asks each participant to bid their preferences over the different social outcomes, and implements one of them (e.g. the one which is most socially beneficial). The challenge is that the preferences of the participants are *private*, and they are either unmotivated to report correctly, or strongly motivated to report them erroneously, if a false report is profitable. A truthful mechanism provides incentives in a way that it is in the best interest of each participant to bid *truthfully.*

The algorithmic nature of mechanism design and the associated computational issues, were brought to light in the seminal 20-year-old paper by Nisan and Ronen [33], which essentially established the area of algorithmic mechanism design. They proposed the scheduling problem on unrelated machines, a fundamental, extensively studied from the algorithmic perspective, optimization problem as a representative specimen to study the limitations of truthful mechanisms. The objective is to incentivize n machines to execute m tasks, so that the maximum completion time of the machines, i.e. the makespan, is minimized. Traditional algorithms, such as the best-known approximation algorithm [27], are not incentive compatible.

Nisan and Ronen applied the famous Vickrey-Clarke-Groves (VCG) mechanism [36, 14, 23] which is a general machinery that truthfully computes the outcome that maximizes the *social welfare*, which for the case of scheduling is the allocation that minimizes the sum of completion times. The VCG is truthful and polynomial-time for scheduling, but with respect to the makespan minimization it has a rather poor approximation ratio, equal to the number

[*]Department of Computer Science, University of Liverpool, UK. Email: `gchristo@liverpool.ac.uk`
[†]Department of Computer Science, University of Oxford, UK. Email: `elias@cs.ox.ac.uk`
[‡]Department of Informatics, Goethe University, Frankfurt M., Germany. Email: `panni@cs.uni-frankfurt.de`

of machines n. Despite this, they conjectured that the VCG is the mechanism with the best approximation ratio for this problem.

Conjecture 1. *There is no deterministic truthful mechanism with approximation ratio better than n for the problem of scheduling n unrelated machines.*

The bound in the conjecture is information-theoretic, in the sense that it should hold for all mechanisms (algorithms), polynomial-time or not. The Nisan-Ronen conjecture has developed into one of the central problems in Algorithmic Game Theory, and despite intensive efforts, very sparse progress has been made towards its resolution. The original Nisan-Ronen paper showed that no truthful deterministic mechanism can achieve an approximation ratio better than 2. This was improved to 2.41 [12], and later to 2.61 [24], which leaves a huge gap with the known upper bound of n. The most general and interesting result, by Ashlagi et al.[3], resolved a restricted version of the conjecture, for the special yet natural class of *anonymous* mechanisms. However, the original conjecture (for non-anonymous mechanisms) is widely open.

The original conjecture was posed for the case of additive valuations, where the total cost of each machine equals the sum of its individual costs of each task. However, most mechanism design settings consider more general valuations such as submodular, subadditive etc. In these valuations, the cost of a machine that takes a set of tasks S is a function of S that satisfies some natural properties. For example, for subadditive valuations the cost of a bundle of tasks S can be any value bounded above by the sum of the costs of the individual tasks in S.

It is natural to pose the Nisan-Ronen question for these extended valuation classes. Despite the importance of the problem, the conjecture has been widely open for all such valuations. The VCG mechanism provides a trivial n-approximation for these classes of valuations, while the best known lower bounds are still the aforementioned constants. In this work, we focus on *submodular cost functions*, where the marginal contribution of a task to the total cost of a machine is a non-increasing function. The class of submodular valuations contains all additive valuations, it is a proper subset of subadditive valuations, and it is one of the most restrictive natural class of valuations for the scheduling question. In the corresponding maximization problem of combinatorial auctions, submodular valuations is the most-studied class in algorithmic mechanism design (see for example [26, 20, 17]). Our results hold also for supermodular valuations.

1.1 Our result

We give the first non-constant lower bound that works for *all* deterministic truthful mechanisms when the cost functions are submodular. Our lower bound assumes *no restriction on the mechanism side*, but an expanded class of valuations. The Nisan-Ronen conjecture has been shown to hold for only *special classes of mechanisms* (local [33], strongly monotone [32], and anonymous [3] mechanisms).

The importance of the Nisan-Ronen conjecture is that it captures in a crisp way the difficulties that incentive compatibility adds to algorithm theory. It is a happy coincidence for the development of algorithmic mechanism design that the optimization problem they chose to study, the unrelated machines scheduling problem, turned out to be the most challenging problem in the area. This work validates the motivating hypothesis of Nisan and Ronen that incentive compatibility adds insurmountable difficulties for minmax optimization. In order to establish our result, we develop new techniques, potentially with wider applicability: a partial multiplayer characterization and lower bounds for linear mechanisms (Section 4.3).

Theorem 1. *There is no deterministic truthful mechanism with approximation ratio better than $\sqrt{n-1}$ for the problem of scheduling n unrelated machines with submodular cost functions.*

Actually we show a stronger result: Let $\mathcal{M}$ be the class of scheduling mechanisms which are deterministic and truthful *when all machines are additive except for one machine, which is* *submodular.* Then no mechanism in $\mathcal{M}$ has an approximation ratio better than $\sqrt{n-1}$ on the set of instances of scheduling unrelated additive machines.

Other valuation classes.

Our results hold also for *supermodular* valuations. There is no difficulty in translating, in fact carbon-copying the proof of our characterization theorem for this case. Interestingly, the class of additive valuations of the original Nisan-Ronen conjecture is exactly the intersection of submodular and supermodular valuations.

In fact, we provide a stronger version of Theorem 1 by considering submodular (or supermodular or any superclass of these) functions which are also ϵ-additive, in the sense that the execution time of a set of tasks is within an arbitrarily small ϵ from the sum of the execution times of its tasks (see Section 2 for precise definitions, and the long version of our paper [9] for details of the proofs).

1.2 Overview of the techniques

We provide an overview of our approach for the lower bound. We consider instances in which every task has a fixed large value, practically infinite, for all except for two machines; one of the two machines is always the submodular player and the other is an additive player. We can assume that each task is allocated to one of its two machines with the non 'infinite' values, otherwise the approximation ratio is sufficiently high. Such restrictions of the allocation to only two players per task, have been previously used (e.g. [12, 24]). The main difference with previous approaches is that we use properties of mechanisms that involve at least three players, the submodular player and two other players. Obtaining such multiplayer statements is the bottleneck for a complete characterization of mechanisms in multiplayer domains.

Two-player characterization.

We first focus on the tasks that can be allocated to a particular additive player and fix the values of the remaining tasks. For each additive player, there are only two such tasks, and the situation is very similar to the two-player and two-task special case in which the valuation of one player is submodular. A core part of our proof, which may be of independent interest, is a characterization of the allocation functions of all truthful mechanisms for this case. We provide a complete characterization for the case of two players with non-negative, submodular valuations, which, for one player, are bounded above by a constant. The latter is essential in our construction to guarantee that the fixed large value of the other machines does not play any role. Since this is a minimization problem, it is the lower bound (i.e., the restriction that the values are non-negative) that creates complications rather than the upper bound on the domain. We also provide a characterization for additive valuations, the actual scheduling domain.

We note that similar two-player characterizations have been provided by previous work [19, 11], for auctions and scheduling domains, but none of these can be used in our approach. In particular, the characterization of [19] for scheduling relies extensively on the bounded approximation for two players, but we need a characterization without this assumption. The reason is that the approximation ratio of any two players is in general unrelated to the approximation ratio of the whole multi-player instance. In the same work, a characterization for auctions with subadditive valuations is provided, but this is also of no use to the minimization we consider here. Finally, the characterization of [11] cannot be used because it allows negative values.

Indeed, as we show in Section 3, the scheduling domain admits truthful mechanisms not present in the previous characterizations of [19, 11], which we call *relaxed affine minimizers.* Such mechanisms are essentially affine minimizers for large values, but for small values they can have *non-linear* boundaries (see Definitions 4, 5 and Figure 2). We summarize here the characterization result, see Theorem 3 for the precise statement and the full version [9] for a proof.

Theorem 2 (Informal)**.** *Every truthful allocation for two tasks and two players where one player is submodular, and the other additive and bounded from above, and both tasks are always allocated, is one of these three types: (1) relaxed affine minimizer, (2) one-dimensional mechanism, or (3) constant mechanism.*

Gluing two-player mechanisms to multi-player linear mechanisms.

From the characterization of two-payer two-task mechanisms and using the fact that we are interested in mechanisms with small approximation ratio for the whole multi-player instances, we are able to exclude all mechanisms except of those that have affine boundaries (roughly: critical values/payments linear in the other player's cost for the task, see Definition 8 and Lemma 3), that is, affine minimizers *whose coefficients may depend on the values of the other tasks.*

One of the main technical steps is that we use the truthfulness of the submodular player to show that the scaling coefficient of these linear boundary functions do not actually depend on the values of the other tasks (Lemma 4). The rest of the proof, which includes the most complicated technical steps of this work, is to analyze the properties of truthful linear allocation algorithms that facilitate the proof of the lower bound (Lemma 6).

Organization of the paper.

After the preliminaries (Section 2), the types of mechanisms appearing in the characterization (Theorem 3) are introduced and their truthfulness is shown in Section 3. Based on Theorem 3, we prove the lower bound result in Section 4.

1.3 Related work

The problem of scheduling unrelated machines is a typical multi-dimensional mechanism design problem. In multi-dimensional mechanism design, the valuation of each player for different outcomes is expressed via a vector (one value for every outcome). In the case of unrelated scheduling, this vector expresses the processing times of a machine for each subset of tasks and can be succinctly represented by an m-valued vector, one value for each task.

An interesting special case, which is well-understood, is the single-dimensional mechanism design in which the values of the vector are expressions of a single parameter. The principal representative is the problem of scheduling *related* machines, where the cost of each machine can be expressed via a single parameter, its *speed*. This was first studied by Archer and Tardos [2] who showed that, in contrast to the unrelated machines version, an algorithm that minimizes the makespan can be truthfully implemented — albeit in exponential time. It was subsequently shown that truthfulness has essentially no impact on the computational complexity of the problem. Specifically, a randomized truthful-in-expectation[1] PTAS was given in [16] and a deterministic PTAS was given in [13, 21]; a PTAS is the best possible algorithm even for the pure algorithmic problem (unless $P = NP$).

The main obstacle in resolving the Nisan-Ronen conjecture is the lack of clear algorithmic understanding of truthfulness for many players in multi-dimensional domains. In contrast, we understand better truthfulness for a single player. Saks and Yu [35] gave a nice, complete characterization of deterministic truthful mechanisms for convex domains which was later extended

[1]This is one of the two main definitions of truthfulness for randomized mechanisms, where truth-telling maximizes the expected utility of each player.

to truthful-in-expectation randomized mechanisms in [1]. This characterization states that the class of allocations of truthful mechanisms is the class of *weakly monotone* algorithms [5]. This is an elegant characterization, but it has not been proved very useful for mechanisms of many players, because it is difficult to combine monotonicity of each individual player into a single global condition. Its direct applications have provided only constant lower bounds for makespan minimization [12, 32, 24, 33]. What would be more useful is a characterization similar to the one provided by the seminal work of Roberts [34] for *unrestricted* domains. It essentially states that the only truthful mechanisms are affine extensions of VCG. Similar characterizations have been provided in [11, 19, 18] for settings with only two players. Extending these characterizations to multiple players for scheduling and combinatorial auctions is notoriously hard, mainly due to lack of externalities in these settings: the valuation of a player for an allocation depends only on the subset of tasks it receives and is indifferent on how the remaining tasks are assigned to the other players[2].

Scheduling is related to combinatorial auctions, where multiple items need to be assigned to a set of buyers. This is a broad and successful area, and the setting shares both aforementioned features of multi-dimensionality and lack of externalities, therefore insights and techniques can be transferred from the one problem to the other. However the difference is that the objective for combinatorial auctions is *social welfare maximization*, and this is known to be achieved by the VCG mechanism, albeit in exponential time. Hence the focus on this rich area is on what can be achieved by computationally efficient mechanisms (see for example [20]). But in the case of the scheduling with the min-max objective, the flavor is more information theoretic, as we know that *not even exponential time* truthful mechanisms can achieve the optimal makespan.

1.3.1 Further related work

Lavi and Swamy [25] proposed an interesting approach to attack the Nisan-Ronen question, by restricting the input domain, but still keep the multi-dimensional flavour of the setting. They assumed that each entry in the input matrix can take only two possible values "low" and "high", that are publicly known to the designer. In this case, they showed an elegant deterministic mechanism with an approximation factor of 2. Surprisingly, even for this special case there is a lower bound of 11/10. Yu [38] extended the results for a range of values, and Auletta et al. [4] studied multi-dimensional domains where the private information of the machines is a single bit.

Randomization has been explored and slightly improved the known guarantees. There are two notions of truthfulness for randomized mechanisms. A mechanism is *universally truthful* if it is defined as a probability distribution over deterministic truthful mechanisms, while it is *truthful-in-expectation*, if in expectation no player can benefit by lying. In [33], a universally truthful mechanism was proposed for the case of two machines, and was later extended to the case of n machines by Mu'alem and Schapira [32] with an approximation guarantee of $0.875n$, which was later improved to $0.837n$ by [30]. Lu and Yu [31] showed a truthful-in-expectation mechanism with an approximation guarantee of $(m+5)/2$. Mu'alem and Schapira [32], showed a lower bound of $2 - 1/m$, for both notions of randomization. Christodoulou, Koutsoupias and Kovács [8] extended the lower bound for fractional mechanisms, where each task can be fractionally allocated to multiple machines. They also showed a fractional mechanism with a guarantee of $(m+1)/2$. A sequence of papers studied randomized mechanisms for the special case of two machines [29, 31] where a tight answer on the approximation factor is still unresolved. Currently, the best upper bound is 1.587 due to Chen, Du, and Zuluaga [7].

Truthful implementations of other objectives have been explored by Mu'alem and Schapira [32] for multi-dimensional problems and by Epstein, Levin and van Stee [21] for single-

[2]For two players, there exist implicit externalities as the tasks one player doesn't get determine what the other player gets.

dimensional ones, giving a PTAS for a wide range of objective functions. Leucci, Mamageishvili and Penna [28] showed high lower bounds for other min-max objectives on some combinatorial optimization problems. In the Bayesian setting, Daskalakis and Weinberg [15] showed a mechanism that is at most a factor of 2 from the *optimal truthful mechanism*, but not with respect to optimal makespan. Chawla et al. [6] provided bounds of prior-independent mechanisms (where the input comes from a probability unknown to the mechanism). Giannakopoulos and Kyropoulou [22] showed that the VCG mechanism achieves an approximation ratio of $O(\log n/\log\log n)$ under some distributional and symmetry assumptions.

2 Preliminaries

There is a set N of n machines and a set M of m tasks that need to be scheduled on the machines. The processing time or cost that each machine i takes to process a subset S of tasks is described by a set function $t_i : 2^m \to \mathbb{R}_{\geq 0}$. In classic unrelated machines scheduling, and in the Nisan-Ronen model, the cost functions (valuations) are additive and the objective is to minimize the *makespan* (min-max objective). In our lower bound construction, we will assume that all cost functions are additive except for one machine, for which we consider more general normalized and monotone cost functions:[3] We mainly focus on *submodular* cost functions, which satisfy the following condition for every $S, T \subseteq M$

$$t_i(S \cup T) + t_i(S \cap T) \leq t_i(S) + t_i(T).$$

In case of *supermodular* functions the inequality is reversed. We also consider valuations which are arbitrarily close to additive, which we call ϵ-additive, such that for every subset S, $\sum_{j\in S}(t_i(\{j\})) - \epsilon \leq t_i(S) \leq \sum_{j\in S}(t_i(\{j\})) + \epsilon$.[4]

We will assume that all cost functions are additive, except for one which is submodular, or ϵ-additive, or both submodular and ϵ-additive. The results carry over to supermodular valuations and classes of valuations that include these, like subadditive, superadditive or arbitrary (normalized, monotone) valuations.

Mechanism design setting.

We assume that each machine $i \in N$ is controlled by a selfish agent (player) that is reluctant to process the tasks and the cost function t_i is private information known only to her (also called the *type* of agent i). In the most general version of the problem, the set $\mathcal{T}_i$ of possible types of agent i consists of all vectors $b_i \in \mathbb{R}_+^{2^m}$. Let also $\mathcal{T} = \times_{i\in N}\mathcal{T}_i$ be the space of type profiles.

A mechanism defines for each player i a set $\mathcal{B}_i$ of available strategies, the player can choose from. We will consider *direct revelation* mechanisms, i.e., $\mathcal{B}_i = \mathcal{T}_i$ for all i, meaning that the players strategies are to simply report their types to the mechanism. A player may report a false cost function $b_i \neq t_i$, if this serves her interests. A mechanism (A, P) consists of two parts:

An allocation algorithm: The allocation algorithm A allocates the jobs to the machines depending on the players' bids $b = (b_1, \ldots, b_n)$. Let $\mathcal{A}$ be the set of all possible partitions of m tasks to n machines. The allocation function $A : \mathcal{T} \to \mathcal{A}$ partitions the tasks into the n machines; we denote by $A_i(b)$ the subset of tasks assigned to machine i for bid profile b.

A payment scheme: The payment scheme $P = (P_1, \ldots, P_n)$ determines the payments also depending on the bid values b. The functions $P_1, \ldots, P_n$ stand for the payments that the mechanism hands to each agent i.e. $P_i : \mathcal{T} \to \mathbb{R}$.

[3]A cost function is *normalized* if $t_i(\emptyset) = 0$, and *monotone* if $t_i(S) \leq t_i(T)$ for $S \subseteq T$.

[4]We are free to use any reasonable definition of being close to additive, e.g., we could have chosen $(1-\epsilon)\sum_{j\in S}(t_i(\{j\})) \leq t_i(S) \leq (1+\epsilon)\sum_{j\in S}(t_i(\{j\}))$, or even the intersection of the latter with the current (additive) ϵ-neighborhood.

The *utility* u_i of a player i is the payment that he gets minus the *actual* time that he needs to process the set of tasks assigned to her, $u_i(b) = P_i(b) - t_i(A_i(b))$. We are interested in *truthful* mechanisms. A mechanism is truthful, if for every player, reporting his true type is a *dominant strategy.* Formally,

$$u_i(t_i, b_{-i}) \geq u_i(t'_i, b_{-i}), \qquad \forall i \in [n],\ \ t_i, t'_i \in \mathcal{T}_i,\ \ b_{-i} \in \mathcal{T}_{-i},$$

where $\mathcal{T}_{-i}$ denotes the possible types of all players disregarding i.

We use as an objective to evaluate the performance of a mechanism's allocation algorithm the makespan, that is the maximum completion time over all machines. The makespan of the allocation algorithm A with respect to a given input t is

$$Mech(t) \overset{\text{def}}{=} \max_{i \in N} t_i(A_i(t)).$$

We aim at minimizing the makespan, hence the optimum is

$$Opt(t) = \min_{A \in \mathcal{A}} \max_{i \in N} t_i(A_i).$$

We are interested in the approximation ratio of the mechanism's allocation algorithm. A mechanism M is *c-approximate*, if the allocation algorithm is c-approximate, that is, if $c \geq \frac{Mech(t)}{Opt(t)}$ for all possible inputs t.

We are looking for truthful mechanisms with low approximation ratio irrespective of the running time to compute A and P. In other words, our lower bounds do not make use of any computational assumptions.

Weak monotonicity.

A useful characterization of truthful mechanisms in terms of the following monotonicity condition, helps us to get rid of the payments and focus on the properties of the allocation algorithm.

Definition 1. An allocation algorithm A is called *weakly monotone (WMON)* if it satisfies the following property: for every two inputs $t = (t_i, t_{-i})$ and $t' = (t'_i, t_{-i})$, the associated allocations A and A' satisfy

$$t_i(A_i) - t_i(A'_i) \leq t'_i(A_i) - t'_i(A'_i).$$

It is well known that the allocation function of every truthful mechanism is WMON [5], and also that this is a sufficient condition for truthfulness in convex domains [35] (which is the case for all domains of this work).

A useful tool in our proof relies on the following immediate consequence of WMON, which holds in additive domains as well as in the other domains that we consider. Intuitively, it states that when you fix the values of all players for a subset of tasks (focus on a cut of your domain), then the *restriction* of the allocation to the rest of the tasks must remain weakly monotone.

Lemma 1. *Let A be a WMON allocation. Let us fix an (S, T) partition of M, and consider only valuations t_i of player i that are additive across S and T, i.e., for every $X \subseteq M$, $t_i(X) = t_i(X \cap S) + t_i(X \cap T)$. Then the restriction of the allocation A on S is weakly monotone for each valuation fixed on the subsets of T.*

Proof. A is weakly monotone, therefore

$$t_i(A_i) - t'_i(A_i) \leq t_i(A'_i) - t'_i(A'_i),$$

and additivity across S and T implies

$$t_i(A_i \cap S) - t'_i(A_i \cap S) \leq t_i(A'_i \cap S) - t'_i(A'_i \cap S),$$

as the values of subsets of T are fixed, $t_i(A_i \cap T) = t'_i(A_i \cap T)$ and $t_i({A'}_i \cap T) = t'_i({A'}_i \cap T)$.

□

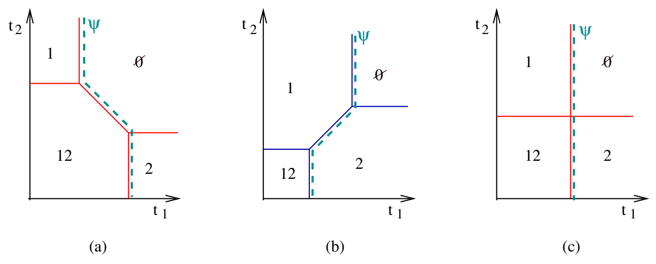

Figure 1: The allocation $A[s]$ to the t-player depending on his own bid vector (t_1, t_2) : (a) quasi-bundling allocation; (b) quasi-flipping allocation; (c) crossing allocation. The interiors of some regions might be empty, but $R_\emptyset \neq \emptyset$ can be assumed w.l.o.g. We marked the functions of critical values $\psi(t_2, s)$ (Definition 7) for receiving task 1 by broken lines.

The following lemma was essentially shown in [33] and has been a useful tool to show lower bounds for truthful mechanisms for several variants (see for example [12, 32, 3]). Although this holds more generally, we only state it (and use it in Section 4) for additive valuations.

Lemma 2. *Let t be a bid vector of additive valuations, and let $S = A_i(t)$ be the subset assigned to player i by a truthful mechanism (A, P). For any bid vector $t' = (t'_i, t_{-i})$ such that only the bid of machine i has changed and in such a way that for every task in S it has decreased (i.e., $t'_i(\{j\}) < t_i(\{j\}), j \in S$) and for every other task it has increased (i.e., $t'_i(\{j\}) > t_i(\{j\}), j \in M \setminus S$). Then the mechanism does not change the allocation to machine i, i.e., $A_i(t') = A_i(t) = S$.*

The main challenge in multi-player settings is that the allocation of the other machines may change and the above condition makes no promise about how this can happen.

3 Characterization for two players

A core element of our lower bound proof is a characterization of truthful mechanisms for two tasks and two players (called t-player, and s-player). The details of the characterization proof can be found in the full version of the paper [9]. There we first provide a characterization for additive valuations $t = (t_1, t_2)$, and $s = (s_1, s_2)$, so that both s_1 and s_2 are bounded by an arbitrarily large but fixed value B. Then we extend it when the t player has submodular valuations $t = (t_1, t_2, t_{12})$ (Theorem 3), which is the main element that we need in the lower bound proof in Section 4. We note that both the characterization and the lower bound result hold analogously when the t-player has arbitrary monotone, ϵ-additive (or submodular *and* ϵ-additive), or supermodular valuations.

In this section we introduce the definitions that we will need in order to state the main result and also introduce some notation which will be used in our lower bound proof in Section 4.

3.1 Basic Definitions

Let (A, P) be a truthful mechanism, where A is the WMON allocation function, and P denotes the payment function. For input (t, s) the allocation is $A(t, s)$. Since we have only tasks 1 and 2, in $A(t, s)$ we can denote the allocation to one of the players as $\alpha_t, \alpha_s \in \{12, 1, 2, \emptyset\}$.

For given $s \in [0, B) \times [0, B)$ the allocation for the t-player as function of his *own* bids (t_1, t_2) (or (t_1, t_2, t_{12})) is denoted by $A[s]$, and symmetrically $A[t]$ is an allocation function for the s-player. For $\alpha_t \in \{12, 1, 2, \emptyset\}$, the allocation *regions* $R_{\alpha_t}(s) \subseteq \mathbb{R}^2_{\geq 0}$ (resp. $\mathbb{R}^3_{\geq 0}$) of $A[s]$ are defined to be the *interior* (wrt. $\mathbb{R}^2_{\geq 0}$) of the set of all t values such that $A(t, s) = \alpha_t$. For the s-player we denote the respective regions by $R^s_{\alpha_s}(t)$.

Additive players.

It is known that in the case of two tasks, the regions in a WMON allocation subdivide $R^2_{\geq 0}$ basically in three possible forms, which are characteristic for the type of the whole allocation-function A (see Figure 1). In fact, it is an *equivalent* condition with WMON that $A[t_{-i}]$ has this form of a geometric representation for every fixed i and t_{-i} [11, 37]. The regions and their boundaries determine the *critical values* for t_1 (as functions of t_2) above which the t-player cannot get task 1, and symmetrically for task 2. These critical value functions are determined by the payment functions $P_\emptyset(s) = 0, P_1(s), P_2(s), P_{12}(s)$ for fixed s. For example, the t-player minimizes her total cost (negative utility) by truth-telling, so for all $t \in R_{12}$ it must hold that (say) $t_1 + t_2 - P_{12} \leq t_2 - P_2$, which yields the vertical boundary position $t_1 = P_{12} - P_2$ between R_{12} and R_2, etc. For every fixed s, the payments can w.l.o.g. be defined in such a way that $(0 =)P_\emptyset \leq P_1 \leq P_{12}$, and $P_\emptyset \leq P_2 \leq P_{12}$. It can be shown that all the payments must be the same in the submodular (ϵ-additive, etc.) domain, as restricted to the additive domain.

Definition 2. For given s, we call the allocation $A[s]$

- *quasi-bundling*, if there are at least two points $t \neq t'$ on the boundary of R_{12} and $R_\emptyset$
- *quasi-flipping*, if there are at least two points $t \neq t'$ on the boundary of R_1 and R_2
- *crossing* otherwise (see Figure 1).

3.2 Mechanisms for submodular players

We now introduce the types of WMON allocations that can occur when the t-player is submodular and the s-player is additive with s_1 and s_2 bounded above by the value B. We denote these particular domains $\mathcal{T}_t \times \mathcal{T}_s$ for two players and two tasks by $V_{submod} \times V_{+,B}$, and refer the reader to the full version for detailed definitions, and also for extension of our results to other domains. We assume that there always exist high enough t values so that the t-player receives no task, i.e., that $R_\emptyset(s) \neq \emptyset$ for every s.[5]

3.2.1 1-dimensional mechanisms.

In a *one-dimensional mechanism* at most two possible allocations are ever realized. Due to the assumption on $R_\emptyset$, one of these must be the allocation $\emptyset$ to the t-player. If the two occuring allocations (for the t-player) are $\emptyset$ and 12, we call the mechanism *bundling mechanism.* The other cases when the allocations to the t-player are $\emptyset$ and 1 (or $\emptyset$ and 2) are degenerate *task-independent* allocations, and can be defined similarly to bundling mechanisms.

Definition 3. In a *bundling mechanism* only the allocations $\emptyset$ and 12 can occur. There is an arbitrary, non-decreasing function $\xi : [0, B) \to [0, \infty)$ so that if $t_{12} > \xi(s_1 + s_2)$ then the t-player gets $\emptyset$, and if $t_{12} < \xi(s_1 + s_2)$ then the t-player gets 12.

If ξ has a jump discontinuity in some point $s_1 + s_2$ then the critical value may depend on the concrete (s_1, s_2) with the given fixed sum, as long as it is between $\xi((s_1 + s_2)^-)$ and $\xi((s_1 + s_2)^+)$ (symmetric statement holds for the s-player, if ξ is constant and therefore ξ^{-1} has a jump discontinuity).

We remark that a bundling mechanism can be considered as a relaxed affine minimizer having $\gamma_1 = \gamma_2 = \infty$ (see next paragraph). A bundling mechanism is truthful with the normalized payments $P_\emptyset(s) = P_1(s) = P_2(s) = 0$, and $P_{12}(s) = \xi(s_1 + s_2)$ for the t-player, and analogously, $P^s_{12}(t) = \xi^{-1}(t_{12})$ for the s-player. For every fixed bid s, the t-player incurs total cost of either

[5]This assumption is without loss of generality for mechanisms with finite approximation of the makespan, even if some 2D cut mechanism of a WMON mechanism with more tasks and/or players is considered.

$t_{12} - P_{12} = t_{12} - \xi(s_1 + s_2)$, or 0. Given that she bids t_{12} truthfully, her total cost will be $\min\{t_{12} - \xi(s_1 + s_2), 0\}$, because the allocation of the mechanism minimizes among these two terms as well. The truthfulness for the s-player is analogous.

3.2.2 Relaxed affine minimizers.

Definition 4. An allocation A is an *affine minimizer*, if there exist positive constants per player μ_t and μ_s, and constants $\gamma_\alpha \in \mathbb{R} \cup \{-\infty, \infty\}$ per allocation (say, here $\alpha = \alpha_t$), so that for every input (t, s) the allocation $A(t, s)$ minimizes over

$$\mu_t t_{12} + \gamma_{12}, \quad \mu_t t_1 + \mu_s s_2 + \gamma_1, \quad \mu_s s_1 + \mu_t t_2 + \gamma_2, \quad \mu_s(s_1 + s_2) + \gamma_\emptyset.$$

Subtracting the last term, and dividing by μ_t yields that the payments of the t-player in an affine minimizer can be $P_\emptyset = 0$, $P_1 = \max\{(\mu_s s_1 + \gamma_\emptyset - \gamma_1)/\mu_t, 0\}$, $P_2 = \max\{(\mu_s s_2 + \gamma_\emptyset - \gamma_2)/\mu_t, 0\}$ and $P_{12} = \max\{(\mu_s(s_1 + s_2) + \gamma_\emptyset - \gamma_{12})/\mu_t, P_1, P_2\}$. Then the allocation of the mechanism minimizes precisely the player's total cost if she bids truthfully (and analogously for the s-player).

Definition 5. An allocation A is a *relaxed affine minimizer*, if there exist positive constants per player μ_t and μ_s, and constants γ_α per allocation $\alpha = \alpha_t$, furthermore an arbitrary non-decreasing function $\xi : [0, \min(\gamma_1, \gamma_2) - \gamma_\emptyset] \to [0, \min(\gamma_1, \gamma_2) - \gamma_{12}]$ (assuming both intervals are nonempty) with $\xi(\min(\gamma_1, \gamma_2) - \gamma_\emptyset) = \min(\gamma_1, \gamma_2) - \gamma_{12}$, so that for every input (t, s)

(*a*) if $\mu_s \cdot (s_1 + s_2) \geq \min(\gamma_1, \gamma_2) - \gamma_\emptyset$, then the allocation $A(t, s)$ is that of an affine minimizer with the given constants

(*b*) if $\mu_s \cdot (s_1 + s_2) \leq \min(\gamma_1, \gamma_2) - \gamma_\emptyset$, then if $\mu_t \cdot t_{12} > \xi(\mu_s(s_1 + s_2))$ then the allocation for the t-player is $\emptyset$ and if $\mu_t \cdot t_{12} < \xi(\mu_s \cdot (s_1 + s_2))$ then it is 12.

The payments of the t-player $P_\emptyset(s)$, $P_1(s)$ and $P_2(s)$ are defined as for affine minimizers. The payment $P_{12}(s)$ is the same as in affine minimizers if $\mu_s \cdot (s_1 + s_2) \geq \min(\gamma_1, \gamma_2) - \gamma_\emptyset$, but it is $P_{12} = \xi(\mu_s(s_1 + s_2))/\mu_t$, if $\mu_s \cdot (s_1 + s_2) \leq \min(\gamma_1, \gamma_2) - \gamma_\emptyset$. Observe that in the latter case $P_1 = P_2 = 0$ must hold, and therefore the regions R_1 and R_2 are empty.[6]

The conditions in the above definition in cases (a) and (b) are consistent. The resulting mechanism is truthful, by the same argument we used above for bundling mechanisms and for affine minimizers. It is crucial that the t-player cannot influence whether case (a) or (b) holds. On the other hand, we could have defined the mechanism symmetrically, from the point of view of the s-player, so for her the mechanism is truthful as well.[7] See Figure 2, for an example of such a mechanism in the additive domain.

3.2.3 Constant mechanisms.

In a *constant mechanism* the allocation is independent of the bids of at least one of the players. This property can also be interpreted as being an affine minimizer with a multiplicative constant $\mu = 0$. Due to the assumption on $R_\emptyset$ we only need to consider constant mechanisms that are independent (at least) of the s-player.

[6]In turn, if $\mu_s \cdot x > \min(\gamma_1, \gamma_2) - \gamma_\emptyset$, then there exist s, so that $x = s_1 + s_2$ but $R_1 \neq \emptyset$ or $R_2 \neq \emptyset$, and this excludes a nonlinear ξ.

[7]The allocation is well-defined even if (say) $\mu_s \cdot (s_1 + s_2) \leq \min(\gamma_1, \gamma_2) - \gamma_\emptyset$, but $\mu_t \cdot (t_{12}) > \min(\gamma_1, \gamma_2) - \gamma_{12}$. Namely, for the t-player the allocation seems bundling, but t_{12} is large and he gets no task; for the s-player the allocation seems to be an affine minimizer, and $s_1 + s_2$ is so small, that s gets both tasks.

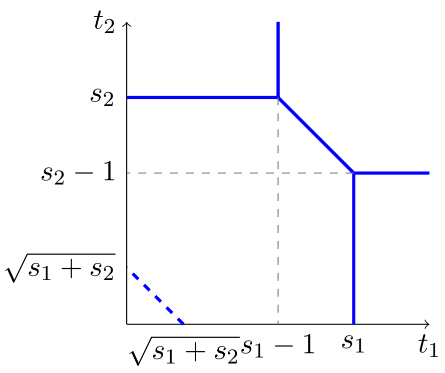

Figure 2: An example of a relaxed affine minimizer, which shows the allocation of the t-player. The solid lines show the boundaries of the allocations for values of the s-player when $s_1+s_2 \geq 1$. The dashed lines show the allocation boundary when $s_1 + s_2 < 1$. Sometimes we refer to this part of relaxed affine minimizers as "bundling tail".

3.3 The main characterization result

The characterization that we use for the lower bound is captured by the following theorem, whose proof can be found in the full version of the paper [9].

Theorem 3. *Every WMON allocation for two tasks and two players with bids $t \in V_{submod}$ and $s \in V_{+,B}$, where both tasks are always allocated, and $R_\emptyset(s) \neq \emptyset$ for every s, is one of these three types: (1) relaxed affine minimizer, (2) one-dimensional mechanism, or (3) constant mechanism.*

In the full version we show that the same characterization (and hence our lower bound) holds for WMON allocations for various domains of valuations of the t-player and in particular for submodular valuations which are arbitrarily close to additive, which we denote by $V_\epsilon \cap V_{submod}$.

4 Lower Bound

In this section, we give a proof of our main theorem (Theorem 1). First in Section 4.1 we describe the domain of instances that we use, and in Section 4.2 we use the characterization for two machines and two tasks (Theorem 3) to establish that the only interesting mechanisms are linear (see the subsection for a precise definition), then in Section 4.3 we explore the linearity property of mechanisms with bounded approximation ratio to establish some useful locality lemmas that are eventually used in Section 4.4 to complete the proof.

4.1 The construction

To prove the lower bound, we focus on the domain of $2(n-1)$ tasks and n players. Player 0 is special and for convenience we use the symbol t for its values; sometimes we refer to it as the t-player. We use the symbol s for the values of the remaining players $1, \dots, n-1$, and sometimes we refer to them as the s-players.

The set of tasks $M = \{1, 1', 2, 2', \dots, n-1, (n-1)'\}$ is partitioned in pairs and each pair $\{i, i'\}$ is associated with player i, $i = 1, \dots, n-1$. We call the two tasks of each pair $\{i, i'\}$ *twin tasks.*

Let $v_i(S)$ denote the cost (valuation) of player i when it takes the subset $S \subseteq M$ of tasks.

- the cost of the s-players is *additive*: $v_i(S)$ is additive for $i \geq 1$.

- the cost of the s-players for tasks not in their associated pair is a sufficiently large fixed constant $\Theta \gg 4n^2$: for distinct $i, j \geq 1$, $v_i(\{j\}) = v_i(\{j'\}) = \Theta$.
- the cost of the t-player for twin tasks is submodular: for every $i = 1, \dots, n-1$, the restriction of $v_0(S)$ to $S \subseteq \{i, i'\}$ is submodular.
- the cost of the t-player is *additive across pairs*: $v_0(S) = \sum_{i=1}^{n-1} v_0(S \cap \{i, i'\})$. Therefore $v_0(S)$ is *submodular*, as the sum of submodular functions.

To simplify the notation we will denote an instance that satisfies the above conditions by

$$(t_i,\, s_i,\, t_{i'},\, s_{i'},\, t_{i,i'})_{i=1}^{n-1}, \tag{1}$$

where

- s_i and $s_{i'}$ are the costs of the i-th s-player for the tasks in its associated pair $\{i, i'\}$,
- t_i or $t_{i'}$, resp., is the cost of the t-player when it takes *only one* of the twin tasks $\{i, i'\}$,
- $t_{i,i'}$ is the cost of the t-player when it takes *both twin tasks* $\{i, i'\}$.

With the exception of costs $t_{i,i'}$ an instance is captured by the following matrix indexed by players and tasks, which shows the cost of each when a player gets no other task.

$$\begin{bmatrix} t_1 & t_{1'} & t_2 & t_{2'} & \cdots & t_{n-1} & t_{(n-1)'} \\ s_1 & s_{1'} & \Theta & \Theta & \cdots & \Theta & \Theta \\ \Theta & \Theta & s_2 & s_{2'} & \cdots & \Theta & \Theta \\ & & & \vdots & & & \\ \Theta & \Theta & \Theta & \Theta & \cdots & s_{n-1} & s_{(n-1)'} \end{bmatrix}.$$

If the valuations of all players were additive, this matrix would be sufficient to determine the cost for all bundles. The instances we consider have more general cost functions and include the submodular valuation of twin tasks for the t-player. Since for two tasks, a function is submodular if and only if it is subadditive, values $t_{i,i'}$ satisfy

$$t_{i,i'} \leq t_i + t_{i'}.$$

It is useful to think of the value Θ as practically infinite, since it is much larger than the other values. On the other hand, to prove our main theorem in its generality, we need this value to be finite and this complicates the characterization of truthful mechanisms.

We focus on instances that satisfy $s_i \in [0,1]$, $s_{i'} = n$, $t_{i'} = 0$, $t_{i,i'} = t_i + t_{i'}$ for all $i \in [n-1]$. Note the subtlety here: while the domain contains instances with $t_{i,i'} \leq t_i + t_{i'}$, for the proof of the lower bound, we consider the subclass of additive instances. This should not be surprising in the sense that lower bound proofs usually employ a subclass of instances. Still, it raises the question whether we could carry out the same proof in the additive domain. The answer is negative, because the sets of mechanisms for additive and submodular (subadditive) domains for two tasks are different; for example, task-independent mechanisms are not truthful for submodular domains.

Definition 6 (Restricted (t,s) instance)**.** Instances of the form $(t_i,\, s_i,\, t_{i'},\, s_{i'},\, t_{i,i'})_{i=1}^{n-1}$ (Equation 1) that satisfy

$$\begin{aligned} &s_i \in [0,1], \quad s_{i'} = n, \quad t_{i'} = 0, \quad t_{i,i'} = t_i + t_{i'}, \qquad \text{for all } i \in [n-1], \\ &(t,s) = ((t_i)_{i=1}^{n-1}, (s_i)_{i=1}^{n-1}) \end{aligned} \tag{2}$$

will be called *restricted* (t,s) *instances.*

Note that, unlike the other values, the t_i values can be arbitrarily high. This will be useful later (Lemma 3). Note also that the optimum makespan of every restricted instance is at most 1. For these instances, any algorithm with approximation ratio less than n, must allocate all the i' tasks to the t-player and every i task either to the associated s_i-player or to the t-player.

4.2 From affine minimizers for twin tasks to linear mechanisms

In this section, we use the characterization of mechanisms (Theorem 3) for *each pair of twin tasks* in an instance $(t_i,\, s_i,\, t_{i'},\, s_{i'},\, t_{i,i'})_{i=1}^{n-1}$, to derive an essential property of mechanisms for the restricted (t,s) instances. By the characterization, there are mechanisms for twin tasks, such as the one-dimensional, constant, or bundling tails of relaxed affine minimizers, that may have non-affine boundaries between different allocations. In this section, we show that mechanisms with non-affine boundaries are also excluded, when we additionally require that the mechanisms have small approximation ratio – on whole instances, not just on a pair of twin tasks.

First we define the notion of boundaries that we consider in the proof of the lower bound.

Definition 7. For a given allocation algorithm, considered on restricted (t,s) instances, we call $\psi_i(s_i, s_{-i}, t_{-i})$ a *critical value* or *boundary* if the t-player receives task i when $t_i < \psi_i(s_i, s_{-i}, t_{-i})$, and does not receive task i when $t_i > \psi_i(s_i, s_{-i}, t_{-i})$ (for example in Figure 1, ψ_1 is shown with dashed lines).

We claim below that a truthful allocation with a reasonable approximation ratio (say, of at most n) for the restricted (t,s) instances satisfies the following important property:

Definition 8 (Linear mechanisms)**.** An allocation algorithm for the restricted (t,s) instances (Equation 2), is called *linear* if the critical values for task i, $i = 1,\ldots,n-1$ are truncated affine functions in s_i. In particular when

$$\psi_i(s_i, s_{-i}, t_{-i}) = \max(0,\, \lambda_i(s_{-i}, t_{-i})\, s_i \,+\, \nu_i(s_{-i}, t_{-i})),$$

for some $\lambda_i(s_{-i}, t_{-i}) > 0$. We call a mechanism linear, if it uses a linear allocation algorithm.

Lemma 3. *A mechanism for the instances $(t_i,\, s_i,\, t_{i'},\, s_{i'},\, t_{i,i'})_{i=1}^{n-1}$ (Equation 1) with approximation ratio less than n must be linear for the restricted (t,s) instances.*

Proof. By Lemma 1, the restriction of the mechanism for the twin tasks i, i' is weakly monotone, and therefore by the characterization theorem (Theorem 3), it is either a relaxed affine minimizer, a one-dimensional, or a constant mechanism. We will show that if the approximation ratio is less than n, the mechanism cannot be one-dimensional, constant, or the bundling part of a relaxed affine minimizer. The reason is that algorithms in all these cases are so inefficient that their makespan only for these two tasks is large enough to show a large approximation ratio for the entire instance.

Let's first argue that it cannot be a one-dimensional or a constant mechanism. We fix all values of the other tasks as in restricted instances. There are (not restricted) values for the twin tasks i, i' for which the algorithm has makespan $\mu \geq 2n$ and approximation ratio at least $2n$. (For each one-dimensional or constant allocation this holds for at least one of the inputs $\left[\begin{smallmatrix} 2n & 0 \\ 0 & 2n \end{smallmatrix}\right]$, $\left[\begin{smallmatrix} 2n & 0 \\ 4n^2 & 0 \end{smallmatrix}\right]$.) Since the optimal makespan of the other tasks of a restricted instance is at most 1, the approximation ratio is at least $\mu/(1+\mu/2n) \geq 2n\mu/2\mu = n$.

As a result, the restriction to twin tasks i, i' is a relaxed affine minimizer. We now argue that if the instance is restricted, and in particular $s_i \leq 1$ and $s_{i'} = n$ hold, then the t-player should not be in the bundling part of the relaxed affine minimizer when the approximation ratio is less than n (his allocation figure is as the solid lines in Figure 2, because $s_i + s_{i'}$ is large). Let us fix the values $s_j \in [0,1]$, $s_{j'} = n$ for *all* $j \in [n-1]$, and the values $t'_j = 0$, $t_{j,j'} = t_j + t_{j'}$ for all $j \neq i$. We observe that the restriction of the mechanism in the cut $t_{-\{i,i'\}}$ (the 2-dimensional partition when we also fix all other values except t_i and t'_i) should definitely contain an area where the t-player gets task i' but not i (in particular, when $t_{i'} = \delta$, for some arbitrarily small δ, and t_i is very large). Otherwise the mechanism has approximation ratio at least n.

This shows that in the domain of restricted (t,s) instances, the mechanisms with approximation ratio less than n are affine minimizers (for the t-player) in every cut of pairs of twin tasks (even if in general, that is, for $s_{i'} < n$ they are relaxed affine minimizers). So, the only mechanisms with approximation ratio less than n are linear for the restricted (s,t) instances. □

4.3 Linearity

From now on we focus on linear truthful mechanisms for restricted (t, s) instances. Establishing linearity of the boundaries (Lemma 3) is not directly useful, because the linear coefficient of the boundary i *may depend on the other values of* t. The next lemma shows that this is not the case for the scaling factor λ_i, although the ν_i term may still depend on the other values of t. Its proof is based on the interaction of pairs of tasks i and j. Note that these are *not* twin tasks but involve at least three players, the t-player and the two associated s-players. Obtaining such multiplayer statements is the bottleneck for a complete characterization of mechanisms in multiplayer domains, and it is perhaps the most crucial part of the proof.

Lemma 4. *For fixed i and j, and for fixed s_{-i}, t_{-ij}, assume that $\psi_i(s_i, s_{-i}, t_{-i})$ is a truncated linear function of s_i, i.e., $\psi_i(s_i, s_{-i}, t_{-i}) = \max(0,\ \lambda_i(s_{-i}, t_{-i})s_i + \nu_i(s_{-i}, t_{-i}))$, for some $\lambda_i(s_{-i}, t_{-i}) \geq 0$. Then $\lambda_i(s_{-i}, t_{-i})$ is the same for all t_j that satisfy $\psi_i(s_i, s_{-i}, t_{-i}) > 0$ (for some s_i).*

Proof. To keep the notation simple, we drop all values except for s_i and t_j. Let $K(s_i) = \{t_j : \psi_i(s_i, t_j) > 0\}$ be the interval of interest. Note first that when we increase s_i, the interval of interest can only expand, because $\psi_i(s_i, t_j)$ is non-decreasing in s_i.

Within interval $K(s_i)$, function $\psi_i(s_i, t_j)$ is positive and therefore equal to $\lambda_i(t_j)s_i + \nu_i(t_j)$. Furthermore, weak monotonicity implies that as a function of t_j, $\psi_i(s_i, t_j)$ is a piecewise linear function with derivative (slope) in $\{0, 1, -1\}$ (see for example, Figure 1).

The piecewise linear function $\psi_i(s_i, t_j)$ is differentiable[8] everywhere except perhaps of the at most two break points. We have

$$\frac{\partial \psi_i(s_i, t_j)}{\partial t_j} = \frac{\partial \lambda_i(t_j)}{\partial t_j} s_i + \frac{\partial \nu_i(t_j)}{\partial t_j}.$$

If $\partial \lambda_i(t_j)/\partial t_j \neq 0$, then by varying s_i, the slope cannot stay in $\{0, 1, -1\}$. We conclude that $\partial \lambda_i(t_j)/\partial t_j = 0$ in each piece, which shows that $\lambda_i(t_j)$ is independent of t_j within each piece and therefore independent of t_j everywhere. □

The following corollary is an essential tool for establishing the lower bound.

Corollary 1. *If the allocation restricted to tasks i, j is quasi-bundling or quasi-flipping, then the constant parts of the piecewise linear function $\psi_i(s_i, s_{-i}, t_{-i})$, as a function of t_j, are independent of s_j, for $j \neq i$.*

Proof. We fix all values except for tasks i and j. By the previous lemma, when we change s_j, the boundary $\psi_j(s_j, s_i, t_i)$ is translated rectilinearly and therefore its break points remain the same. Since the constant parts of $\psi_i(s_i, s_j, t_j)$ are determined by the break points of $\psi_j(s_j, s_i, t_i)$, they also remain the same. See Figure 3 for an illustration. □

Corollary 2. *If the allocation restricted to tasks i, j is quasi-bundling, then the piecewise linear function $\psi_i(s_i, s_{-i}, t_{-i})$ is either non-decreasing in t_j or non-decreasing in s_j.*

Proof. When we fix all other values and consider the $t_{-\{i,j\}}$ cut, it is either quasi-flipping or crossing, in which case $\psi_i(s_i, s_j, t_j)$ is non-decreasing in t_j, or quasi-bundling, in which case 2D geometry shows that it is non-decreasing in s_j (when s_j decreases the diagonal part shifts downwards, and the rectilinear parts remain fixed). See Figure 3 for an illustration.

□

[8] It is not hard to see that if $\psi_i(s_i, t_j)$ is differentiable then so are $\lambda_i(t_j)$ and $\nu_i(t_j)$; alternatively, one could use small differences instead of derivatives.

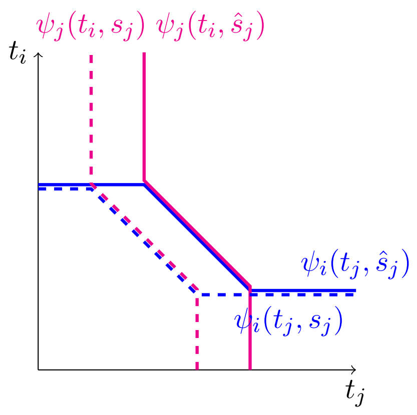

Figure 3: Consider the dashed lines that show $\psi_i(t_j, s_j)$ and $\psi_j(t_i, s_j)$ (these functions may depend on other values but they play no role and we can ignore them). The regular lines show $\psi_i(t_j, \hat{s}_j)$ and $\psi_j(t_i, \hat{s}_j)$, for some $\hat{s}_j > s_j$. Notice that linearity implies that $\psi_j(t_i, \hat{s}_j)$ is a shift to the right of $\psi_j(t_i, s_j)$; in particular, the break points stay at the same height. As a consequence, the blue horizontal parts remain at the same height and the blue diagonal part shifts to the right. Therefore $\psi_i(t_j, \hat{s}_j) \geq \psi_i(t_j, s_j)$.

4.4 The main theorem

We now have most of the ingredients to prove the next theorem which directly implies the main lower bound (Theorem 1).[9]

Theorem 4. *The approximation ratio of linear truthful algorithms on restricted* (t, s) *instances (Equation 2) with* n *machines is at least* $\sqrt{n-1}$.

To prove this theorem, we fix some linear truthful algorithm and we focus on a particular set of instances, which we call s-inefficient. The set of s-inefficient instances depends on the linear algorithm and each instance consists of two types of tasks: either a task is unimportant (i.e., it has very small value for one of the machines), or its s value is significantly higher than its t value, yet the algorithm allocates the task to the s-player.

Definition 9 (s-inefficient instances)**.** Let's call a task i *trivial* when either $s_i = 0$ or $t_i \in (0, \delta_0]$, for some fixed (sufficiently small) δ_0. We call a restricted (t, s) instance *s-inefficient* for a mechanism, if it contains at least one non-trivial task, every non-trivial task i satisfies $s_i/t_i > \sqrt{n-1}$, and the mechanism allocates all non-trivial tasks to the s-player.

The heart of the proof is to show that if the set of s-inefficient instances is non-empty, there exists an s-inefficient instance with exactly one non-trivial task. From this, it immediately follows that if s-inefficient instances exist, then the algorithm has approximation ratio at least $\sqrt{n-1}$.

However, it may be that for a given linear truthful algorithm there are no s-inefficient instances. But then we can use weak monotonicity to easily derive a $\sqrt{n-1}$ lower bound on the approximation ratio as the following lemma shows.

Lemma 5. *If for a given truthful algorithm the set of s-inefficient instances is empty, then its approximation ratio is at least* $\sqrt{n-1}$.

Proof. Towards a contradiction, consider an algorithm that has approximation ratio $\sqrt{n-1}-\delta$, for some $\delta > 0$, for which the set of s-inefficient instances is empty. We consider the instance

[9]We should emphasize that Lemma 4 and Theorem 4 hold independent of the domain of the t-player (whether additive or submodular), whereas Lemma 3 does not hold for all truthful mechanisms when the t-player is additive.

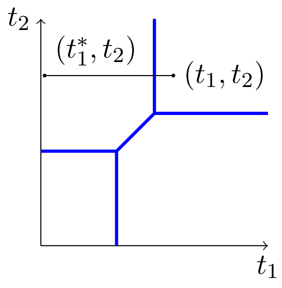

Figure 4: When the cut is quasi-flipping or crossing, and neither task is given to the t-player in the allocation for (t_1, t_2) (w.l.o.g. (t_1, t_2) is not on the boundary), then the second task is still not allocated to the t-player in the allocation for $(t_1^* \approx 0, t_2)$.

with $t_i = \alpha = 1/\sqrt{n-1} - \delta/n$ and $s_i = 1$, for all $i \in [n-1]$.

$$\begin{bmatrix} \alpha & \alpha & \cdots & \alpha \\ 1 & & & \\ & 1 & & \\ & & \vdots & \\ & & & 1 \end{bmatrix} \tag{3}$$

Note that at least one task is allocated to the s-players, because if all tasks are assigned to the t-player, the makespan is $(n-1)\alpha = \sqrt{n-1} - (n-1)\delta/n > \sqrt{n-1} - \delta$, the optimum makespan is 1, and the approximation ratio is strictly greater than $\sqrt{n-1} - \delta$.

Now for every task i which is allocated to the t-player, we lower its value from α to some small value in $(0, \delta_0]$ and increase the t value of every other task to $\alpha + \delta/(2n) < 1/\sqrt{n-1}$. By weak monotonicity (Lemma 2), the allocation of the tasks for the t-player remains the same. Since at least one task is allocated to the s-players, we end up with an s-inefficient instance, a contradiction. $\square$

The proof of the next lemma (Lemma 6), which provides a very useful property of linear mechanisms, is the most critical part of the proof. It shows that linear weakly monotone algorithms satisfy a locality property, that bears some resemblance to the locality property in [33]. But unlike [33], our proof does not assume this property, but it derives it from weak monotonicity for the special class of instances that we consider.

Lemma 6. *If for a given linear truthful algorithm the set of s-inefficient instances is non-empty, then there is an s-inefficient instance with exactly one non-trivial task.*

Proof. Fix a linear truthful algorithm and consider an s-inefficient instance (t, s) with the minimum number of non-trivial tasks. If the number of non-trivial tasks is at least two, let us assume without loss of generality that tasks 1 and 2 are non-trivial. We will derive a contradiction by reducing the number of non-trivial tasks.

Consider the boundary function of the first task, $\psi_1(s_1, s_{-1}, t_{-1}) = \max(0, \lambda_1(s_{-1}, t_{-1})s_1 + \nu_1(s_{-1}, t_{-1})$, for some positive λ_1. The crux of the matter is that we can reduce either the value of s_1 to 0, or t_1 to at most δ_0, and guarantee that the t-player will keep not getting the second task. This guarantees that the second task is non-trivial and is given to the s-player, while the first task becomes trivial.

If $\psi_2(s_2, s_{-2}, t_{-2})$ as a function of t_1 is non-decreasing (i.e., the $t_{-\{1,2\}}$ cut is quasi-flipping or crossing), we set $t_1^* \in (0, \delta_0]$. Since $\psi_2(s_2, s_{-2}, t_{-2})$ is non-decreasing in t_1, in the new instance the second task is still allocated to the s-player (Figure 4).

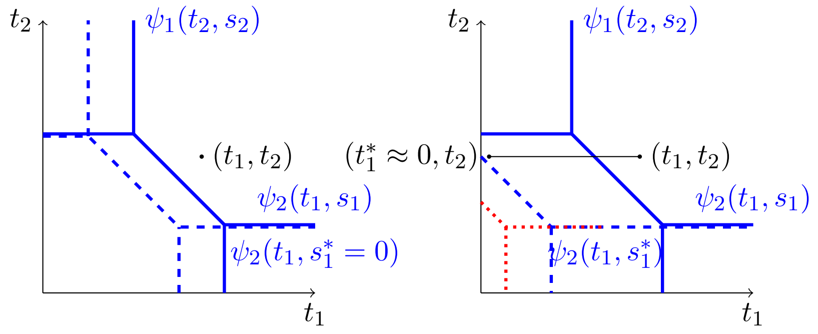

Figure 5: Left side case (a), when s_1^* decreases to 0: the boundary $\psi_2(t_1, s_1^*)$ is lower than the boundary $\psi_2(t_1, s_1^*)$. Both tasks are still allocated to the s-player. Right side case (b), when the s_i decreases to $s_1^* = -\nu_1(s_{-1}, t_{-1})/\lambda_1(s_{-1}, t_{-1})$ and t_1 changes to $t_1^* \approx 0$. The new (t_1^*, t_2) remains in the region in which both tasks are allocated to the s-player.

Otherwise (i.e., the $t_{-\{1,2\}}$ cut is quasi-bundling), by Corollary 2, function $\psi_2(s_2, s_{-2}, t_{-2})$ is non-decreasing in s_1. In this case, we change the instance as follows (see Figure 5 for an illustration of the first two cases):

a) if $\nu_1(s_{-1}, t_{-1}) \geq 0$, we set $s_1^* = 0$ and $t_1^* = t_1$,

b) otherwise, if $\psi_1(s_1, s_{-1}, t_{-1}) > 0$, we set

$$s_1^* = -\nu_1(s_{-1}, t_{-1})/\lambda_1(s_{-1}, t_{-1})$$

and $t_1^* \in (0, \delta_0]$, small enough to make task 1 trivial,

c) and if $\psi_1(s_1, s_{-1}, t_{-1}) = 0$, we set $s_1^* = s_1$ and $t_1^* \in (0, \delta_0]$, small enough to make task 1 trivial.

In the first two cases, we lower the s_1 value to s_1^* until either s_1^* becomes 0 (case a), or $\psi_1(s_1^*, s_{-1}, t_{-1})$ becomes 0 and then we set $t_1 = t_1^* \approx 0$ (case b). The third case is when $\psi_1(s_1, s_{-1}, t_{-1})$ is already 0.

In all cases, the first task becomes a trivial task with the new values. Furthermore, in all cases the new value of s_1 is not greater than the original value: $s_1^* \leq s_1$. This is clearly true in the first and third case. To see that this is true in the second case, observe that the boundary $\psi_1(s_1, s_{-1}, t_{-1})$, which is a non-decreasing function on s_1, moved from a positive value to $\psi_1(s_1^*, s_{-1}, t_{-1}) = 0$.

We now argue that *the second task is still given to the s-player after the change.* The argument is based on Corollary 1, which guarantees that the changes can only shift the boundary $\psi_2(s_2, s_{-2}, t_{-2})$ rectilinearly: in Figure 5, the slanted part moves only horizontally.

For case (a), by Corollary 2, the boundary $\psi_2(s_2, s_{-2}, t_{-2})$ did not increase and therefore the second task is still given to the s-player (left part of Figure 5). For the other two cases, this is not sufficient because t_1 changed and therefore t_{-2} changed as well. For case (b), the change shifts the slanted boundary (right part of Figure 5). In its new position, the slanted boundary meets the boundary of the positive orthant at $(0, t_2)$ which is dominated by the point (t_1^*, t_2). Therefore, the t-player gets neither task, so the s-player gets the second task. Case (c) is simpler and similar to the second one; the difference is that the slanted boundary starts at the leftmost position and it does not need to move at all (see the dotted lines in the right part of Figure 5).

The change of values of the first task did not change the allocation of the second task, but it may have changed the allocation of the remaining non-trivial tasks. But we can use weak monotonicity to further change the instance to obtain an s-inefficient instance. If some non-trivial tasks changed allocation and were given to the t player, we reduce their t values to 0 (this will make them trivial), and increase slightly the t values of the other non-trivial tasks without violating the constraint $s_i/t_i > \sqrt{n-1}$. By weak monotonicity (Lemma 2), this preserves the allocation of the first player for the other non-trivial tasks. The resulting instance is s-inefficient and has fewer non-trivial tasks, a contradiction. $\square$

The proof of the main result of this section follows directly from the last two lemmas.

Proof of Theorem 4. By the last two lemmas, either a linear truthful algorithm has approximation ratio at least $\sqrt{n-1}$, or there exists an s-inefficient instance with one non-trivial task. The approximation ratio of such an instance is trivially at least $\sqrt{n-1}$, and the proof is complete. $\square$

5 Conclusions

We showed a lower bound of $\Omega(\sqrt{n})$ for the approximation ratio achieved by deterministic truthful mechanisms for the problem of scheduling n unrelated machines with submodular cost functions, when the objective is makespan minimization.

It remains open whether the lower bound can be improved to n, which we conjecture to be the right answer, as in the Nisan-Ronen conjecture. However, we should note that the instances used in our lower bound construction (Form (1) in Section 4.1) have approximation ratio $\Theta(\sqrt{n})$. This is so, because the weighted VCG mechanism that assigns a weight of $\sqrt{n}$ to the t-player and 1 to the rest of the players can achieve approximation ratio $O(\sqrt{n})$.

It should be emphasized that submodularity is essential to establish linearity, and hence to achieve the claimed lower bound. An obstacle in extending our results to the case of additive players is the existence of other, non-linear, mechanisms. In fact, in a follow up paper [10], we show a new mechanism that achieves a constant approximation for the corresponding additive instances of the lower bound construction.

References

[1] Aaron Archer and Robert Kleinberg. Truthful germs are contagious: A local-to-global characterization of truthfulness. *Games and Economic Behavior*, 86:340–366, Jul 2014.

[2] Aaron Archer and Éva Tardos. Truthful mechanisms for one-parameter agents. In *Proc. of the 42nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 482–491, 2001.

[3] Itai Ashlagi, Shahar Dobzinski, and Ron Lavi. Optimal lower bounds for anonymous scheduling mechanisms. *Mathematics of Operations Research*, 37(2):244–258, 2012.

[4] Vincenzo Auletta, George Christodoulou, and Paolo Penna. Mechanisms for scheduling with single-bit private values. *Theory Comput. Syst.*, 57(3):523–548, 2015.

[5] Sushil Bikhchandani, Shurojit Chatterji, Ron Lavi, Ahuva Mu'alem, Noam Nisan, and Arunava Sen. Weak monotonicity characterizes deterministic dominant-strategy implementation. *Econometrica*, 74(4):1109–1132, Jul 2006.

[6] Shuchi Chawla, Jason D. Hartline, David L. Malec, and Balasubramanian Sivan. Prior-independent mechanisms for scheduling. In *STOC*, pages 51–60. ACM, 2013.

[7] Xujin Chen, Donglei Du, and Luis Fernando Zuluaga. Copula-based randomized mechanisms for truthful scheduling on two unrelated machines. *Theory Comput. Syst.*, 57(3):753–781, 2015.

[8] George Christodoulou, Elias Koutsoupias, and Annamária Kovács. Mechanism design for fractional scheduling on unrelated machines. *ACM Trans. Algorithms*, 6(2):38:1–38:18, 2010.

[9] George Christodoulou, Elias Koutsoupias, and Annamaria Kovacs. On the Nisan-Ronen conjecture for submodular valuations, 2019.

[10] George Christodoulou, Elias Koutsoupias, and Annamária Kovács. Truthful graph balancing. Manuscript, 2020.

[11] George Christodoulou, Elias Koutsoupias, and Angelina Vidali. A characterization of 2-player mechanisms for scheduling. In *ESA*, volume 5193 of *Lecture Notes in Computer Science*, pages 297–307. Springer, 2008.

[12] George Christodoulou, Elias Koutsoupias, and Angelina Vidali. A lower bound for scheduling mechanisms. *Algorithmica*, 55(4):729–740, 2009.

[13] George Christodoulou and Annamária Kovács. A deterministic truthful PTAS for scheduling related machines. *SIAM J. Comput.*, 42(4):1572–1595, 2013.

[14] Edward H. Clarke. Multipart pricing of public goods. *Public Choice*, 11(1):17–33, Sep 1971.

[15] Constantinos Daskalakis and S. Matthew Weinberg. Bayesian truthful *Mechanisms* for job scheduling from bi-criterion approximation *Algorithms.* In *SODA*, pages 1934–1952. SIAM, 2015.

[16] Peerapong Dhangwatnotai, Shahar Dobzinski, Shaddin Dughmi, and Tim Roughgarden. Truthful approximation schemes for single-parameter agents. *SIAM J. on Computing*, 40(3):915–933, 2011.

[17] Shahar Dobzinski. Breaking the logarithmic barrier for truthful combinatorial auctions with submodular bidders. In *STOC*, pages 940–948. ACM, 2016.

[18] Shahar Dobzinski and Noam Nisan. Multi-unit auctions: Beyond roberts. *J. Econ. Theory*, 156:14–44, 2015.

[19] Shahar Dobzinski and Mukund Sundararajan. On characterizations of truthful mechanisms for combinatorial auctions and scheduling. In *EC*, pages 38–47. ACM, 2008.

[20] Shahar Dobzinski and Jan Vondrák. Impossibility results for truthful combinatorial auctions with submodular valuations. *J. ACM*, 63(1):5:1–5:19, 2016.

[21] Leah Epstein, Asaf Levin, and Rob van Stee. A unified approach to truthful scheduling on related machines. *Math. Oper. Res.*, 41(1):332–351, 2016.

[22] Yiannis Giannakopoulos and Maria Kyropoulou. The VCG mechanism for bayesian scheduling. *ACM Trans. Economics and Comput.*, 5(4):19:1–19:16, 2017.

[23] Theodore Groves. Incentives in teams. *Econometrica*, 41(4):617–631, 1973.

[24] Elias Koutsoupias and Angelina Vidali. A lower bound of $1+\phi$ for truthful scheduling mechanisms. *Algorithmica*, pages 1–13, 2012.

[25] Ron Lavi and Chaitanya Swamy. Truthful mechanism design for multidimensional scheduling via cycle monotonicity. *Games Econ. Behav.*, 67(1):99–124, 2009.

[26] Benny Lehmann, Daniel Lehmann, and Noam Nisan. Combinatorial auctions with decreasing marginal utilities. *Games Econ. Behav.*, 55(2):270–296, 2006.

[27] Jan Karel Lenstra, David B Shmoys, and Eva Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical programming*, 46(1-3):259–271, 1990.

[28] Stefano Leucci, Akaki Mamageishvili, and Paolo Penna. No truthful mechanism can be better than n approximate for two natural problems. *Games Econ. Behav.*, 111:64–74, 2018.

[29] Pinyan Lu. On 2-player randomized mechanisms for scheduling. In *WINE*, volume 5929 of *Lecture Notes in Computer Science*, pages 30–41. Springer, 2009.

[30] Pinyan Lu and Changyuan Yu. An improved randomized truthful mechanism for scheduling unrelated machines. In *25th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 1 of *LIPIcs*, pages 527–538, 2008.

[31] Pinyan Lu and Changyuan Yu. Randomized truthful mechanisms for scheduling unrelated machines. In *4th International Workshop on Internet and Network Economics (WINE)*, pages 402–413, 2008.

[32] Ahuva Mu'alem and Michael Schapira. Setting lower bounds on truthfulness. *Games Econ. Behav.*, 110:174–193, 2018.

[33] Noam Nisan and Amir Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35:166–196, 2001.

[34] Kevin Roberts. The characterization of implementable choice rules. *Aggregation and Revelation of Preferences*, pages 321–348, 1979.

[35] Michael E. Saks and Lan Yu. Weak monotonicity suffices for truthfulness on convex domains. In *EC*, pages 286–293. ACM, 2005.

[36] William Vickrey. Counterspeculations, auctions and competitive sealed tenders. *Journal of Finance*, 16:8–37, 1961.

[37] Angelina Vidali. The geometry of truthfulness. In *5th Workshop on Internet and Network Economics (WINE'09), Springer*, pages 340–350, 12 2009.

[38] Changyuan Yu. Truthful mechanisms for two-range-values variant of unrelated scheduling. *Theoretical Computer Science*, 410(21-23):2196–2206, 2009.