Truthful Mechanisms for One-Parameter Agents

Aaron Archer* Éva Tardos†

Abstract

In this paper, we show how to design truthful *(dominant strategy) mechanisms for several combinatorial problems where each agent's secret data is naturally expressed by a single positive real number. The goal of the mechanisms we consider is to allocate* loads *placed on the agents, and an agent's secret data is the cost she incurs per unit load. We give an exact characterization for the algorithms that can be used to design truthful mechanisms for such load balancing problems using appropriate side payments.*

We use our characterization to design polynomial time truthful mechanisms for several problems in combinatorial optimization to which the celebrated VCG mechanism does not apply. For scheduling related parallel machines ($Q||C_{\max}$), we give a 3-approximation mechanism based on randomized rounding of the optimal fractional solution. This problem is NP-complete, and the standard approximation algorithms (greedy load-balancing or the PTAS) cannot be used in truthful mechanisms. We show our mechanism to be frugal*, in that the total payment needed is only a logarithmic factor more than the actual costs incurred by the machines, unless one machine dominates the total processing power. We also give truthful mechanisms for maximum flow, $Q||\sum C_j$ (scheduling related machines to minimize the sum of completion times), optimizing an affine function over a fixed set, and special cases of uncapacitated facility location. In addition, for $Q||\sum w_j C_j$ (minimizing the weighted sum of completion times), we prove a lower bound of $\frac{2}{\sqrt{3}}$ for the best approximation ratio achievable by a truthful mechanism.*

1 Introduction

In economics, social choice theory addresses the problem of aggregating individuals' preferences to make a group decision. As indicated by Arrow's impossibility theorem for satisfactory voting systems [2], this is a thorny problem. It is further complicated by the possibility that the participants (usually called *players* or *agents*) might try to manipulate the system by misrepresenting their preferences. The field of *mechanism design* recognizes this game theoretic aspect and aims to arrange things so that a rational player will never find it in her self-interest to lie. Mechanisms that do this are called *strategyproof* or *truthful*.

Because of some stifling negative results that apply when the agents' preferences can be arbitrary, it is common to restrict the domain of preferences by assuming *additive separability*. Each agent is assumed to incur some intrinsic benefit or loss (called its *valuation*) depending on the outcome of the mechanism, and this valuation is expressible in some common unit of currency. The mechanism also makes *payments* to the agents in this currency, and each agent aims to maximize the sum of her valuation and payment. The most famous positive result in this area is the Vickrey-Clarke-Groves (VCG) mechanism [31, 4, 12].

Nisan and Ronen [24] considered discrete optimization problems in this game theoretic context, where the correct data is not directly available to the algorithm. Instead, there are several economic agents who each know some of the data and report it to the algorithm, but they might lie. Nisan and Ronen apply this framework to some standard problems in computer science, including shortest paths, minimum spanning trees, and scheduling on unrelated machines. They make a significant conceptual departure from the bulk of the economics literature in that the mechanism's objective function may have nothing to do with social welfare. Here, the agents' preferences are relevant to the goal of the mechanism only because both are tied to the agents' secret data, and because they determine the agents' strategies.

In this paper, we show how to design truthful (dominant strategy) mechanisms for problems where each agent's secret data is naturally expressed by a single positive real number. Our mechanisms allow general objective functions but restrict the form of the valuations. This is in contrast to VCG mechanisms, which allow arbitrary valuations but apply only to the *utilitarian* objective, which is the sum of the agents' valuations. The output of the mechanisms we consider will always define some set of *loads* placed on the

*Operations Research Department, Cornell University, Ithaca, NY 14853. Email: aarcher@orie.cornell.edu. Supported by the Fannie and John Hertz Foundation.

†Computer Science Department, Cornell University, Ithaca, NY 14853. Email: eva@cs.cornell.edu Research supported in part by NSF grant CCR-9700163 and ONR grant N00014-98-1-0589.

agents (e.g. the total size of jobs assigned to a machine, or the total flow through a network link). An agent's secret data will always be the cost she incurs per unit load, and this data will generally also have some physical significance (e.g. processing speed of the machine, or capacity of the link). Her goal is to maximize her profit, which is her payment minus her cost.

In Section 4 we characterize which output functions can be used to design truthful mechanisms. Our mechanisms use side payments to induce the agents to tell the truth. The idea is that if revealing the true parameter would result in an increased load for the agent, we can compensate for this increased load by a payment. However, for some output functions, no side payments can make the resulting mechanism truthful. We prove that the output functions that can be used in truthful mechanisms are exactly those in which the load assigned to an agent decreases monotonically as her announced cost increases, and the payment is given by an explicit formula involving an integral of the load curve. We will use this characterization to design truthful mechanisms for some non-utilitarian objective functions. Our characterization can also be used to design polynomial time truthful approximation mechanisms for utilitarian objective functions, when the VCG mechanism is impractical because the optimal output is hard to compute.

Our main example is the problem of scheduling jobs on related parallel machines to minimize makespan. This problem is commonly denoted $Q||C_{\max}$ in the scheduling literature, and is NP-hard. Each job j has a processing requirement p_j (the amount of work it represents), and each machine i runs at some speed s_i. If job j is scheduled on machine i, it takes p_j/s_i units of time to complete. The goal is to allocate the jobs to machines so that the last job finishes as soon as possible. Each machine is a distinct economic agent, which incurs a cost proportional to the total time it spends processing, and only the machine i knows the true value of s_i.

Our mechanism will ask each agent i to report its speed s_i, then allocate the jobs to machines using some algorithm and hand a payment P_i to each machine. The machines know the allocation algorithm and payment scheme in advance, and we assume each machine wants to choose its strategy (i.e. what speed it reports) in order to maximize its profit (the payment it receives minus the cost it incurs from running the jobs assigned to it). Our challenge as the mechanism designer is to find an allocation algorithm and payment scheme that yields a good makespan according to the reported rates and motivates rational agents to report their true rates. Notice that truthful mechanisms are not easy to design even with unlimited computational power. However, we also want to be able to compute the allocation and payments in polynomial time. Since $Q||C_{\max}$ is NP-hard, we will use an approximation algorithm to find the allocation.

In Section 5 we show several applications of our characterization of output functions that can be used to design truthful mechanisms. Designing a truthful mechanism now reduces to designing allocation algorithms with decreasing load curves. Our main application is for the problem $Q||C_{\max}$ discussed above, where we give a randomized 3-approximation mechanism. The problem is NP-complete, but a greedy load-balancing scheme provides a 2-approximation, and there is also a PTAS[1] based on rounding and dynamic programming [16]. However, these types of combinatorial approximation algorithms do not provide monotone work-curves, as the effect of changing the parameter of a single agent is hard to control throughout the algorithm. Our 3-approximation mechanism is based on randomized rounding using an optimal solution for the corresponding fractional problem. We also give an optimal truthful mechanism using unlimited computational power.

We use our characterization to design polynomial time truthful mechanisms for several other combinatorial problems. We design optimal mechanisms for maximum flow, $Q||\sum C_j$ (scheduling related machines to minimize the sum of completion times), optimizing an affine function over a fixed set, and special cases of uncapacitated facility location. We also get a constant approximation mechanism for the general uncapacitated facility location problem, provided the facility costs come from a bounded interval.

In contrast to our optimal truthful mechanism for $Q||\sum C_j$, we prove in Section 6 that no truthful mechanism can achieve an approximation ratio better than $\frac{2}{\sqrt{3}}$ for $Q||\sum w_j C_j$ (scheduling related machines to minimize the weighted sum of completion times).

In the problems discussed above and throughout the paper of Nisan and Ronen [24], the mechanism cares only about the outcome, and the payments exist only to induce truth-telling by the agents. In Section 7, we consider the issue of *frugality* – whether truthful mechanisms can keep the total payment low, by some measure. The shortest path mechanism of Nisan and Ronen behaves poorly in this regard, as some cases force the mechanism to pay $\Omega(n)$ times the cost of the shortest path, even when there is an alternate path of similar cost. Surprisingly, we show in [1] that *every* reasonable mechanism for this problem exhibits this bad behavior. In contrast, we show here that our mechanism for $Q||C_{\max}$ pays out only a logarithmic factor more than the actual costs incurred by the machines, so long as no single machine dominates the total processing power.

2 Terminology and notation

We now introduce our notation. There are m agents, represented by the index set I. Each agent i has some private

[1] A PTAS is a family of algorithms that, for fixed ϵ yields a $1+\epsilon$ approximation in polynomial time.

data consisting of a single parameter $t_i \in \mathbb{R}$ that describes the agent. We call this the agent's *true data* or *true value*. In the literature, it is sometimes called the agent's *type*. Only agent i knows t_i. Everything else is public knowledge. Each agent will report some value b_i to the mechanism. We call this the agent's *bid*. Let t denote the vector of true values, and b the vector of bids.

There is some allowable set of outcomes O that the mechanism is allowed to choose. The mechanism's *output algorithm* computes a function $o(b) \in O$ according to the agents' bids. The mechanism tries to maximize or minimize some function $g(o, t)$, but of course it does not know t directly. An algorithm that computes an output whose value is guaranteed to be within an α factor of the optimum is called an *α-approximation algorithm*. An *α-approximation mechanism* is one whose output algorithm is an α-approximation.

Each agent i incurs some monetary *cost*, $\mathrm{cost}_i(t_i, o)$, depending on the output and its private data. In order to offset these costs, the mechanism makes a *payment* $P_i(b)$ to agent i (a negative payment means the agent pays money to the mechanism). We assume that each agent i always attempts to maximize her *profit*, $\mathrm{profit}_i(t_i, b) = P_i(b) - \mathrm{cost}_i(t_i, o(b))$. Notice that agent i cares about the other agents' bids only insofar as they influence the outcome and the payment. While t_i is known only to agent i, the function cost_i is public.

In this paper we will assume that the costs have a particularly nice form. Namely, our outcomes o will assign some amount of *load* or *work* $w_i(o)$ to each agent i, and we will assume $\mathrm{cost}_i(t_i, o) = t_i w_i(o)$. Thus, agent i's private data t_i measures her cost per unit work.

Let b_{-i} denote the vector of bids, not including agent i. We sometimes write b as (b_{-i}, b_i). We say that *truth-telling* is a *dominant strategy* for agent i if bidding t_i always maximizes her profit, regardless of what the other agents bid. That is, $\mathrm{profit}_i(t_i, (b_{-i}, t_i)) \geq \mathrm{profit}_i(t_i, (b_{-i}, b_i))$ for all b_{-i} and b_i. We are interested in designing mechanisms such that truth-telling is a dominant strategy for each agent. We call such a mechanism *truthful*.

Formally, the mechanism $\mathcal{M}$ consists of the pair $\mathcal{M} = (o, P)$, where o is the output function and P is the payment *scheme*, i.e. the vector of payment functions P_i. We say that an output function *admits* a truthful payment scheme if there exist payments P such that the mechanism $\mathcal{M} = (o, P)$ is truthful. Some output functions admit a truthful payment scheme, and some do not. Our goal is to choose an output function that both admits a truthful payment scheme and achieves (or approximates) the optimal value of $g(o, b)$. In addition, we will usually require that we can compute the output and payments in polynomial time.

One could consider games in which the agents act in a more complicated way than just submitting a bid, instead selecting their courses of action from some broader class of *strategies*. We would then try to design mechanisms where each agent has a dominant strategy. However, it is easy to see that we lose no generality by restricting to mechansims in which agents directly reveal their parameters [22].

3 Related work

The economics and game theory literature contains an enormous body of work on mechanism design, also called *implementation theory* or the *theory of incentives*. See [22, ch. 23] or [26, ch. 10] for an introduction to the field, or the surveys [20, 13]. The Gibbard-Satterthwaite theorem [8, 28] is the main negative result, which states that truthful non-dictatorial mechanisms do not exist, when the players' domain of possible preferences is sufficiently rich.

In light of this, it is common to specialize by allowing side payments to the players, and assuming each player tries to maximize the sum of her payment plus her intrinsic valuation of the outcome. The celebrated Vickrey-Clarke-Groves (VCG) mechanism [31, 4, 12] is the main general positive result here. It handles arbitrary valuation functions, but only the utilitarian objective function, which maximizes the sum of the agents' valuations. Nevertheless, this objective function captures some interesting combinatorial problems [24], in addition to the more usual social welfare functions. For example, the shortest path in a graph with respect to edge costs maximizes social welfare because it minimizes the total cost incurred. For the utilitarian objective function, [11] proves that VCG is the only optimal truthful mechanism. In the case of one-parameter agents with some differentiability assumptions, [19] gives a simplified proof.

Much work has addressed computational issues surrounding VCG mechanisms. The main difficulty is that in many settings, the VCG mechanism is NP-hard to compute, since it requires finding an optimal output. One approach is to compute the output using a fast heuristic, and still try to use the VCG payment scheme. Such mechanisms are studied in [18], which gives three properties of the allocation algorithm that will allow the VCG payments to induce truth-telling. However, [25] exhibits a broad class of problems for which no mechanism that uses VCG payments is truthful, if its output algorithm is suboptimal. On the bright side, it also shows that if the mechanism lets the agents suggest ways to improve the output, these mechanisms can be made to satisfy a modified notion of truthfulness. A different approach, taken in [21], is to use a heuristic for the output and use a non-VCG payment scheme to induce truth-telling. They consider a simple type of auction in which computing the socially optimal assignment of goods is hard, and propose a greedy allocation algorithm with a non-VCG payment scheme. Even though their bids are two-dimensional, their problem essentially boils down to a one-parameter problem that is a special case of ours, as we show in the full version of this paper. While all of these papers depart from the standard VCG mechanism, they still aim to maximize the utilitarian objective, whereas we look at general objective functions.

Both [14] and [3] consider cases where the VCG mechanism can be computed in polynomial time, and address how to speed up this run time. While we are interested in polytime computable mechanisms, we make no attempt to optimize the run time.

Nisan and Ronen [24] applied the mechanism design framework to some standard optimization problems in computer science, suggesting general objective functions. Some subsequent algorithmically-oriented work involves cost-sharing mechanisms for multicast trees [7, 17], auctions for digital goods [10], and the use of auctions to elicit information [30]. The digital goods paper [10] is notable because it explicitly chooses *not* to maximize the social welfare. In their model, the marginal cost of creating an extra copy of the good is negligible, so the socially optimal allocation is to sell this good to everyone, but they do not do this because it generates no revenue. Highlighting the fact that revenue is a major concern, [27] suggests looking at auctions of a single good that do not necessarily maximize the social welfare, and characterizes all truthful mechanisms for this problem. His characterization is a special case of ours, for 0-1 load functions, and it also appears implicitly in [21] and [10]. The paper [30] also ignores the social welfare, instead attempting to compute various functions of the agent's valuations (such as the order statistics) using auctions of minimal communication complexity.

The paper of Nisan and Ronen [24] is closest in spirit to our work. While our main example is the problem $Q||C_{\max}$, scheduling on machines with speeds, their main focus is a similar NP-hard problem $R||C_{\max}$, scheduling on unrelated machines. In that problem, each machine has n items of private data, the amounts of time it would take for it to process each job (so our one-parameter results do not apply). The output is an allocation of jobs to machines, and the cost to a machine equals the total time it spends processing its jobs. Nisan and Ronen provide a simple truthful mechanism (consisting of a separate Vickrey auction for each job) that yields an m-approximation. They conjecture that *no* truthful mechanism has a better approximation guarantee, although the best lower bound they prove is 2. With strong additional restrictions on the types of payment schemes allowed, they prove a lower bound of m. Note the large gap between the best approximation factors known for a polytime algorithm (2) and for a truthful mechanism (m). We have a similar gap for $Q||C_{\max}$ between the PTAS of [16] and the truthful 3-approximation of Theorem 5.4.

The lower bound of 2 for $R||C_{\max}$ stands in contrast to our truthful (non-polytime) mechanism that exactly solves $Q||C_{\max}$. Ronen and Nisan do give a mechanism that solves $R||C_{\max}$ exactly, but only in a much stronger model in which the mechanism is allowed to observe the machines process their jobs and compute the payments *afterwards*, which makes it easy to penalize lying agents.

While revenue is heavily studied in auctions, the important corresponding issue of frugality for task allocation problems is not addressed in [24]. We contribute the first positive frugality result in this area.

4 Characterization of truthful mechanisms

Here we completely characterize which output functions do and do not admit truthful payment schemes for mechanism design problems where the cost to agent i is of the form $t_i w_i(o)$, its privately-known cost per unit work times the amount of work assigned. We also characterize the accompanying truthful payment schemes.

In order to motivate our theorem, we first assume all our functions are smooth, and use calculus to derive a formula for the payments and a condition on the output algorithm. Theorem 4.2 below shows that these conditions are actually necessary and sufficient to obtain a truthful mechanism, whether or not the functions are smooth. Let us assume that mechanism $\mathcal{M} = (o, P)$ is truthful and each payment $P_i(b_{-i}, b_i)$ and load $w_i(b_{-i}, b_i)$ is twice differentiable with respect to b_i, for all values of b_{-i}. We fix some agent i and derive a formula for P_i. Fixing the other agents' bids b_{-i}, we can consider the payment P_i, work w_i, and profit to be functions of just agent i's bid b_i. Since agent i's profit is always maximized by bidding truthfully, the derivative is zero and the second derivative is non-positive at t_i. Since this holds no matter what the value of t_i is, we can integrate to obtain an expression for P_i. Specifically, $\mathrm{profit}_i = P_i - t_i w_i$, so the first order condition gives

$$\left[\frac{dP_i(b_i)}{db_i} - t_i \frac{dw_i(b_i)}{db_i}\right]\Bigg|_{b_i=t_i} = 0 \qquad (1)$$

for all values of t_i. Integrating by parts gives

$$P_i(b_i) = P_i(0) + b_i w_i(b_i) - \int_0^{b_i} w_i(u)du. \qquad (2)$$

The second order condition says $P_i''(t_i) - t_i w''(t_i) \leq 0$. Using (1), this reduces to $w'(t_i) \leq 0$. Thus, in order to be truthful, the mechanism should have decreasing "work curves" w_i, and the payments should be given by (2).

Definition 4.1 *With the other agents' bids b_{-i} fixed, consider $w_i(b_{-i}, b_i)$ as a single-variable function of b_i. We call this the* work curve *or* work profile *for agent i. We say the output function o is* decreasing *if each of the associated work curves is decreasing (i.e. $w_i(b_{-i}, b_i)$ is a decreasing function of b_i, for all i and b_{-i}).*

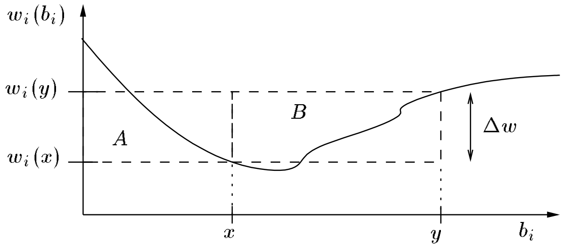

Figure 1. This graph shows why we cannot allow the work curve to increase.

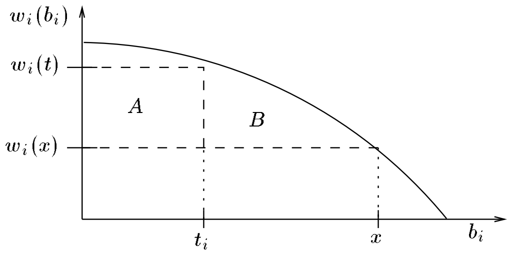

Figure 2. This picture shows why agent i never gains by overbidding.

Theorem 4.2 *The output function $o(b)$ admits a truthful payment scheme if and only if it is decreasing. In this case, the mechanism is truthful if and only if the payments $P_i(b_{-i}, b_i)$ are of the form*

$$h_i(b_{-i}) + b_i w_i(b_{-i}, b_i) - \int_0^{b_i} w_i(b_{-i}, u)du \qquad (3)$$

where the h_i are arbitrary functions.

Proof sketch: We explain the pictorial proofs of Figures 1 and 2. In Figure 1, A and B denote the areas of the rectangles they label. If i's true value is y, she would save cost $A + B$ by bidding x. If her true value is x, she would incur an extra cost of A by bidding y. To motivate truth-telling, the extra payment for bidding y instead of x should be at least $A+B$ and at most A, which is impossible since $B > 0$. Therefore, the work curve must decrease monotonically.

In Figure 2, the work curve is decreasing and the payments are given by (3). Geometrically, the payment to i if she bids x is a constant minus the area between the work curve and the horizontal line at height $w_i(x)$. If agent i's true value is t_i and she bids $x > t_i$, then her cost decreases by A from the decreased load, but her payment decreases by $A + B$. Since $B \geq 0$, she never benefits from overbidding. Similarly, she never benefits from underbidding.

To prove that all truthful payment schemes take the form (2) even when w_i is not smooth, we follow essentially the same reasoning as in the calculus derivation. ■

By Theorem 4.2, the only flexibility we have when designing a truthful payment scheme is in the additive constant terms $h_i(b_{-i})$. Consider the profit for a truth-telling agent if we set all of these terms to zero. Her cost is $t_i w_i(t_i)$, which exactly cancels out the second term in the payment formula (3). On net, she incurs a loss equal to the area under her work curve from zero to t_i. Since the agents cannot even hope for a profit under this scheme, they presumably would not participate in such a mechanism unless they were coerced. This motivates the following definition.

Definition 4.3 *A mechanism satisfies the* voluntary participation *condition if agents who bid truthfully never incur a net loss, i.e.* $\mathrm{profit}_i(t_i, (b_{-i}, t_i)) \geq 0$ *for all agents i, true values t_i, and other agents' bids b_{-i}.*

We want to design mechanisms satisfying voluntary participation. To do this, we need to set $h_i(b_{-i})$ to a constant that is larger than the area under the work curve to the left of t_i, no matter what the value of t_i is. If the total area under the work curve is infinite then no such constant exists. If the area is finite then we can set $h_i(b_{-i})$ to be this area, in which case a truth-telling agent i is guaranteed a profit equal to the area under the work curve to the right of t_i.

Theorem 4.4 *A decreasing output function admits a truthful payment scheme satisfying voluntary participation if and only if $\int_0^\infty w_i(b_{-i}, u)du < \infty$ for all i, b_{-i}. In this case, we can take the payments to be*

$$P_i(b_{-i}, b_i) = b_i w_i(b_{-i}, b_i) + \int_{b_i}^{\infty} w_i(b_{-i}, u)du. \qquad (4)$$

Remarks Our characterization of truthful mechanisms in terms of monotone decreasing outputs should not be confused with other uses of the word "monotone." In particular, a theorem in [6] characterizes truthful mechanisms in terms of "independent person-by-person monotonicity" (IPM). In our context, IPM would be a property of the output and payments together, whereas the beauty of Theorem 4.2 is that it allows us to focus only on the output function.

Our result yields the low-bid Vickrey auction as a special case. Here, the agents are bidding their costs to perform some job, so the load is either 0 or 1. The auction assigns the job to the lowest bidder, and pays her the amount of the second lowest bid. This is the same payment given by (4).

We also note that the obvious analog of Theorem 4.2 holds for the case where t_i denotes agent i's *benefit* per unit load, i.e. where $\mathrm{profit}_i(t_i, b) = P_i(b) + t_i w_i(o)$.

5 Designing truthful mechanisms

In this section we utilize Theorem 4.2 to design truthful mechanisms for several problems with one-parameter agents. Theorem 4.2 neatly separates the problem of designing the output function and the payment scheme – we just have to design an output that assigns decreasing work to agent i as her announced cost per unit work increases. Thus, the challenge is to find an output function o that optimizes (exactly or approximately) our function of interest, $g(o(b), b)$, such that the work curves are all decreasing.

5.1 Scheduling to Minimize Makespan

We consider the problem $Q||C_{\max}$, which we mentioned in the introduction. This problem is NP-hard, although there is a PTAS [16]. We are given n jobs and m machines. The jobs represent amounts of work $p_1 \geq \ldots \geq p_n$. The output is an assignment of jobs to machines. Machine i runs at some speed s_i, so it must spend $\frac{p_j}{s_i}$ units of time processing each job j assigned to it. The load on machine i is $w_i(b) = \sum p_j$, where the sum runs over jobs j assigned to i. Each machine incurs a cost proportional to the time it spends processing its jobs. For simplicity of presentation, we choose our unit of currency so that the constant of proportionality is one.[2] We take the true data to be $t_i = \frac{1}{s_i}$ so that the machines' costs are of the correct form, $\mathrm{cost}_i(t_i, o(b)) = t_i w_i(b)$. The mechanism's goal is to minimize the completion time of the last job on the last machine, i.e. $g(o(b), t) = C_{\max} = \max_i t_i w_i(b)$.

The mechanism design problem for $Q||C_{\max}$ contrasts sharply with the mostly negative results of Nisan and Ronen [24] (see Section 3). We show that truthfulness alone does not prohibit achieving the optimal allocation. Then we give a randomized polytime truthful mechanism that yields a 3-approximation for $C_{\max}$.

Definition 5.1 *A vector* $(w_1, \ldots, w_m)$ *is smaller than* $(\bar{w}_1, \ldots, \bar{w}_m)$ lexicographically *if, for some* i, $w_i < \bar{w}_i$ *and* $w_k = \bar{w}_k$ *for all* $k < i$.

Proposition 5.2 *There is a truthful mechanism (not polytime) that outputs an optimal solution for* $Q||C_{\max}$ *and satisfies voluntary participation.*

Proof sketch: Among the optimal allocations of jobs, our algorithm selects the one in which the load vector $(w_1, \ldots, w_m)$ is lexicographically minimum. Clearly, a machine raising its bid b_i (i.e. announcing it is slower) will not cause the allocation to change unless that machine is the bottleneck. In this case raising b_i will only cause machine i to get less work. Thus, the output function o is decreasing, so by Theorem 4.2 it admits a truthful payment scheme given by (3). As we just argued, $w_i(b_{-i}, \cdot)$ is constant except for jumps at the breakpoints where machine i becomes the bottleneck, so P_i is easily computed. Moreover, a sufficiently slow machine receives no work, so by Theorem 4.4 we can choose P to satisfy voluntary participation. ■

We now move to polytime mechanisms. We cannot simply use any existing approximation algorithm because the work assigned to agent i typically changes in complicated ways as her bid b_i changes. In particular, the PTAS in [16] relies on dynamic programming and rounding the job sizes. If a machine were to announce a slightly slower speed, causing it to receive a different set of jobs, the load could actually increase because of the rounding. The greedy load balancing algorithm also fails to be monotone. Consider scheduling three jobs on two machines of almost equal speeds, where $p_1 = 2$ and $p_2 = p_3 = 1 + \epsilon$. First job 1 is assigned to the faster machine, then jobs 2 and 3 both go on the slower machine, so it gets more work. We need to construct an approximation algorithm with decreasing work curves.

We first note that our problem is equivalent to bin packing with uneven bins, which leads to a lower bound on $C^*_{\max}$, the optimal makespan. This bound is implicit in the $\frac{3}{2}$-approximation algorithm of [29]. Given a guess T at the value of $C^*_{\max}$, we create a bin of size T/b_i for each machine i. The size of a machine's bin is the maximum load we can assign to it if the machine is to finish all its jobs by time T. Then $T \geq C^*_{\max}$ if and only if there exists an assignment of jobs to bins such that each bin is at least as large as the total size of all the jobs assigned to it. We can relax this requirement by allowing fractional assignments. A *fractional assignment* of jobs to bins consists of a partition of each job j into pieces whose sizes sum to p_j and an assignment of these pieces to the bins. A fractional assignment is *valid* if each bin is at least as large as the total size of all fractional jobs assigned to it, and every bin receiving a piece of a job is large enough to contain that entire job. The smallest T for which there exists a valid fractional assignment is a lower bound on $C^*_{\max}$. We now derive a formula for this lower bound.

If a valid fractional assignment exists, the following greedy algorithm clearly finds it. Number both the bins and jobs from largest to smallest, i.e. $b_1 \leq \ldots \leq b_m$ and $p_1 \geq \ldots \geq p_n$. Assign jobs $1, 2, \ldots, (k-1)$ to bin 1, where k is the first job that would cause the bin to overflow. Then assign to bin 1 a piece of job k exactly as large as the remaining capacity in bin 1. Continue by assigning jobs to bin 2, starting with the rest of job k, and so on.

Under what conditions is the greedy assignment valid? For each job j, let $i(j)$ denote the last bin that is at least as large as job j. The greedy assignment is valid if and only if, for each j, the total capacity of the first $i(j)$ bins is at least the total size of the first j jobs. So if the greedy assignment is valid and i is the last bin to which job j is partially as-

[2] Everything that follows still works if we let the constant vary from machine to machine, so long as the constants are known to the mechanism (not part of the private data).

signed, then $T \geq \max\{b_i p_j, \sum_{k=1}^{j} p_k / \sum_{l=1}^{i} 1/b_l\}$. Thus,

$$T_{LB} = \max_j \min_i \max \left\{ b_i p_j, \frac{\sum_{k=1}^{j} p_k}{\sum_{l=1}^{i} \frac{1}{b_l}} \right\} \quad (5)$$

is our lower bound on $C^*_{\max}$. For each job j, $i(j)$ is at least as large as the i that attains the min for job j in equation (5), so the first $i(j)$ bins are large enough to accommodate the first j jobs, and the greedy assignment is valid.

Lemma 5.3 *Sizing the bins according to T_{LB}, the greedy algorithm yields a valid fractional assignment such that each bin contains some number of full jobs plus at most two partial jobs.*

Now a natural algorithm suggests itself. Starting with the greedy assignment, round each split job to the faster of its two machines. The load on each machine is now the total size of the jobs fully assigned to that bin in the fractional assignment, plus at most one more job. Since the fractional assignment is valid, the rounded one overflows each bin by at most a factor of 2, so this algorithm is a 2-approximation.

Unfortunately, the algorithm does not yield decreasing work curves. Suppose $b_i p_j$ is the bottleneck term in (5) with $i > 1$, $j < n$, and job j exactly finishing off bin i. If i perturbs its bid upwards then T_{LB} increases, so job $j+1$ gets split across bins i and $i+1$ then rounded to i, increasing i's load. It seems difficult to overcome this problem with a deterministic algorithm, so we turn to randomized ones.

There is flexibility in defining what it means for a randomized algorithm to be truthful. Here we assume that each agent aims to maximize her *expected* profit. Thus, truth-telling is a dominant strategy for agent i if bidding t_i maximizes her expected profit regardless of what the other agents bid, and a mechanism is truthful if truth-telling is always a dominant strategy for each agent.[3] We now interpret w_i as the *expected* load on agent i. By Theorem 4.2, our randomized output algorithm admits a truthful payment scheme if and only if the expected load on i is a decreasing function of i's bid b_i. We choose our payment scheme to be given by formula (3) deterministically, but notice that it would be enough for our payments to be random variables whose expectation is given by this formula.

We use randomization to obtain a monotone work curve. Starting with our greedy fractional assignment of jobs to bins, we randomly assign jobs as follows. Job j is assigned to machine i with probability equal to the proportion of j that is fractionally assigned to bin i.

Theorem 5.4 *The randomized allocation described above admits a truthful payment scheme satisfying voluntary participation, and deterministically yields a polytime 3-approximation mechanism for $Q||C_{\max}$.*

Proof: Since the fractional assignment is valid and the rounding gives each machine at most 2 extra jobs, each bin is at most triply full. Thus, our allocation is a 3-approximation, no matter how the random choices turn out.

We now show that the expected load on each machine i decreases as i bids higher (i.e. claims to be slower). The expected load on i is precisely the load in the greedy fractional assignment. For full bins this is T_{LB}/b_i, for the (at most) one partially full bin it is the work left over from the full ones, and for the empty bins it is 0. Suppose some machine claims she is slower, replacing her bid b_i with αb_i, where $\alpha > 1$. This yields a new lower bound T'_{LB} from (5). Clearly $T'_{LB} \geq T_{LB}$, but also $T'_{LB} \leq \alpha T_{LB}$, since shrinking bin i by a factor of α then blowing up *all* bins by α would allow for a valid fractional assignment. Thus, the overall effect of increasing i's bid is to enlarge the other bins while shrinking bin i, so the greedy fractional assignment gives i less work. The expected load $w_i(b_{-i}, b_i)$ is a decreasing function of b_i, so by Theorem 4.2, we can design a truthful payment scheme. Since machines bidding sufficiently high receive no jobs, we can choose the payments to satisfy voluntary participation, by Theorem 4.4.

To compute the payments, we must compute the function $w_i(b_{-i}, \cdot)$ and the integral $\int_{b_i}^{\infty} w_i(b_{-i}, x)dx$. Let $T_{LB}(x)$ denote our lower bound when agent i bids x and the others bid b_{-i}. For small bids (fast speeds) bin i is full, so $w_i(b_{-i}, x) = T_{LB}(x)/x$. For large bids the load is zero. For the interval inbetween, the load is just the leftover work from the larger bins. Thus, we just need to find $T_{LB}(x)$. On different intervals it is either constant, of the form cx, or of the form $\frac{c}{d+1/x}$ (where c and d are constants), depending on which term is the bottleneck in formula (5). Breakpoints occur only when x coincides with another agent's bid or when two of the terms inside the braces in (5) (considered as a function of x) cross. Thus the number of intervals is polynomial and the integral over each interval is a closed-form expression, so the mechanism is polytime computable.[4] ■

This mechanism has the peculiar feature that we introduced the randomness not to improve the objective function, but to cause the expected load to decrease monotonically as the bid increases.

5.2 Affine Functions of the Loads: LP and Uncapacitated Facility Location

Here we consider a general class of problems admitting truthful mechanisms. The main result is the existence of a

[3] A more restrictive definition used in [24] requires truth-telling to be the best strategy, regardless of the outcome of the algorithm's random coin flips.

[4] The closed form expressions giving the payments in the mechanism described above may contain natural logarithms, so our model of computation must allow us to compute these if we wish to obtain a numerical answer for the payment.

truthful mechanism; at this level of generality we cannot say whether we can compute it. Suppose the mechanism wishes to minimize some affine function of the loads

$$g(o, b) = d(o) + \sum_{i \in I} c_i(b_i) w_i(b), \tag{6}$$

where d is some function of the output not depending on the bids, $c_i(b_i)$ is an increasing function for each agent i, and the set of allowable outputs o does not depend on the bids. One special case is linear programming, where some of the decision variables are the loads w_i, $d(o)$ is the part of the objective depending on the other variables, the cost coefficient on w_i depends on i's bid b_i, and the feasible set is given by linear inequalities *not* depending on the bids. Another special case is uncapacitated facility location, where each facility is an agent whose private data is the facility cost, and $d(o)$ is the (publicly known) transportation cost.

Assume that, for every set of bids, there exists an optimal solution. Fix an ordering of the agents.

Theorem 5.5 *For the problem stated above, if each coefficient $c_i(b_i)$ is strictly increasing in b_i, then any optimal output function $o(b)$ admits a truthful payment scheme. Otherwise, any output function that gives an optimal solution whose vector of loads $(w_1, \ldots, w_m)$ is lexicographically minimal admits a truthful payment scheme.*

Proof sketch: Raising i's bid only raises the cost of w_i, which can only lower the optimal w_i, since the set of feasible outputs is fixed. ■

We can apply this theorem to the uncapacitated facility location problem. In the standard problem, we are given a set of facilities and a set of customers. We need to select some subset of the facilities to open, and then assign each customer to be served by some open facility. Each facility has a *facility cost* associated with opening it, and for each customer j and facility i, there is a *transportation cost* incurred if customer j is assigned to facility i. There are no capacities, so an open facility may serve an arbitrary number of customers. The goal is to minimize the sum of the facility costs and the transportation costs. As explained above, we can apply Theorem 5.5.

Theorem 5.6 *Any algorithm that solves the uncapacitated facility location problem optimally admits a truthful payment scheme.*

Uncapacitated facility location is NP-hard, so we cannot expect to find a polynomial time algorithm to solve it. However, there are some special cases that can be solved in polynomial time, such as if the facilities and customers lie on a line, circle, or tree. Slight generalizations of these cases are solved in [9]. In these cases, we can also compute each payment easily, as it just involves finding the threshold bid at which a facility would no longer be open (i.e. where $w_i(b_{-i}, \cdot)$ jumps from 1 to 0).

We can use the algorithm of [23] as the basis for a truthful mechanism for facility location in an arbitrary metric space. Since it considers each facility one at a time and opens it with probability inversely proportional to its cost, the load curve decreases with the bid.

Theorem 5.7 *There is a constant-approximation truthful mechanism for uncapacitated facility location where every customer point is also a potential facility, and all the facility costs are known to lie in an interval $[c_1, c_2]$, where c_2/c_1 is bounded by a constant.*

5.3 Sum of Completion Times and Max Flow

Here we briefly mention two other problems to which we can apply Theorems 4.2 and 4.4.

The problem of scheduling on related machines to minimize the sum of completion times, commonly denoted $Q||\sum C_j$, can be solved optimally by a simple algorithm [5]. We can prove that this algorithm results in work curves that decrease monotonically to zero. Thus, by Theorems 4.2 and 4.4, we obtain a truthful mechanism that solves $Q||\sum C_j$ exactly and satisfies voluntary participation.

Now we consider the maximum flow problem. We are given a directed graph with source and sink nodes. Each edge e is an agent and has a finite non-zero capacity c_e, known only to itself. The mechanism wishes to find a maximum flow from source to sink respecting the capacity constraints on all edges. We assume that each edge incurs a cost equal to the congestion on that edge. That is, if we send f_e units of flow on an edge, that agent incurs a cost of f_e/c_e. In order for this problem to fit the form we have been considering (i.e. cost equals $t_e w_e$, the private data times the load), we take the private data to be $t_e = 1/c_e$, and the load on edge e to be $w_e = f_e$. Thus, in truthful mechanisms the flow on edge e must decrease as its announced capacity decreases.

We can guarantee this property by using max flows that are lexicographically minimal (in the sense of Theorem 5.5). We can compute such a flow using m max flows (where m is the number of edges). There is a closed-form expression for the payments in terms of the flow, so we can solve max flow exactly in polynomial time. Unfortunately, the work curves have infinite integrals, so we cannot satisfy voluntary participation. However, we could choose to ignore edges of capacity below ϵ/m, in which case the work curves would drop to zero at that point, so we could satisfy voluntary participation and obtain a flow within an additive ϵ of optimal.

6 Lower bounds

We can use our characterization theorem to prove lower bounds on approximability by truthful mechanisms. In particular we consider scheduling on machines with speeds to minimize the weighted sum of completion times, usually denoted $Q||\sum w_j C_j$. The idea behind the lower bound is that in an optimal allocation, the fast machines should get the important jobs, whereas in a truthful allocation the fast machines should get the bulk of the work. If we arrange the job weights w_j so that these two principles conflict, then the truthful allocation will be suboptimal.

Theorem 6.1 *No truthful mechanism for $Q||\sum w_j C_j$ can achieve an approximation ratio better than $\frac{2}{\sqrt{3}}$, even on instances with just two jobs and two machines.*

Proof: Consider an instance with two jobs and two machines. Job 1 has weight and processing requirement 1, while job 2 has weight w and processing requirement p, where $p > 1$. Suppose machine 1 runs at speed 1, and machine 2 runs at speed s. In order for our mechanism to be truthful, the load on machine 2 must increase monotonically as its bid decreases (i.e. as its speed increases). To show that any truthful mechanism is suboptimal, we must select p and w so that in the optimal schedule, the load on machine 2 is non-monotone in s. To this end, we set p and w such that $pw < 1$.

In the optimal schedule, for small s, both jobs will be assigned to machine 1. As we raise s, the jobs will eventually be split between machines, the machines will swap jobs, then eventually machine 2 will get both jobs, for large enough s. When the jobs are split, the job with larger weight-processing product goes on the faster machine. Since $pw < 1$, job 2 goes to the slower machine. Thus, the load on machine 2 is non-monotone in the optimal assignment. It is easy to check that the optimal assignment is to give both jobs to machine 1 when $s \in (0, \frac{p}{p+1})$, put job 1 on machine 1 and job 2 on machine 2 when $s \in (\frac{p}{p+1}, 1)$, swap the jobs when $s \in (1, 1 + \frac{1}{p})$, and put both jobs on machine 2 for $s \in (1 + \frac{1}{p}, \infty)$. Whenever both jobs are on the same machine, job 1 goes first (by Smith's rule).

Now we reason about the behavior of any truthful mechanism, as s increases from 0 to ∞. If the mechanism is to achieve a finite approximation ratio, then it must assign both jobs to machine 1 when $s \ll 1$ and both jobs to machine 2 when $s \gg 1$. For intermediate values of s it may split the jobs. The key is that if machine 2 gets job 2 for some value of s, then it must keep job 2 for all larger values of s, since the load may only increase and $p > 1$. The optimal schedule violates this. Because $pw < 1$, we have $\frac{1}{1+w} \in (\frac{p}{p+1}, 1)$. Thus, when $s = \frac{1}{1+w}$, the optimal schedule gives job 2 to machine 2, but when $s = 1 + w$, it gives only job 1 to machine 2. Therefore, the truthful mechanism is suboptimal either when $s = \frac{1}{1+w}$ or when $s = 1 + w$. We discuss the first case. The second is symmetric.

Since we are assuming the truthful mechanism does not split the jobs the optimal way when $s = \frac{1}{1+w}$, the best possible schedule it can use is to split the jobs the other way. Thus, the mechanism gives a schedule with objective function value at least $pw + (1 + w)$, while the optimal schedule has value $1 + pw(1 + w)$. For any fixed p, the ratio of these two values is maximized when $w = \frac{-p+\sqrt{2p^2+p}}{p^2+p}$. The ratio increases as $p \downarrow 1$, approaching a limit of $\frac{2}{\sqrt{3}}$. ■

7 Frugal mechanisms

To this point, we have viewed payments only as an inducement to the agents to bid truthfully, while the mechanism cared about minimizing some unrelated objective function. We are also interested in mechanisms whose payments are small by some measure. We describe these qualitatively as *frugal*. Since subtracting a constant from the payment functions preserves truthfulness, it is only interesting to consider mechanisms satifying voluntary participation. The total cost incurred by the agents is then a lower bound on the total payment.

We show in [1] that the shortest path mechanism of [24] can be forced to pay $\Omega(n)$ times the cost of the shortest path, even when there is an alternate path of similar cost. Surprisingly, this pitfall is intrinsic to the problem, since we also prove that *every* reasonable mechanism exhibits this bad behavior. In contrast, our 3-approximation algorithm for $Q||C_{\max}$ never pays more than a logarithmic factor more than the expected costs incurred by the machines, provided no single machine dominates the processing power.

Theorem 7.1 *The payment to each machine $i \geq 2$ is at most $T_{LB}(1+2\ln\frac{x}{b_i})$, where $x = b_1(p_1 + \ldots + p_n)/p_n$ and T_{LB} is given by* (5)*. The payment to machine* 1 *is at most $T_{LB}(\frac{b_2}{b_1} + 2\frac{b_2}{b_1}\ln\frac{x_1}{b_2})$, where $x_1 = b_2(p_1 + \ldots + p_n)/p_n$.*

Proof sketch: The payment to machine i consists of two terms (formula (4)). The first term $b_i w_i(b_i)$ exactly compensates machine i for its expected cost, which is T_{LB} for all machines that are full in the greedy fractional assignment of Lemma 5.3. The second term, $\int_{b_i}^{\infty} w_i(u)du$, is the (expected) profit. We always have $w_i(u) \leq T_{LB}(u)/u$, where $T_{LB}(u)$ is the lower bound on the optimal makespan $C^*_{\max}$ computed in formula (5). Equality holds as long as bin i is full in the fractional assignment. But (for $i > 1$) $T_{LB}(u)$ stays approximately constant, since machine i constitutes at most half of the processing power, so decreasing its speed can at most double $T_{LB}(u)$. That is, $T_{LB}(\infty) \leq 2T_{LB}$. Moreover, $w_i(u)$ drops to zero at some point y. Thus, the integral is at most $2T_{LB}\ln\frac{y}{b_i}$.

This argument breaks for machine 1 if it is much faster than all the rest combined, since the load on this machine stays nearly constant as its announced speed decreases to that of the second fastest machine. Therefore, our bound on its profit depends on the ratio between the speeds of the fastest two machines.

It then remains to bound y. If machine $i > 1$ has speed $1/x = p_n/(b_1(p_1 + \ldots + p_n))$, then placing even the smallest job on i would take longer than processing all jobs on machine 1. Thus, when i bids x, it gets no work, so $y \leq x$. Similarly, when machine 1 bids x_1, it gets no work. ■

Corollary 7.2 *If the sizes of all n jobs differ by a factor of at most r_1, and the speeds of the two fastest machines differ by a factor of r_2, then the payment given by the mechanism exceeds the total expected cost incurred by all the agents by a factor of at most $O(r_2 \ln(r_1 n))$.*

Proof: We bound the ratio of payment to expected cost, machine by machine. For each machine i whose bin is full in the greedy fractional assignment, the cost is T_{LB}. There is at most one machine with a partially full bin. We cannot bound its ratio, but clearly its payment is no greater than that of the next faster machine. For machine 1, the ratio is at most $r_2(1 + 2\ln(r_1 n))$, and for each other full machine the ratio is at most $1 + \ln(\frac{r_1 n}{r_2})$. ■

Acknowledgments We thank David Shmoys for several helpful discussions on scheduling, and the anonymous reviewers for many excellent suggestions.

References

[1] A. Archer and É. Tardos. Frugal path mechanisms. Unpublished manuscript.

[2] K. Arrow. *Social Choice and Individual Values*. Wiley, 1951.

[3] S. Bikhchandani, S. de Vries, J. Schummer, and R. Vohra. Linear programming and vickrey auctions. Unpublished manuscript, 2001.

[4] E. Clarke. Multipart pricing of public goods. *Public Choice*, 8:17–33, 1971.

[5] R. Conway, W. Maxwell, and L. Miller. *Theory of Scheduling*. Addison-Wesley, 1967.

[6] P. Dasgupta, P. Hammond, and E. Maskin. The implementation of social choice rules: Some general results on incentive compatibility. *Rev. Econ. Stud.*, 46:185–216, 1979.

[7] J. Feigenbaum, C. Papadimitriou, and S. Shenker. Sharing the cost of muliticast transmissions. In *STOC*, 218–227, 2000.

[8] A. Gibbard. Manipulation of voting schemes: A general result. *Econometrica*, 41(4):587–601, 1973.

[9] M. Goemans and M. Skutella. Cooperative facility location games. In *SODA*, 76–85, 2000.

[10] A. Goldberg, J. Hartline, and A. Wright. Competitive auctions and digital goods. In *SODA*, 735–744, 2001.

[11] J. Green and J.-J. Laffont. Characterization of satisfactory mechanisms for the revelation of preferences for public goods. *Econometrica*, 45(2):427–438, 1977.

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

[13] T. Groves. On theories of incentive compatible choice with compensation. In Hildenbrand [15], 1–29.

[14] J. Hershberger and S. Suri. Vickrey pricing in network routing: Fast payment computation. In *FOCS*, 2001.

[15] W. Hildenbrand, ed. *Advances in Economic Theory*. Cambridge UP, 1982.

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

[17] K. Jain and V. Vazirani. Applications of approximation algorithms to cooperative games. In *STOC*, 364–372, 2001.

[18] N. Kfir-Dahav, D. Monderer, and M. Tennenholtz. Mechanism design for resource bounded agents. In *Intl. Conf. on MultiAgent Systems*, 2000.

[19] J.-J. Laffont and E. Maskin. A differential approach to dominant strategy mechanisms. *Econometrica*, 48(6):1507–1520, 1980.

[20] J.-J. Laffont and E. Maskin. The theory of incentives: An overview. In Hildenbrand [15], 31–94.

[21] D. Lehmann, L. O'Callaghan, and Y. Shoham. Truth revelation in rapid, approximately efficient combinatorial auctions. In *ACM Conf. on Electronic Commerce*, 96–102, 1999.

[22] A. Mas-Collel, M. Whinston, and J. Green. *Microeconomic Theory*. Oxford UP, 1995.

[23] A. Meyerson. Online facility location. In *FOCS*, 2001.

[24] N. Nisan and A. Ronen. Algorithmic mechanism design. In *STOC*, 129–140, 1999.

[25] N. Nisan and A. Ronen. Computationally feasible VCG mechanisms. In *ACM Conf. on Electronic Commerce*, 242–252, 2000.

[26] M. Osborne and A. Rubinstein. *A Course in Game Theory*. MIT Press, 1994.

[27] A. Ronen. Algorithms for rational agents. In *Conf. on Current Trends in Theory and Practice of Informatics*, 56–70, 2000.

[28] M. Satterthwaite. Strategy-proofness and Arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *J. Econ. Theory*, 10:187–217, 1975.

[29] D. Shmoys, J. Wein, and D. Williamson. Scheduling parallel machines on-line. *SIAM J. Comp.*, 24(6):1313–1331, 1995.

[30] Y. Shoham and M. Tennenholtz. On rational computability and communication complexity. *Games and Econ. Behavior*, 35:197–211, 2001.

[31] W. Vickrey. Counterspeculation, auctions and competitive sealed tenders. *J. Finance*, 16:8–37, 1961.