Testing Truthfulness for Single Parameter Agents

Nir Andelman* Yishay Mansour†

April 8, 2005

Abstract

We consider the task of designing truthful mechanisms for single parameter agents. We prove a general sufficient condition for truthfulness when the valuation function of the agents of any outcome is one dimensional and continuous. For certain types of natural valuation functions, our condition is also necessary. One of the main advantages of our characterization is that it provides a computationally efficient method for testing whether a given algorithm admits a truthful payment scheme.

We then demonstrate the simplicity of testing our condition by showing that classical criteria for truthfulness in combinatorial problems such as auctions and machine scheduling can be derived from our condition. In addition, we use our condition to derive results for new single parameter problems, which have not been previously analyzed.

We also consider combinatorial problems where the true types of agents affect the valuation of each other, such as in load balancing. In such cases there are no efficient dominant strategy mechanisms. We show that the same condition can be used to design mechanisms which are ex-post truthful, meaning that the outcome where all agents cooperate and report their true type is a Nash equilibrium. We demonstrate the power of this condition by applying it on the problem of machine scheduling with strategic job owners, which was not addressed before. We prove upper and lower bounds for ex-post truthful mechanisms for this problem.

1 Introduction

1.1 Background

At the core of Mechanism Design is the desire to define a system such that selfish agents interacting with the system would reach the desired outcome. Example of such setting is a centralized decision maker with regulatory power (government, employer, etc.) that wishes to optimize an objective function by aggregating signals on private information collected from agents (buyers, sellers, workers, etc.). The private information of an agent, which is also referred as its type, may include its demand curve, income reports, potential labor power, etc. The information (e.g., bids) that the agents submit signals their types, however they are expected to strategically choose their signal such that they will affect the decision making in a manner that will optimize their own utility.

Mechanism design aims to give the agents an incentive to simply reveal their information to the decision maker, and avoid strategic signaling. This goal is achieved by assigning payments from or to the mechanism, which encourage the agents' cooperation by making truthtelling a (weakly) dominant strategy. I.e., for each agent, regardless of the other agents' actions, reporting its type is a best response strategy. Settings where each agent has a dominant strategy are more convenient to the central decision maker, since it is likely that all agents will indeed follow their dominant strategies, and this frees the central decision maker from considering the implications of strategic behaviors of the agents. By the

*School of Computer Science, Tel-Aviv University. Email: andelman@cs.tau.ac.il

†School of computer Science, Tel-Aviv University. The work was done while the author was a fellow in the Institute of Advance studies, Hebrew University. This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778, by a grant no. 1079/04 from the Israel Science Foundation and an IBM faculty award. This publication only reflects the authors' views.

revelation principle, if dominant strategy mechanisms exist, it is sufficient to consider mechanisms where truthtelling is the dominant strategy (see, e.g., [17]). Under the reasonable assumption that each agent is aware of its own type, truthtelling is the simplest strategy for an agent to follow, which motivates making it a dominant strategy.

Algorithmic mechanism design [20] extends mechanism design into the realm of discrete algorithms, by considering computational limitations on the algorithmic process that calculates the decision rule. Much of the algorithmic mechanism design literature is devoted to studying a wide range of different utility and objective functions and design appropriate efficient mechanisms for them.

One can hope that rather than verifying that a given algorithm and a payment scheme construct together a truthful mechanism, we would rather have a criteria for verifying whether a given algorithm can be melded into a truthful mechanism by adding a proper payment scheme (i.e., if it is rationalizable). Such criteria was given by Rochet in [23]. Additionally, if a decision rule is rationalizable, there is a mathematical formulation to the suitable payment functions. The drawback of Rochet's characterization of rationalizable decision rules is that it does not provide a computationally practical method for testing rationalizability. A simpler condition, referred as *weak monotonicity* [14] or as *2-cycle inequality* [10], is a necessary condition for rationalizability, but not a sufficient condition, as demonstrated in [14].

Given a restricted domain of single parameter agents, we search for a sufficient condition, which is not necessarily a necessary one. On one hand, we aim for a condition that is simple enough to be easily applied to test rationalizability. On the other hand, the condition should be general enough to be useful in constructing truthful mechanisms. We focus on mechanisms for single parameter agents, which although restricted, include a wide range of mechanisms, and weak monotonicity alone does not imply rationalizability by itself.

In the case that the output of the mechanism is continuous, the *Mirrlees-Spence condition* [25, 18] is a simple necessary and sufficient condition that characterizes all truthful single parameter mechanisms. The requirement that the output space is continuous is a sever limitation. In algorithmic mechanism design, the outcome space for many problems is discrete. (Example of such problems include auctions, machine scheduling, routing, etc.) In such settings the Mirrlees-Spence condition is inapplicable, since in discrete spaces the preliminary requirements required for applying this condition immediately fail. Currently, algorithmic mechanism design requires deriving rationalizability conditions for each model, separately. A replacement to the Mirrlees-Spence condition that holds for discrete models would simplify this process in single parameter setups.

1.2 Our Results

We present a generalization to the Mirrlees-Spence condition, which relaxes prior assumptions on the structure of the valuation functions and the decision rule. We define a property named *Halfway Monotone Derivative (HMD)*, prove that all HMD algorithms are rationalizable and characterize the structure of the payment function, which is derived from a weaker property, *Interway Monotone Derivative (IMD)*. We also prove that for some valuation functions, HMD is also a necessary condition.

We apply our condition on several sample algorithmic problems and derive simple conditions for truthfulness, as well as simple structures of the payment functions. Specifically, we show that in all single commodity auctions, the critical value condition is equivalent to the HMD condition. We show that in machine scheduling problems where agents (machines) have a cost per unit of work function, the monotonicity condition of Archer and Tardos [4] is an outcome of the HMD condition. We define a setting where an agent's utility from getting service before some deadline is proportional to the time before the deadline that the service was given, and analyze conditions for rationalizability. We analyze a single parameter version of an auction with budget constraints [2] and give a piecewise monotonicity condition for rationalizability. We then present rationalizable approximation algorithms for this problem.

We then extend our work to models where the valuation of each agent is also affected by the true types of the other agents. Since it may not be possible to achieve a reasonable truthful implementation, we settle with an *ex-post truthful* implementation, where there is a Nash equilibrium where all agents reveal their true type. We show that the HMD condition holds for this model, and then apply it on the problem of scheduling on related machines with strategic job owners, which is a dual problem to scheduling on strategic related machines [4]. We give a 2 approximation which is ex-post truthful, and give a lower bound of 1.19 for all ex-truthful mechanisms. Finally, we consider the problem where both machines and jobs are strategic, and show that the 5 approximation in [1] is ex-post truthful for this problem.

1.3 Related Work

A truthful mechanism for any optimization problem where the objective function of the algorithm is to maximize the sum of utilities (maximize the social welfare) is the *VCG* mechanism [8, 9, 26], which can also be generalized to any maximization problem of an affine function. However, for other objective functions VCG is not applicable. Additionally, optimally solving an affine maximization problem may be computationally hard, which motivates developing truthful mechanisms that only approximate the optimal solution. However, the VCG mechanism does not cope well with approximate solutions [21].

Given that the valuation function of the agents is taken from some restricted domain, several papers have attempted to characterize function spaces for which weak monotonicity would also be a sufficient condition for truthfulness. Roberts [22] has proved sufficient and necessary conditions for truthfulness if the valuation functions of the agents are unrestricted (i.e., any valuation function is possible) and there are at least 3 possible outcomes. The equivalence of these conditions to weak monotonicity is shown in [14]. Several restricted function domains for which weak monotonicity is a sufficient condition are given in [10, 14].

For unrestricted domains, only affine maximizers admit truthful payments [22]. For some restricted domains presented in [14], assuming several properties on the preference function (the most central of them is independence of irrelevant alternatives), a preference function admits a truthful payment only if it is an affine maximizer, over large enough input valuations.

2 Preliminaries

2.1 Mechanisms

The system consists of a decision rule (an algorithm) and n agents. Each agent submits a bid (signal) $b_i \in T$, and then an outcome $\omega \in \Omega$ is calculated by an algorithm $A(b)$, where b denotes the bid vector.

The bid vector without the i-th bid is denoted by b_{-i}. Additionally, (β, b_{-i}) denotes the bid vector b with the i-th bid replaced with β. When it is clear from the context that A and b_{-i} are fixed, we shall let $\omega_{b_i} = A(b_i, b_{-i})$ denote the outcome when agent i bids b_i.

Definition 2.1 *A* decision rule *is a function $A : T^n \mapsto \Omega$ that given a vector b of n bids returns an outcome $\omega \in \Omega$.*
A payment scheme *P is a set of payment functions $P_i : \Omega \times T^n \mapsto \mathbb{R}$, where P_i determines the payment of agent i to the mechanism, given the output ω and the bid vector b.*
A mechanism *$M = (A, P)$ is a combination of a decision rule A and a payment scheme P.*

2.2 Utilities

Each agent has a private value $t_i \in T_i$, which is called its *type*. For simplicity, we will assume that all agents have their types taken from the same space as the bids (signaling) space, T.

Each agent has a valuation function $v_i : \Omega \times T \mapsto \mathbb{R}$ that reflects the utility from an outcome $\omega \in \Omega$, given that the type of the agent is t_i. The agents have a *quasi-linear* utility, meaning that their utility can be shifted linearly by monetary payments. Therefore, the utility of agent i from an outcome ω and a payment p_i is $u_i(\omega, t_i, p_i) = v_i(\omega, t_i) - p_i$.

In some cases we will be interested in the partial derivative of a valuation function by the agent's type. Therefore, for simplicity, we denote ${v_i}' = \frac{\partial v_i}{\partial t_i}$.

2.3 Truthfulness

For truthful mechanisms we will concentrate on payment functions of the form $P_i : \Omega \times T^{n-1} \mapsto \mathbb{R}$, which don't depend on the i-th bid.

Definition 2.2 *Algorithm A* admits a truthful payment *if there exists a payment scheme P such that for any set of fixed bids b_{-i}, and for any two types $s, t \in T$*

$$v_i(\omega_t, t) - P_i(\omega_t, b_{-i}) \geq v_i(\omega_s, t) - P_i(\omega_s, b_{-i}) \quad (1)$$

In this case, A is also called rationalizable*, and the mechanism $M = (A, P)$ is called* truthful.

Literally, a mechanism is truthful if for every agent, reporting its true type as a bid is a weakly dominant strategy. A mechanism is *strongly truthful* if truthtelling is the only dominant strategy.

If an algorithm A is randomized, then it can be viewed as a distribution over deterministic algorithms. Therefore, a randomized mechanism $M = (A, P)$, which is a distribution over truthful mechanisms is also truthful (this property is also referred as *universal truthfulness*). A weaker notion of truthfulness is to carry the randomization into the mechanism, and require truthtelling to always be a best response strategy in expectation only. A mechanism of this type is *truthful in expectation*.

Rochet [23] presented a necessary and sufficient condition for rationalizable decision rules. We refer to a slightly different presentation of this condition, which appears in [10]:

Theorem 2.3 *[23, 10] Given an agent i and having all other bids b_{-i} held fixed, let $G(i, b_{-i}) = (V, E)$ be a weighted directed graph such that $V = T$, $E = T \times T$ and the weight of every edge is $w(s, t) = v_i(\omega_t, t) - v_i(\omega_s, t)$. An allocation algorithm admits a truthful payment if and only if for every agent i and for every vector of fixed bids b_{-i}, the graph $G(i, b_{-i})$ has no finite negative cycles.*

In addition to Theorem 2.3, if an decision rule is rationalizable, the following is a suitable payment function for the i-th agent: For every vector of fixed bids b_{-i}, choose an arbitrary type t_0. The payment from agent i to the mechanism if it bids t is:

$$p(t, b_{-i}) = \inf \left\{ \sum_{i=0}^{k} w(t_i, t_{i+1}) | k \geq 0,\ t_1, \ldots, t_{k+1} \in T, t_{k+1} = t \right\} \quad (2)$$

Literally, the payment function is the infimum on all finite paths from an arbitrary type t_0 to the actual bid t.

Theorem 2.3 does not provide an efficient computational method for testing whether an algorithm is rationalizable. A simpler condition, which should be easier to test, is whether the graph contains a negative cycle of length 2. Formally, a graph $G(i, b_{-i})$ does not have negative cycles of length 2 if and only if for every two types $t, s \in T$,

$$v_i(\omega_t, t) - v_i(\omega_s, t) \geq v_i(\omega_t, s) - v_i(\omega_s, s) \quad (3)$$

This necessary (but not sufficient) condition is referred as *weak monotonicity* [14] or as *2-cycle inequality* [10].

2.4 Single Parameter

Definition 2.4 *An agent i is a* single parameter *agent with respect to Ω if there exists an interval $S_i \subseteq \mathbb{R}$ and a bijective transformation $r_i : T \mapsto S_i$ such that for any $\omega \in \Omega$, the function $\widehat{v_i}(\omega, s_i)$ is continuous and piecewise differentiable in s_i, where $\widehat{v_i}(\omega, s_i) = v_i(\omega, {r_i}^{-1}(s_i))$.*

The purpose of $r_i(\cdot)$ is to overcome the difficulty of having different representations for the same type space. The single parameter property should be indifferent to the chosen representation. For simplicity, we shall assume the type representation allows ignoring $r_i(\cdot)$. For example, if the type space is $\mathbb{R}$ itself, or an interval $T = [t_0, t_1]$, and the utility function is continuous and piecewise differentiable in T for every $\omega \in \Omega$, then $r_i(\cdot)$ is the identity function and can be ignored. We therefore, slightly abuse the notation and assume $v_i = \widehat{v_i}$.

Definition 2.5 *A mechanism (algorithm) is a* mechanism (algorithm) for single parameter agents *if all agents are single parameter.*

3 Halfway Monotone Derivative

The *Mirrlees-Spence condition* [25, 18, 23] characterizes all truthful single parameter mechanisms, where the notion of single parameter is more restricted than the one we use. Assume the output space Ω is a continuous interval in $\mathbb{R}$, the type space T is a segment $T = [t_0, t_1]$. The condition requires that for every b_{-i} we have that $A(\cdot, b_{-i})$ is differentiable and v_i is twice differentiable, and that

$$\forall t, b_i \in T \ \frac{\partial^2}{\partial t \partial A} v_i(A(b_i, b_{-i}), t) > 0 \tag{4}$$

then the mechanism is truthful if and only if the decision rule $A(\cdot, b_{-i})$ is not decreasing in the bid b_i of the agent.

Our condition is a generalization of the Mirrlees-Spence condition, as it relaxes the assumptions on Ω and the differentiability of the valuation and decision functions. It is a sufficient but not necessary condition, which we call *Halfway Monotone Derivative (HMD)* condition. We first define a weaker property called *Interway Monotone Derivative (IMD)*.

Definition 3.1 *A valuation function v_i is* Interway Monotone Derivative (IMD) *if for every fixed bid vector b_{-i}, the following holds: For every two types $s, t \in T$ such that $s < t$, $\forall u \in [s, t]$ $v_i{}'(\omega_s, u) \leq v_i{}'(\omega_t, u)$, except for possibly a countable number of points.*

The set of points where the inequality may not hold is required since the valuation function may be only piecewise differentiable. Clearly, IMD implies weak monotonicity. HMD is defined as a special case of IMD:

Definition 3.2 *A valuation function v_i is* Halfway Monotone Derivative (HMD) *if for every fixed bid vector b_{-i},* one of the following *holds:*

1. *For every two types $s, t \in T$ such that $s < t$, $\forall u \geq s$ we have, $v_i{}'(\omega_s, u) \leq v_i{}'(\omega_t, u)$, except for possibly a countable number of points.*
2. *For every two types $s, t \in T$ such that $s < t$, $\forall u \leq t$ we have, $v_i{}'(\omega_s, u) \leq v_i{}'(\omega_t, u)$, except for possibly a countable number of points.*

The following condition gives a characterization to a family of rationalizable decision rules.

Theorem 3.3 *A single parameter decision rule $A(b) : T^n \mapsto \Omega$ is rationalizable when all valuation functions are HMD.*

Proof: We shall prove only that the first condition of HMD leads to rationalizability, as the proof for the second condition is analogues. Assume by way of contradiction that the condition holds yet A is not rationalizable. By Theorem 2.3 there is a some graph $G(i, b_{-i})$ which has a negative cycle $t_0, t_1, \ldots, t_k, t_{k+1} = t_0$. We shall first show that the graph must also contain a negative 2-cycle, and then infer that the condition is violated. If $k = 1$ then the claim is trivial. Otherwise, let t be the node in the cycle such that $\forall 0 \leq i \leq k \ t \geq t_i$, and let s and u be the nodes in the cycle which have cycle arcs in and from t, respectively. The length of the path from s to u through t is:

$$\begin{aligned} w(s,t) + w(t,u) &= v_i(\omega_t, t) - v_i(\omega_s, t) + v_i(\omega_u, u) - v_i(\omega_t, u) \\ &= v_i(\omega_t, t) - v_i(\omega_t, u) - v_i(\omega_s, t) + v_i(\omega_s, u) - v_i(\omega_s, u) + v_i(\omega_u, u) \\ &= \int_u^t v'(\omega_t, x)dx - \int_u^t v'(\omega_s, x)dx - v_i(\omega_s, u) + v_i(\omega_u, u) \\ &= \int_u^t \left(v'_i(\omega_t, x) - v'_i(\omega_s, x)\right) dx + w(s,u) \geq w(s,u) \end{aligned}$$

The last integral is non negative since $t > u$ and since the difference between the derivatives is positive due to the first condition of HMD. From the inequality we infer that a shorter negative cycle with $k-1$ nodes can be constructed with a shortcut from s to u. By repeating this process, we get that the graph has a negative 2-cycle. Let s and u be the nodes in this cycle, and assume without loss of generality that $s < u$. We infer from IMD (which is implied by HMD), that:

$$\begin{aligned} w(s,u) + w(u,s) &= v_i(\omega_u, u) - v_i(\omega_s, u) + v_i(\omega_s, s) - v_i(\omega_u, s) \\ &= \int_s^u {v'}_i(\omega_u, x)dx - \int_s^u {v'}_i(\omega_s, x)dx \\ &= \int_s^u ({v'}_i(\omega_u, x) - {v'}_i(\omega_s, x))dx \geq 0, \end{aligned}$$

which is in contrast to the cycle being of negative length. □

For some types of valuation functions, HMD is also a necessary condition. The following Theorem proves this property for a simple case where the partial derivative is a constant in the agent's type. (The proofs of the remaining theorems in this section appear in Appendix A.)

Theorem 3.4 *If for every agent i, fixed bid vector b_{-i}, and bid b_i, ${v_i}'(\omega_{b_i}, x)$ does not depend on x, then HMD is a necessary and sufficient condition for rationalizability.*

We now show a simple structure for the payment function in truthful IMD mechanisms (and therefore, also in HMD mechanisms):

Theorem 3.5 *A suitable payment scheme for agent i in a single parameter rationalizable decision rule $A : T^n \mapsto \Omega$ that is IMD is*

$$P_i(t, b_{-i}) = c(b_{-i}) + v_i(\omega_t, t) - \int_{t_0}^t {v_i}'(\omega_x, x))dx \tag{5}$$

Where b_{-i} is held fixed, t_0 is some arbitrary type. (Note that c does not depend on t.)

In many combinatorial problems the outcome space Ω is finite or countable, and combinatorial algorithms have the following property: For each agent i, when b_{-i} is fixed, the type space T can be divided into a finite or countable number of intervals, such that for each interval, the algorithm outputs the same outcome for any bid within the interval. We denote such algorithms as *piecewise continuous*. Let $T_{b_{-i}}$ denote the set of endpoints of these intervals. For any endpoint $s \in T_{b_{-i}}$, let s^- and s^+ denote the intervals whose upper and lower endpoints are s, respectively. Clearly, for any endpoint s, either $s \in s^-$ or $s \in s^+$. Given an interval $[t_1, t_2]$, $T_{b_{-i}}/[t_1, t_2]$ denotes the set of endpoints whose surrounding intervals intersect with $[t_1, t_2]$ (i.e., all the endpoints within (t_1, t_2), and possibly t_1 and t_2, if they are endpoints and $t_1 \in t_1^+$, $t_2 \in t_2^-$. The integral in Theorem 3.5 reduces into an alternative representation, which may be more convenient to implement:

Corollary 3.6 *A suitable payment scheme for agent i in a single parameter piecewise continuous rationalizable decision rule $A : T^n \mapsto \Omega$ that is IMD is*

$$P_i(t, b_{-i}) = c(b_{-i}) + \begin{cases} \sum_{s \in T_{b_{-i}}/[t_0, t]} v_i(\omega_{s^+}, s) - v_i(\omega_{s^-}, s) & t > t_0 \\ \sum_{s \in T_{b_{-i}}/[t, t_0]} v_i(\omega_{s^-}, s) - v_i(\omega_{s^+}, s) & t < t_0 \\ 0 & t = t_0 \end{cases}, \tag{6}$$

where b_{-i} is held fixed, t_0 is some arbitrary type.

Literally, the non constant part of the payment is given by accumulating the values of the transitions between two consecutive outcomes, over all the endpoints between t_0 and t.

Example: To illustrate our payment function consider the following case for player i, when all the other bids b_{-i} are fixed. If $b_i = x \in \mathbb{R}^+$ then $A(x, b_{-i}) = \lceil x \rceil = \omega_{\lceil x \rceil}$. The valuation is $v_i(\omega_{\lceil x \rceil}, t) = t + \lceil x \rceil$. Choosing $t_0 = 0$ and $c(b_{-i}) = 0$, the payment in this case would be $p_i(\omega_{\lceil x \rceil}) = \lceil x \rceil$. If we slightly change the valuation function to $v_i(\omega_{\lceil x \rceil}, t) = t \cdot \lceil x \rceil$ then the payment would be $p_i(\omega_{\lceil x \rceil}) = \sum_{k=1}^{\lceil x \rceil - 1} k = O(x^2)$.

4 Applications of HMD Mechanisms

We show applications of the HMD condition for several single parameter mechanisms. We first apply the theorem on some well known mechanisms, and show that conditions derived for these mechanisms can also be derived as applications of HMD mechanisms. We then present new single parameter mechanisms and demonstrate the usage of the HMD condition for them. (Most of the proofs in this section appear in Appendix B.)

4.1 Single Commodity Auction

For simplicity, we concentrate on single commodity auctions, where each agent has a unit demand. However, the results hold for a more general setup of an auction with known single minded bidders [19], which is a multi commodity combinatorial auction where each agent is interested in a specific bundle of commodities, which is publicly known.

In an auction, an agent's type is a private value t_i, which is the value the agent associates with a good that it is bidding on. From the point of view of a specific agent, there are two possible outcomes: winning and losing. When winning, the value of the agent is t_i, and in the losing outcome, the value is 0.

The second price auction [26] is a classical method for auctioning a single commodity: The highest bidder wins and pays the bid of the second highest bid. Although it is a popular example, it is not the only truthful mechanism for auctioning.

A well known result on rationalizable deterministic auctions is the existence if a *critical value* for each bidder, unless the bidder has no winning bid. A bidder's critical value is determined by the bids of the other agents. The bidder wins if its bid is above the critical value and loses if the bid is under it. Given that losing bidders pay 0, the winning bidders pay exactly their critical value. The second price auction is truthful since the critical value of each bidder is the highest bid among the other bidders.

We show that the HMD condition is equivalent to the critical value condition in auctions, and show how the HMD condition extends to randomized auctions:

Claim 4.1 *A deterministic auction fulfills the critical value condition iff it fulfills HMD.*

In randomized auctions, the relevant outcome for an agent is a probability $0 \leq p \leq 1$ for winning the good. The randomized auctioning can be viewed as a distribution over deterministic auctions. If for each and every one of these deterministic auctions we have a critical value, then we have a universal truthfulness. Otherwise, we can settle with truthfulness in expectation. In this case, the following holds:

Claim 4.2 *A randomized auction is rationalizable iff each bidder's probability for winning is weakly monotone in its bid.*

4.2 Machine Scheduling

A central mechanism assigns n jobs to m machines, with the goal of minimizing the *makespan*, which is the longest completion time (also known as $Q\|C_{max}$). Machines are related by speeds $s_1, \ldots, s_m$, and the jobs have weights $w_1, \ldots, w_n$. Therefore, running the i-th job on the j-th machine requires $\frac{w_i}{s_j}$ time. It is sometimes more convenient to refer to a machine's cost per unit of work $c_j = \frac{1}{s_j}$ rather than to its speed.

If all speeds and weights are known to the mechanism, then it is known that the optimization problem is NP-Complete [11], however there is a $PTAS$ to this problem [13]. Since the problem is strongly NP hard, this is the best approximation result possible, unless $P = NP$. In case the number of machines is constant then there exists an $FPTAS$ to the problem [12].

4.2.1 Mechanism Design for Machine Scheduling

If the machines' speeds are known privately to the owners of the machines, then the owners (agents) may report false values to the mechanism in order to lower their costs. A machine j incurs a cost proportional to the amount of work assigned to it, i.e. $C_j = (\sum_{i \in I_j} w_i) c_j$, where I_j is the set of jobs assigned to machine j.

Archer and Tardos [4] have shown that a scheduling algorithm is rationalizable if and only if it is monotone. Monotonicity means the amount of work assigned to a machine cannot decrease if it raises its speed, while the rest of the inputs remain constant. Although traditional algorithms for job assignment are not monotone, several monotone algorithms have been given to this problem.

There is a mechanism that solves the optimization problem optimally and is monotone. Obviously, its running time is not polynomial. However, there is a 2-approximation polynomial time randomized mechanism that is monotone in expectation, and therefore achieves truthfulness in expectation [5] (this result improves a previous 3-approximation [4]). There is also a 5-approximation polynomial time deterministic mechanism that is truthful [1]. If the number of machines is constant, there is a monotone $FPTAS$ [1] (this result improves a previous $4+\epsilon$-approximation [6]).

The monotonicity condition holds for any problem where the agents have a cost function which is linear in the amount of work assigned to them. This has been demonstrated in [4] for strategic versions of various allocation problems such as maximal flow and facility location. We show that the monotonicity condition is equivalent to HMD.

Claim 4.3 *A scheduling algorithm is monotone iff it is HMD.*

By Theorem 3.5, when choosing $t_0 = \infty$ (or any large enough type that assures a zero assignment) and $c = 0$, the payment from the machine to the mechanism is $P_j(b_j, b_{-j}) = -b_j A_j(b_j, b_{-j}) - \int_{\infty}^{b_j} [-A_j(x, b_{-j})]dx = -b_j A_j(b_j, b_{-j}) - \int_{b_j}^{\infty} A_j(x, b_{-j})dx$, where $A_j(x, b_{-j})$ is the amount of work assigned to machine j when the bid vector is (x, b_{-j}). Since the payment is negative, the mechanism actually pays the machine $-P_j$, which is exactly the payment function derived in [4].

The natural dual problem for strategic machine scheduling is scheduling with strategic job owners which is discussed in Section 5.

4.3 Scheduling with Deadlines

The following problem is an example for single parameter agents, where the derivative of the valuation function is not a constant (but still simple): n agents apply to get service from a central mechanism. An agent's type is a deadline $t_i \in \mathbb{R}^+$, which it must be served by to have a positive valuation. The output is a service time $\omega_i \in \mathbb{R}^+ \cup \{-\infty\}$. The valuation function of an agent is $v_i(\omega_i, t_i) = \max\{0, t_i - \omega_i\}$, i.e., it linearly decreases until it drops to 0 at the deadline, and then remains fixed. The infinity outcome represents the case where an agent is never served.

Since the HMD condition relates only to the valuation function, we don't consider here the objective function of the mechanism, or the set of feasible outcomes, which are needed for the algorithmic design, but settle on deriving the rationalizability condition:

Theorem 4.4 *Given that a server never serves an agent after its declared deadline, then it is rationalizable if and only if for each agent, either $\omega_i = \infty$ for every b_i, or it has a critical time c_i such that if $b_i < c_i$ then $\omega_i = \infty$, and if $b_i > c_i$ then $\omega < c_i$.*

The assumption that an agent is never served after its deadline simplifies the condition for rationalizability. It is also a reasonable assumption since a cooperative agent is indifferent between late service and denial of service.

4.4 Auctions with Budget Constraints

A budget constraint is a special case of submodular [15] valuations. Given n items and m agents (bidders), p_{ij} denotes the valuation of the i-th agent for the j-th item. Each agent has also a budget constraint t_i. Given a bundle of items I, the valuation of the i-th agent of the bundle is $v_i(I, t_i) = \min\{t_i, \sum_{j\in I} p_{ij}\}$. For simplicity we assume that $\max_j\{p_{ij}\} \leq t_i \leq \sum_j p_{ij}$. We also assume that the allocation algorithm does not have to allocate all of the items. We define the objective function of the mechanism to be the total valuation of all agents.

This optimization problem is known to be NP-Complete, but has several approximations: A simple greedy algorithm gives a 2-approximation for a wider class of valuations [15], an LP-rounding gives a 1.58-approximation, and there is a PTAS when the number of bidders is constant [2].

4.4.1 Strategic Budgets

The following is a single parameter problem: Assume that all valuations are known, but the budgets are privately held by the agents. Although this is a somewhat artificial setting, it demonstrates the HMD condition well: We will show that the valuation function is very similar to the valuation function in Section 4.3, and the rationalizability conditions are different.

Definition 4.5 *An allocation algorithm for auctions with budget constraints is* piecewise monotone *if for every agent i and every budget t_i such that the outcome ω has $v_i(\omega, t_i) = t_i$, bidding any value above $t > t_i$ will output an allocation ω', such that $v_i(\omega', t_i) = t_i$.*

Literally, piecewise monotonicity requires the total value of allocated items to be weakly monotone if it exceeds the budget constraint. When the budget constraint is not reached, the total value does not have to be monotone but cannot go below the most recent budget constraint that was exceeded.

Theorem 4.6 *Any piecewise monotone allocation rule is rationalizable.*

Proof: From the point of view of the i-th agent, the exact allocation is not important, only the total value of items assigned to him (denoted by ω), and the budget constraint. Having ω fixed, $v_i(\omega, t_i) = t_i$ if $\omega < t_i$ and $v_i(\omega, t_i) = \omega$ otherwise. The derivative is therefore 1 if $\omega < t_i$ and 0 otherwise.

We now show that piecewise monotonicity leads to HMD: Given a budget constraint b_0, first assume that the outcome ω_0 does not exceed b_0 in its total value. For any type $t > b_0$, ${v_i}'(\omega_0, t) = 0$, so b_0 does not induce any constraints on the outcomes for larger bids. If ω_0 exceeds b_0 in its total value, then the derivative is 1 until ω_0, therefore to fulfill the HMD condition, for any type $t > b_0$ the outcome must be at least ω_0. This is achieved due to weak monotonicity, and therefore we also have the HMD condition, which by Theorem 3.3 is a sufficient condition for rationalizability. $\square$

We note that unlike previous examples, HMD is not a necessary condition here. For example, consider VCG [26, 8, 9] mechanisms. Since the objective function is to maximize the total valuation, these mechanism are truthful (and therefore, any algorithm that outputs an optimal allocation is rationalizable). If all the budgets are low such that all budget constraints are exceeded and there are still redundant items left, VCG does not mind how these items are allocated. Therefore, by slightly changing one of the budgets, it is possible to choose a different allocation of the redundant items that is optimal, yet violates the piecewise monotonicity property.

Since finding the optimal allocation is NP-Hard, we show that there are piecewise monotone (hence, rationalizable) allocation algorithms that output approximate allocations. The approximation algorithms for the non-strategic form of the problem [2, 15] are not piecewise monotone. We give in Appendix C approximation algorithms that are also examples of two special subcases of piecewise monotone algorithms:

Corollary 4.7 *Any allocation algorithm that never exceeds the budget constraint is rationalizable.*

Corollary 4.8 *Any allocation algorithm that the total value of items it allocated to each agent is weakly monotone in its budget constraint is rationalizable.*

5 Ex-Post Equilibrium

The theory of mechanism design assumes that the valuation function of each agent depends on its true type and on the output of the mechanism, which may depend on the bids of the other agents. The true types of all other agents, have no effect on the valuation function, once the outcome is fixed. In such setups, it is reasonable to try and construct a mechanism where truthtelling is a dominant strategy, since each agent is invariant to whether the other agents bid truthfully or strategically.

This assumption fails to hold when valuation functions are affected by the true types of other bidders and not only their actual bids and the mechanism outcome. When an agent lies about its type, it not only affects the outcome, but also inserts great uncertainty into the system, since the valuations of the other agents depend on the true type of that agent.

Since having truthtelling be a dominant strategy may be an unreasonable aim, we settle on designing mechanisms which are ex-post truthful, meaning that there is an equilibrium where all agents reveal their true type. This is a weaker requirement than dominance, which is necessary due to the complexity of the valuation functions, which depend on the private parameters of the other agents. Formally, the valuation function of an agent is now $v_i : \Omega \times T^n \mapsto \mathbb{R}$. The outcome space is now $\Omega \times T^{n-1}$ instead of Ω, i.e. a combination of the algorithm's decision and the true types of the other agents. We define an *ex-post truthful mechanism* as follows:

Definition 5.1 *A mechanism $M = (A, P)$ is* ex-post truthful *if for every agent i, for every fixed set of bids for the other agents b_{-i} and for every types $s, t \in T$*

$$v_i(A(t, b_{-i}), (t, b_{-i})) - P_i(A(t, b_{-i}), b_{-i}) \geq v_i(A(s, b_{-i}), (t, b_{-i})) - P_i(A(s, b_{-i}), b_{-i}) \quad (7)$$

Literally, a mechanism is ex-post truthful if any setting where all agents report their true type is a Nash Equilibrium. For any agent, assuming that all other agents report their true values, its valuation function reduces to a function that does not depend directly on the other agents' types, as those are assumed to be equal to the bids, and are now part of the extended outcome space. Let $\Omega' \subset \Omega \times T^{n-1} = \{(\omega, b_{-i}) | \exists b_i,\ A(b_i, b_{-i}) = \omega\}$ denote the set of possible outcomes if indeed the other agents bid their types. Replacing Ω with Ω', we can now extend the HMD and IMD conditions for valuation functions which depend on the types of the other agents, and give a ex-post truthful characterization and payment functions similar to Theorems 3.3 and 3.5.

Theorem 5.2 *If a decision rule A supports the HMD condition then $M = (A, P)$ is an ex-post truthful mechanism, where*

$$P_i(t, b_{-i}) = c(b_{-i}) + v_i(A(t, b_{-i}), (t, b_{-i})) - \int_{t_0}^{t} v_i{}'(A(x, b_{-i}), (x, b_{-i}))dx$$

HMD is a necessary condition if for each agent, the partial derivative of the valuation function by the agent's type does not depend on the type.

5.1 Scheduling with Strategic Job Owners

Consider a machine scheduling setting where machine speeds are publicly known, but the job owners have to report their weights. Each job owner has a negative utility from waiting until the completion of the job (we assume that there is also a positive utility from having the job executed, which would motivate job owners to participate, but we ignore this issue here). The valuation of having a job of weight w_i executed on a machine with a cost per unit of c_j (i.e., a speed of $s_j = 1/c_j$) is $v_i = -c_j(W_{ij} + w_i)$, where W_{ij} denotes the total weight of jobs that are executed on machine j, not including the i-th job. Clearly, W_{ij} depends on the actual weights, which are unknown to the mechanism, and only assumed to be equal to the bids.

First, we like to show that unless the scheduling algorithm is degenerate, there is no dominant strategy for the job owners. A degenerate scheduler is a scheduler that always assigns the same jobs together on the same machine (or on a machine with the same speed), regardless of their bids, given that the jobs are assigned on the same machine for some bid. Clearly, degenerate scheduling may be considerably inefficient.

Assume that given the bids of the other agents, if a job owner bids its true type b_1 then it is assigned to machine j_1 together with the jobs in set S_1, and if the bid is b_2 then the assignment is machine j_2 with the set S_2 of jobs. Assume that there is a job $i \in S_1$ such that i is not in S_2. The value of the first outcome is $V_1 = -(b_1 + W_{1j_1})/s_{j_1}$, and the value of the second outcome is $V_2 = -(b_1 + W_{1j_2})/s_{j_2}$. However, since the actual weight of the i-th job is unknown, $V_2 - V_1$ is unbounded, and therefore no payments p_1, p_2 exists such that $V_1 - p_1 \geq V_2 - p_2$, for every possible weight of the i-th job. A similar impossibility result can be shown if $S_1 = S_2$ but are non empty, and $s_{j_1} \neq s_{j_2}$. This motivates searching for mechanisms that are ex-post truthful.

Observe that the partial derivative is of the i-th valuation function is $v_i{}' = -c_j$. By Theorem 5.2, the necessary and sufficient HMD condition is that when a job raises its declared weight (having all other jobs fixed and bidding truthfully), the job cannot move to a slower machine. Surprisingly, the constraint does not depend on W_{ij}. This is because if indeed all other agents bid truthfully, then $c_j W_{ij}$ is known to the mechanism and its effect can be reversed by setting the payment appropriately.

Follows is a construction of a fractional job assignment from [4], which is used to both lower bound the makespan of the optimal integral assignment, and serves as a starting point for constructing rationalizable scheduling algorithms for strategic machines [4, 5, 1].

1. Sort the jobs in non increasing order
2. Set a threshold of maximal load
3. Assign the jobs to machines, from the fastest to the slowest, splitting a job whenever the threshold is reached

The threshold that serves as a lower bound for the optimal integral allocation is the smallest value that is large enough such that there is enough volume to allocate all the jobs, and no fraction of a job is allocated to a machine that cannot run the full job within the threshold limit. It appears that this threshold can be computed efficiently, in polynomial time.

By rounding each fractionally assigned job towards the faster machine we get a 2-approximation, since each machine gets at most one extra fraction, and the load it incurs is lower than the threshold. Although this approximation is not rationalizable for strategic machines, it can be used to construct an ex-post mechanism for strategic job owners (The proofs of Theorems 5.3 and 5.4 appear in Appendix D).

Theorem 5.3 *There exists an ex-post mechanism for machine scheduling with strategic job owners that is a* 2*-approximation*

The scheduling mechanism described above is polynomial in its running time, since the assignment algorithm runs in polynomial time, and the payment function requires repeating the algorithm for each job in all the breakpoints that cause the job to change a machine, or change the jobs that are assigned with it to the same machine. It can be shown that finding the next breakpoint can be accomplished in polynomial time, as well as the number of breakpoints until the job is so large it has to be assigned alone to the fastest machine.

When only the machines are strategic, there exists a (non-polynomial) truthful mechanism that is optimal [4]. However, this is not the case when jobs are controlled by agents, even if the machines' speeds are publicly known.

Theorem 5.4 *No ex-post truthful mechanism for machine scheduling on uniformly related machines with strategic job owners has an approximation ratio lower than* 1.19*.*

If both jobs and machines are strategic, then any ex-post truthful mechanism must fulfill a dual HMD condition, for both jobs and machines.

Among the several rationalizable approximation algorithms for strategic machines, the 5 approximation in [1] is also monotone from the jobs' perspective, and therefore can be part of an ex-post truthful mechanism for strategic machines and job owners. Using similar ideas to that of [1] one can derive the following theorem. (A short discussion on monotone scheduling algorithms appears in Appendix E.)

Theorem 5.5 *There exists an ex-post truthful mechanism for uniformly related machine scheduling, for both strategic machines and job owners, which achieves a* 5*-approximation.*

References

[1] N. Andelman, Y. Azar and M. Sorani. Truthful Approximation Mechanisms for Scheduling Selfish Related Machines. In *22nd Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 69–82, 2005.

[2] N. Andelman and Y. Mansour. Auctions with Budget Constraints. In *9th Scandinavian Workshop on Algorithm Theory (SWAT)* pages 26–38, 2004

[3] A. Archer, C. Papadimitriou, K. Talwar and E. Tardos, An Approximate Truthful Mechanism for Combinatorial Auctions with Single Parameter Agents. In *Proc. 14th Annual ACM Symposium on Discrete Algorithms (SODA)*, 2003.

[4] A. Archer and E. Tardos. Truthful mechanisms for one-parameter agents. In *Proceedings of the 42nd IEEE symposium on Foundations of Computer Science*. IEEE Computer Society, 2001.

[5] A Archer. *Mechanisms for Discrete Optimization with Rational Agents*. PhD thesis, Cornell University, 2004.

[6] V. Auletta, R. De Prisco, P. Penna, and P. Persiano. Deterministic truthful approximation mechanisms for scheduling related machines. In *Proc. 21th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 608–619, 2004.

[7] C. Chekuri and S. Khanna. A PTAS for the Multiple Knapsack Problem. In *11th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 213–222, 2000.

[8] E. H. Clarke. Multipart Pricing of Public Goods. In *Public Choice*, pages 17–33, 1971.

[9] T. Groves. Incentives in Teams. In Econometrica, pages 617–631, 1973.

[10] H. Gui, R. Müller and R. V. Vohra. Dominant Strategy Mechanisms with Multidimentional Types. Working paper.

[11] M. R. Garey and D. S. Johnson. *Computers and Intractability: a Guide to the Theory of NP-completeness*. Freeman, San-Francisco, 1979.

[12] E. Horowitz and S. Sahni. Exact and Approximate Algorithms for Scheduling Nonidentocal Processors. In *Journal of the Association for Computing Machinery 23*, pages 317-327, 1976.

[13] D. Hochbaum and D. Shmoys. A polynomial approximation scheme for scheduling on uniform processors: Using the dual approximation approach. *SIAM Journal on Computing*, 17(3):539–551, 1988.

[14] R. Lavi, A Mu'alem and N. Nisan. Towards a Characterization of Truthful Combinatorial Auctions. In *44th Symposium on Foundations of Computer Science (FOCS)*, pages 574–583, 2003.

[15] B. Lehmann, D. Lehmann and N. Nisan, Combinatorial Auctions with Decreasing Marginal Utilities. In *3rd ACM Conference on Electronic Commerce (EC)*, pages 18-28, 2001.

[16] J. K. Lenstra, D. B. Shmoys and E. Tardos. Approximation Algorithms for Scheduling Unrelated Parallel Machines. In *Mathematical Programming* 46, pages 259-271, 1990.

[17] A. Mas-Collel, W. Whinston and J. Green. *Microeconomic Theory*. Oxford University press, 1995.

[18] J. A. Mirrlees. Optimal Tax Theory: A Synthesis. In *J. of Public Economics* 6:327–358, 1976.

[19] A. Mu'alem and N. Nisan. Truthful Approximation Mechanisms for Restricted Combinatorial Auctions. In *18th National Conf. on Artificial Intelligence and 14th Conf. on Innovative Applications of Artificial Intelligence (AAAI/IAAI)*, pages 379–384, 2002.

[20] N. Nisan and A. Ronen. Algorithmic Mechanism Design. In *31st ACM Symp. on Theory of Computing (STOC)*, pages 129–140, 1999.

[21] N. Nisam and A. Ronen. Computationally Feasible VCG Mechanisms. In *2nd ACM conference on Electronic commerce (EC)*, pages 242–252, 2000.

[22] K. Roberts. The Characterization of Implementable Choice Rules. In J. J. Laffont, ed., *Aggregation and Revelation of Preferences. Papers presented at the first European Summer Workshop of the Econometric Society*, pages 321–348. North-Holland, 1979.

[23] J. C. Rochet. A Necessary and Sufficient Condition for Rationalizability in a Quasi-Linear Context. In *J. of Mathematical Economics*, 16:191–200, 1987.

[24] D. B. Shmoys and E. Tardos. An Approximation Algorithm for the Generalized Assignment Problem. In *Mathematical Programming*, 62:461–474, 1993.

[25] M. Spence. Competitive and Optimal Responses to Signals: An Analysis of Efficiency and Distribution. In *J. of Economic Theory*, 7:296–332, 1974.

[26] W. Vickrey. Counterspeculation, Auctions and Competiitve Sealed Tenders. In *J. of Finance*, pages 8–37, 1961.

A Proofs from Section 3

Proof of Theorem 3.4: Sufficiency is by Theorem 3.3, so it remains to prove necessity. Assume by the way of contradiction that HMD is violated. Then there is an agent i, a bid vector b_{-i} and types $s < t$ such that $v_i'(\omega_s, x) > v_i'(\omega_t, x)$ for some x. Since both functions don't depend on x, the inequality holds for every x, and specifically in the interval $[s, t]$. By integrating both sides of the inequality, we get:

$$\int_s^t v_i'(\omega_s, x)dx \quad > \quad \int_s^t v_i'(\omega_t, x)dx \tag{8}$$

$$v_i(\omega_t, s) - v_i(\omega_s, s) \quad > \quad v_i(\omega_t, t) - v_i(\omega_s, t) \tag{9}$$

Which is a violation of weak monotonicity. Hence, the decision rule cannot be rationalizable. □

Proof of Theorem 3.5: The following proof holds for $t \geq t_0$. The proof for $t \leq t_0$ is analogous. By Theorem 2.3, a suitable payment function is

$$P_i(t, b_{-i}) = \inf\left\{\sum_{i=0}^{k} w(t_i, t_{i+1}) | k \geq 0,\ t_1, \ldots, t_{k+1} \in T, t_{k+1} = t\right\}$$

We first prove that for any given path, adding an intermediate node inside an arc cannot make it longer. Let (t_i, t_{i+1}) be an arc in a given path. Assume $t_{i+1} > t_i$ and let $s \in [t_i, t_{i+1}]$ be an intermediate node. We have from IMD that

$$\begin{aligned} w(t_i, s) \quad + \quad & w(s, t_{i+1}) = v_i(\omega_s, s) - v_i(\omega_{t_i}, s) + v_i(\omega_{t_{i+1}}, t_{i+1}) - v_i(\omega_s, t_{i+1}) \\ = \quad & -\int_s^{t_{i+1}} v'_i(\omega_s, x)dx + \int_s^{t_{i+1}} v'_i(\omega_{t_i}, x)dx + v_i(\omega_{t_{i+1}}, t_{i+1}) - v_i(\omega_{t_i}, t_{i+1}) \\ = \quad & \int_s^{t_{i+1}} \left(-v'_i(\omega_s, x) + v'_i(\omega_{t_i}, x)\right) dx + w(t_i, t_{i+1}) \leq w(t_i, t_{i+1}) \end{aligned}$$

A similar claim holds when $t_{i+1} < t_i$. For every arc leaving or entering the interval $[t_0, t]$ we add an intermediate node at either t_0 or t (depending on which one of the endpoints of the interval is an intermediate node). For arcs that skip the entire interval $[t_0, t]$ we add both t_0 and t as intermediate point. The resulting path is not longer, and all the nodes outside the interval $[t_0, t]$ are contained in cycles, starting and ending at either t_0 or at t. Since there are no negative cycles in the graph, we can remove the cycles and end with a path that uses only nodes within the interval $[t_0, t]$.

In a similar technique we remove all the backward arcs (t_i, t_{i+1}) such that $t_{i+1} < t_i$. Let $j > i$ be the index of the first node t_j that appears after t_{i+1} such that $t_j \geq t_i$. The arc that ends at t_j must be a forward arc (i.e. $t_{j-1} < t_j$), since by definition of j, $t_{j-1} \leq t_i$. By adding t_i as an intermediate node within the arc (t_{j-1}, t_j), we get a cycle from t_i to itself, which can be removed. Removing the cycle also removes the backward arc (t_i, t_{i+1}).

We now have that the payment is the infimum on paths from t_0 to t that only use forward arcs. By adding intermediate points to all arcs, we get that

$$\begin{aligned} P_i(t, b_{-i}) &= \lim_{\Delta \to 0} \left(\sum_{i=0}^{[(t-t_0)/\Delta]-1} w(t_0 + i\Delta, t_0 + (i+1)\Delta) \right) \\ &= \lim_{\Delta \to 0} \left(\sum_{i=0}^{[(t-t_0)/\Delta]-1} v_i(\omega_{t_0+(i+1)\Delta}, t_0 + (i+1)\Delta) - v_i(\omega_{t_0+(i+1)\Delta}, t_0 + i\Delta) \right) \\ &= v_i(\omega_t, t) - v_i(\omega_{t_0}, t_0) - \lim_{\Delta \to 0} \left(\sum_{i=0}^{[(t-t_0)/\Delta]-1} v_i(\omega_{t_0+(i+1)\Delta}, t_0 + i\Delta) - v_i(\omega_{t_0+i\Delta}, t_0 + i\Delta) \right) \\ &= v_i(\omega_t, t) - v_i(\omega_{t_0}, t_0) - \lim_{\Delta \to 0} \left(\sum_{i=0}^{[(t-t_0)/\Delta]-1} \int_{t_0+i\Delta}^{t_0+(i+1)\Delta} v_i'(\omega_{t_0+i\Delta}, x)dx \right) \\ &= v_i(\omega_t, t) - v_i(\omega_{t_0}, t_0) - \int_{t_0}^{t} v'(\omega_x, x)dx \end{aligned}$$

Since $v_i(\omega_{t_0}, t_0)$ does not depend on t it can be replaced by any other function $c(b_{-i})$ without affecting the truthfulness. □

B Proofs from Section 4

Proof of Claim 4.1: In a deterministic auction, when winning, the value of the i-th agent is t_i, and the derivative is 1. In the losing outcome, the value is the constant 0, and so is the derivative. Since both derivatives are constant, by Theorem 3.4, HMD is a necessary and sufficient condition.

For any type t_i the derivative of the winning outcome is higher than the losing outcome. Having b_{-i} fixed, all deterministic HMD mechanisms must either decide that the i-th agent never wins, or have a critical value c_i, for which the i-th agent loses if it's type is lower than c_i, and wins if it is higher (in case of equality, both outcomes are acceptable). By Corollary 3.6, the payment in the winning outcome should be c_i plus the payment in the losing outcome (which is usually set to 0. However, this is not a requirement for achieving truthfulness). □

Proof of Claim 4.2: If the probability for winning is p the expected valuation of an agent with type t is pt, and the derivative is p. In order to have HMD, the probability for winning has to be weakly monotone in the valuation. Since for a given probability p the derivative is constant, HMD is a necessary and sufficient condition. □

Proof of Claim 4.3: The value of an assignment to a machine j is $v_j = -c_j W_j$, where $W_j = \sum_{i \in I_j} w_i$. Deriving by the cost per unit c_j, the derivative is $-W_j$. The monotonicity criteria immediately follows: Since $-W_j$ does not depend on the type c_j, by Theorem 3.4 HMD is a sufficient and necessary condition, and requires that $-W_j$ would increase if the reported cost increases, which means that when a machine reports a slower speed (higher cost), its load decreases ($-W_j$ becomes less negative). □

Proof of Theorem 4.4: Sufficiency is proved by showing the HMD condition from Theorem 3.3: If $\omega_i = \infty$, then v_i is 0 for every t_i, and so is the derivative $v_i{}'$. If ω_i is finite, then $v_i(\omega_i, t_i) = t_i - \omega_i$ if $t_i > \omega_i$ and 0 otherwise. The derivative is therefore 1 if $t_i > \omega_i$, 0 if $t_i < \omega_i$, and undefined for $t_i = \omega_i$.

If an agent is never served, then the derivative is always 0 and HMD holds trivially. Assume that for some deadline t_0, the agent is served at $\omega_0 \leq t_0$. Therefore, $v_i{}'(\omega_0, t_0) = 1$ for every $t > t_0$, which means that if an agent is served before or at t_0 for any bid $b_i > t_0$, then the HMD condition holds. This implies that either the agent is never served, or that it is served before some critical time, given that the deadline is after it.

Necessity of HMD (under the constraint of never serving after the declared deadline), is since the violation of the critical time also violates weak monotonicity: Let b_0 and b_1 be two deadlines, and let ω_0 and ω_1 be the service times. Additionally, assume that $\omega_0 \leq b_0 < b_1$ and that $b_0 < \omega_1$. If $\omega_1 = \infty$ then $v_i(\omega_1, b_1) - v_i(\omega_0, b_1) = \omega_0 - b_1 < \omega_0 - b_0 = v_i(\omega_1, b_0) - v_i(\omega_0, b_0)$. If $b_0 < \omega_1 \leq b_1$ then $v_i(\omega_1, b_1) - v_i(\omega_0, b_1) = \omega_0 - \omega_1 < \omega_0 - b_0 = v_i(\omega_1, b_0) - v_i(\omega_0, b_0)$. □

C Piecewise Monotone Algorithms

Under the restriction of never exceeding the budget constraint, the allocation problem reduces to a subcase of the MAX-GAP (Generalized Assignment Problem). MAX-GAP is a generalization of the Multiple Knapsack problem [7], where each item has a size and profit, and the items are packed into bins (knapsacks) of limited capacity. The goal is to maximize the total profit of the packed items, under the capacity constraints. In the MAX-GAP problem, each item has a different size and profit for each bin. MAX-GAP is NP-Hard to solve, but there is a 2-approximation, based on a reduction to MIN-GAP [24], which is a similar problem where items have a cost instead of a profit. An instance to an auction with budget constraints is reduced to a MAX-GAP instance by having a bin for each agent, with a capacity equal to the budget constraint of that agent. The value of an assigning the j-th item to the i-th item is both the size and the profit of packing the j-th item in the i-th bin. The 2-approximation of the MAX-GAP problem is preserved under the auction with budget constraint problem, since the approximation is in comparison to a fractional assignment, which is optimal in both problems. By Corollary 4.7, this approximation algorithm is also rationalizable. Choosing that an agent with an empty assignment won't pay the mechanism, the payment function in Corollary 3.6 reduces to paying the exact value of the allocated items.

Input : a matrix $\{p_{ij}\}$ of values of items for agents and a bid vector $b = (b_1, b_2, .., b_m)$ of budget constraints, such that $b_i \geq p_{ij}\ \forall i, j$

Output: An allocation $\{S_i\}_{i=1}^m$ of items to agents

1. $I \leftarrow \{1, 2, \ldots, m\}$, $S_i \leftarrow \emptyset$ $1 \leq i \leq m$
2. For $j \leftarrow 1$ to n;
   (a) If $I = \emptyset$ Return $\{S_i\}_{i=1}^m$
   (b) $i \leftarrow \arg\max_{i \in I}\{p_{ij}\}$
   (c) $S_i \leftarrow S_i \cup \{j\}$
   (d) If $\sum_{j \in S_i} p_{ij} > b_i$ Then $I \leftarrow I \setminus \{i\}$
3. Return the allocation $\{S_i\}_{i=1}^m$

Algorithm 1: Algorithm Lazy Greedy

Algorithm *Lazy Greedy* outputs a weakly monotone assignment and requires only $O(mn)$ operations, which is linear in the input size. Lazy Greedy allocates each item to the highest bidder, ignoring the budget constraint until after the assignment. If the budget constraint of an agent has been exceeded, then the agent is removed from the rest of the allocation process. Ignoring the budget constraint until after the allocation is required for weak monotonicity: An agent that raises its budget constraint will receive at least the same items. By Corollary 4.8, this implies rationalizability. When the budget is raised to a critical value needed to win another item, the extra valuation of this item is zero. Therefore, the payment function in Corollary 3.6 reduces to a constant payment, regardless of the actual assignment.

It remains to show that Lazy Greedy is a outputs a reasonable approximation. In the analysis, we use the following definition:

Definition C.1 *The* virtual profit *of an agent is the difference between the total value of the items assigned and their valuation, i.e.* $\sum_{j \in S_i} p_{ij} - v_i(S_i, t_i)$. *Similarly, the virtual profit of an allocation is the sum of virtual profits over all agents.*

We now prove several properties on the virtual profit:

Lemma C.2 *If all agents bid truthfully, the virtual profit of an allocation is at most the total value of the allocation*

Proof: An agent who hasn't exceeded its budget constraint does not have a virtual profit. Once an agent exceeds its budget constraint, it is not allocated any more items, and therefore the virtual profit of an agent is at most the value of the last item allocated to it, which is at most equal to the agent's budget. The actual profit of the agent is exactly the budget constraint, since it has been exceeded. Since the claim holds for each agent separately, it holds for the entire allocation, too. □

Lemma C.3 *If all agents bid truthfully, the value of the optimal allocation is at most twice the value of Lazy Greedy plus its virtual profit.*

Proof: We split each item that increases the virtual profit into two items with a total value equal two the original item, such that the first fraction exactly fills the budget constraint of the agent who receives it. We set the valuations of the other agents for these fractions proportionally to the proportions chosen for the winning agent. For the altered input, Lazy Greedy outputs an allocation with the same value, while the optimum can only improve from the change.

We can now split the items into two groups: The first group includes items that allocating them is within the constraints of the budget, and items that are not allocated at all. The second group includes the items that contribute entirely to the virtual profit. We observe how an optimal allocation performs with respect to Lazy Greedy, by observing the allocation of the items in the first group before the items in the second group.

By the end of the allocation of the items in the first group, the total value of the optimal allocation is at most twice two total value of Lazy Greedy. The proof is by induction on the number of items in this group. If the first group is empty then the claim trivially holds. Assuming the claim holds for $n-1$ items, if there are n items then let i denote the agent who wins the first item (according to Lazy Greedy), and let p denotes its bid. by allocating the first item to highest bidder, Lazy greedy increases its marginal value by at least the increase in the optimal allocation (the budget constraint is not exceeded, and therefore the marginal increase equals p). Allocating the item decreases the available budget of agent i by p. By changing the decision of the optimal allocation to the same agent, the marginal value from this item cannot decrease, but the total value of the rest of the allocation can decrease by at most p, due to the decrease in the available budget. Therefore, the value of the Lazy Greedy allocation is p plus the value of the rest of the allocation, while the value of the optimal allocation is at most $2p$ plus the value of an optimal allocation of the rest of the items, given that the budget of agent i is decreased by p. By the inductive claim, an optimal allocation achieves on the rest of the items in the first group at most twice than the rest of the value of Lazy Greedy, and therefore the claim holds.

After allocating the items in the first group, the optimal allocation has been changed to make the same allocations as Lazy Greedy, but compensated by doubling its value. Lazy Greedy allocates the items to agents who have already reached their budget constraint, and achieves for each item a virtual profit which is larger than the bids of agents who haven't reached their budget constraints yet. For each item in the second group, the altered optimal allocation increases its value by at most the virtual profit achieved by Lazy Greedy for this item, since otherwise Lazy Greedy would have chosen a different agent for this item. This phase contributes at most the virtual profit of Lazy Greedy to the value of the altered optimal allocation. Therefore, the value of the optimal allocation is bounded by twice the value of Lazy Greedy plus its virtual profit. □

We now prove a tight bound on the approximation ratio of Lazy Greedy:

Theorem C.4 *The approximation ratio of Lazy Greedy is* 3

Proof: The upper bound is directly derived from Lemmata C.2 and C.3. The following setup gives a tight lower bound:

Given 3 agents and 3 items, let the price vectors of the agents be $(1,1,1)$, $(1,0,0)$ and $(0,1,0)$, and let all budget constraints be 1. An optimal allocation allocates one item to each agent, achieving a value of 1 for each agent and a total value of 3. Lazy Greedy allocates the first two items two the first agent (with a total value of 1) and the third item to one of the other agents (who bid 0 for it). The total value of this allocation is 1. We note that one can avoid the tie breaks in this example by slightly perturbing the bids, and achieving a ratio that asymptotically reaches 3. □

One may argue that both algorithms presented here give a weak incentive to cooperation, since the mechanisms are not strongly truthful. There are strategies different than truthtelling, that may change the allocation, without affecting the utility (the importance here is on changing the allocation, since small perturbations in the bids that never affect the outcome can hardly be considered as different strategies). In the MAX-GAP based mechanism, since the payment equals the valuation, there is no incentive to avoid underbidding. The Lazy Greedy based mechanism does not give an incentive to avoid overbidding, since the payment is constant.

This difficulty is an outcome of the structure of the valuation functions, but can be overridden by a convex combination of these mechanisms, i.e. a randomized mechanism that with probability p applies the MAX-GAP reductions, and with probability $1-p$ applies Lazy Greedy. This mechanism is universally truthful (a distribution over truthful mechanisms), has an approximation ratio of 3 in the worst case, and $3-p$ in expectation.

D Proofs from Section 5

Proof of Theorem 5.3: We show that the approximation described above is monotone in the jobs, meaning that when a job increases in weight, it cannot move to a slower machine. The existence of an ex-post mechanism follows from Theorem 5.2.

Given an increase in the weight of the i-th job, we observe the effect of the increase in stages: As long as the order of the sorted jobs does not change, the threshold of the fractional assignment can only increase, increasing the capacity of all machine. The total weight of jobs 1 to $i-1$ is unchanged, and therefore the machine where the first fraction (whether there are one or two fractions) of the i-th job is allocated is either a faster machine, or the same one. The integral solution rounds to the fastest machine with a fraction of the job, which is the machine that holds the first fraction, and therefore the chosen machine cannot be slower.

In breakpoints where the increase in the weight of the i-th job cause it to swap places in the sorted job vector with the job before it, the same allocation is generated, except for swapping between two jobs. Obviously, the i-th job is shifted towards the faster machines, and cannot be allocated to a slower one. □

Proof of Theorem 5.4: Observe the following setting: There are 2 machines with speeds 1 and 1.84, and 3 jobs with weights 1, 1.84 and x. If $x < 1$, the optimal assignment is to assign 1.84 and x on the fast machine and 1 on the slow machine. When $1 < x < 1.84$, job x swaps places with 1 and goes to the slow machine. For $x > 1.84$, the job returns to the fast machine (first with job 1, after $x > 4.23$ it remains alone).

The behavior of the optimal allocation is non-monotone, as job x moved at first from the fast machine to the slow machine. A monotone allocation algorithm would have to choose between always allocating x to the fast machine and between allocating x to the slow machine when it is small. If when $x = 0.543$ it is still allocated to the slow machine, then the best assignment has a makespan of 1.543, while the optimal one has 1.295, and the lower bound is approximately 1.19.

Otherwise, if x moved early to the fast machine, then it is still there when $x = 1.543$. At this stage, no assignment can do better than 1.838 with x on the faster machine, while moving it to the slower machine allows 1.543. Again, the ratio is above 1.19. □

E Discussion on Monotone Scheduling

The 5-approximation scheduling algorithm, taken from [1], sorts the jobs and allocates them starting from the fastest machine, until exceeding a given threshold (different from the fractional threshold used for the 2 approximation, but based on it). Monotonicity follows from similar ideas as in the proof of Theorem 5.3.

In contrast, not every rounding technique based on the fractional optimal fractional assignment is monotone in the jobs' perspective. The randomized 3 approximation in [4], as well as its improved 2 approximation in [5] calculate an optimal fractional assignment and then round the fractions of each job that is split between two machines with probabilities proportional to the sizes of the fractions. The valuation function is therefore the expected delay time, and its derivative is a convex combination of the types of the machines. Increasing a job's weight may cause the probability to shift towards the slower machine, which contradicts the monotonicity property.

We note that since a job's weight may unbounded, and since the payment function does not depend directly on this weight (only indirectly, through the outcome that is chosen), no payment function can always compensate job owners fully for their waiting time. Therefore, the assumption that job owners have a significantly large utility from having their jobs executed (large enough to overcome any waiting time, and any payment), is needed to assure that all valuations are positive, and that participation is a rational choice. One would expect the machines to get paid for the services, the job owners to pay for the execution, and the mechanism to be left with a small as possible positive balance from these transactions. The structure of such payment schemes and the conditions for their existence is an open question.